Slight Heterogeneity in Multi-core Architecture: An Experimental & Comparative Study

Abdullah Al Mamun, Hilal H. Nuha, Sultan Anwar, Hassan Ali

Abstract— there is a growing consensus that heterogeneous multicores are the future of CPUs. These processors would be composed of cores that are specifically adapted or tuned to particular types of applications and use cases, thereby increasing performance. The move from homogeneous to heterogeneous multicores causes the design space to explode, however. An architect of a heterogeneous processor must make design decisions per processor core rather than once for the entire processor as before. Currently, there are no methods for handling this design complexity to yield a processor that performs well for real workloads. As a step forward, we propose weak heterogeneity. A weakly heterogeneous processor is one whose cores are different, but not significantly so. The cores share an ISA and major microarchitectural features, differing only in minor details. Limiting the design space in this way allows us to explore the heterogeneous space without becoming overwhelmed by its s ize. We show preliminary results suggesting that a design space so constrained still has interesting trade-offs among performance, power consumption, and area.

Index Terms— Multicore, Processor, GPU, Microprocessor, Cache, Gem5, Processing Simulation.

1 Introduction

Increasing processor performance by adding more cores on a single chip is not a long term solution since chip temperature will restrict the number of core. Because of power limit, multicore scaling grows no more than 50% even though 8nm is used. Heterogeneous processors can solve this restriction partially. Each different CPU will have a different specification so that Heterogeneous processors will choose the suitable CPU for a certain job. Weakly Heterogeneous design is one of the initial approaches for Heterogeneous CPU. [1] Weak Heterogeneous Design means a multicore processor with slightly different cores. The cores use similar ISA and micro-architecture but different in small details. A typical multicore architecture is shown in fig (1).Increasing processor performance by adding more cores on a single chip is not a long term solution since chip temperature will restrict the number of core. Because of power limit, multicore scaling grows no more than 50% even though 8nm is used. Heterogeneous processors can solve this restriction partially. Each different CPU will have a different specification so that Heterogeneous processors will choose the suitable CPU for a certain job. Weakly Heterogeneous design is one of the initial approaches for Heterogeneous CPU.[1] Weak Heterogeneous Design means a multicore processor with slightly different cores. The cores use similar ISA and micro-architecture but different in small details. A typical multicore architecture is shown in fig (1). We wish to evaluate a weakly heterogeneous design performance by performing benchmark in gem5 simulator. In our case, we wish to implement multicore with the same ISA but different cache size.

- *Abdullah Al Maun, Hassan Ali and Sultan Anwar are currently pursuing masters degree program in Computer engineering in King Fahd University of Petroleum & Minerals, Dammam,KSA. PH-0594968041. E-mail: g201403680@kfupm.edu.sa*
- *Hilal H. Nuha is currently pursuing Phd degree program in electric power engineering in King Fahd University of Petroleum & Minerals, Dammam, KSA. E-mail: g201309210@kfupm.edu.sa*

2 Gem5 Simulator and Other Simulators

We choose gem5[2] because of its supports on many types of ISA. The gem5 simulator is a fusion of the best features of the M5 [3] and GEMS [4] simulators. GEMS provide a simulation with a detailed and flexible memory system. M5 gives us a highly configurable simulation framework, multiple ISAs, and diverse CPU models. Nowadays, gem5 supports most popular ISAs (x86, ARM, ALPHA, MIPS, Power, and SPARC).

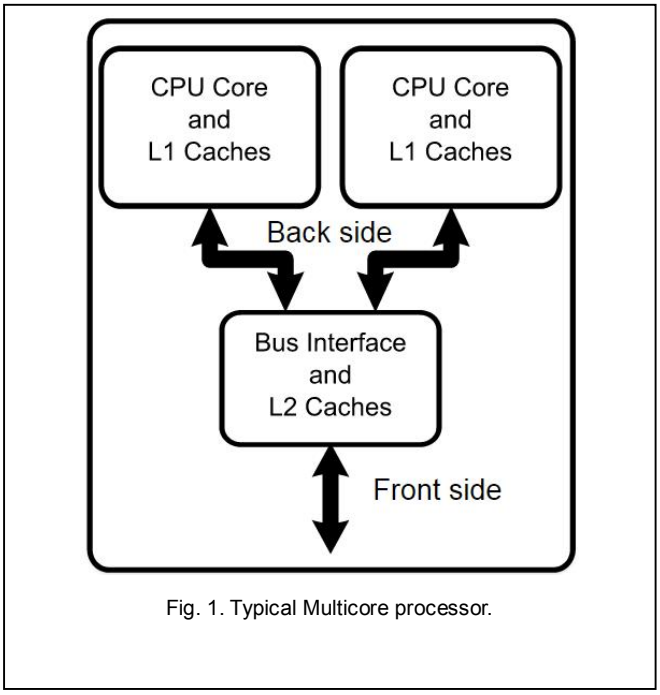

Fig. 1. Typical Multicore processor.

