Advanced Architectures: Open-source release of the software simulator for the emerging memory technologies

D4.5 v1.0

WP4 – Advanced Architectures: D4.5 Open-source release of the software simulator for the emerging memory technologies

Dissemination Level: Confidential

Lead Editor: Adrian Cristal

Date: 13/10/2014

Status: Final

Description of Work:

Open-source release of the software simulator for the emerging memory technologies

Contributors:

Adrian Cristal (AC)
Adria Armejach (AA)
Nehir Sonmez (NS)

Internal Reviewer(s):

UL: Janez Demsar
2ndQ: Martin Marques

Version History

Version	Date	Authors	Sections Affected
0.1	13/10/2014	AC, AA, NS	Initial version
0.2	15/10/2014	NS	Internal revision
1.0	31/10/2014		Final version

Table of Contents

List of Figures

List of Tables

Abbreviations

BSC	Barcelona Supercomputing Center
DBMS	Database Management System
HPC	High-performance Computing
TPC	Transaction Processing Council
NVM	Non-Volatile Memory
DRAM	Dynamic Random Access Memory
IPC	Instructions Per Cycle
MPKI	Misses Per Kilo-Instruction
PCM	Phase-Change Memory

1. Summary

This deliverable describes the work done during the second year regarding the simulation platform for emerging memory technologies, leading to its public open source release. The first year description of the work done on the simulation platform was included in Deliverable 4.4 - submitted in month 12.

The drawbacks detailed in the first year description have been addressed by changing the simulation infrastructure substantially. The current infrastructure can now simulate workloads that spawn additional processes, such as multi-query PostgreSQL; a wide variety of workloads, including native multi-process simulations; and mimics a modern architecture based on the AXLE server machine hosted at the Barcelona Supercomputing Center (BSC).

2. Analysis/ Description of work done

In this section we describe the work done during the second year in order to improve the simulation infrastructure.

2.1. General overview of the simulation platform

Our simulator platform has been built with the objective to simulate complex Database

Management System (DBMS) workloads. These workloads can be substantially more complex than typical High-performance Computing (HPC) workloads for a number of reasons, i.e., different server-client configurations, large dataset sizes, and execution times that are several orders of magnitude longer.

The simulation platform presented in Deliverable D4.4 (submitted in month 12) was based on SniperSim [SniperSC11] and met most of the requirements. However, as was noted in the deliverable, SniperSim is not able to handle workloads that spawn new processes, such as our DBMS of choice - PostgreSQL. For this reason simulating multiple queries that execute in parallel was not possible, being limited to single-query runs. In order to address this limitation and have the capability to execute multiple queries in parallel, the simulator platform has been modified substantially.

The simulator is now based on a different simulator infrastructure, also based on Pin [Pin], which fulfills all the requirements including the simulation of multiple queries in parallel. The base simulator now employed - Zsim [ZsimISCA13] - was also mentioned in deliverable D4.4 as the main candidate to overcome the issues that were found in the initial release. Zsim source code became available in March 2014 and after initial testing we decided to port our infrastructure to it, which proved to be a straightforward process due to the fact that both simulators leverage Pin's dynamic binary translation mechanism.

The current simulator platform can execute PostgreSQL with multiple concurrent clients executing different decision support queries from the TPC-H [TPCH] workload. Moreover, it allows for significantly higher simulation speeds, being able to simulate very large samples of 100 billion instructions - several tens of seconds of real execution time on a modern server machine - in a reasonable amount of time; comprised within a few hours of simulation.

2.2. Integration of emerging memory technologies in the simulation platform

Similar to SniperSim, Zsim is a complete architectural simulator with accurate out-of-order core models as well as models for typical memory hierarchies, able to simulate common systems with 3 levels of caches and a multi-channel DRAM subsystem. However, it also lacks models for 3D integration and Non-Volatile Memory (NVM) technologies.

