An empirical validation of power-performance scaling: DVFS vs. multi-core scaling in big.LITTLE processor

Seehwan Yoo[a)]
College of International Studies, Dankook University,
Juk-jeon ro, Su-ji gu, Yong-in si, Korea
a) *seehwan.yoo@dankook.ac.kr*

Abstract: In this letter, we present a power-performance scaling in asymmetric multi-core embedded microprocessor. Asymmetric multi-core processor draws attention in embedded systems because the design tries to catch both energy-efficiency and high-performance. In this letter, we revise a multi-core power-performance scaling study with more practical parameters, and present empirical validation in a real embedded processor.

Keywords: embedded processor, big.LITTLE CPU, power-performance, Amdahl's law
Classification: Electron devices, circuits, and systems

References

[1] ARM: Juno ARM Development Platform (2014) http://www.arm.com/products/tools/development-boards/versatile-express/juno-arm-development-platform.php.
[2] M. D. Hill and M. R. Marty: Computer **41** [7] (2008) 33. DOI:10.1109/MC.2008.209
[3] D. H. Woo and H.-H. S. Lee: Computer **41** [12] (2008) 24. DOI:10.1109/MC.2008.494
[4] A. S. Cassidy and A. G. Andreou: IEEE Trans. Comput. **61** (2012) 1110. DOI:10.1109/TC.2011.169
[5] H. Esmaeilzadeh, T. Cao, Y. Xi, S. M. Blackburn and K. S. McKinley: ACM ASPLOS (2011) 319. DOI:10.1145/1950365.1950402
[6] H. Esmaeilzadeh, E. Blem, R. St. Amant, K. Sankaralingam and D. Burger: ACM Trans. Comput. Syst. **30** [3] (2012) 27. DOI:10.1145/2324876.2324879
[7] O. Azizi, A. Mahesri, B. C. Lee, S. J. Patel and M. Horowitz: ACM ISCA (2010) 26. DOI:10.1145/1815961.1815967
[8] E. L. Sueur and G. Heiser: USENIX HotPower (2010) 1.
[9] E. Blem, J. Menon and K. Sankaralingam: IEEE HPCA (2013) 1. DOI:10.1109/HPCA.2013.6522302
[10] S. J. Cho, S. H. Yun and J. W. Jeon: IEICE Electron. Express (2015) 20141128. DOI:10.1587/elex.12.20141128
[11] M. Pricopi, T. S. Muthukaruppan, V. Venkataramani, T. Mitra and S. Vishin: IEEE CASES (2013) 1. DOI:10.1109/CASES.2013.6662519
[12] J. M. Kim, S. K. Seo and S. W. Chung: IEEE International Workshop on Parallelism in Mobile Platforms (2014) 1.

© IEICE 2015
DOI: 10.1587/elex.12.20150236
Received March 5, 2015
Accepted March 23, 2015
Publicized April 3, 2015
Copyedited April 25, 2015

[13] A. Carroll and G. Heiser: IEEE RTAS (2014) 287. DOI:10.1109/RTAS.2014.6926010
[14] A. Miyoshi, C. Lefurgy, E. Van Hensbergen, R. Rajamony and R. Rajkumar: ACM ICS (2002) 35. DOI:10.1145/514191.514200

1 Introduction

Recently, ARM has proposed a new embedded processor design, called big.LITTLE. big.LITTLE draws attention because the design tries to focus on important two goals in the modern embedded systems at the same time; low-power and high-performance. In general, high-performance processors struggles against limited power budget; low-power design suffer from limited performance. big.LITTLE processor leverages asymmetric multi-cores, challenging low-power design optimization for energy-efficient embedded systems as well as high-performance design for cpu-intensive workloads.

Recent big.LITTLE such as Juno platform [1] consists of two different core clusters: a big cluster and a little cluster. A little cluster has multiple little cores that implements low-power micro-architecture, so multiple little cores enables us to perform fine-grained power management. On the other hand, a big cluster has one or two big cores that can maximize performance with additional power consumption. Big cores outperform little cores by adopting performance-oriented micro architectural features. Big cores have larger cache, TLB, BTB, deep pipeline, and floating-point hardware, etc. Namely, big and little cores are on the different power-performance engineering points, presenting different power-performance characteristics even though they share the same ISA (instruction set architecture).