International Journal of Scientific & Engineering Research Volume 6, Issue 10, October-2015
ISSN 2229-5518

3 EXPEIMENTAL SETUP

Simulations are based on the x86 detailed and arm_detailed processor model distributed with gem5. As benchmark, we will use matrix multiplication and 8-Queens problem, and evaluate the running time performance. We run the simulation in SE mode.

- X86 2 cores, CPU type: detailed, combination
  - P0: Both core 64kB, associativity 2.
  - P1: Core 1: 32kB, Core 2: 64 kB, associativity 2.
  - P2: Core 1: assoc 2, Core 2: assoc 1, both: 64kB
  - P3: Core 1: assoc 1, Core 2: assoc 1, both: 64kB
- ARM 2 cores, CPU type: arm_detailed, combination
  - P4: Both core 64kB, assoc 2
  - P5: Core 1: 32kB, Core 2: 64 kB
  - P6: Core 1: assoc 2, Core 2: assoc 1, both: 64kB
  - P7: Core 1: assoc 1, Core 2: assoc 1, both: 64kB

We used gem5 hello world, matrix multiplication with dimension 32x32 and N-Queen Problem where N=8 as benchmark program.

4 RESULT AND ANALYSIS

4.1 Processor Run Time

We conduct some initial experiments. We apply some ISA with different configurations and different benchmark program. We have ARM, x86, and x86 with 2 cores. For benchmark program we have helloworld, matmul1 is a 32 x 32 matrix multiplication, and matmul2 is a double 32x32 matrix multiplication. As performance evaluation, we evaluate Number of tick which represent the clock cycle of the CPU.

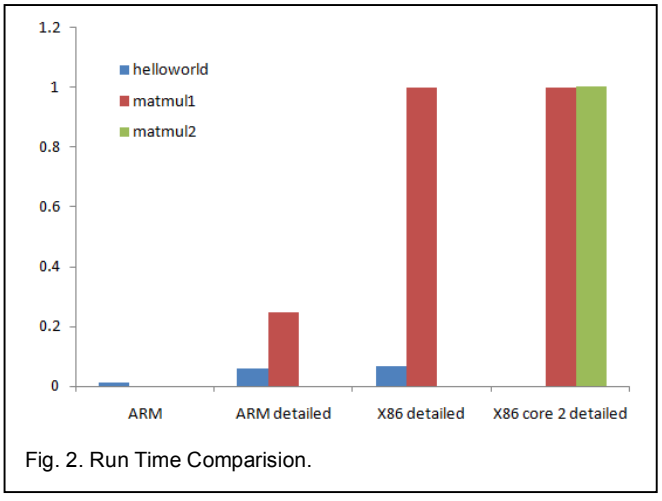

Fig. 2. Run Time Comparision.

Run Time comparison among many different processor architectures are shown in fig 2. We normalized the running Number of tick for each benchmark. One most important result is x86 core 2 on matmul1 and matmul2. As shown in fig 2, indicates that x86 with two core execute double matrix multiplication almost as fast as single core executing single matrix multiplication. Both cores work simultaneously to calculate the matrix calculation. Each matrix is distributed to each core and each core works on a matrix as if single core works at single matrix. In this paper, we did performance evaluation for 8-Queens problem for different processor architecture individually. The runtime of processors x86 and ARM are shown respectively in fig 3 and fig 4, clearly indicates that processor 3 takes much time whereas others takes only half of P3. In contrast, P4 and P5 have similar run time in case of processor ARM. However, in both processors, changing the value of associativity has no impact on run time. Only 8 queen problems are considered as benchmark for all following performance test.