To address this, we integrated a modern cycle-accurate main memory simulator - NVMain [NVMain] - that adds all the required capabilities required to fulfill the project objectives, such as 3D stacking and models for different NVM technologies that can be used in different ways, e.g., as a replacement for main memory. In addition, NVMain also includes power models for all the technologies it implements, which is a feature that will be of great interest within the project and that we anticipate that will also be of importance to potential users.

The integration has been done by adding additional functionality to Zsim, which allows the use of NVMain as the memory simulator. This functionality includes additional configuration options to specify whether or not NVMain is used and what kind of system will NVMain

simulate. Additionally, new classes that interface the Zsim memory system with NVMain have been added. This allows us to leave the NVMain source code untouched, easing usability.

While testing and validating the simulation infrastructure, a number of bugs were found in the NVMain source tree, and these have been fixed and contributed back to the project. Moreover, additional changes in the NVMain source tree have been submitted and accepted so that compilation with Zsim happens seamlessly by just setting an environment variable. Further input on how to use and configure the simulator is provided in the upcoming sections.

2.3. Simulator validation using DBMS workloads

In an effort to ensure that the results we extract from the simulator are representative of real executions, we compared a number of metrics from simulated runs and real runs of PostgreSQL executing TPC-H queries. Zsim was already validated using common CPU-intensive workloads [ZsimISCA13] such as SPEC CPU2006 and PARSEC, with positive results. However, given the addition of NVMain into the simulation infrastructure and the fact that DBMS workloads are known to behave substantially different, further validation was deemed necessary.

To perform the validation we configured the simulator to mimic a modern Xeon machine that resembles the AXLE server hosted at the Barcelona Supercomputing Center (BSC). For these runs the simulator uses a conventional multi-channel DRAM architecture, modeled by the integrated NVMain simulator, so that an accurate comparison with the real hardware is possible. Table 1 shows the architectural description of the server machine and the simulated system employed to do the validation.

Component	**Description**
Processor	32nm Intel Xeon E5-2670, 2,60GHz, HT and TurboBoost disabled
Cores	8 out-of-order cores, 4-wide issue and retire
L1 cache	32KB 4-way, split I/D, 4-cycle access latency
L2 cache	256KB 8-way per core, 8-cycle access latency
L3 cache (LLC)	20MB 16-way shared, 28-cycle access latency
Memory	256GB DDR3-1600, 4 channels, delivering up to 51.5GB/s

Table 1: Real and simulated architectural parameters.

We developed a powerful profiling simulation infrastructure using hardware performance counters based on *perf*, in order to extract a number of meaningful metrics from real runs in the server. These include instruction per cycle (IPC) values, misses per kilo instruction (MPKI) at different cache levels, and many others. We use this information to correlate it with the

simulated runs. This profiler has also been used as a part of Task 4.5 which will be detailed in Deliverable D4.6 in month 36.

For validation we mainly used the IPC metric taken at short instruction intervals, giving a sense of how the system behaves throughout the execution of the different TPC-H queries in both the real machine and in the simulator. We are interested in identifying changes in IPC and corroborate that the simulator is in line with real runs. Figures 1 and 2 present two examples of TPC-H queries using a dataset of scale 100, which yields a database of almost 200GB of data. The figures show IPC values while executing a sample of 100 billion instructions - tens of seconds of real execution time. As it can be seen, the simulator is in line with the results obtained with real runs. The error observed between real runs and simulated runs is within an acceptable range, and more importantly, the simulator behaves correctly when sudden changes (peaks/drops) in IPC happen, following the behaviour of the real runs.

