Utilization-based Power Modeling of Modern Mobile Application Processor

Abstract **Power modeling of a modern mobile application processor (AP) is challenging because of its complex architectural characteristics. Previous work on power modeling barely supports the modern mobile AP equipped with multicore CPU and other hardware components. In this paper, we propose a power model of a mobile AP used in modern smartphones. The proposed model estimates the power consumption of multicore CPU and GPU, based on the utilization information acquired from each hardware component. We evaluated the proposed model using our test applications as well as real applications. The evaluation results indicate that our power model estimates the power consumption of an AP, with an average error of 7.1% and 4.1% for CPU and GPU, respectively.**

Keywords: Application processor, multicore CPU, integrated GPU, power modeling, mobile device

I. INTRODUCTION

The SoC-based mobile application processor (AP) typically includes CPU and GPU cores, multimedia processors, and other necessary hardware components to effectively support mobile applications. In particular, with the availability of a multicore-based mobile AP, the performance of mobile devices such as smartphones and tablets has improved significantly over the years.

The processor power model is important for studies that predict and estimate an application's energy behavior. However, the complicated nature of modern mobile AP architecture makes power modeling difficult; therefore, the power estimation of a mobile application is also difficult. In fact, despite active research on power modeling and estimation, current studies [1, 2] rarely support a multicore-based mobile AP due to lack of effective power models. Previous research [3–6] analyzed the power characteristics of mobile devices, including CPU; however, no consideration has been given to multicore CPU or GPU. Recently, multicore CPU has been studied for power management [7]. The focus is on establishing a new policy for Dynamic Voltage and Frequency Scaling (DVFS) by controlling active cores, rather than analyzing the power consumption of the processor. Kim et al. [8] proposed power models of multicore CPU for mobile devices, but other subsystems on mobile AP such as GPU were not considered in the study. Recent mobile platforms such as Android actively utilize GPU for improving user experience. Therefore, power modeling of the mobile AP should be expanded to support complex subsystems employed in modern architecture.

The OS-level energy management techniques [9, 10] and application-level power monitoring systems [1, 2] require power models to estimate the power consumption for each running process. To build an energy-efficient system based on energy accounting for each process, the power model should be able to estimate power consumption accurately at each process level. [11, 12, 13, 22] proposed accurate power models considering circuit designs and hardware performance counters; yet their scheme does not estimate process-level power consumption. Pathak et al. [6] proposed a finite state-machine-based power model for analyzing power consumption of running processes in [14]. However, the FSM power model is not utilization-based; hence, it is difficult to estimate power consumption for processes that run simultaneously.

Accurate modeling of mobile AP power consumption is an important technique that is necessary for constructing energy-efficient mobile systems. In this paper, we propose a power model of a mobile AP used in modern smartphones. The proposed model is utilization-based and supports multicore CPU and GPU. The model also considers power behaviors that are related to system-level power management, such as DVFS, OS-level power governor [15], and low-power hardware states. To estimate process-level power consumption of a mobile AP, our model requires only per-process hardware utilization information that could be acquired at the software level. The proposed model contributes to the development of advanced techniques [1, 2] that require the estimation of process-level power consumption. The following are the main technical contributions of our study:

- We present a new utilization-based power model of a mobile AP characterized by multicore CPU and GPU, which are commonly found in modern mobile devices.
- The proposed power model predicts the process-level power consumption by using hardware utilization information. To the best of our knowledge, our study is the first to construct a power model of a multicore-based mobile AP to support estimation of process-level power consumption.

The remainder of this paper is organized as follows: Section II presents our methodologies for modeling the power consumption of multicore CPU and GPU. Section III presents the evaluation results. Section IV discusses previous studies

related to power modeling and estimation. Section V concludes the paper.

