Asymptotic Performance Optimization of Manycore Architectures Under Process Variation

Thi-Yen Phuong, Young-Woong Ko, Jungmin So and Jeong-Gun Lee

Dept. of Computer Science, Hallym University, Chuncheon, South Korea
{ yanni, yuko, jso1, jeonggun.lee}@hallym.ac.kr

Abstract

A manycore processor architecture integrating multiple cores onto a single die has been widely used in almost all computer systems and academia and industry have worked on the architecture for high-performance and low-power implementation. The manycore architectures have been proposed and designed for overcoming diminishing return and for efficiently utilizing the exponentially increasing number of transistors available in nano-meter technology. Although the architecture has been investigated extensively, under the process variation that is now considered as a critical design problem in the nano-meter technology, performance characteristics and benefits of the manycore architecture are not well studied. In this paper, we develop an asymptotic analysis model for better understanding the performance characteristics of manycore processor architectures using Amdahl's law under process variation in order to foresee their performance impact for a given workload characteristics (e.g. available parallelism). Through the asymptotic analysis based on the models proposed in this paper, we can make the architectural design decisions such as "the number of cores" and "core size", and further we can probe the possible research direction of optimizing the performance of manycore architectures at the future of high process variation era.

Keywords: *We would like to encourage you to list your keywords in this section*

1. Introduction

CMOS semiconductor technology has been improved continuously and rapidly during the last decade. Many design problems regarding energy consumption and reliability have been occurred under this highly scaled deep-submicron / nano-meter technology [1, 2]. Moreover, the performance of traditional monolithic core processors is getting harder to improve as the number of transistors integrated on the monolithic core processors increases due to the limited instruction level parallelism. The approach of integrating more functional units to a single core and increasing clock frequencies has met the saturating limit of performance. This phenomenon is called "diminishing return" [3]. In order to cope with the severe diminishing return problem and better utilize the large number of available transistors provided by advanced CMOS technology, industries have started to produce manycore processors providing high-performance, energy-efficient and reliable computation. More than one hundred cores are expected to be integrated into a single chip by 2015 [1]. At the side of Graphics Processing Unit (GPU) designs, already more than thousand cores have been integrated to best exploit the enormous amount of parallelisms in graphics and scientific computing applications [4].

Particularly, in a manycore architecture, supply/threshold voltages and clock frequencies of individual cores can be managed and controlled without considering the other cores in a whole system. This makes the manycore architectures to be able to exploit more flexible power/energy management by controlling the individual cores independently. Such a fine-grain controllability in manycore architectures gives more room for optimizing various design objectives: performance, energy, fault-tolerance and process variation [5-8].

Though the manycore architectures are getting more popular and considered as a very promising alternative to a traditional monolithic core architecture, under a process variation operating environment, their performance still has not been well studied and characterized owing to the high design and simulation complexity of manycore architectures and process variation models. With modern processor simulation tools, it takes several hours to simulate single application benchmark even with a monolithic core architecture model [9].

In this paper, in order to cope with the performance simulation for manycore architectures under process variation, asymptotic analytical performance models are derived for exploring a high-level design space of a manycore architecture. In particular, we issue design problems such as “how many cores and how powerful cores does a manycore system have to use for a given workload and a given technology for best performance?” and try to answer the question by using the proposed analytical models. The model based design space exploration is very important to understand the fundamental relationship between the manycore configurations and their performance. In this paper, we derive a set of equations based on Amdahl’s law asymptotically capturing the performance benefits of a manycore processor. Through the asymptotic analysis based on the models proposed in this paper, we can make the architectural design decisions such as “the number of cores” and “core size”, and further we can probe the possible research direction of optimizing the performance of manycore architectures for the future technology in which high process variation will be observed.

This paper is organized as followings. In Section 2, preliminary and related work will be described briefly. In Section 3, we will develop a performance model of manycore architectures, and analyze the models for finding optimum manycore configurations such as optimum number of cores and their size. Finally, in Section 4 we conclude this paper with summary and future work.

2. Preliminary and Related Work

In this section, the basics of the Amdahl’s law and process variation are explained briefly for the understanding of this paper together with corresponding related previous work.