A goal of this letter is to characterize big.LITTLE's power-performance scaling. Traditionally, Amdahl's law has been extended by several studies [2, 3, 4, 5, 6], particularly for multi-core processors. Among them, Woo and Lee [3] link Amdahl's performance scaling model with power consumption in asymmetric core design, in respect to energy-efficiency. They depicted a brief power-performance trend line of three different many-core designs. This letter smoothly revises their model, and presents empirical validation through experiments on a real big.LITTLE microprocessor. Throughout our model, we present power-performance scaling law in big and little clusters, bridging dynamic voltage and frequency scaling (DVFS) and multi-core scaling with idleness.

2 Related work

Woo and Lee [3] comparatively presented Amdahl's law in three different many-core designs: many superscalar cores (P^*), many power-efficient small cores (c^*), and asymmetric many-core ($P+c^*$). Based upon the Amdahl's parallel execution model, they derive theoretical maximum speedup and power consumption. Their model is numerically evaluated with the assumption of parallel-izable fraction of execution code (f). An important part of the work is that links performance scaling model with power consumption model. They distinguish idle power from active power, and additionally consider idle core's power consumption.

© IEICE 2015
DOI: 10.1587/elex.12.20150236
Received March 5, 2015
Accepted March 23, 2015
Publicized April 3, 2015
Copyedited April 25, 2015

In addition, they presented the benefits from asymmetry in many-core design. In their model, one big superscalar core plus small multi-core combination (P+c*) shows the best power-performance characteristics in many reasonable conditions. Our work extends the work in a sense that we provide a more accurate model considering different intra-cluster scaling for big and little clusters.

Along with multi-core power-performance scaling, DVFS is another important power-performance throttling techniques. By adjusting the operating CPU clock frequency and input voltage, we can throttle up and down the power and performance [7]. Despite the pessimistic studies on DVFS [8], it is one of widely used in modern processors. In particular, it is essential mechanism to control power consumption for energy-limited mobile systems as well as desktop computers or servers [9, 10].

Although power-performance of asymmetric multi-core has been presented in several studies, up to our knowledge, asymmetric scaling law for big and little cluster has not been thoroughly investigated. Pricopi et al. presented power-performance model in asymmetric multi-core using big-little core in [11]. However, their model is evaluated with single-core-per-cluster. Jaemin et al. [12] presented symmetric big.LITTLE that has the same number of big and little cores in each cluster; Big and little cluster would have different number of cores, so that they follow different scaling law. Carroll focused on linking DVFS and multi-core scaling in [13]. However, their model is not general to incorporate asymmetric multi-core clusters, and their hardware platform does not fully support heterogeneous multi-core scheduling (HMP), in software and hardware perspectives.

3 Energy-performance scaling in little cluster

In this section, we model energy-performance scaling in little cluster. Little cluster has multiple little cores, each of which consumes relatively less power. Thus, we can control power consumption in a small granularity, by turning on/off the active cores. Namely, we can use previous studies that extends multi-core Amdahl's law.

In the previous power-performance model by Woo and Lee, the authors assume power and energy model for the effective execution time; the amount of time that a program makes actual progress, except for idle period. The model accounts power-performance only for the effective execution. However, system-level power management should take idle power into account because the system runs continuously until it is explicitly stopped by users. To derive the actual energy gain, we consider this idle power consumption, we use energy-performance model, instead of power-performance.

To accurately model the average power consumption, we, at first, derive an energy model. To consider idle time, we identify the slowest execution among all configurations. Then, we measure the total energy consumption during the same amount of time.