II. METHODOLOGY

Fig. 1 shows a block diagram of the Exynos-4412 [16] multicore SoC, which our study is based upon. The AP implements quad-core CPU based on ARM Cortex-9 and includes specialized subsystems such as GPU and multimedia processors. In this study, we specifically focus on the power consumption characteristics of both CPU and GPU, which account for a large proportion of overall AP power consumption. For the experimental analysis, we used a Samsung Galaxy S3 GT-I9300 (S3; Samsung Exynos-4412, 4.5-inch AMOLED display) [17] smartphone.

A. Power Analysis of Multicore CPU

The power consumption of an AP depends on its operating frequency. The operating system dynamically adjusts the frequency to conserve energy. This technique is commonly known as DVFS. According to the CPU frequency governor, DVFS sets the CPU frequency using a predefined frequency-voltage table. In the case of the S3, the frequency-voltage table is classified with 13 scaling steps, and all the cores operate at the same frequency. The power consumption of the CPU depends on the utilization at a particular frequency. The utilization is the percentage of processor usage involved in running a task in unit time. Multiple power modes are available on Exynos-4412; however, the S3 employs two power modes that are represented as two-level C-states on the Linux kernel. The power modes are managed by the idle governor of the Linux kernel. If there is one active core and its utilization is zero, the idle governor sets the power mode as C-state 1 from C-state 0; then, the processor goes into a low-power mode by stopping the internal clock.

In the case of a single-core CPU, the power model considers operating frequency and utilization [4, 5]. The power consumption of a multicore CPU varies with the number of active cores, and the default CPU frequency governor of the S3—"*pegasusq*"—controls the activation of each core by observing the utilization and job queues of the scheduler. Hence, we analyze the power consumption characteristics by considering each core as an individual subsystem. By considering the individual core, the following simplified CPU power model is obtained:

$$P_{CPU} = \sum_{c=0}^{n} \left(\beta^{c}_{freq_i} \times u^{c} + \beta^{c}_{idle_i} \times (1 - u^{c})\right) + P_{base},$$

$$\mathrm{c} = 0,1,2 \cdots, n, i = 0,1,2 \cdots, m, 0\% \le u \le 100\%, \quad (1)$$

where c is the index of the active core, i the index of the frequency-voltage table, $\beta^{c}_{freq_i}$ the maximum power consumption of the c-th core in the i-th operating frequency, u^{c} the utilization of the c-th core, $\beta^{c}_{idle_i}$ the idle power consumption of the c-th core in the i-th operating frequency, and P_{base} the base power consumption for the operating CPU.

To determine β_{freq} and β_{idle} upon varying c and i in (1), we conducted experiments to measure the change of power that depends on the number of cores and operating frequencies. Further, we implemented a test application that controls the activation state of cores and operating frequencies to set a specific power consumption condition. The test application is an attempt to stabilize power consumption at each condition for over 20 seconds by maintaining utilization and the C-state. To minimize the power consumption of other hardware modules, the test application turns off display, GPU, all network interfaces, and the sensors in the system.

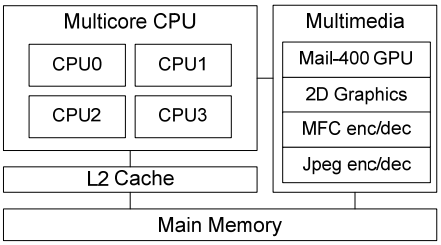


Fig. 1. Block diagram of the important components of Exynos-4412

The test application performs the Coremark benchmark [18] to stabilize the high CPU utilization. The computational complexity and memory usage of the Coremark are not modified, and the experiment is conducted with the utilization of all cores maintained at 100% and the memory usage maintained at 50%. To measure the idle power consumption for every case, the test application freezes both operating frequency and active cores and then maintains the utilization of all cores to almost 0. Additionally, we wrote a new *idle governor* to prevent CPU from entering C-state 1. All results of the experiment were measured using Monsoon Power Monitor [19].