2.1. Amdahl's Law

The Amdahl’s law is a very well-known law in the parallel computing society and it has been used for estimating the upper bound of speedup improvement by adopting parallel multiprocessor systems. When a program have ‘s’ portion of serial code and ‘$1 - s$’ portion of parallel code, the Amdahl’s law states that the overall speedup of exploiting N multiple processing units will be described in the following equation [10].

$$Sp = \frac{1}{s + \frac{1-s}{N}} \tag{1}$$

The Amdahl’s law has been used to give the maximum speedup we can achieve from manycore architectures [3, 11]. Very recently, Amdahl’s law based performance and energy analyses have been published in [12-15]. In [12, 13], simple architecture models for three different types of heterogeneous manycore processors were constructed and corresponding performance equations were derived. Finally numerical performance simulations were performed. In [14], different clock frequencies were assumed to be used for executing serial code and parallel code. Then, two optimal clock frequencies for running serial code and parallel code were derived in order to minimize energy consumption. Their main focus was the impact of dynamic clock frequency scaling on energy consumption of manycore architecture. In [15], more rigorous analysis for power/energy model derivation has been performed with dynamic voltage scaling.

This trend of the researches based on asymptotic analysis has been performed continuously for figuring out the fundamental characteristics of manycore processor architectures.

2.2. Process Variation

As a semiconductor technology advances, we have observed rapidly increasing impact of semiconductor process, operation voltage and working temperature (PVT) on the performance and energy/power consumption of circuits.

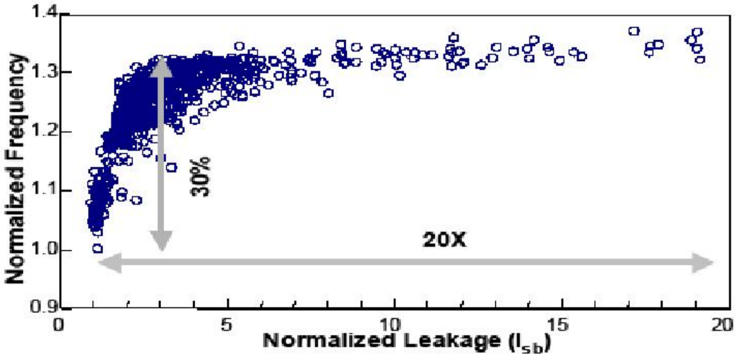


Figure 1. Performance and Leakage Current Variations under 0.18um CMOS Technology [5]

Figure 1 shows the performance and leakage current variations observed under 0.18um CMOS technology. As shown in the figure, 20X leakage variation and 30% frequency variation have been observed due to the high process variation. With this high variation of chip operation, when binning chips, high frequency but too leaky chips or low leakage chips with too low frequency must be discarded. Consequently chip yield rate will be significantly degraded. Since the graph is obtained at the 0.18um technology, more high variation will be observed in the nano-meter advanced technologies.

To overcome the yield loss caused by the process variations and to increase reliability of a processor under the process variation, we need a way of increasing yield and reliability. Significant amount of work has been performed to solve the problems caused by the process variation in the technology, circuit, processor architecture societies. A manycore processor architecture can be an unique solution for the problem thanks to the redundant cores integrated onto a processor.

2.3. Core Dependent Clock Frequency

Under the existence of process variation, clock frequency is strongly depending on the size of a chip. When a chip size increases, there will be higher probability of having a larger number of critical paths. Then, it leads to the increase of clock frequency since the clock frequency of a chip is determined by the worst case critical path delay in a traditional synchronous circuit design.

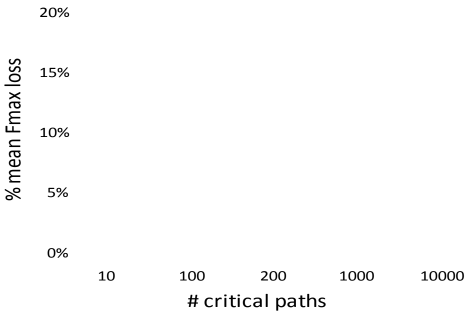


Figure 2. The Increase of Clock Frequency when the Number of Critical Paths Increase in a Chip [16]

Figure 2 shows the loss of clock frequency when the number of critical paths increase from 10 to 10000 in a chip. The y-axis is the mean value of loss in maximum clock frequency. As shown in the figure, when the number of critical path in a chip is around 10, the loss is around 6% of the critical path delay. However, the loss is significantly increased up to more than 15% when 10000 critical paths exist in the chip. It implies that small chip area is good for best clock frequency performance without experiencing the performance loss of the clock frequency determined by critical paths. It is noteworthy that the small chip has lower performance owing to the lack of high performance circuitry employed by larger chip. In consequence, it is important to find out the optimal core size at the viewpoint of micro-architecture and clock frequency.