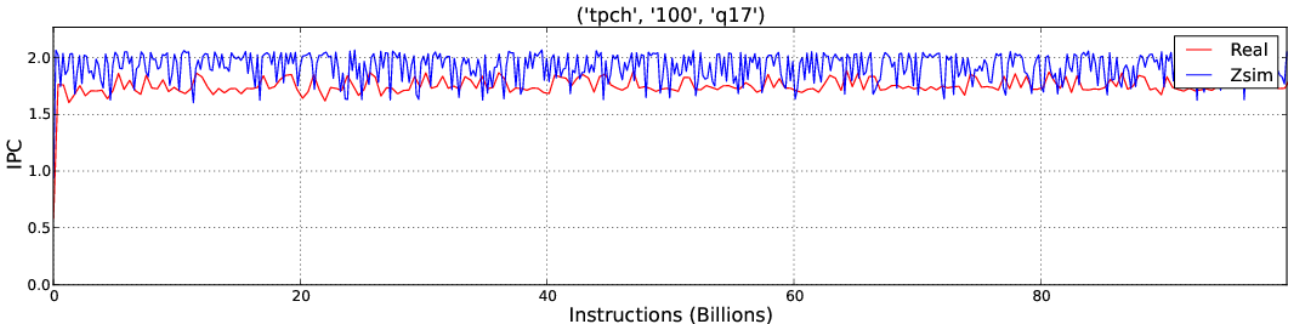

Figure 1: Query 17 of TPC-H with a scale of 100. IPC throughout a 100 billion instruction sample in real and simulated runs.

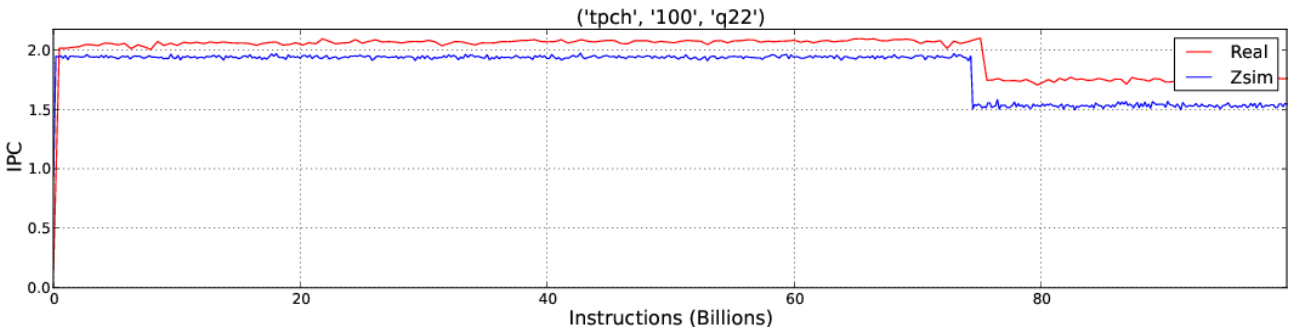

Figure 2: Query 22 of TPC-H with a scale of 100. IPC throughout a 100 billion instruction sample in real and simulated runs.

With these charts and additional data regarding MPKI values from different cache levels we therefore consider our simulator infrastructure to be robust and representative when compared to real execution runs.

2.4. Simulator preconfigured files

The simulator comes with a number of example configuration files that were part of the original Zsim distribution. Additionally, we provide 3 configuration files, including the one we used to do our validation study. These files all model the base architectural parameters shown in Table 1, which match the Sandy Bridge specifications of the machine hosted at BSC. The difference between the configuration files lays in the memory subsystem they simulate. The three different configurations model the following memory architectures:

1. Multi-channel DRAM: This is the configuration file we used for the validation study. It models a modern 4-channel DRAM memory subsystem. The configuration file can be found in:

   'tests/AXLE-sandy-dram.cfg'

2. State-of-the-art 3D stacked DRAM cache: This configuration will set up a system featuring a state-of-the-art 3D stacked DRAM cache [3DCache], and utilizes a multi-channel DRAM system, similar to the one discussed above, as main memory. The configuration file can be found in:

   'tests/AXLE-sandy-3dcache.cfg'

3. Non-Volatile memory as main memory: This configuration replaces the typical DRAM modules with Phase-Change Memory (PCM) modules. Yielding a system that has storage-class main memory. Such systems may allow for substantial benefits by making applications exploit the non-volatility characteristics of memory, simplifying the applications design and lowering overheads related to ensure data persistency. The configuration file can be found in:

   'tests/AXLE-sandy-nvm.cfg'

The simulation infrastructure is not limited to these configurations, different configurations with mixed technologies can be used and are easy to configure. NVMain allows setting a different memory technology for each channel, so hybrid solutions are also made possible.