Fig. 2 shows the results of per-frequency power consumption of the CPU depending on the number of active cores when utilization is 100%. As the operating frequency increases, the power consumption also increases in the form of a quadratic curve. In addition, the power consumption at the same operating frequency linearly increases with the number of active cores. Therefore, the estimated power coefficient β_{freq} of each core is almost the same. Fig. 3 shows the measurement results of per-frequency idle power consumption depending on the number of active cores. The power

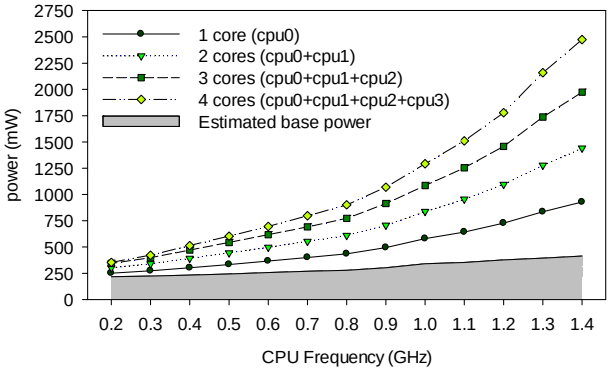
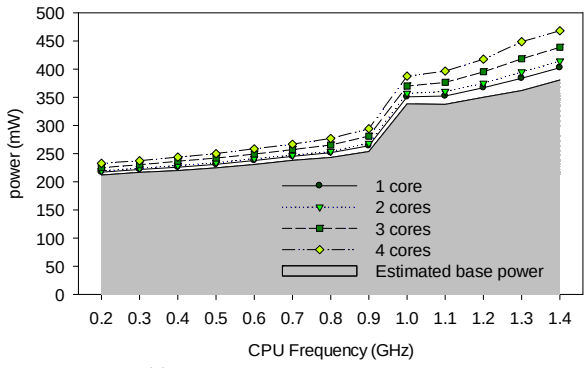
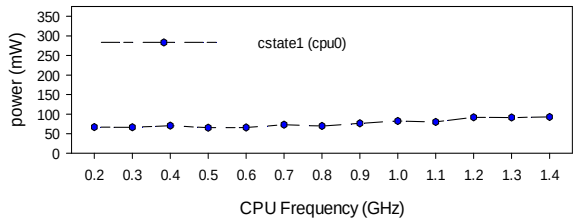


Fig. 2. The measurement result of power consumption versus operating frequency per number of active cores



(a) Idle power consumption on C-state 0



(b) Idle power consumption on C-state 1

Fig. 3. The measurement result of idle power consumption versus operating frequency per number of active cores

consumption of C-states is measured separately through the idle governor control. Note that C-state 1 is available when there is only one active core (CPU 0). The power consumption at the same operating frequency is proportional to the number of active cores.

B. Utilization-based CPU Power Model

To estimate the CPU power consumption of running processes, we need to obtain P_{base} in (1). Because all the cores cannot be inactivate simultaneously, it is not possible to measure P_{base} directly; hence, we estimated P_{base} by considering the relationship between the number of active cores and the increase in power consumption. According to the results of the experiment in Section III-A, there is a consistent increase in the power consumption with the number of active cores at a certain frequency; hence, we assume that the power consumption of each core is the same. The power consumption of the first core (CPU 0) includes the base power consumption; thus, we estimated P_{base} by subtracting the average of the increased power consumption from the CPU power consumption. The red area represents the estimated base power in Fig. 2; Fig. 3(a) and Table I list the power coefficient values for each case. In summary, we model the power consumption of a multicore CPU in the following manner:

$$P_{CPU} = \begin{cases} P_{cstate0} + P_{base}, & if \text{ condition} \\ P_{cstate1}, & else \end{cases} \quad (2)$$

$$\text{where the C} - \text{state level of CPU 0 is 0}$$

$$P_{cstate0} = \sum_{c=0}^{n} \{\beta_{freq_i} \times u^c + \beta_{idle_i} \times (1 - u^c)\},$$

$$P_{base} = \beta_{freq_i}^{base} \times u + \beta_{idle_i}^{base} \times (1 - u),$$