Next, we define the total time as serial (t_{ser}), parallel (t_{par}), and idle (t_{idle}) time. We further assume effective execution time (or execution time) as the sum of serial and parallel execution time (i.e. $t_{exec} = t_{see} + t_{par}$). Note that regardless of the

© IEICE 2015
DOI: 10.1587/elex.12.20150236
Received March 5, 2015
Accepted March 23, 2015
Publicized April 3, 2015
Copyedited April 25, 2015

number of cores, or effective execution time, total time (c) is the same (i.e. $t_{ser} + t_{par} + t_{idle} = c$).

To derive idle time, we use execution time ratio from the Amdahl's law. When we leverage multiple cores, the execution time is reduced by the amount of speedup factor (Perf in [3]). Thus, the processor is in idle state for $(1 - 1/\mathrm{Perf}) \cdot c$ time because total time c is the same. We assume power consumption (k) at idle time. We take this idle power consumption into account, and revise the average power model, which accounts for critical power slope of a processor [14].

In addition, idling power in our big.LITTLE platform does not increase as the number of idle cores. That is because all cores in the same cluster are in the same power domain [13], in which case, power-gating for each idle core is not feasible. Thus, idle power is only observed when there is no active core. (Namely, $k = 0$, when $n \geq 1$).

In our model, power consumption of cluster consists of per-cluster static power and per-core additive power. Since a cluster has some shared hardware resources such as L2 cache, and TLB, among cores, static power (c_s) and dynamic power (c_d) are modeled separately. When one core is active, the average power consumption becomes $c_s + c_d$. When n cores becomes active, then the average power consumption becomes $c_s + n \cdot c_d$.

In general, energy is time-integration of power consumption, which can be expressed as follows:

$$Energy = \int P(t)\,\mathrm{d}t = \int_{t_{ser}} P_{ser}(t)\,\mathrm{d}t + \int_{t_{par}} P_{par}(t)\,\mathrm{d}t + \int_{t_{idle}} P_{par}(t)\,\mathrm{d}t, \tag{1}$$

where P(t) is average power consumption, and P(t) can be differently defined according to the serial, parallel, idle execution status as $\mathrm{P}_{ser}(\mathrm{t})$, $\mathrm{P}_{par}(\mathrm{t})$, and $\mathrm{P}_{idle}(\mathrm{t})$, respectively. Note that Energy measures all the energy for the total time, c.

From the above equation, we revise the energy model, which calculates the normalized energy consumption with n cores over that with single core as follows:

$$\mathrm{Energy}_{(1-core \rightarrow n-core)} = \frac{\mathrm{Energy}_{n-core}}{\mathrm{Energy}_{1-core}} \tag{2}$$

$$= \frac{(c_s + c_d) \cdot (1 - f) \cdot c + (c_s + n \cdot c_d) \cdot \frac{f}{n} \cdot c + k \cdot \left\{1 - \left(1 - f + \frac{f}{n}\right)\right\} \cdot c}{(c_s + c_d) \cdot c} \tag{3}$$

$$= J = 1 + kf(1 - 1/n) - \frac{n-1}{n} \cdot c_s \cdot f, \tag{4}$$

where f is fraction of parallel execution, c is total time (i.e. c = effective execution time + idle time), n is the number of cores, k is idle power consumption normalized by $c_s + c_d (= 1)$.

In the above equation, Energy_{n-core} is normalized by the energy consumption of single core, Energy_{1-core}, where all the execution is serial (n = 1, f = 0). Because serial part (the first term in the equation) utilizes only one core, the cluster consumes the same amount of power with the serial execution. For parallel part (the second term in the equation), the average power consumption increases as the number of active core. However, the execution time is reduced by the number of

© IEICE 2015
DOI: 10.1587/elex.12.20150236
Received March 5, 2015
Accepted March 23, 2015
Publicized April 3, 2015
Copyedited April 25, 2015

cores, energy consumption is reduced by that factor. Thus, for effective execution, multi-core utilization has limited power gain.

However, after the completion, all the cores are in idle state, and leakage power is observed. During the idle time (the final term in the equation), compared with the all serial execution (1 − (1 − f + f/n)), the cluster consumes average idle power k.

