International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878 (Online), Volume-8 Issue-4, November 2019

OS Level Power Optimization in LCMP

Vanlalmuansangi Khenglawt, Zonunmawii, Vanlalhruaia

Abstract*: With the increasing levels of transistor count and clock rate of microprocessors there is a significant increase in power dissipation. Reducing power consumption in both high power consumption and high performance has developed into one of the main target in designing a system for various devices. As the chip multiprocessor (CMP) are integrating more cores on the die, it will leads to the extent of Large scale CMP (LCMP) architectures with potentially hundreds of thread on the die and thousands of cores. Therefore, we proposed an approach of OS level power optimization in LCMP to optimize the heat dissipation rate and increase computing power under some considerations. To satisfy the main goal of our work, the heat dissipation should be optimizing with increase in computing power. The approach of optimizing the heat dissipation is done at the synthesis level. There are three approaches for modifying the synthetic benchmark: Singly Synthesis, Hierarchical Synthesis and Group Synthesis. The result is that the power dissipation of Group synthesis is equally distributed without giving more loads to only one processor as compared to Hierarchical Synthesis and Singly Synthesis. Therefore, from our result we can conclude that in Group Synthesis power is equally distributed hence heat dissipation is optimized. The future work will be to further optimize the result of the Synthesis level using thread migration. Thread Migration can increase the system throughput; it relies on multiple cores that vary in performance capabilities.

Keywords* : Large Scale Chip Multiprocessor (LCMP), Power consumption, Power optimization.

I. INTRODUCTION

With the speedy and wide spread of non-traditional computing platforms, particularly movable computing devices and mobile, the dissipation of energy could be a major concern because of the lifetime of battery on the market [7]. Thermal and power-delivery problems have become a lot of vital for superior computing systems. The performance edges of the many technologies are step by step changing into over shadowy by accumulated style complexness and power dissipation [5].

A. *Multiprocessor*

Multiprocessor is that the use of more than one central processing unit (CPUs) into a single computing machine. The term additionally assign to the power of a system to reinforce greater than one processor and/or the capability to allot tasks between them. In multiprocessing system different processors collaborates to produce higher performance. Multiprocessing additionally refers to the execution of different concurrent software processes in a system which is opposite to a one process at any instant [8].

B. *LCMP*

With the performance of a Chip Multiprocessor (CMP) increase continuously as the variety of cores continues to extend with the advancement of technology. CMPs use comparatively easy single-thread processor cores to take advantage of solely moderate amounts of parallelism among any one thread, while executing different threads in parallel across multiple processor cores. As the chip multiprocessor (CMP) are integrating more and more cores on the die, it will ends up to the extent of Large scale CMP (LCMP) architectures with thousands of cores and potentially countless of thread on the die. LCMP is an attractive platform for high bandwidth communication and low latency on the die. [10].

II. METHODOLOGY

This section describes the processor configuration for heterogeneous systems and the experimental methodology. A system is assumed to be simple homogeneous processor cores on the same chip with heterogeneous power-performance capabilities. As with heterogeneous power performance capability, the different approaches at the synthesis level can improve the execution time and performance in the overall system [6].

A microprocessor simulator is build which can be much easier, faster, use less power and be more reliable than fabricating a new microprocessor which is very expensive to design. In this approach SESC simulator is used [4]. There are two requirements that SESC should achieve: First, it has to be modifiable. Second, the code has to be easy to test.

The flowchart of benchmark synthesis and simulation is shown in Fig 1. All these applications are fetch parallel along with the synthesis specification. The application programs and the synthesis specification file are synthesized which gives synthetic benchmark as output file. The synthetic benchmark runs with SESC simulator that gives a required statistics report of the applications. It is mainly concern about the optimization of power, so three different approach are proposed so as to give the desire output : Singly Synthesis, Hierarchical Synthesis and Group Synthesis.

