# A Comparison of x86 Computer Architecture Simulators

Ayaz Akram and Lina Sawalha*
Dept. of Electrical and Computer Engineering
Western Michigan University, Kalamazoo, MI

**S**imulation is used as a primary performance evaluation methodology in most computer architecture publications. There is not much literature dealing with the evaluation of simulators by comparing them to each other and to the state-of-the-art processors. The absence of performance validation of simulators may cause experimental errors that can lead to incorrect conclusions. This work provides a simulation accuracy and performance comparison of four modern x86 computer architecture simulators: gem5 [1], Sniper [2], MARSSx86 [3] and ZSim [4]. We configured these simulators to model one of Intel's high-performance processor, Core-i7 *Haswell* microarchitecture based CPU. Then we quantified the experimental errors. The selected simulators for this study have diverse design strategies with respect to detail and abstraction. All of them are contemporary simulators with active development.

Validation effort for gem5 simulator exists for ARM based systems [1, 5], however; no validation for x86 systems exists. Latest validation effort for Sniper was done for Intel Nehalem microarchitecure processor showing a single-core error of 11.1% for a subset of SPLASH-2 benchmarks [2]. ZSim has been validated against an Intel Westmere system showing an average error of 10% [6]. We did not find any rigorous validation effort for MARSSx86. Our previous work compared the accuracy and speed of some of the aforementioned simulators, and few others, for single core simulations only [7]. This work, compares more simulators for single-core and multicore runs with real hardware platform runs.

The experimental system that we used to test the four simulators is similar to Haswell microarchitecture (i7-4770 CPU, 3.40 GHz), see Table 1. As all the exact configurations for our target processor are not published by Intel, we tried our best to model similar features based on some Intel documentation [8] and other sources [9, 10, 11]. We compared the instruction per cycle (IPC), L1 data cache miss, L3 cache miss and branch misprediction values from the simulation results with that of real hardware runs for MiBench and SPEC-CPU2006 benchmark suites.

Figure 1 shows single-core's IPC, branch mispredictions and L1 data cache misses, and dual-core's and quad-core's IPC runs normalized to results obtained on real hardware, using hardware monitoring counters. For dual and quad-core runs, benchmark combinations are randomly selected from SPEC-CPU2006 benchmark suite. The figure shows two types of average errors for all evaluated metrics: one including all benchmarks, *avgERROR*, and the other without outliers, $avgERROR - NO$; where an outlier corresponds to more than 50% inaccuracy in a metric.

**Table 1: Target Configurations.**

| Parameter | Core i7 Like |
|---|---|
| Pipeline | Out of Order |
| Fetch width | 6 instructions per cycle |
| Decode width | 4-7 fused $\mu$-ops |
| Decode queue | 56 $\mu$-ops |
| Rename and issue widths | 4 fused $\mu$-ops |
| Dispatch width | 8 $\mu$-ops |
| Commit width | 4 fused $\mu$-ops per cycle |
| Reservation station | 60 entries |
| Reorder buffer | 192 entries |
| Number of stages | 19 |
| L1 data cache | 32KB, 8 way |
| L1 instruction cache | 32KB, 8 way |
| L2 cache size | 256KB, 8 way |
| L3 cache size | 8 MB, 16 way |
| Cache line size | 64 Bytes |
| L1 cache latency | 4 cycles |
| L2 cache latency | 12 cycles |
| L3 cache latency | 36 cycles |
| Operation latency | Based on [8, 9] |
| Branch predictor, BTB entries | Tournament, 4K |
| Branch misprediction penalty | 14 cycles |

For embedded benchmarks, the mean absolute percentage error (MAPE) in IPC values (without outliers) compared to real hardware runs is: 20.6%, 37.6%, 33.03% and 24.3% for Sniper, gem5, MARSSx86 and ZSim respectively. The MAPE (without outliers) for integer benchmarks for Sniper, gem5, MARSSx86 and ZSim is 17.6%, 37.1%, 22.16% and 22.59% respectively. For floating point benchmarks, the MAPE excluding outliers is 24.8%, 35.4%, 32.0% and 27.5% for Sniper, gem5, MARSSx86 and ZSim respectively.