TABLE I. POWER COEFFICIENT VALUES OF EXYNOS-4412 CPU

i	$\beta_{freq_i}^{base}$	β_{freq_i}	$\beta_{idle_i}^{base}$	β_{idle_i}	$\beta_i^{cstate1}$
0	413.15	515.16	380.66	21.88	93.05
1	394.74	440.99	362.07	21.68	91.7
2	376.78	350.38	350.14	16.89	92.12
3	352.72	289.63	337.46	14.82	80.22
4	340.85	237.97	336.48	12.30	82.6
5	302.47	191.85	253.53	10.20	76.56
6	279.08	155.80	243.14	8.41	69.82
7	268.71	132.30	237.92	7.19	73.07
8	257.21	109.43	230.60	6.96	65.87
9	242.85	90.16	224.59	6.31	65.49
10	231.84	70.45	219.86	5.97	70.54
11	223.61	49.54	216.53	5.18	66.44
12	216.65	34.29	211.80	5.21	67.11

TABLE II. THE FREQUENCY-VOLTAGE SCALING TABLE OF MALI-400 GPU

Frequency Index	Frequency(MHz)	Voltage(V)	Utilization Threshold	
			Up	Down
0	160	0.875	70	0
1	266	0.900	90	62
2	350	0.950	90	85
3	440	1.025	100	90

$$P_{cstate1} = \beta_i^{cstate1}$$

$$\text{c} = 0, 1, 2 \cdots, n, i = 0, 1, 2 \cdots, m, 0\% \leq u \leq 100\%$$

where c is the index of active core, i the index of the frequency-voltage table, β_{freq_i} the maximum power consumption of each core, and β_{idle_i} the idle power consumption of each core in the i-th operating frequency, u^c the utilization of the c-th core, and $\beta_i^{cstate1}$ the power consumption at C-state 1 at the i-th operating frequency.

C. Power Consumption Analysis of GPU

The Exynos-4412 SoC is equipped with ARM Mali-400 GPU [20] cores, which has four levels of operating frequencies. All cores should turn on and off simultaneously, and the operating frequency is controlled based on the utilization at every second. Table II shows the frequency-voltage scaling table of Mali-400 GPU.

To analyze the characteristics of GPU power consumption, we implemented a test application using OpenGL ES to control utilization. The GPU test application sets a certain operating frequency, then changes utilization by adjusting the number of 3D cube objects and their spin speeds. Separately, we also implemented a Linux kernel module to minimize the power consumption caused by the display. When the GPU test began, the test application set the device to airplane mode and our kernel module turned the screen off. Since CPU power consumption is inevitable while measuring power consumption, we used the CPU power model expressed in (2) to estimate the GPU power consumption. During the GPU test, we simplified CPU behavior by fixing the operating frequency at maximum and allowing only one core to run. We logged both the CPU and GPU utilization at every second. The CPU utilization was

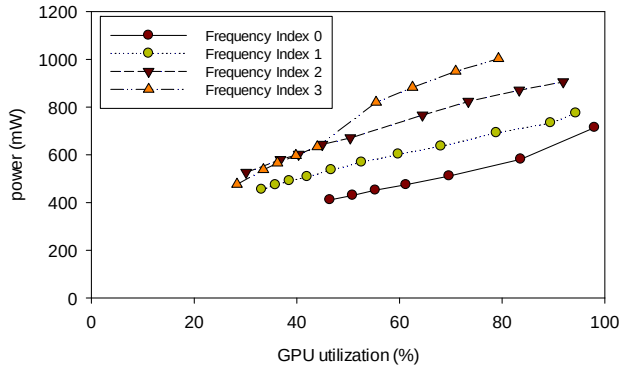


Fig. 4. The measurement result of GPU power consumption versus utilization

approximately 15% for every measurement. By subtracting the CPU power estimation result from the total power consumption, we estimated the GPU power consumption versus GPU utilization for every available frequency.