2.5. How to use the simulator

The simulator sources include the original readme file 'README.md' which has been augmented with additional information regarding the integration with NVMain, and an explanation of the provided configuration files and how to use them. Furthermore, an installation script 'install.sh' is provided. This script installs all the necessary tools and libraries the simulator requires and provides instructions on how get the NVMain sources and set up the appropriate environment variables.

The simulator sources are publicly available at the AXLE project github site:

https://github.com/AXLEproject/axle-zsim-nvmain

In addition, a blog entry has been created in the project's website (www.axleproject.eu) announcing the simulator release, including a link to the github site and a direct download link to the simulator sources: http://axleproject.eu/latest-news/

2.6. Additional simulator capabilities

The initial simulator platform released in month 12 was able to execute DBMS workloads with small datasets of scale 1, which provides databases of a few gigabytes. Due to the faster simulation speeds that the new infrastructure can reach, we are now able to simulate very large samples of queries executing on larger, more realistic and representative datasets, that exceed the 100GB mark (scale: 100). In addition, the current simulator has been validated against a more modern architecture based on Sandy Bridge, which features a larger number of cores.

One of the main distinctions with respect to the initial release made in month 12, is that we can now simulate workloads that spawn new processes, which was not possible before; and multi-process workloads natively without having to resort to trace-based simulations. In this regard, we have successfully simulated setups with up to 8 simultaneous TPC-H queries running, one for each processing core; and multi-programmed workloads where different instances of the SPEC CPU2006 benchmarks are used concurrently. In addition, a number of useful statistics have been added to the simulation infrastructure, such as the memory footprint used during simulation, and a histogram of how many times addresses are accessed at the main memory level to determine if there is little or high data reuse at that level.

3. Conclusions / Future work

The current simulator infrastructure fulfills all requirements to perform the tasks relevant to the AXLE project. In addition to relevant DBMS workloads, we also tested the SPEC CPU2006 benchmark suite, which is one of the most commonly used set of benchmarks in computer architecture research, making the simulator appealing for a wider audience.

The simulator is robust and provides representative results when compared to real hardware executions as our validation study shows. We perform this validation using large simulation samples, than are equivalent to several tens of seconds of real time execution; and with dataset sizes, of over 100GB, that are typically not employed in simulation environments.

Future work on the simulation infrastructure will include additional support for statistics that are deemed necessary, as well as improvements, novel architectural proposals, and bug fixes that come up while developing other tasks within the project.

4. References

[SniperSC11] T. E. Carlson, W. Heirman, and L. Eeckhout, “Sniper: Exploring the level of abstraction for scalable and accurate parallel multi-core simulations,” in International Conference for High Performance Computing, Networking, Storage and Analysis (SC), Nov. 2011.

[ZsimISCA13] Daniel Sanchez and Christos Kozyrakis. 2013. ZSim: fast and accurate microarchitectural simulation of thousand-core systems. In Proceedings of the 40th Annual International Symposium on Computer Architecture (ISCA '13). ACM, New York, NY, USA, 475-486.

[Pin] Pin - A Dynamic Binary Instrumentation Tool, Intel Corp. http://software.intel.com/en-us/articles/pin-a-dynamic-binary-instrumentation-tool

[TPCH] http://www.tpc.org/tpch/

[NVMain] NVMain: An Architectural-Level Main Memory Simulator for Emerging Non-Volatile Memories, Matt Poremba, Yuan Xie, IEEE Symposium on VLSI (ISVLSI), 2012.

[3DCache] Moinuddin K. Qureshi and Gabe H. Loh. 2012. Fundamental Latency Trade-off in Architecting DRAM Caches. In Proceedings of the 2012 45th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO-45). IEEE Computer Society, Washington, DC, USA, 235-246.