For large n, we get the following approximation.

$$n \cdot c_d + c_s \approx n \cdot (c_d + c_s) \tag{5}$$

$$J \approx 1 + kf(1 - 1/n), \tag{6}$$

Accordingly, energy-performance (Perf/J) is approximated as follows:

$$\mathrm{Perf}/J \approx \frac{1}{1 - f + f/n} \cdot \frac{1}{1 + kf(1 - 1/n)}. \tag{7}$$

4 Energy-performance scaling in big cluster

In this section, we present energy-performance scaling in big cluster. Asymmetric many-core model in [3] comparatively present big core's power-performance using multi-core model (P*). However, a big cluster has only one or two cores, multi-core scaling (P*) does not fit with our big-cluster model. In our big cluster model, we added an assumption with respect to DVFS. DVFS scales performance up and down by changing operating core frequency. Along with the frequency change, DVFS throttles energy-efficiency by adjusting the input voltage.

To work with DVFS, a core defines operating performance points (OPPs) table, that includes operating voltage-frequency pairs. Because all the cores in the same cluster are in the same power domain, the input voltage is the same for all the cores in the cluster (both big and little clusters); thus, applying DVFS for different cores in the same cluster is possible, but has limited impact.

In the original model, the authors assume a fixed scaling ratio between big and little cores; in terms of power consumption (w_c) and performance scaling (s_c). However, when applying DVFS, the ratio changes dynamically and significantly. More importantly, the changing ratio of power and performance is not linear so that existing model needs revise. The reason is that power consumption is correlated with frequency and voltages^2, as follows:

$$P \propto c \cdot F_i \cdot V_i^2, \tag{8}$$

where F_i and V_i represent frequency and voltage for OPP$_i$, respectively. For the same core, operating frequency is inverse proportional to the execution time. Thus, performance gain (Speedup$_{(0\to j)}$) by changing OPP from OPP$_0$ to OPP$_j$ can be described as follows[1]:

$$\mathrm{Speedup}_{(0\to j)} = \frac{\text{execution time}_0}{\text{execution time}_j} = F_j/F_0. \tag{9}$$

Since the execution time is changed by DVFS, we further need to consider idle time as well as effective execution time.

From the above equations, the average power consumption ratio in big cluster can be calculated as follows:

[1]Let us assume that $j > 0$, and $F_0 < F_j$

© IEICE 2015
DOI: 10.1587/elex.12.20150236
Received March 5, 2015
Accepted March 23, 2015
Publicized April 3, 2015
Copyedited April 25, 2015

$$\text{Energy}_{(0\to j)} = J = \frac{\text{Energy}_j}{\text{Energy}_0} \tag{10}$$

$$= \frac{\int_{t_{exec}} P_j(t)\,\mathrm{d}t + \int_{t_{idle}} P_j(t)\,\mathrm{d}t}{\int_{t_{exec}} P_0(t)\,\mathrm{d}t + \int_{t_{idle}} P_0(t)\,\mathrm{d}t} \tag{11}$$

$$= \frac{F_j \cdot V_j{}^2 \cdot (F_0/F_j) + k \cdot (1 - F_0/F_j)}{F_0 \cdot V_0{}^2 \cdot \{F_0/F_j + (1 - F_0/F_j)\}} \tag{12}$$

$$= \frac{F_0 \cdot V_j{}^2 + k \cdot (1 - F_0/F_j)}{F_0 \cdot V_0{}^2}, \tag{13}$$

where k is idle power consumption, t_{exec} and t_{idle} presents effective execution time and idle time. $P_j(t)$ and $P_0(t)$ presents average power consumption in OPP_j and OPP_0, which follows the relationship in 8.

