An Evaluation Methodology for Exascale Memory Systems

Ronald G. Dreslinski, Bharan Giridhar, Michael Cieslak, Yiping Kang,
David Blaauw, Trevor Mudge, Jeffrey Vetter*, Robert Schreiber^

University of Michigan *Oak Ridge National Labs ^Hewlett-Packard

Abstract
Evaluating potential Exascale memory systems is difficult because of the scale of the target system. While full-workload simulation techniques work at the core and cache level granularity, the shielding effects of the cache require extremely large simulation times to simulate a significant number of memory accesses to assess performance. The current approach to simulate such systems is the use of memory trace files, but these are difficult to generate and are inflexible in their ability to capture important data dependencies, which may be important in assessing revolutionary memory subsystems.

Our methodology takes a two-pronged approach to estimate power, performance, and resiliency of large scale memory systems. We first improve the trace-based approach to measure power and resiliency, and then take a hardware approximation approach to evaluate performance for proxy-applications. Our design is useful for exploring traditional DRAM, 3D-Stacked DRAM, non-volatile memories, or any combination.

1. Proposed Methodology

Our proposed methodology is composed of two techniques. First, we use proxy-application trace-based simulation to evaluate memory system power and resiliency. Second, we use real hardware measurements to approximate performance.

1.1 Power Modeling: To evaluate the total power of an Exascale system we use proxy-applications on real servers to generate trace files. We first instrument the proxy-applications (written using MPI) to execute within the PIN binary instrumentation package [1]. PIN is then used to output every load and store to a trace file. This allows us to evaluate systems with different cache configurations. If the trace file is too large, PIN itself can be augmented to simulate the non-shared levels of cache to reduce the trace size. However, this requires that the applications be run for each possible cache configuration. Once we have the core trace files, we run them through a configurable cache simulator (including prefetch options) to generate the main memory references and store those in another trace file.

We have also created an XML interpreter that allows for flexible definition of the memory system. The XML language defines how memory addresses are distributed in the memory system. For example, it defines how ports, ranks, banks, sub-arrays of a DRAM are organized. The XML also defines the number, size, and how row-buffers are organized. The memory simulator takes the XML configuration and runs the memory trace through that configuration. It tracks the number of CAS and RAS operations that are performed. Once the simulation has finished, the row-buffer hit rates (Ratio of CAS to RAS) is used to scale to a 100PB exascale system with a peak bandwidth of 100PB/s. The simulation organization is diagramed in Figure 1.

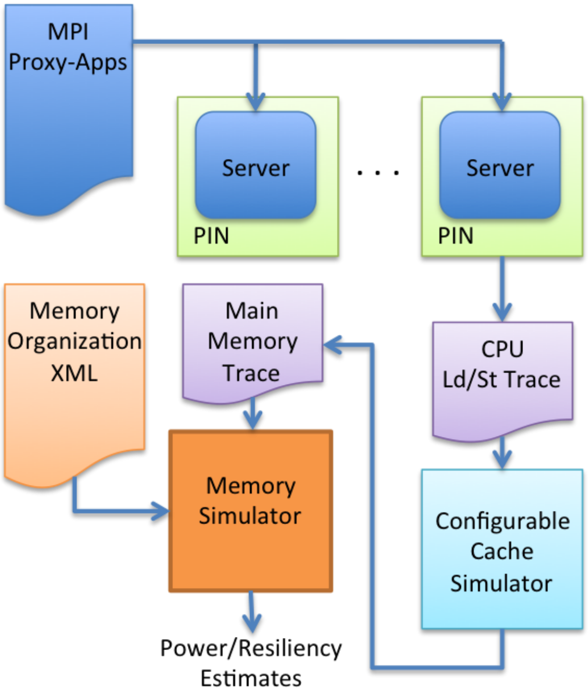

Figure 1: Simulation Infrastructure

The memory organization XML file can also be configured to model multi-level memory systems where a DRAM read/write buffer is inserted in front of non-volatile memory structures.

1.2 Resiliency Impact: To measure the impact of resiliency the XML file is augmented with the ECC information of each type of memory, the expected error-rates and breakdowns (transient, intermittent, and permanent), and checkpointing information. With this information, the simulator can output the power impact of checkpointing/recovery and can indicate how often an error occurs.