3. Optimal Core Configuration under Process Variation

In this section, we describe the process variation model used in this paper and how the process variation can affect the performance. Then, we discuss process-variation aware issues including a thermal aware issue and their impacts on the performance-optimal core partitioning on manycore architectures.

3.1. Process Variation Model

In general, parameter variation in a semiconductor chip can be classified into two main components: die-to-die (D2D) and within-die (WID) [5]. The D2D variation is variation among the different chip dies. On the other hand, The WID variation is the variation occurred the location of circuits within a single die and it is composed of *random* and *systematic* components. The random variation component is affected by variable dopant density and line edge roughness while the system variation component is affected by lens aberrations, mask deformities, thickness variation and photo-lithograph [5, 17]. Finally, the total process variation can be modeled in the following equation:

$$\Delta P = \Delta P_{D2D} + \Delta P_{WID} = \Delta P_{D2D} + \Delta P_{rand} + \Delta P_{sys} \qquad (2)$$

In this paper, WID variation is only considered because we are only concerned in performance variation within a die. The D2D variation can be modeled by variation offset among the chip dies. We also assume that the two components of WID process variation follow normal distributions [17]. Two key parameters, V_{th} and L_{eff} variations, are considered in our performance optimization study because they are directly affecting the leakage power/energy consumption and frequency of a chip. In this paper, we use a VARIUS variation modeling tool that has been developed at the UIUC [17]. The VARIUS tool is developed with *R-Package* that is a well-known visualization tool for numerical and statistical computations.

Figure 3 shows the variation map of a threshold voltages that are calculated from the VARIUS tool. The VARIUS tools take three input parameters: μ, σ and φ. μ is the average value of an parameter object such as V_{th} and L_{eff} while σ is corresponding standard deviation. φ is used for denoting a fraction of the chip's width. The large value of φ implies that large sections of the chip are correlated with each other. The small value of φ means that small sections of the chip are correlated. The value of φ is a normalized value between 0 and 1 [17]. Simply we can project that the σ is likely to increase in the future.

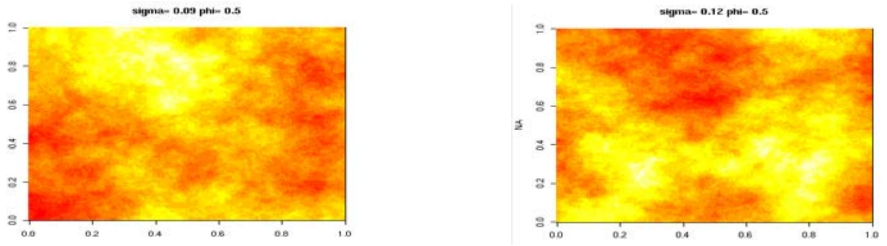


Figure 3. Variation Maps for V_{th} with Two Different σ Values: 0.09 and 0.12 with φ = 0.5

Figure 3 shows the two V_{th} variation maps that are generated from VARIUS tool. For the variation map, φ is set to 0.5 and σ is set to 0.09 and 0.12 for each case. The V_{th} variations can be observed from those maps and V_{th} has different values at different location on a die. In addition to V_{th} variation, L_{eff} variation also can be modeled in a similar manner.

3.2. Performance Trends under Process Variation

Figure 4 shows the impact of a core size on the critical path delay. The bright (yellow in color) area means that the transistors on the area work with lower V_{th} so that the circuits mapped onto the area can be operating at faster speed but with higher leakage current consumption. On the other hand, the dark (red in color) area are implies that the transistors on the area operate with higher V_{th} so that the circuits mapped onto the area can be working at slower speed but with lower leakage current consumption.