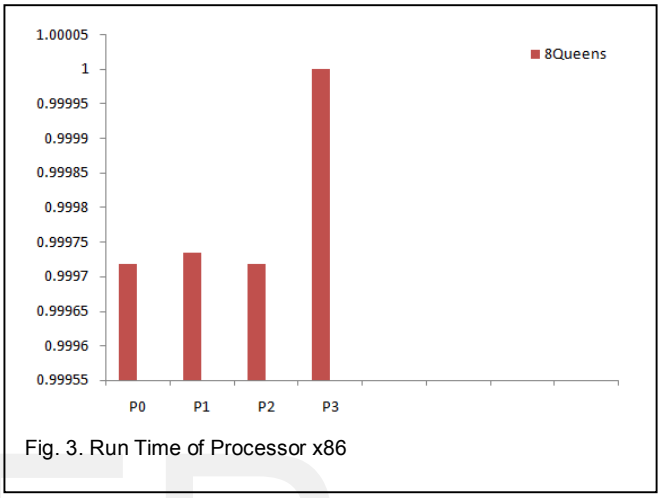

Fig. 3. Run Time of Processor x86

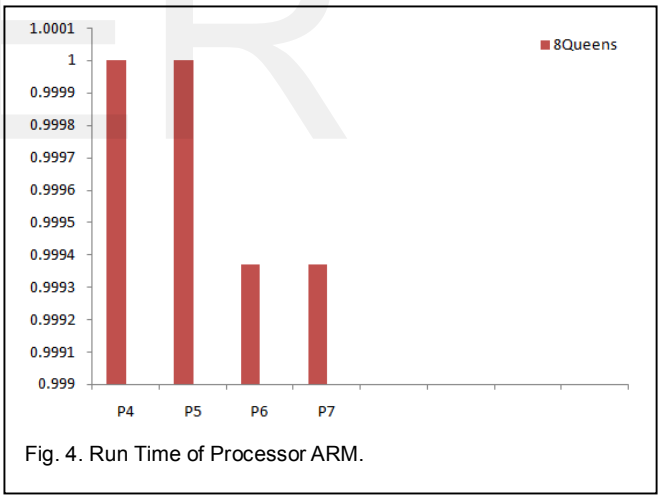

Fig. 4. Run Time of Processor ARM.

4.2 Miss Rate (Data Cache)

To evaluate miss rate, we consider data cache only because of having some modification on configuration for data cache. When processor want to read and write data from cache executing load and store instructions, if cache hit happen then we have no miss processor gets data and continue its next executions. Miss happens when processors don't get data or write data.

International Journal of Scientific & Engineering Research Volume 6, Issue 10, October-2015
ISSN 2229-5518

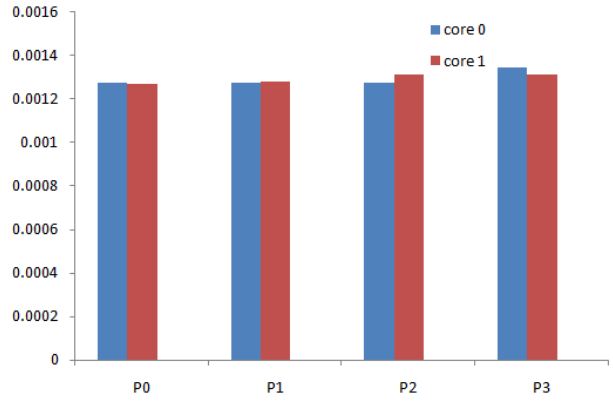

Fig. 5. Cache Miss Rate of processor x86

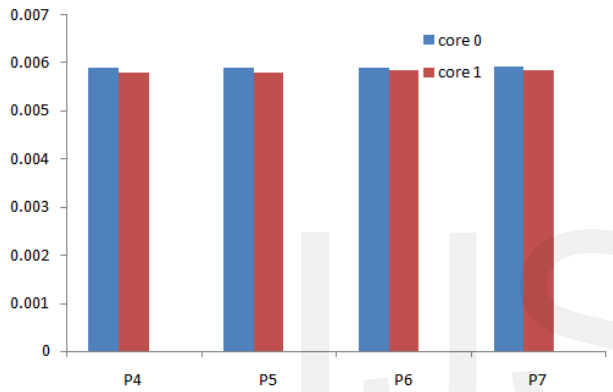

Fig. 6. Cache Miss Rate of processor ARM.

4.3 Number of Instructions

Number of instruction to be executed is very important to calculate processor performance. It is depends on which ISA is used. In this test, we count number of instructions in both processors x86 and ARM that are shown below in fig 7 and 8. However, in case of x86 all processor execute same number of instructions for a given benchmark program on the other hand core 1 and core 0 of ARM are slightly different than x86.