1.3 Performance Estimation: While the trace based infrastructure is good for modeling the power/resilience of the memory system, it is unable to model performance accurately. This happens because the PIN instrumentation tool inserts overhead into the simulation and because the specific trace captured for any given execution is fixed in time. For the power measurements, we only use the row-buffer and multi-level hit rates to estimate power and scale it to the size and peak performance of an Exascale system.

Fixed time traces are difficult to scale performance because the traces are inflexible. If a new memory design is simulated it may have a different access latency. In the worst case every access is delayed by the increase in DRAM latency. This means that if we increase the DRAM latency by *N* cycles, and there are *A* dram accesses, the total runtime is increased by *A*N* cycles. In reality some of these accesses do not have dependencies and can be run in parallel. These parallel accesses come from cache evictions, out-of-order cores issuing multiple requests, prefetchers, and multiple cores/threads.

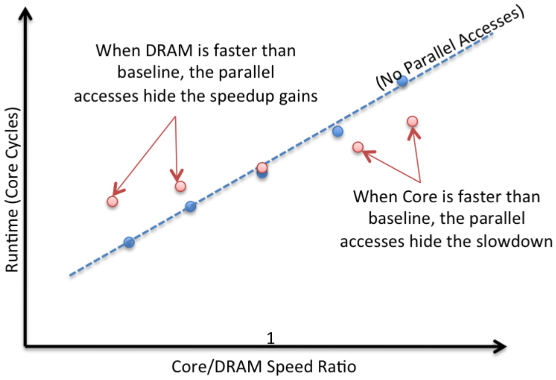

Figure 2: Impact of Parallel Accesses on Performance

To quantify the performance impact of the modified system, we propose using the proxy-applications to determine the amount of parallel accesses that occur. To do this we run the proxy-applications on the system without the PIN instrumentation and measure total runtime (in cycles). To quantify the impact of faster/slower memory systems we modify the servers. We can generate a wide range of core to memory speed ratios by using the DVFS feature of the cores and by modifying the bus speed to DRAM. After measuring the system at a variety of Core/DRAM ratios we are able to determine the amount of memory-level-parallelism in each benchmark and can approximate the overall performance impact by using the memory simulator to determine the average memory latency. Figure 2 summarizes the methodology. The dotted line represents a memory system where every memory access must wait for the previous one to finish. It represents the the worst case slowdown (longer runtime) when the memory latency is increased (to the right of 1 in the figure). The measured data (red dots) will give us an indication how the actual benchmarks performance.

2. Questions from CFPP

2.1 Challenges addressed: This approach addresses power/performance/resiliency tradeoffs in memory design that allow us to evaluate non-volatile memory alternatives to DRAM in Exascale systems.

2.2 Maturity: We have used the infrastructure to evaluate the impact of 3D-DRAM technologies, such as the hybrid-memory cube from Micron, successfully. We are in the process of using the same methodology to evaluate memristors as an Exascale solution.

2.3 Uniqueness: This approach is not unique to Exascale and could be used to evaluate any form of memory organization. The design was needed because existing approaches do not scale well to the PB memory system range.

2.4 Novelty: The approach for power simulation is quite similar to other trace-based approaches, but differs in scale. The performance evaluation approach is novel compared to elastic-style traces, which are too difficult to simulate at the Exascale level.

2.5 Applicability: This infrastructure is applicable to a wide range of memory system designs, which is becoming important with the growing research into the non-volatile memory space.

2.6 Effort: Much of the effort is complete on the power simulation infrastructure, as we have used it for initial 3D DRAM evaluation. Some additional work needs to be done to support multi-level non-volatile memory solutions. The performance evaluation is underway and is not difficult to design. The major complexity is the resiliency modeling where we need breakdowns of how often transient, intermittent, and permanent failure impact bitcells. To date these models do not exist and would require the most effort to generate.

References:

[1] C.-K. Luk, R. Cohn, R. Muth, H. Patil, A. Klauser, G. Lowney, S. Wallace, V.J. Reddi, and K. Hazelwood, “Pin: building customized program analysis tools with dynamic instrumentation,” SIGPLAN Not., 40(6):190-200, 2005, 10.1145/1064978.1065034.