Fig. 4 presents the estimation results of GPU power consumption with varying utilization. All the experimental results are the average value of over five repetitions at each frequency and are conducted within the available utilization value that is generated by our GPU test application. In each operating frequency, GPU power consumption increases linearly with utilization; thus, it can be stated that power consumption increases with operating frequency. However, in the case of frequency index 3 (440 MHz), GPU power consumption is lower than that of other frequencies where the utilization is under 43%.

D. Utilization-based GPU Power Model

According to the result of the experiment in Section III-C, the power consumption of GPU increases proportionally with utilization at a certain frequency; therefore, we applied a linear regression for the results of each frequency in Fig. 4. Consequently, we model the power consumption of GPU in the following manner:

TABLE III. ESTIMATION ERRORS OF CPU POWER MODEL

# of active cores	Mean err (mW)	Std. Dev	Std. Err ($Std.Dev/\sqrt{n}$)	Estimation Error (%)
1	21.3	38.2	1.1	4.8
2	30.0	78.4	2.7	5.8
3	39.9	133.1	5.0	7.1
4	10.3	184.7	7.2	1.9

TABLE IV. POWER COEFFICIENT VALUES FOR MALI-400 GPU

i	$\beta_{freq_i}^{GPU}$	$\beta_{base_i}^{GPU}$
0	5.60	138.99
1	5.00	298.36
2	6.20	355.17
3	9.87	153.10

$$P_{GPU} = \beta_{freq_i}^{GPU} \times u_{GPU} + \beta_{base_i}^{GPU}, \quad i = 0, 1, 2, 3 \quad (3)$$

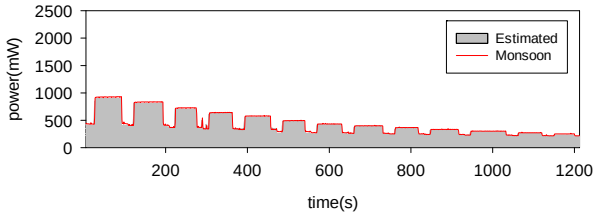
where i is the index of operating frequency, $\beta_{freq_i}^{GPU}$ the gradient of the linear regression result for frequency index i, u_{GPU} the utilization of GPU, and $\beta_{base_i}^{GPU}$ the y-intercept of the linear regression result for the frequency index i. Table III summarizes the power coefficient values of the GPU power model.

III. EVALUATION

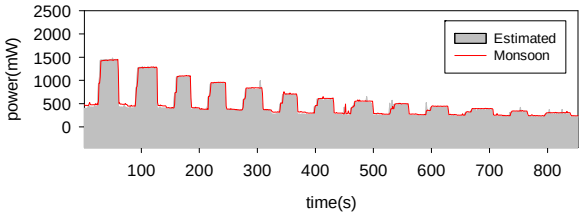
We evaluate the accuracy of the presented power model using our CPU test application and the Android GPU benchmark application. The evaluation was conducted on the S3 with Android platform version 4.0. The Monsoon Power Monitor was used as an external power meter.

A. Validation of the CPU Power Model

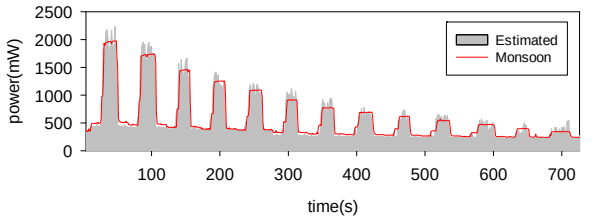
The validation of the CPU power model should be conducted in a condition where other hardware modules have a minimal influence on the power consumption. We implemented a CPU test application that does not use display, GPU, sensors, and other network devices. The application sets the number of active cores and changes the operating



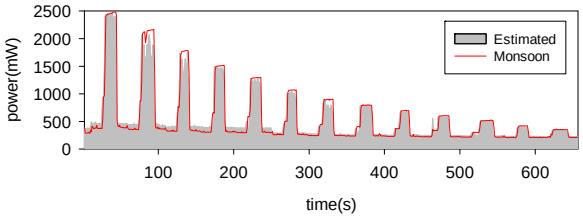
(a) One active core



(b) Two active cores



(c) Three active cores



(d) Four active cores

Fig. 5. Results of comparison of estimation using the CPU power model and measurement using Monsoon Power Monitor

frequency step by step sequentially from 1.4 GHz to 0.2 GHz, then assigns workloads. While our application tested the CPU for each frequency, we logged the CPU utilization at every second to estimate the power consumption based on (2). Simultaneously, we measured real power consumption by using the external power monitor.