Manuscript published on November 30, 2019.
* Correspondence Author
Vanlalmuansangi Khenglawt*, Faculty in Department of Information Technology, Mizoram University, Aizawl, Mizoram, India. Email: k.sangtei09@gmail.com
Zonunmawii, Faculty in Department of Electronics and Communication Engineering, Mizoram University, Aizawl, Mizoram, India. Email: zonunimiller@gmail.com
Vanlalhruaia, Faculty in Department of Computer Engineering, Mizoram University, Aizawl, Mizoram, India. Email: hruaia_a56@yahoo

In this methodology, any numbers of applications are possible to run on SESC simulator. Here in our SESC simulation, W1, W2...Wn refers to different workload and eight workloads are consider for our experiment.

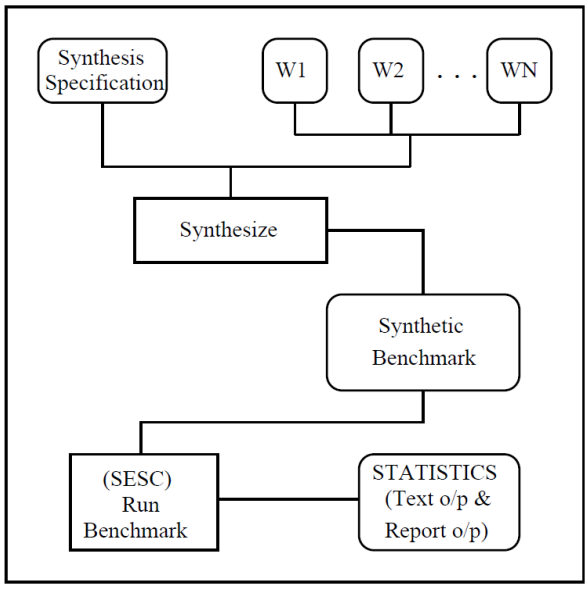

Fig 1: Flowchart of Benchmark Synthesis & Simulation

A. *Singly Synthesis*

Fig 2 shows the pictorial diagram of the singly synthesis. Each workload are running independently on their assigned processors. Each processor runs as long as the assigned workload remains and gets idle when the assigned workload is over. But one of the processor is assigned for running the 'main' thread, it runs from the start of the first workload till the last workload ends. So, the processor assigned for 'main' thread runs till the workloads finish which increase the execution time of the system. Finally, it affects the overall performance of the system.

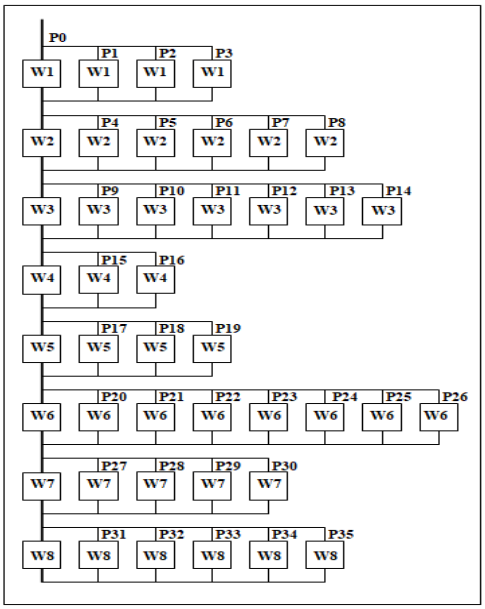

Fig 2: Workload Distribution on Singly Synthesis

B. *Hierarchical Synthesis*

In hierarchical synthesis, the workloads are nested so as to modify the synthetic benchmark. As shown in Fig 3, the workloads which are running on a system call other workloads which again call other workloads in a nested fashion. But there is also a burden in 'main' thread, since it runs from the starting of the first workload till the end of the last workload. As compared to singly synthesis, even though there is a huge power dissipation in the 'main' thread, the power dissipated by each workloads are distributed among themselves since they are arrange in a nested fashion of workload.