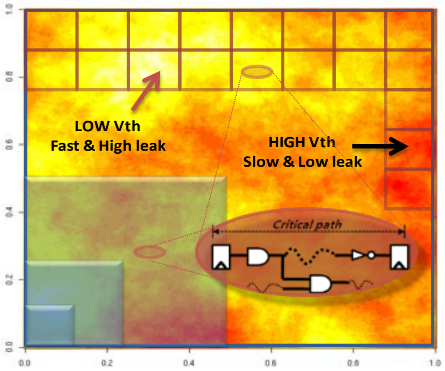


Figure 4. A Variation Map for V_{th}. The Delay of a Critical Path changes depending on the Values of V_{th} that are changing According to the Circuit's Placement

It is noteworthy that the core size also has impact on the critical path delay of a circuit in addition to process variation. This is because that worst case delay model is used in a synchronous circuit and the worst critical path in the area of a core determines its working frequency.

As larger core size increases, the number of critical paths increases. Consequently, the critical path delay will be increases. On the other hand, as the size of core deceases, the number of critical paths decreases and it leads to the smaller loss of a clock frequency. The clock frequency variation will be higher if we consider temperature variation as well.

With constrained transistor resources, we can integrate more cores into a processor die for exploiting parallelism. Then it leads to the smaller size of a core that is operating with the higher speed of clock frequency.

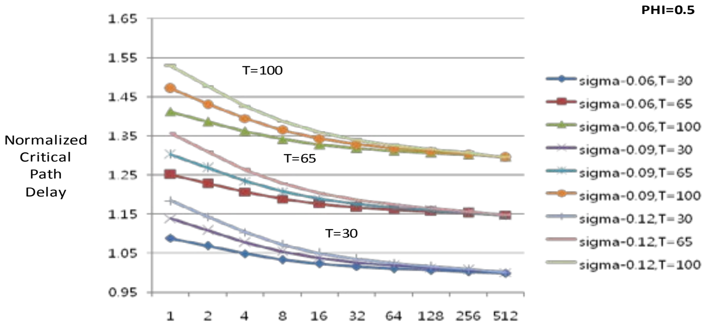
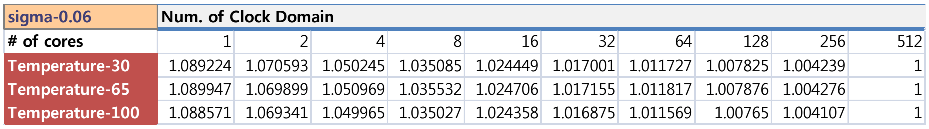


Figure 5. Variation Maps for V_{th} with Two Different σ Values: 0.09 and 0.12 with φ = 0.5

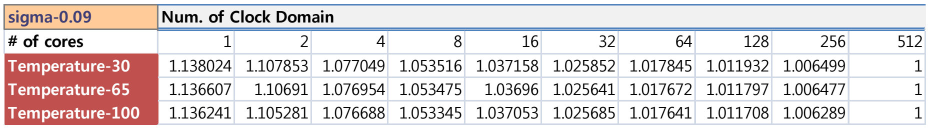
Figure 5 shows the performance variation of a processor with different size of manycore integration and different values of σ and working temperatures, T (30°C, 65°C and 100°C). The normalized critical path delays are displayed at y-axis while the number of cores integrated into the processor is shown in the x-axis. The reference performance for the normalization is the performance of the processor in which 512 independent cores are working at 30°C. The value of the x-axis can be thought as the number of independent clock domains in the processor. Detailed numerical values are presented in Table 1 which are obtained by changing the values of σ and the number of cores. For an examples, when an operating temperature is assumed to be 30°C, the critical path delay at a single core processor is 18% slower than the critical path at 512-core processor.