Fig. 5 shows the results of CPU power consumption, depending on the number of active cores, from our estimation and external measurement. In each plot, the idle state is characterized by low power consumption, while the full working state is characterized by high power consumption. In fact, the idle state is not completely idle because of the CPU usage by kernel and other processes. All the estimation results using (2) are similar to the results obtained using the external power meter. However, if three cores are activated, our estimation is relatively higher than the measurement result, while the CPU runs with a maximum workload. In the case in which all cores are activated, the estimation result of (2) is higher than the measurement result of the external power meter in the idle state at certain high frequencies. This demonstrates the limitation of our assumption that the power coefficients of every active core are the same. The mean estimation errors in Fig. 5 are 21.3, 30.0, 39.9, and 10.3 mW. Table IV summarizes various statistics such as mean estimation error, standard deviation, standard error, and estimated energy error of each case. The standard deviation and standard error are found to increase with the number of active cores.

B. *GPU Power Model Validation*

We evaluated the accuracy of the GPU power model using the Mobile GPU Mark distributed via Google Play Market. For this analysis, we logged the GPU operating frequency and its utilization at every second to estimate GPU power consumption according to (3). Since GPU power consumption cannot be measured separately, we additionally utilized the CPU and OLED [21] power models to analyze the accuracy of the GPU power model.

Fig. 6 shows the results for the estimated power consumption and the external measurement. To minimize the effect of the OLED power model [21], we used measurement duration when the screen color was not changing. Each colored area in Fig. 6 represents the estimated result of each hardware module. Except for the portion depicting GPU power estimation, we observed a constant estimation error of approximately 267.1 mW per second on average. However, after applying the GPU power model, the estimation result improved with a mean error of 45 mW (4.1%).

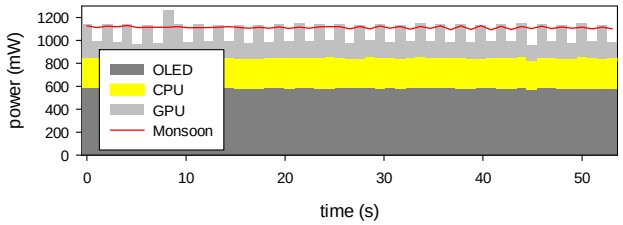


Fig. 6. Result of comparison between estimation and measurement by using the Monsoon Power Monitor