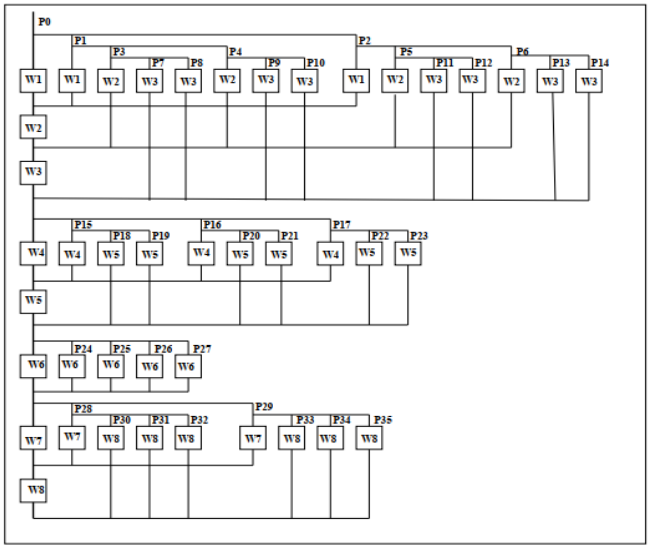

Fig 3: Workload Distribution on Hierarchical Synthesis

C. *Group Synthesis*

In Group Synthesis, unlike the former two approaches, only one processor which is assigned for the ''main' thread does not take the entire burden. In this approach, different workloads are group together, so as to modify the synthetic benchmark which will leads to the minimization of the power dissipation. As the Fig 4, the workloads are grouped into four groups with two workloads each. The 'main' thread still runs from start to end, but a grouping exist which takes up the burden of one processor and divide the power consumption among the workloads.

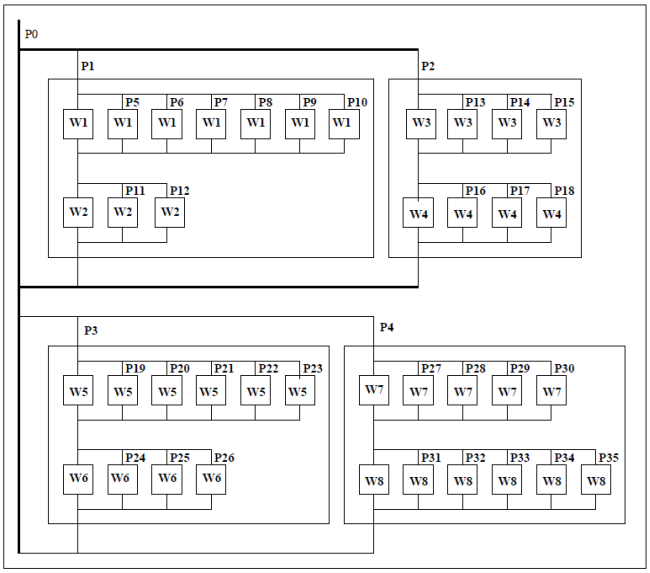

Fig 4: Workload Distribution on Group Synthesis

Retrieval Number: D9552118419/2019©BEIESP
DOI:10.35940/ijrte.D9552.118419
Journal Website: www.ijrte.org

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

III. RESULTS AND ANALYSIS

For simplicity and easy to visualize in this experiment, the processors are arrange in a 6X6 metrics form with eight workloads running on it.

Each of these workloads can allocate multiple processors depending on the users input. The amount of power dissipation of each of the processors are categorize into three groups as shown in Fig 5: low power dissipation (horizontal pattern), medium power dissipation (diagonal pattern) and high power dissipation (cross pattern).