Table. 1. Numerical Values of Relative Performance at Different σ Values and Different Number of Cores. In this Table, φ is set to 0.5



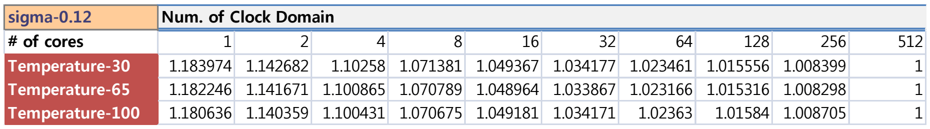
sigma-0.06	Num. of Clock Domain									
# of cores	1	2	4	8	16	32	64	128	256	512
Temperature-30	1.089224	1.070593	1.050245	1.035085	1.024449	1.017001	1.011727	1.007825	1.004239	1
Temperature-65	1.089947	1.069899	1.050969	1.035532	1.024706	1.017155	1.011817	1.007876	1.004276	1
Temperature-100	1.088571	1.069341	1.049965	1.035027	1.024358	1.016875	1.011569	1.00765	1.004107	1

(a)



sigma-0.09	Num. of Clock Domain									
# of cores	1	2	4	8	16	32	64	128	256	512
Temperature-30	1.138024	1.107853	1.077049	1.053516	1.037158	1.025852	1.017845	1.011932	1.006499	1
Temperature-65	1.136607	1.10691	1.076954	1.053475	1.03696	1.025641	1.017672	1.011797	1.006477	1
Temperature-100	1.136241	1.105281	1.076688	1.053345	1.037053	1.025685	1.017641	1.011708	1.006289	1

(b)



sigma-0.12	Num. of Clock Domain									
# of cores	1	2	4	8	16	32	64	128	256	512
Temperature-30	1.183974	1.142682	1.10258	1.071381	1.049367	1.034177	1.023461	1.015556	1.008399	1
Temperature-65	1.182246	1.141671	1.100865	1.070789	1.048964	1.033867	1.023166	1.015316	1.008298	1
Temperature-100	1.180636	1.140359	1.100431	1.070675	1.049181	1.034171	1.02363	1.01584	1.008705	1

(c)

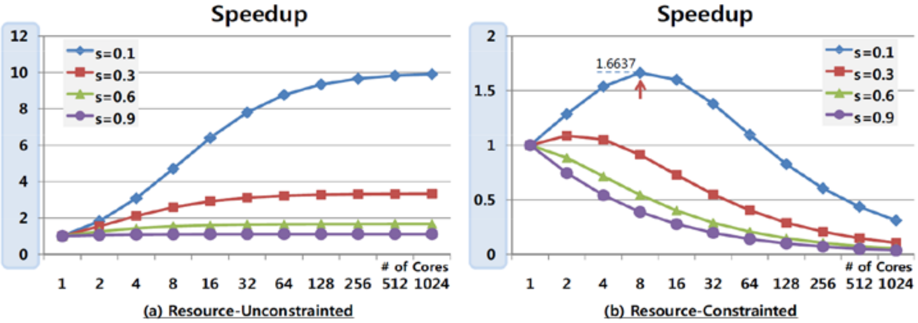
3.3. Optimizing Core Partitioning Under Process Variation

When asymptotic models are derived for evaluating the performance improvement of a manycore architecture, if "*perf(r)* = $k{\cdot}r^{\beta}$" is assumed then the speedup equation based on an Amdahl's law for a resource-constrained manycore architecture can be rewritten in the followings [15]:

$$Sp = \frac{\dfrac{IC}{k \cdot n^{\beta}}}{\dfrac{s \cdot IC}{k \cdot r^{\beta}} + \dfrac{(1-s) \cdot IC}{\left(\dfrac{n}{r}\right) \cdot k \cdot r^{\beta}}} = \frac{\dfrac{1}{n^{\beta}}}{\dfrac{s}{r^{\beta}} + \dfrac{(1-s)}{\left(\dfrac{n}{r}\right) \cdot r^{\beta}}} \qquad (3)$$

Figure 6 shows speedups obtained from the resource unconstrained case using Eq. 1 and the resource constrained case manycore architectures using Eq. 3 [15]. In the case of resource unconstrained manycore processors, the performance of cores remains as the number of cores increases thanks to the unlimited transistor resources. In the resource unconstrained case, around 10 times speedup is achieved when 90% parallelism is extractable from applications. On the other hand, in resource-constrained manycore architectures (β is set to 0.5 for Figure 6(b)), it is interesting to see that the speedup is quite smaller than that of the resource-unconstrained case. Maximum speedup is around 1.66 (66% improvement) at the system of employing eight cores when applications are exploiting 90% parallelism from their instruction codes [15]. This is a very disappointing result when compared with the resource unconstrained case. This implies that the real performance benefit of a resource constrained manycore architecture is not that big and manycore architectures, in particular, will not be best architectural templates for the applications of having parallelism smaller than around 50%. Finally, optimized number of core, Sp_{Opt} and nc_{Opt}, can be described as follows [15]:

$$\begin{aligned} Sp_{Opt} &= \left(\frac{1-\beta}{s}\right)^{1-\beta} \cdot \left(\frac{1}{1-s}\right)^{\beta} \\ nc_{Opt} &= \frac{(1-s)\cdot(1-\beta)}{s \cdot \beta} \end{aligned} \qquad (4)$$