To understand some sources of inaccuracies of the simulators, we looked at cache misses and branch mispredicition behaviors. As can be seen in Figure1, many cases exist where the average error for cache misses and branch misprediction goes above 100% for gem5. This helps to understand very high underestimation of IPC values by simulators for some benchmarks. For example, most of the outliers in gem5 ($h264ref$, *gcc_200*, *gobmk*, *perlbench*, *namd*, *povray*) have much higher branch mispredictions and cache misses than real hardware runs. The benchmarks that have high overestimated branch mispredictions are those that have a high percentage of branch instructions in their dynamic instruction mix (20% or more). Furthermore, some of the IPC inaccuracies can be due to the way some of the x86 instructions are decoded and implemented in gem5.

Other simulators show similar results of overestimated branch mispredictions and cache misses associated with benchmarks with high IPC inaccuracies, for example: $h264ref$, *libquantum*, *milc* and *povray* on MARSSx86 and *gamess*, *libquantum* and $mcf$ on ZSim. Note that MARSSx86 is a full-system only simulator, while the other simulators are run using application-mode. The results of MARSSx86 in-

* Corresponding author email: lina.sawalha@wmich.edu

© 2016 ACM. ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

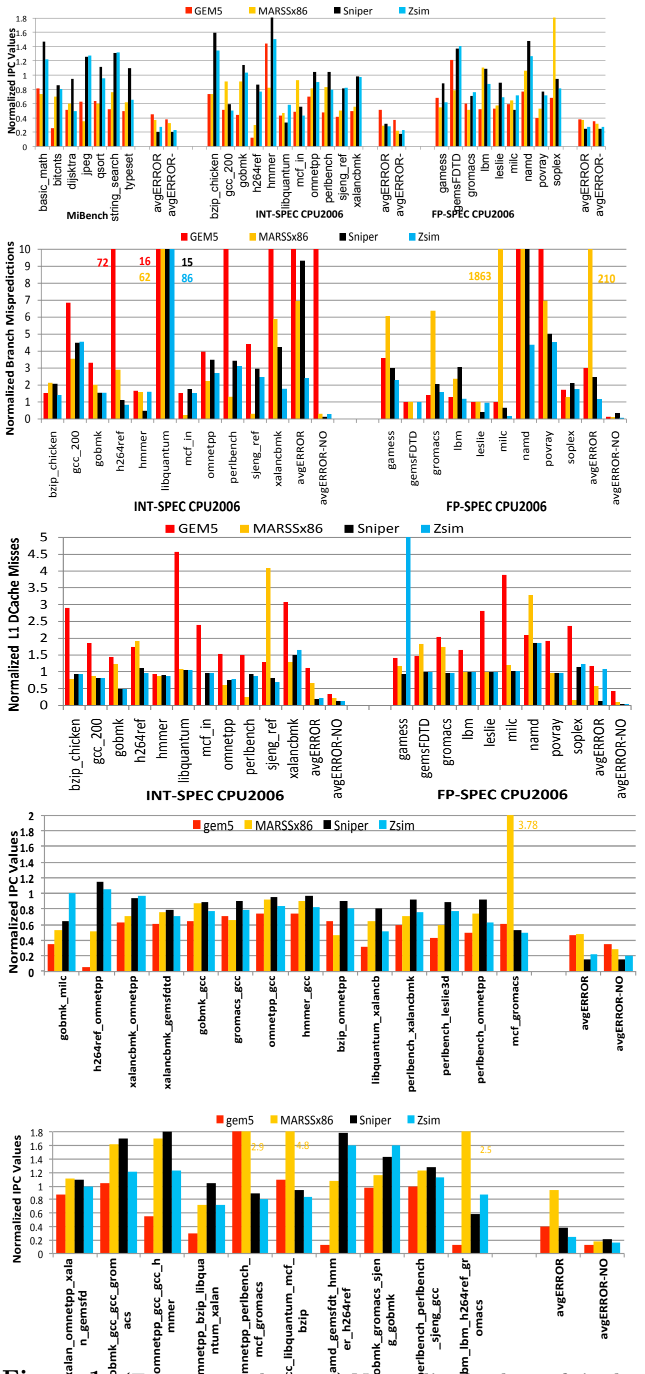

**Figure 1:** (From top to bottom) Normalized values of single-core IPC, branch mispredictions, L1-dcache, relative IPC change, dual-core and quad-core IPC.