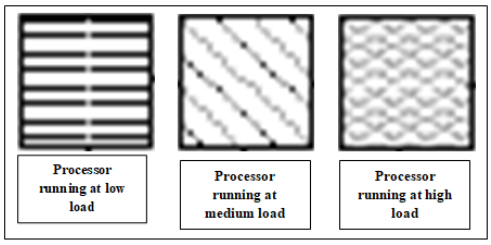

Fig 5: Processor running at low, medium and high load

A. *Power Dissipation of Singly Synthesis*

The workload runs on each processors allocated by the users. The 'main' thread is automatically allocated at Processor 0 (i.e P0). This 'main' thread handles all the workloads and it runs from the starting of the workloads till all the eight workloads runs completely.

In Fig 6, the mapping visualization of cores are shown on 6X6 arrange in matrix form. Here it indicates that the first node P0 is cross pattern (high power dissipation) and all other nodes are horizontal pattern (low power dissipation). Fig 7 gives the power dissipation of each thread. Since all the eight workloads are handle by the 'main' thread i.e thread1, the power dissipation of the first thread is very high with 105.347 watts. All the other workloads run with 16 watts approx. each.

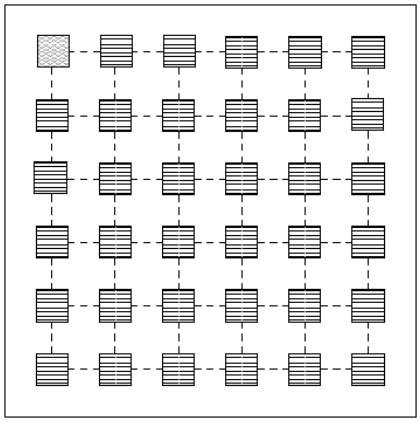

Fig 6: Processor allocation for Singly Synthesis

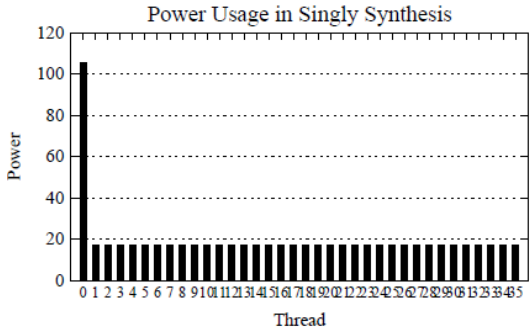

Fig 7: Power dissipation in Singly Synthesis

B. *Power Dissipation of Hierarchical Synthesis*

The workload runs in nesting form. The first workload runs which calls for the second workload and it calls for the third workload and so on. The 'main' thread is again automatically allocated at Processor 0 (i.e P0). This 'main' thread handles all the nesting of the eight workloads and it runs from the starting of the workloads till all the eight workloads runs completely.

In Fig 8, the mapping visualization of cores are shown on 6X6 arrange in matrix form. Here it indicates that the first node P0 is cross pattern (high power dissipation) and all other nodes are horizontal pattern (low power dissipation). Fig 9 gives the power dissipation of each thread. Since all the eight workloads are handle by the 'main' thread i.e thread1, the power dissipation of the first thread is very high with 215.547 watts. All the other workloads run with 18 watts approx. each.