(a) Resource-Unconstrainted

(b) Resource-Constrainted

Figure 6. Speedup as the Number of Cores Increases when Resources are constrained [15]

Process variation effect can be imposed on the performance of a core whose size is 'r'. Then, we modify the "*perf(r)*" to include the process variation parameters, σ, φ and T. To find an equation for modeling process variation, we use a fitting function that is approximating the process variation with σ, φ and T and core size r.

In order to derive a fitting equation easily, we set σ, φ and T to constant values and then the approximated fitting functions are derived for each combination of σ, φ and T. In this paper, for the simplicity of analysis, we only use the fitting function derived from the parameter configuration: 0.09 and 0.5 and 65°C as σ, φ and T, respectively. The following equation (Eq. 5) can be derived from VARIUS simulation results.

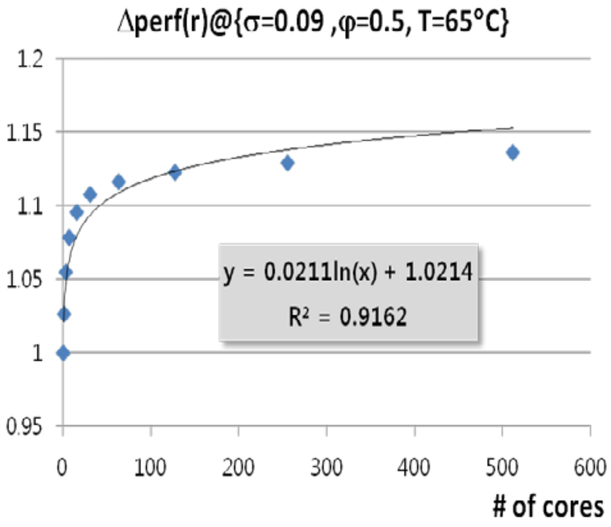


Figure 7. Speedup as the Number of Cores Increases when Resources are Constrained

$$\Delta perf(r)|_{<\sigma,\varphi,T>=<0.09,0.5,65^{\circ}C>} = 0.0211\cdot\log(n/r)+1.0214 \quad \text{with } R^2 = 0.9162 \qquad (5)$$

Since the fitting function has an error that is not ignorable, R^2=0.9162, we use a *table lookup* for calculating the function *Δperf(r)* in our performance model. When all the resource is used for a single core processor, then *Δperf(n)* is evaluated as '1'. *Δperf(r)* is then reduced to the value less than 1 as r is reduced less than n. The process-variation aware perf(r) can be derived from combining Eq. 3 and Eq. 5 as follows:

$$\text{perfWithPV}(r) = \text{perf}(r)\cdot\Delta\text{perf}(r) \qquad (6)$$

Then perf(r) is replaced with new performance function, perfWithPV(r), then final speedup equation can be expressed in the following equation:

$$Sp = \frac{\dfrac{IC}{k\cdot n^{\beta}\cdot\Delta perf(n)}}{\dfrac{s\cdot IC}{k\cdot r^{\beta}\cdot\Delta perf(r)}+\dfrac{(1-s)\cdot IC}{\left(\dfrac{n}{r}\right)\cdot k\cdot r^{\beta}\cdot\Delta perf(r)}} = \frac{\dfrac{1}{n^{\beta}}}{\dfrac{s}{r^{\beta}\cdot\Delta perf(r)}+\dfrac{(1-s)}{\left(\dfrac{n}{r}\right)\cdot r^{\beta}\cdot\Delta perf(r)}} \qquad (7)$$

Figure 8-9 shows the final comparison of optimal core partitioning with/without considering process variation. Figure 8 shows the performance of a single core as its size decrease for increasing total number of cores in a manycore processor. Figure 9 shows the performance of the manycore processor integrating all the performance of individual cores for a given degree of parallelism (For the graph, we use 0.1 as 's'). The better performance benefit of employing a manycore architecture is obtained as the number of cores increases in the process-variation aware case than in the process-variation unaware case.