Then, performance per Joules change is as follows:

$$\frac{Perf}{J}(0 \to j) = \text{Speedup}_{(0\to j)}/\text{Energy}_{(0\to j)} \tag{14}$$

$$= \frac{F_j}{F_0} \cdot \frac{F_0 \cdot V_0{}^2}{F_0 \cdot V_j{}^2 + k \cdot (1 - F_0/F_j)} \tag{15}$$

$$= \frac{F_j \cdot V_0{}^2}{F_0 \cdot V_j{}^2 + k \cdot (1 - F_0/F_j)}. \tag{16}$$

5 Empirical validation on big.LITTLE energy-performance

To characterize energy-performance of big.LITTLE design, we need to compare performance per watt for big and little cluster. To validate our analysis, we measure the energy consumption and performance in a real hardware platform [1]. The Juno platform has a big core cluster and a little cluster. The big cluster has two big cores (Cortex-A57), and the little cluster has four little cores (Cortex-A53). The cores share the same 64 bit ISA, armv8-a so that any task can freely migrate to another core. We use Dhrystone benchmark as a standard CPU workload, which constitutes of CPU-only operations.

To present power-performance scaling of big and LITTLE cores, we apply DVFS for the big cluster, and multi-core utilization for the small cluster. For big cluster, we turned on two cores on, and we measure the performance and the energy consumption as we change the clock speed and frequency. For little cluster, we set all cores run at the same clock speed, 450 Mhz, and we measure the performance and the energy consumption as we change the number of active cores. As a workload, we measure achieved Dhrystone million instructions per second (DMIPS). Energy consumption is separately measured for during the effective execution, during idle, until the slowest execution completion. Note that Dhrystone is a CPU-intensive benchmark that has little serial execution part. Thus, we run Dhrystone on each core to simulate multi-core execution. Namely, we assume perfect parallel execution for the entire code (i.e. $f = 1$).

At first, we present power consumption during effective execution time. Fig. 1 presents the power-performance during effective execution that does not include power consumption during idle.

In the figure, X-axis presents the average power consumption during execution in milliWatt, and Y-axis presents achieved performance in DMIPS (Dhrystone

© IEICE 2015
DOI: 10.1587/elex.12.20150236
Received March 5, 2015
Accepted March 23, 2015
Publicized April 3, 2015
Copyedited April 25, 2015

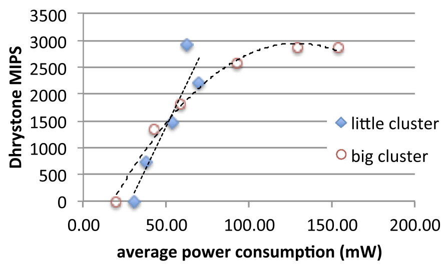


Fig. 1. Power-performance of big.LITTLE during effective execution

million instructions per second). The higher curve presents better throughput for the same power consumption. X-value when Y is 0 represents leakage power when the cluster is idle.

In the graph, big and little clusters show different trends: big cluster's power-performance curve decreases gradually as we change OPP. On the other hand, little cluster's power-performance curve linearly increases as we increase the number of cores. A reason for linear performance scaling is because we provide almost parallel workload ($f = 1$).

In the graph, in small power range, big and little cluster presents similar performance per Watt. That represents big and little clusters have comparable performance-per-power efficiency. For closer look, two curves exchange when core number is 3 and 4. When the core number is one or two, (and idle), little cluster's curve lies beneath the big cluster's curve, which implies little cluster is less efficient in respect to performance-per-Watt. When the number of core is more than 3, little cluster becomes more efficient in terms of performance-per-Watt.

To consider energy-efficiency, we additionally consider the energy consumption during idle. Fig. 2 shows energy consumed during idle. To comparatively present idle and effective energy, we measure the the longest execution time among all configurations. Then, separately measure the energy during the effective execution and the idle energy. Usually, the slowest execution time is observed at the single core little cluster execution.

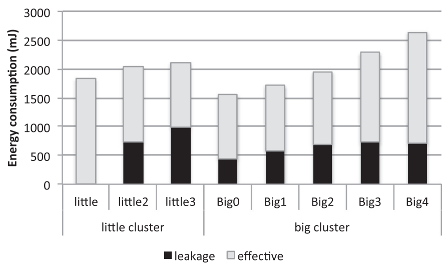


Fig. 2. Energy consumption during effective execution time and idle time

As shown in the figure, idle energy for little cluster increases significantly as the number of cores. On the other hand, idle energy for big cluster changes marginally; the variation is much smaller than that of little cluster. This results in a large change in overall energy consumption, and overall efficiency.