clude both kernel and application instructions. Another source of inaccuracy for all simulators can be a result of the lack of support of fused $\mu$-ops, and $\mu$-op cache of Haswell (significantly reduces the effective pipeline depth in case of $\mu$-op cache hit). Sniper and ZSim show close accuracy for multicore runs (less experimental error than gem5).

In addition to comparing the absolute performance of simulators, we studied the effect of changing pipeline widths (half of their values from Table 1) for MiBench benchmarks. The simulators show different sensitivity to this change, see Figure 2. We also compared these simulators on the basis

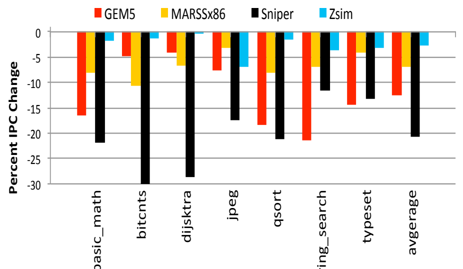

**Figure 2:** Percent change in IPC with half pipeline stages' widths.

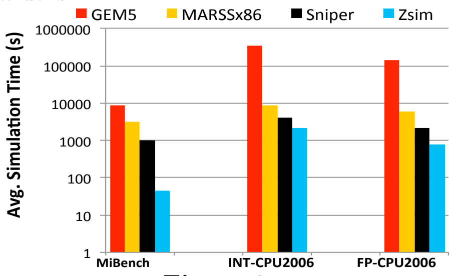

**Figure 3:** Simulation Time.

of the time simulators took to perform these simulations. Figure 3 shows average simulation time in addition to fast forwarding time for the benchmarks running on the simulators. The results show that ZSim is the fastest simulator.

In summary, this study emphasizes on the importance of validating simulators, and aims to help the community to point out sources of inaccuracies in simulators that can be modified later in future work. Our experiments indicate a correlation between the accuracy of simulators and the existence of thorough validation and calibration of simulators for a particular target architecture. Errors due to abstraction and lack of details in simulators do not necessarily imply inaccuracy, as validated simulators can still achieve acceptable relative performance. In future, we plan to dig deep into more sources of inaccuracies, and potentially fix them, and thoroughly study relative performance accuracy.

## 1. REFERENCES

[1] A. Butko et al., "Accuracy Evaluation of GEM5 Simulator System," in *ReCoSoC*, pp. 1–7, 2012.

[2] T. E. Carlson et al. ,"An evaluation of high-level mechanistic core models," *ACM TACO*, vol. 11, no. 3, p. 28, 2014.

[3] A. Patel et al.,'MARSS-x86: A Qemu-Based Micro-Architectural and Systems Simulator for x86 Multicore Processors," in *DATE*, pp. 29–30, 2011.

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

[5] A. Gutierrez et al., "Sources of Error in Full-System Simulation," in *ISPASS*, pp. 13–22, 2014.

[6] http://zsim.csail.mit.edu/tutorial/slides/validation.pdf.[Online; accessed 9-Oct.-2016].

[7] A. Akram and L. Sawalha, "x86 computer architecture simulators: A comparative study," in *ICCD*, pp. 638–645, 2016.

[8] http://www.intel.com/content/www/us/en/processors/architectures-software-developer-manuals.html. [Online; accessed 9-Oct.-2016].

[9] http://www.agner.org/optimize/instruction_tables.pdf.[Online; accessed 9-Oct.-2016].

[10] http://www.realworldtech.com/haswell-cpu/. [Online; accessed 9-Oct.-2016].

[11] http://www.anandtech.com/show/6355/intels-haswell-architecture/6. [accessed 9-Oct.-2016].