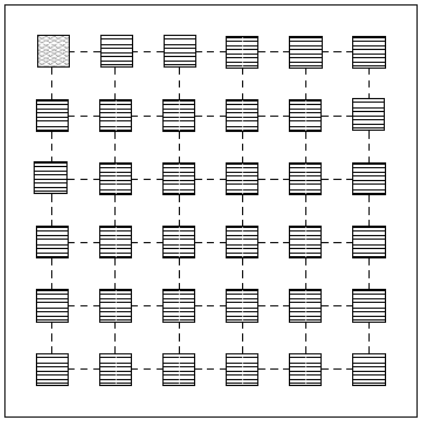

Fig 8: Processor allocation for Hierarchical Synthesis

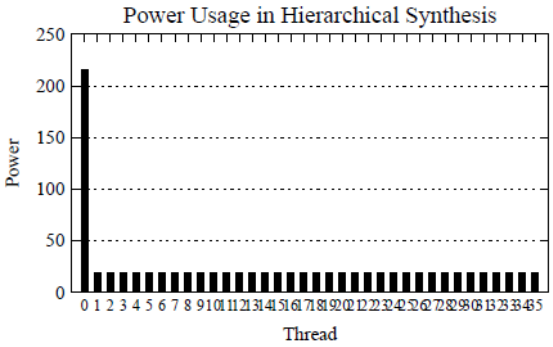

Fig 9: Power dissipation in Hierarchical Synthesis

C. *Power Dissipation of Group Synthesis*

Two workloads each are group together to minimize the power dissipation of the 'main' thread which is allocated at Processor 0 (P0). Since there are eight workloads, they are separated into four groups. Each of these groups is handled by different processors. The thread which handles these four groups is then automatically allocated at Processor P1, P2, P3 and P4.

In Fig 10 the mapping visualization of cores are shown on 6X6 arrange in matrix form. Here it indicates that the first node P0 is horizontal pattern (low power dissipation) and all other nodes are diagonal pattern (medium power dissipation). Fig 11 gives the power dissipation of each thread. Since all the eight workloads are handle by four groups i.e thread2, thread3, thread4 and thread5, the power dissipation of the first thread is very low with 6.969 watts.

The four threads give a power dissipation of 36 watts approx. and all the other workloads run with 20 watts approx. each.

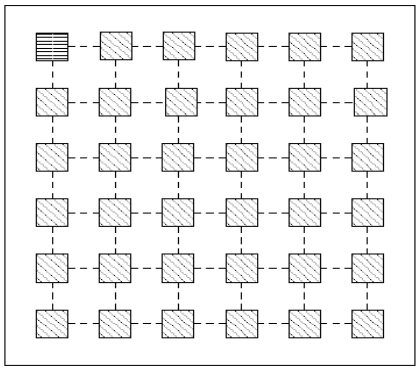

Fig 10: Processor allocation for Group Synthesis

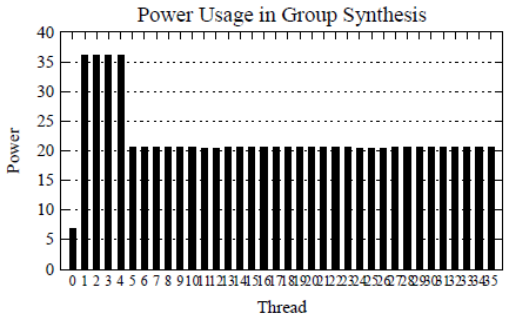

Fig 11: Power dissipation in Group Synthesis

D. *Comparisons of Synthesis in Power Dissipation.*

From the above three approaches the power dissipation of the first thread is shown in the following table i.e Table 1

Approach	Power Dissipation of First Thread
Singly Synthesis	105.347 watts
Hierarchical Synthesis	215.547 watts
Group Synthesis	6.969 watts

Table 1: Power Dissipation of the First Thread

IV. CONCLUSION

The main goal of our work is to optimize the heat dissipation. The approach of optimizing the heat dissipation is done at the synthesis level. Here power is optimize by modifying the synthetic benchmark and run on top of SESC simulator. Three approaches have been proposed for modifying the synthetic benchmark - Singly synthesis, Hierarchical synthesis and Group synthesis. From the result shown in Fig 7, Fig 9 and Fig 11, the power dissipation of Group synthesis is equally distributed without giving more loads to only one processor as compared to hierarchical synthesis and singly synthesis. The result found in Table 1 also shows that on comparing to Singly synthesis, the power dissipation of the first thread on Group Synthesis have an improvement of 93% approximately and also on comparing to Hierarchical synthesis, the power dissipation of the first thread on Group synthesis have an improvement of 96% approximately. Therefore we can conclude that heat dissipation is best optimized in Group synthesis. So in terms of power dissipation, grouping of the workloads using Group synthesis is the most preferable approach and it can thus improve the execution time and performance in the overall system.