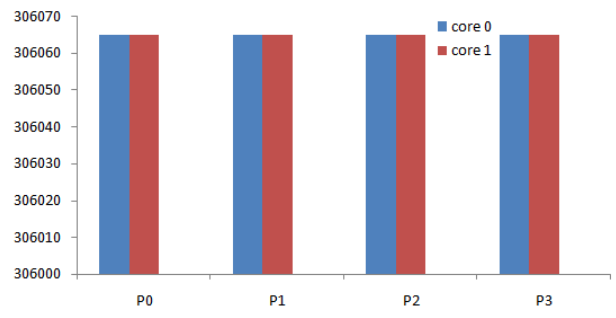

Fig. 7. x86 Number of Simulated Instruction

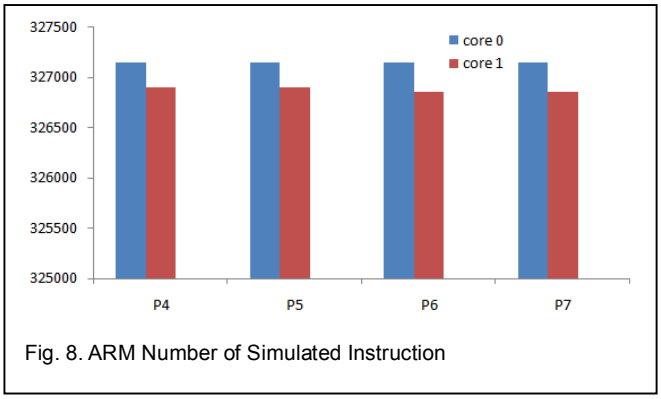

Fig. 8. ARM Number of Simulated Instruction

4.4 CPU Clock Cycle

It refers to the frequency of multi core processor. The clock cycle is different with the size of cache in processor. If miss rate is growing high then CPU clock rate becomes also high. The CPU clock time is shown in fig 9 and 10 for both processors. Highest CPU time needs for P3 in x86 and lowest CPU time taken for P7 of ARM.

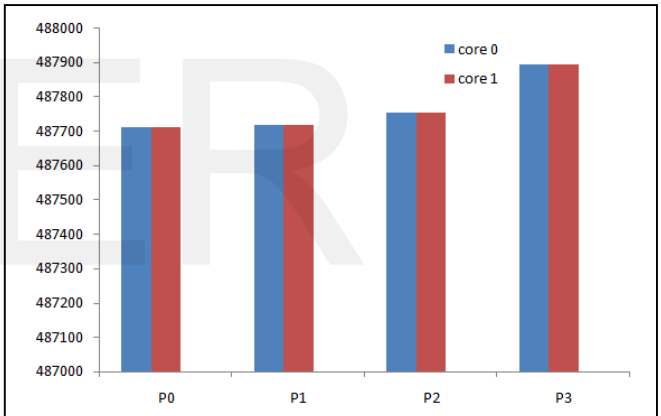

Fig. 9. CPU clock cyple of processor x86

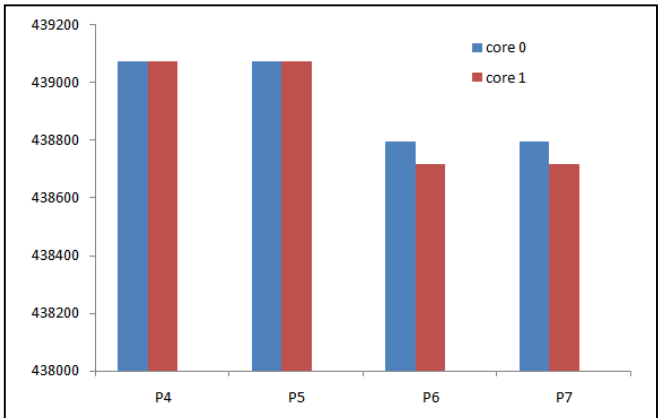

Fig. 10. CPU clock cycle of processor ARM

International Journal of Scientific & Engineering Research Volume 6, Issue 10, October-2015
ISSN 2229-5518

4.5 Clock per Instruction CPI

CPI is most important indicator to measure performance of a processor. It plays an important role while comparing among different processor. According to fig 11 and fig 12, ARM has less CPI to complete 8-Queen problem than x86 that is only 1. It is a good indication of fastness.

