A Comparison of x86 Computer Architectural Simulators

Ayaz Akram and Lina Sawalha

Electrical and Computer Engineering Department, Western Michigan University

ABSTRACT

Computer architecture simulators are widely used by computer architects to evaluate different design options. This work (continuation of earlier work [1]) explores different x86 computer architecture simulators and quantifies the simulation inaccuracies. x86 is one of the oldest and widely used instruction set architectures (ISAs) used in desktops and servers. We selected gem5 [2], Sniper [3], MARSSx86 [4] and ZSim [5], and configured them to model one of the state-of-the-art high-performance processors, Intel's *Haswell* microarchitecture, to compare and evaluate the accuracy of such simulators to model actual products.

SELECTED SIMULATORS

gem5

- gem5 [2] is a full system simulator that supports many ISAs (x86, ARM, SPARC, Alpha and MIPS) with various CPU models (non-pipelined, in order pipelined, out-of-order pipelined).
- Pipelined models can be configured to simulate different number of pipeline stages, issue widths and number of hardware threads.

Sniper

- Sniper [3] is a parallel simulator for simulating large scale multicore systems using interval simulation, which provides a balance between detailed cycle-level simulation and one-IPC (single issue pipeline model) simulation.
- 'Instruction window centric' core model was added to the simulator later on to improve its accuracy.

MARSSx86

- MARSSx86 [4] is a cycle-level full system x86-64 simulator.
- It is based on PTLsim and QEMU.
- It can model heterogeneous multi-core processors.

ZSim

- ZSim [5] is a parallel and scalable x86-64 simulator.
- It extensively uses dynamic binary translation and focuses on simulating detailed memory hierarchies.

Table 1
Feature Comparison of Selected Simulators

Feature	Gem5	Sniper	MARSSx86	Zsim
Platform / target Support	P++	P	P	P
Full System	✔	X	✔	X
Fast forwarding & cache warmup	✔	✔	✔	✔
Checkpointing	✔	X	✔	✔
Details of stats.	D++	D	D+	D+
Energy/power	E++	E	E	E
HMP support	M,G,S	S	S	S
GPU modelling	✔	X	X	X
In Order Pipeline	✔	✔	✔	✔
Community support	C++	C++	C++	C+

Note: [feature's 1st letter]++ is better than [feature's 1st letter] + which is better than [feature's 1st letter] which is better than [feature's 1st letter] - , S=Single-ISA, M=Multi-ISA, G=GPU

SIMULATORS VERIFICATION METHODOLGY

- All simulators configured to model hardware configuration similar to Intel *Haswell* (Intel i7-4770 cpu, 3.4 GHz).
- SPEC-CPU2006 and a subset of MiBench embedded benchmark suites simulation timing and performance results compared to real hardware runs.
- SPEC benchmarks executed for 500 million instructions chosen from a statistically relevant portion of the program (based on SimPoint), after a warming up period of 100 million instructions.
- IPC (instructions per cycle), branch misprediction and cache miss ratios measured on real hardware using hardware monitoring counter tools (PAPI).
- Use of 64 bit binaries of all workloads for all simulators.

Table 2
Target Configurations

Parameter	Core i7 Like
Pipeline	Out of Order
Fetch width	6 instructions per cycle
Decode width	4-7 fused uops
Decode queue	56 uops
Rename and issue widths	4 fused uops
Dispatch width	8 uops
Commit width	4 fused uops
Reservation station	60 entries
Reorder buffer	192 entries
Stages	19
L1 data cache	32KB, 8 way
L1 instruction cache	32KB, 8 way
L2 cache size	256 KB, 8 way
L3 cache size	8 MB, 16 way
L1, L2 and L3 cache latency	4,12 and 36 cycles
Branch predictor	Tournament
Branch misprediction penalty	14 cycles
BTB and RAS entries	4K and 16

RESULTS

NOTE: *avg-E* : average absolute error
***avg-E-NO*: average absolute error with no outlier (more than 50% error)**