REFERENCES

1. Jungseob Lee, Chi-Chao Wang, Hamid Ghasemi, Lloyd Bircher, Yu Cao,and Nam Sung Kim. Workload-adaptive process tuning strategy for power efficient multi-core processors. In *Low-Power Electronics and Design (ISLPED), 2010 ACM/IEEE International Symposium on*, page 225-230.
2. F. Gatti, et al., "Low Power Control Techniques for TFT LCD Displays," *Proc. Int'l Conf. Compilers, Architecture, and Synthesis for Embedded Systems (CASES)*, 2002.
3. R. Bergamaschi, Guoling Han, A. Buyuktosunoglu, H. Patel, I. Nair, G. Dittmann, G. Janssen, N. Dhanwada, Zhigang Hu, P. Bose, and J. Darringer. Exploring power management in multi-core systems. In *Design Automation Conference, 2008. ASPDAC 2008. Asia and South Pacific*, pages 708-713.
4. Pablo Montesinos Ortego and Paul Sack. Sesc: Superescalar simulator. Technical report, 2004.
5. D. Brooks and M. Martonosi. Dynamic thermal management for high performance microprocessors. In *High-Performance Computer Architecture. The 7th International Symposium on*, pages 171-182, 2001.
6. Koushik Chakraborty, Philip M. Wells, and Gurindar S. Sohi. Computation spreading: employing hardware migration to specialize CMP cores on-the-fly. In *Architectural Support for Programming Languages and Operating Systems*, pages 283-292, 2006.
7. Min-Sik Gong, Yeong Rak Seong, and Cheol-Hoon Lee. On-line dynamic voltage scaling on processor with discrete frequency and voltage levels. 44 In *Convergence Information Technology, International Conference*, pages 1824-1831, 2007.
8. John L. Hennessy and David A. Patterson. *Computer Architecture; A Quantitative Approach*. Morgan Kaufmann Publishers Inc., 1st edition, 1992.
9. Arindam Mallik, Bin Lin, Gokhan Memik, Peter Dinda, and Robert P.Dick. User-driven frequency scaling. *Computer Architecture Letters*, page 16, 2006.
10. Li Zhao, R. Iyer, J. Moses, R. lllikkal lllikkal lllikkal lllikkal, S. Makineni, and D. Newell. Exploring Large-Scale CMP Architectures UsingManySim. *Micro, IEEE*, pages 21-33, 2007.
11. Chongmin Li, Haixia Wang, Yibo Xue, Xi Zhang, and Dongsheng Wang. Fast Hierarchical Cache Directory: A Scalable Cache Organization for Large-Scale CMP. In *Networking, Architecture and Storage (NAS), 2010 IEEE Fifth International Conference on*, pages 367-376.

AUTHORS PROFILE

Vanlalmuansangi Khenglawt has done her B.E in Computer Science and Engineering from Jorhat Engineering College in the year 2008, MTech in Computer Science and Engineering from IIT Guwahati from 2009-2011. She is currently working as an Assistant Professor in the Department of Information Technology, Mizoram University. Her research interests are Machine translation and Natural Language Processing.

Zonunmawii has done her B.E in Electronics and Communication Engineering from Dr. Ambedkar Institute of Technology, Bangalore and M.Tech in Electronics and Communication Engineering from Lovely Professional University, Punjab. She is currently working as an Assistant Professor in the Department of Electronics and Communication Engineering, Mizoram University, Tanhril, Aizawl, Mizoram, India.

Vanlalhruaia graduated from Jorhat Engineering college, Dibrugarh University in 2005 in Computer Science and Engineering, after working for four years as lecturer in Mizoram polytechnic, Lunglei He continued for higher study and got Mater degree in Information Technology from tezpur university. He is working as an Assistant Professor in the Department of Computer Engineering, Mizoram University. His area of interest is Formal Language and Web Technology.