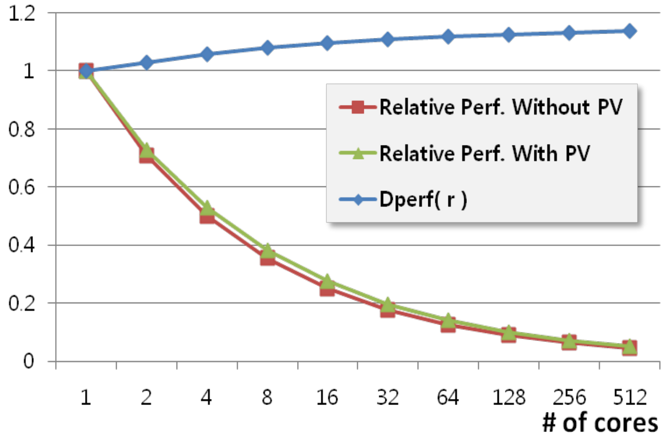
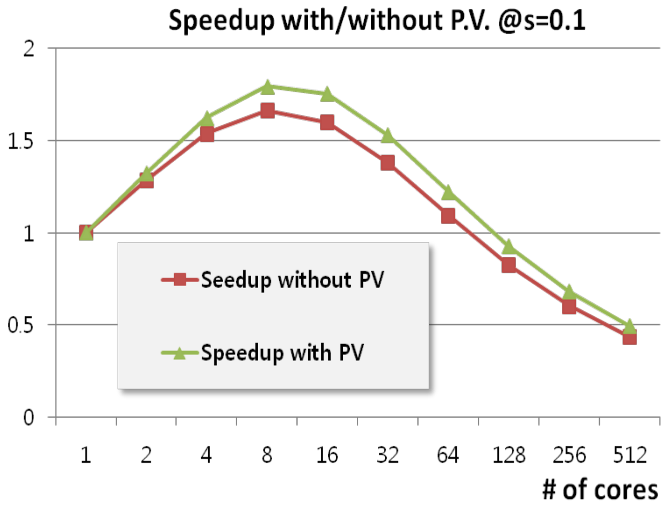


Figure 8. The Performance of a Single Core with/without Process Variation Effect as the Number of Cores Increases in a Resource-constrained Manycore System (i.e., the size of an individual core decreases)



(b)

Figure 9. Speedup with/without Process Variation Effect as the Number of Cores Increases in a Resource-constrained Manycore System

4. Conclusion

Modern nano-meter CMOS technology scaling suffers from process variation. In such a system, a critical path determining clock frequency remarkably is changing by process variation in accordance with increasing design complexity to improve cores performance. A larger core includes more critical paths and there is much probability to increase those path delay by process variation. In consequence, smaller core size will be preferred since the critical path delay increases as core size increases. In this paper, we consider an extension of manycore performance analysis model to include process variation performance impact.

Through the asymptotic analysis based on the models proposed in this paper, we can make the architectural design decisions such as "the number of cores" and "core size", and further we can probe the possible research direction of optimizing the performance of manycore architectures at the future of high process variation era.

Acknowledgements

This research was supported by Hallym University Research Fund, HRF-201209-025.

References

[1] Semiconductor Industry Association, International Technology Roadmap for Semiconductors, **(2007)**.

[2] S. Borkar, “Design Challenges of Technology Scaling”, IEEE Micro, **(1999)** July-August, pp. 23-29.

[3] S. Borkar, “Thousand Core Chips A Technology Perspective”, ACM/IEEE Design Automation Conference (DAC), **(2007)** June, pp. 746-749.

[4] Wikipedia, Graphics Processing Unit. http://en.wikipedia.org/wiki/Graphics_processing_ unit.

[5] S. Borkar, “Parameter Variations and Impact on Circuits and Microarchitecture”, Proceedings, DAC, **(2003)**.