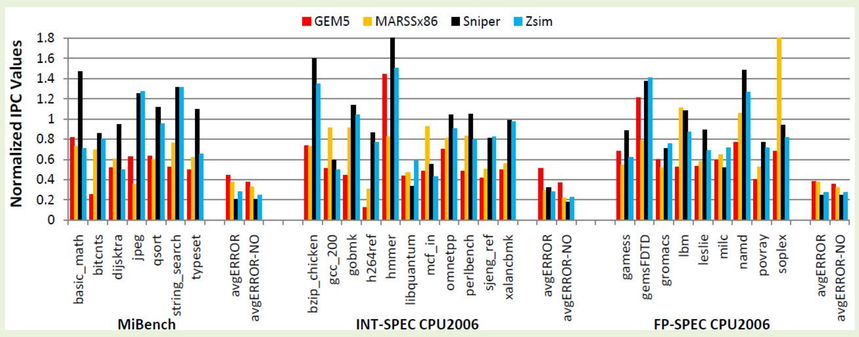

Figure 1. Normalized IPC values for single core runs

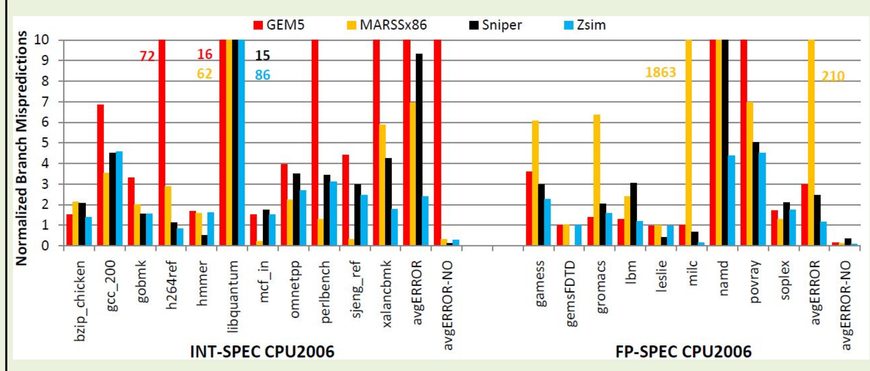

Figure 2. Normalized branch mispredictions for single core runs

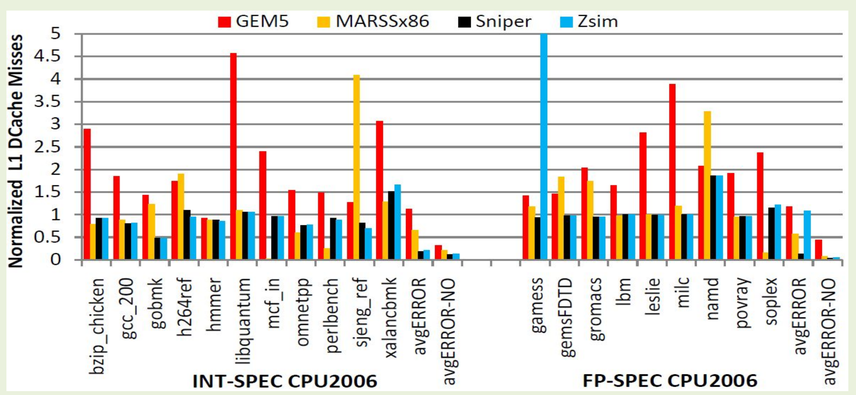

Figure 3. Normalized L1-d cache misses for single core runs

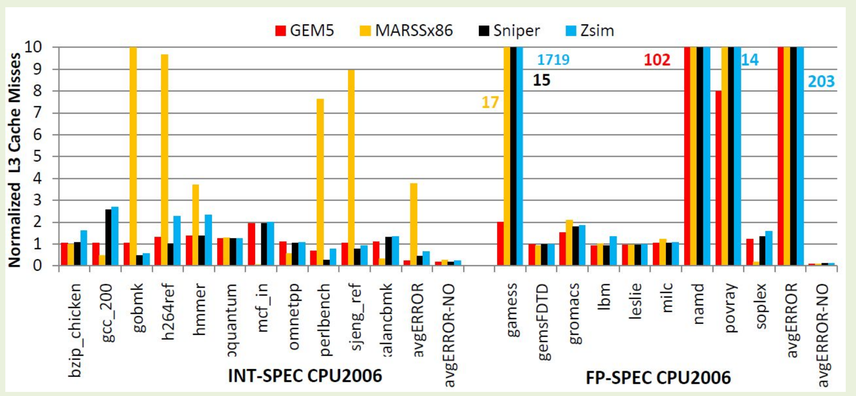

Figure 4. Normalized L3 cache misses for single core runs

- Sniper is the most accurate for all types of workloads.
- The mean absolute percentage error (MAPE) in IPC values compared to hardware runs (excluding outliers) for MiBench is: 20.6%, 37.6%, 33.03% and 24.3% for Sniper, gem5, MARSSx86 and ZSim.