© IEICE 2015
DOI: 10.1587/elex.12.20150236
Received March 5, 2015
Accepted March 23, 2015
Publicized April 3, 2015
Copyedited April 25, 2015

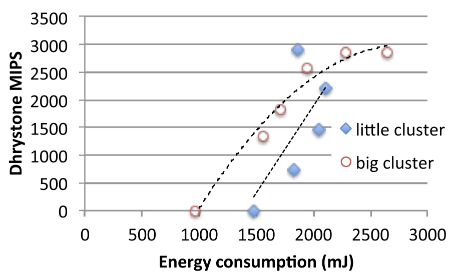


Fig. 3. Energy-performance of big.LITTLE

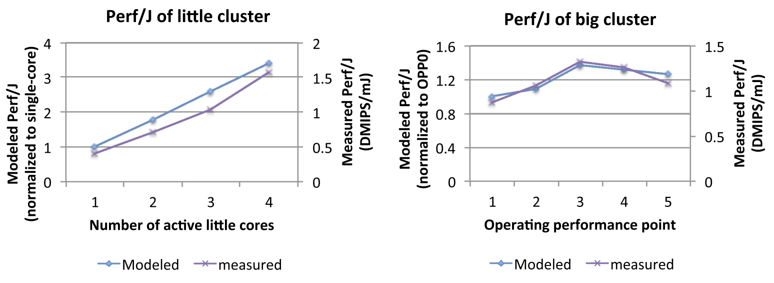
Fig. 3 presents the energy-performance for the entire execution time + idle time. Due to the energy consumption in idle time, the efficiency of big cluster is largely enhanced. In the figure, big cluster's curve lies over the most of the little cluster's points. The result implies the big cluster's DVFS presents better power-performance scaling than multi-core scaling in the little cluster.

Due to our parallel workload, multi-core efficiency seems to be very close to optimal. For other real workloads, the efficiency will be degraded when serial execution part exists.

Despite the presented power-performance efficiency result in little cluster, it seems that there are room for future enhancements because the little cluster's curve is much stiff and rapidly grow than that of big cluster. Thus, little cluster seems to be a feasible approach as the number of cores increases.

Fig. 4 comparatively presents our modeled Perf/J model with big and little cluster. In both graphs, modeled Perf/W and measured values present the almost same trend. The result shows that our model catches important characteristics of the power-performance scaling of big.LITTLE design.

Table I summarizes the actual parameters used with our model.



(a) little cluster (b) big cluster

Fig. 4. Model-and-measured Perf/W

Table I. Parameters used for big.LITTLE Juno

Little cluster	Big cluster	
k: 0.80834	k: 0.46806	
c_s: 0.56757	v_0: 0.84 v	F_0: 450 Mhz
c_d: 0.43243	v_1: 0.87 v	F_1: 625 Mhz
	v_2: 0.91 v	F_2: 800 Mhz
	v_3: 0.97 v	F_3: 950 Mhz
	v_4: 1.04 v	F_4: 1100 Mhz

© IEICE 2015
DOI: 10.1587/elex.12.20150236
Received March 5, 2015
Accepted March 23, 2015
Publicized April 3, 2015
Copyedited April 25, 2015

6 Conclusion and future work

In this letter, we characterize the power-performance scaling in big.LITTLE asymmetric embedded multi-core processor. We revise the previous extended Amdahl's law, so that the proposed model can accurately characterize performance and power-efficiency and their scaling trends. In addition, the model has empirically validated using Dhrystone benchmark. Though our work is theoretically sound, it requires thorough validation for parallel applications, which has much impact on multi-core execution. In addition, we are extending our model for concurrent P+c* execution, and P*+c* model.

Acknowledgement

The present research was conducted by the research fund of Dankook University in 2014.

© IEICE 2015
DOI: 10.1587/elex.12.20150236
Received March 5, 2015
Accepted March 23, 2015
Publicized April 3, 2015
Copyedited April 25, 2015