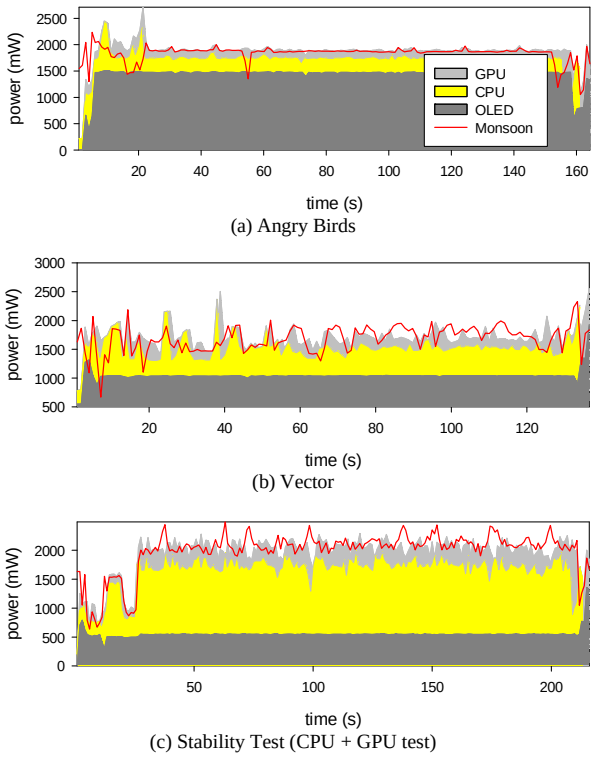


Fig. 7. Trace results of power consumption for Android Market Applications

C. *Real Application Test*

To evaluate the usability of our presented AP power model, we analyzed the estimation results using real applications. For this analysis, we selected three applications that adequately utilize CPU and GPU.

Fig. 7 shows the estimated power consumption of Angry Birds (game), Vector (game), and StabilityTest (CPU/GPU benchmark) by using our AP power model. We used the OLED power model [21] to estimate the display power consumption. The presented power model generally showed accurate estimation results as compared with the measurement results. At 18, 58, and 155 seconds, in Fig. 7(a), the Monsoon result is smaller than the estimation result. At these points, Angry Birds caused a darkening in the overlay screen to show loading, scoring, and exit confirmation dialogs. The opposite cases are observed in Fig. 7(c), which are caused by detection error of screen color, which is an important factor in the OLED power model. Table V summarizes the mean and overall estimation errors of each application in comparison with the Monsoon results. In the case of Vector, the estimation error is higher than others. It is evident that the background screen of Vector changes very often, but the estimated power of display is

TABLE V. ESTIMATION ERRORS OF ANDROID APPLICATIONS

App.	Average Estimation Error (mW)	Estimation Error (%)
Angry Birds	33.5	2.0
Vector	537.4	31.8
Stability Test	45.5	4.0

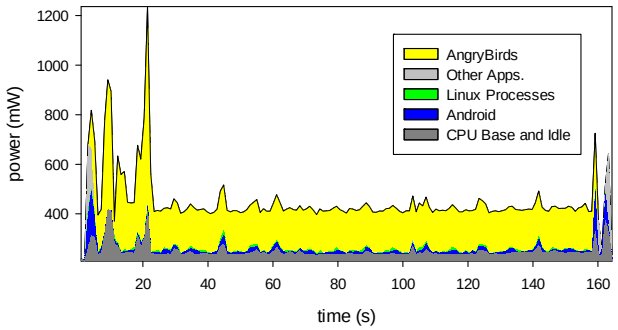


Fig. 8. Estimation result obtained by decomposing the power consumption of AP for each application or processes group

almost constant during the playing, as shown in Fig. 7(b).

D. Per-process Power Estimation

To analyze the practicality of the presented power model, we re-estimated the power consumption of Angry Birds by using process-specific CPU and GPU utilization at every second. Fig. 8 presents the re-estimation result of Fig. 7(a), except for the display power consumption. Due to space limitations in Fig. 8, we grouped the processes as Angry Birds, other applications (i.e., Launcher, Market, and browser), Linux processes, and Android. Each color represents the estimation result of each process group, except for the dark gray area that represents the estimated base power consumption of CPU and idle power consumption.

IV. Related Work

Power modeling of smartphones is an important research topic. Active research is found in the literature that is particularly related to the CPU power model because CPU power consumption is relatively large in the overall power consumption of a system. Most research on the power analysis of multicore CPU has been conducted for server computers, while there is barely any research on mobile multicore CPU. Basmadjian [13] proposed a multicore CPU power model, which is derived on the basis of hardware design in die-level. Further, Bircher and Bertran [22, 23] proposed accurate CPU power models by using the hardware performance counter. Although these models accurately estimate CPU power consumption, they are not applicable to the estimation of process-level power consumption. In recent research, PowerTutor [1], AppScope [2], and Eprof [14] support the estimation of application power consumption. They, however, do not support modern multicore-based smartphones due to the absence of appropriate power models.