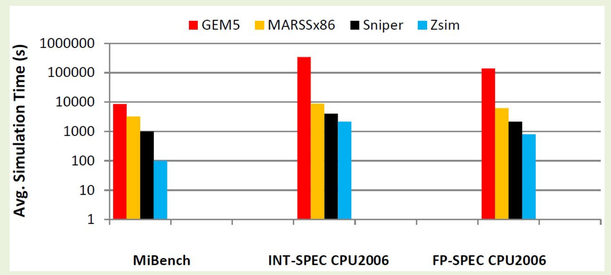

Figure 5. Average simulation time for all simulators

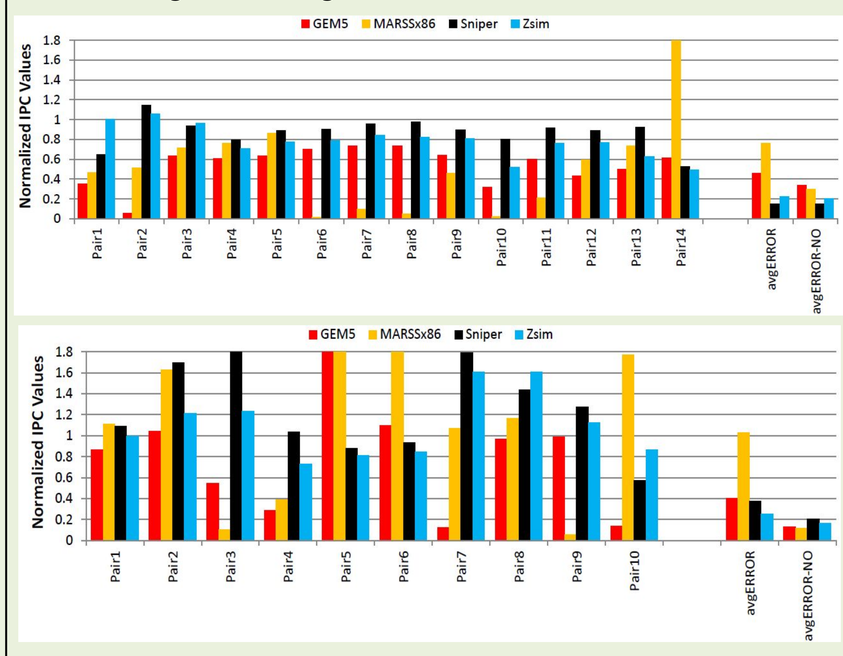

Figure 6. (a) Normalized IPC values for 2 core and (b) 4 core runs

- MAPE for integer benchmarks is 17.6%, 37.1%, 22.16% and 22.59% for Sniper, gem5, MARSSx86 and ZSim respectively.
- For floating point benchmarks, the MAPE excluding outliers is 24.8%, 35.4%, 32% and 27.5% for Sniper, gem5, MARSSx86 and ZSim respectively.
- Sniper and Zsim show close accuracy for dual core and quad core runs (more accurate than the other two simulators).
  .
- As shown in the figures, the average error in total cache misses and branch mispredictions goes above 100% for various cases. This results in high underestimation of IPC values for some benchmarks. Examples are most of the outliers in gem5 (*h264ref, gcc_200, gobmk, perlbench, namd, povray*). Examples for MARSSx86 include: *h264ref, milc, libquantum, povray*. Most of these benchmarks have significant number of committed branch instructions (20% or more) which exposes the inefficiency of modeled branch predictor when compared to *Haswell* branch predictor. Some of the IPC inaccuracies can be due to the way some of the x86 instructions are decoded and implemented in gem5.
- Some of the inaccuracies can be a result of lack of support of fused u-ops, and u-op cache of *Haswell* (significantly reduces the effective pipeline depth in case of u-op cache hit).
- Comparison of simulation times shows ZSim to be the fastest simulator out of the studied simulators.

CONCLUSIONS

The experimental results indicate that the simulators which have been validated for Intel Haswell like architectures show better accuracy as compared to the ones which are not validated and calibrated for such targets. Sniper and ZSim showing more accuracy have been validated for Intel *Nehalem* and *Westmere* systems [6,7]. In future, we plan to dig deep into issues with these simulators and potentially fix them.

REFERENCES

[1] A. Akram and L. Sawalha, "x86 Computer Architecture Simulators: A Comparative Study", in IEEE *ICCD,* 2016.

[2] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, et al., "The gem5 Simulator," *SIGARCH Comp. Arch. News*, vol. 39, pp. 1–7, 2011.

[3] T. E. Carlson, W. Heirman, and L. Eeckhout, "Sniper: Exploring the Level of Abstraction for Scalable and Accurate Parallel Multi-Core Simulation," *in* ACM *Int. Conf. for High Perf. Comp., Net., Storage and Analysis*, pp. 1 – 12, 2011.

[4] A. Patel, F. Afram, and K. Ghose, "MARSS-x86: A Qemu-Based Micro-Architectural and Systems Simulator for x86 Multicore Processors," in *DATE* p. 29–30, 2011.

[5] D. Sanchez and C. Kozyrakis, "ZSim: Fast and Accurate Microarchitectural Simulation of Thousand-Core Systems," in *ISCA*., vol. 41, pp. 475–486, 2013

[6] T. E. Carlson, W. Heirman, S. Eyerman, I. Hur, and L. Eeckhout, "An evaluation of high-level mechanistic core models," ACM *TACO,* vol. 11, no. 3, p. 28, 2014.

[7] http://zsim.csail.mit.edu/tutorial/slides/validation.pdf