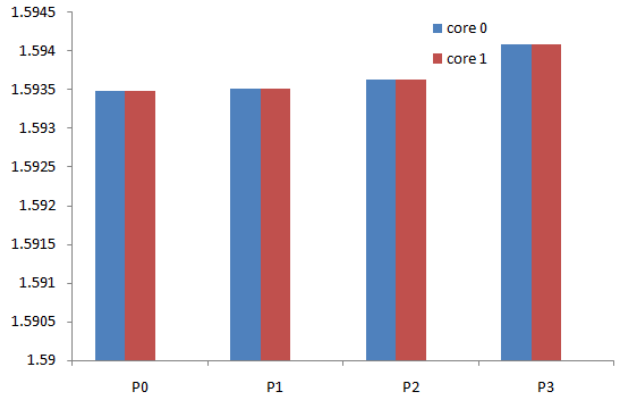

Fig. 11. CPI of processor x86

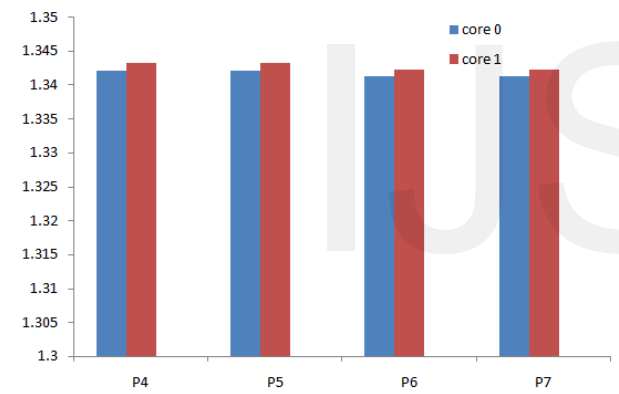

Fig. 12. CPI of processor ARM

4.6 Idle Cycle

It refers the total number of cycles that the CPU has spent unscheduled due to idle. As we can see in fig 13 and fig 14, clearly observed that core 0 is busier than core 1.

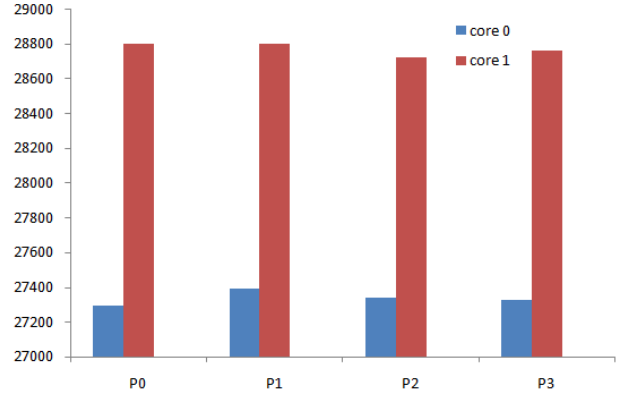

Fig. 13. Idle of processor x86

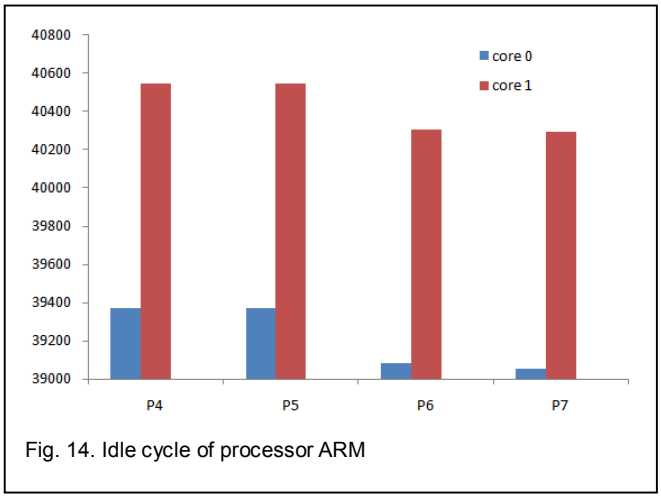

Fig. 14. Idle cycle of processor ARM

4 Conclusion

Based on our simulation, it is possible to put different configuration for each core without significantly affecting the performance. We want to continue our investigation on this project. We will different configuration using more advance benchmark and more combinations in future.

References

[1] Tomusk, Erik, and Michael O'Boyle. "Weak heterogeneity as a way of adapting multicores to real workloads." Proceedings of the 3rd International Workshop on Adaptive Self-Tuning Computing Systems. ACM, 2013.

[2] Binkert, Nathan, et al. "The gem5 simulator." ACM SIGARCH Computer Architecture News 39.2 (2011): 1-7.

[3] Binkert, Nathan L., et al. "The M5 simulator: Modeling networked systems." IEEE Micro 26.4 (2006): 52-60.

[4] Martin, Milo MK, et al. "Multifacet's general execution-driven multiprocessor simulator (GEMS) toolset." ACM SIGARCH Computer Architecture News 33.4 (2005): 92-99.

[5] Esmaeilzadeh, Hadi, et al. "Dark silicon and the end of multicore scaling." Computer Architecture (ISCA), 2011 38th Annual International Symposium on. IEEE, 2011.93. (Thesis or dissertation)