V. Conclusion

In this study, we proposed a utilization-based AP power model by analyzing the power characteristics of Exynos-4412 SoC. The proposed model supports multicore CPU and GPU and enables the estimation of application-level AP power consumption. Recently, mobile operating systems and applications have begun to aggressively use multicore CPU and GPU to improve user experience. The results of our research are expected to contribute to process-level or application-level power estimation in the development of energy-aware mobile systems.

References

[1] PowerTutor, http://powertutor.org

[2] C. Yoon, D. Kim, W. Jung, C. Kang, and H. Cha, "AppScope: Application energy metering framework for Android smartphone using kernel activity monitoring," *USENIX ATC*, 2012

[3] A. Shye, B. Scholbrock, and G. Memik, "Into the wild: Studying real user activity patterns to guide power optimizations for mobile architectures," *MICRO*, 2009

[4] L. Zhang and et al., "Accurate online power estimation and automatic battery behavior based power model generation for smartphones," *CODES/ISSS*, 2010.

[5] W. Jung, C. Kang, C. Yoon, D. Kim, and H. Cha, "DevScope: A nonintrusive and online power analysis tool for smartphone hardware components," *CODES/ISSS*, 2012.

[6] A. Pathak, Y. C. Hu, M. Zhang, P. Bahl, and Y. Wang, "Fine-grained power modeling for smartphones using system call tracing,"*EuroSys*, 2011.

[7] W. L. Bircher and L. John, "Predictive Power Management for Multi-core Processors," *LNCS*, vol. 6161, pp. 243-255, 2010.

[8] M. Kim, J. Kong, and S. W. Chung, "Enhancing online power estimation accuracy for smartphones," *Consumer Electronics, IEEE Transactions on*, vol.58, pp. 333-339, 2012

[9] H. Zeng, X. Fan, C. Ellis, A. Lebeck, and A. Vahdat, "ECOSystem: Managing energy as a first class operating system resource," *ASPLOS*, 2002.

[10] A. Roy, and et al., "Energy management in mobile devices with the Cinder operating system," *EuroSys*, 2011

[11] K. Singh, M. Bhadauria, and S. McKee, "Real time power estimation and thread scheduling via performance counters," *ACM SIGARCH Computer Architecture News*, 37(2):46-55, 2009.

[12] Y. Xiao, and et al., "A system-level model for runtime power estimation on mobile devices," *GreenCom-CPSCom*, 2010.

[13] R. Basmadjian, and H. Meer, "Evaluating and modeling power consumption of multi-core processors," *e-Energy*, 2012.

[14] A. Pathak, Y. C. Hu, and M. Zhang, "Fine grained energy accounting on smartphones with Eprof," *EuroSys*, 2012.

[15] Z. Mwaikambo, A. Raj, R. Russell, and J. Schopp, "Linux kernel Hotplug CPU support," *Linux Symposium*, 2004

[16] Exynos-4412, http://www.samsung.com/global/business/semiconductor/file/product/Exynos_4_QUAD-0.pdf

[17] Samsung Galaxy S3, http://pdadb.net/index.php?m=specs&id=3534&c=samsung_gt-i9300_galaxy_s_iii_16gb__galaxy_s3

[18] CoreMark, http://www.coremark.org

[19] Monsoon Power Monitor, http://www.msoon.com/LabEquipment/PowerMonitor

[20] ARM Mali-400, http://www.arm.com/products/multimedia/mali-graphics-hardware/mali-400-mp.php

[21] D. Kim, W. Jung, and H.Cha, "Runtime power estimation of mobile AMOLED displays," *DATE*, 2013.

[22] W. L. Bircher and K. John, "Complete system power estimation using processor performance events," *Computers, IEEE Transactions On*, vol.61, pp.563-577, 2012

[23] R. Bertran, M. Gonzalez, Z. Martorell, N. Navarro and E. Ayguade, "A systematic methodology to generate decomposable and responsive power models for CMPs," *Computers, IEEE Transactions On*, vol.99, pp. 1-14, 2012.