[6] F. Chaix, G. Bizot, M. Nicolaidis and N.-E. Zergainoh, “Variability-aware task mapping strategies for many-cores processor chips”, IEEE 17th International Symposium on On-Line Testing, **(2011)**, pp. 55-60.

[7] S. Majzoub, R. Saleh, S. Wilton and R. Ward, “Energy Optimization for Many-Core Platforms: Communication and PVT Aware Voltage-Island Formation and Voltage Selection Algorithm”, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 29, no. 5, **(2010)** May, pp. 816-829.

[8] Teodorescu and J. Torrellas, “Variation-Aware Application Scheduling and Power Management for Chip Multiprocessors”, International Symposium on Computer Architecture, **(2008)**.

[9] T. Austin, E. Larson and D. Ernst, “Anant Agarwal: SimpleScalar, an Infrastructure for Computer System Modeling, IEEE Computer, **(2002)** February.

[10] G. M. Amdahl, “Validity of single-processor approach to achieving large-scale computing capability”, Proceedings of AFIPS Conference, **(1967)**, pp. 483-485.

[11] K. Asanovic, R. Bodik, B. Catanzaro, J. Gebis, P. Husbands, K. Keutzer, D. Patterson, W. Plishker, J. Shalf, S. Williams and K. Yelick, “The Landscape of Parallel Computing Research: A View from Berkeley”, UCB Technical Paper, **(2006)**.

[12] M. D. Hill, M. R. Marty, Amdahl’s Law in the Multicore Era. Univ. of Wisconsin Computer Sciences Technical Report CS-TR-2007-1593, **(2007)** April.

[13] J.-G. Lee, E. Jung, and D.-W. Lee, “Asymptotic Performance Analysis and Optimization of Resource-Constrained Multi-Core Architectures”, Proceedings of the IEEE International Conference on Microelectronics (ICM), **(2008)**, December 14-17.

[14] S. Cho and R. Melhem, “Corollaries to Amdahl’s Law for Energy”, IEEE Computer Architecture Letters, **(2007)** December.

[15] J.-G. Lee, W. Shin, S.-J. Kim and E Jung, “A Performance/Energy Analysis and Optimization of Multi-Core Architectures with Voltage Scaling Techniques”, IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, vol. E93-A, no. 6, **(2010)** June, pp. 1215-1225.

[16] T. Karnik, S. Borkar and V. De, “Probabilistic and Variation-Tolerant Design: Key to Continued Moore's Law”, ACM/IEEE International Workshop on Timing Issues, **(2004)**.

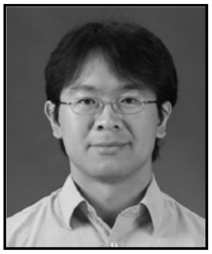
[17] S. R. Sarangi, B. Greskamp, R. Teodorescu, J. Nakano, A. Tiwari and J. Torrellas, “VARIUS: A Model of Process Variation and Resulting Timing Errors for Microarchitects”, IEEE Transactions on Semiconductor Manufacturing, **(2008)**.

Authors

Thi-Yen Phuong is currently a master student in Computer Engineering at Hallym University in South Korea. She received her bachelor's degree of Electrical and Electronics Engineering in University Tenaga National, Malaysia in 2011. Her research interests are many-core microprocessor, system-on chip design and GPU-based parallel processing.

Young-Woong Ko received both a M.S. and Ph.D. in computer science from Korea University, Seoul, Korea, in 1999 and 2003, respectively. He is now a professor in Department of Computer engineering, Hallym University, Korea. His research interests include operating system, embedded system and multimedia system..

Jungmin So received his B.S. degree in computer engineering from Seoul National University in 2001, and Ph.D. degree in Computer Science from University of Illinois at Urbana-Champaign in 2006. He is currently an assistant professor in Department of Computer Engineering, Hallym University. His research interests include wireless networking and mobile computing.



Jeong-Gun Lee received his B.S. degree in computer engineering from Hallym University in 1996, and M.S. and Ph.D degree from Gwangju Institute of Science and Technology (GIST), Korea, in 1998 and 2005. He is currently an assistant professor in the Computer Engineering department at Hallym university. Prior to joining the faculty of Hallym University in 2008, he was a postdoctoral researcher of the Computer Lab. at the University of Cambridge, UK.