Evaluation of emerging memory technologies for HPC, data intensive applications

Amoghavarsha Suresh
San Diego Supercomputing Center
La Jolla, CA, USA
amoghavs@sdsc.edu

Pietro Cicotti
San Diego Supercomputing Center
La Jolla, CA, USA
pcicotti@sdsc.edu

Laura Carrington
San Diego Supercomputing Center
La Jolla, CA, USA
lcarring@sdsc.edu

***Abstract*— DRAM technology has several shortcomings in terms of performance, energy efficiency and scaling. Several emerging memory technologies have the potential to compensate for the limitations of DRAM when replacing or complementing DRAM in the memory sub-system. In this paper, we evaluate the impact of emerging technologies on HPC and data-intensive workloads modeling a 5-level hybrid memory hierarchy design. Our results show that 1) an additional level of faster DRAM technology (i.e. EDRAM or HMC) interposed between the last level cache and DRAM can improve performance and energy efficiency, 2) a non-volatile main memory (i.e. PCM, STTRAM, or FeRAM) with a small DRAM acting as a cache can reduce the cost and energy consumption at large capacities, and 3) a combination of the two approaches, which essentially replaces the traditional DRAM with a small EDRAM or HMC cache between the last level cache and the non-volatile memory, can grant capacity and improved performance and energy efficiency. We also explore a hybrid DRAM-NVM design with a partitioned address space and find that this approach is marginally beneficial compared to the simpler 5-level design. Finally, we generalize our analysis and show the impact of emerging technologies for a range of latency and energy parameters.**

Keywords—component; formatting; style; styling; insert (*key words****)***

I. INTRODUCTION

It is anticipated that DRAM technology will be inadequate for future large scale systems due to a number of factors. First, the evolution of DDR DRAM has not been able to keep up with the bandwidth demand of multicore processors (i.e. *the memory wall* [1]) and this gap in the performance of CPU and memory sub-systems will only increase further for future many-core systems [2]. Second, there is a growing capacity gap due to the limited scaling of DRAM, power, and cost limitations [3, 4]. Finally, at the current DDR3 power-performance level of approximately 600 MW/GB/s, in an Exascale system, the memory alone would draw 600MW [2]. Consequently, advancements in memory technology are sought in all the significant metrics of performance, power/energy efficiency, density, and scaling [2].

A number of memory technologies are emerging as viable alternatives that address one or more of these shortcomings of DRAM. *Volatile* memories are one such technology, which have retention times comparable to DRAM (order of nanosecond) and also need frequent refresh, similarly to DRAM. They are based on DRAM cell technology but improve on the architecture and the interface to deliver higher performance and energy efficiency than DRAM [5, 6]. Also emerging as a technology solution are *non-volatile* memories (NVM), which have much longer retention times than DRAM (order of years), are based on new cell technologies and offer improvements mostly towards capacity [7].

A significant body of research has explored the use of NVM technologies as storage and as an alternative to DRAM, with many research efforts focusing on the feasibility and design of the NVM devices, the architecture of hybrid memory systems, data and access partitioning between DRAM and NVM, and high level interfaces to NVM (see Section II.B).

In this paper we evaluate the impact of emerging technologies to identify opportunities for complementing DRAM to improve the overall performance and energy-efficiency of the memory system. Our work focuses on understanding the impact of these technologies on High Performance Computing (HPC) and data intensive applications, and uses this understanding to define the performance and energy efficiency levels that memory technologies have to satisfy to be viable solutions. In particular, we consider using volatile technologies as a last level cache (LLC) and non-volatile technologies as main memory; in the former case, the focus is on improving performance and energy efficiency, in the latter case the focus is on increasing the capacity while still preserving performance and energy efficiency.

One concern with NVM is its asymmetry in read-write performance and energy consumption (writes are typically slow and energy demanding operations). To address this we also investigate a hybrid main memory system with DRAM and NVM. The hybrid design supports a partitioned address space, in which frequently accessed and updated objects are stored in DRAM, while the rest are stored in NVM.

We evaluate these technologies on a set of HPC and data-intensive workloads. Our evaluations are based on

performance and energy models that leverage online simulations of the memory hierarchy during the execution of the workload.

We make the following contributions:

- We evaluate 5-level memory hierarchies that employ eDRAM and HMC as LLC,
- we evaluate 5-level memory hierarchies that employ PCM, STT-RAM, and FeRAM as main memory,
- we evaluate 5-level memory hierarchies that employ eDRAM and HMC as LLC and PCM, STT-RAM, and FeRAM as main memory,
- we evaluate a hybrid DRAM-NVM design and propose a methodology for partitioning the address space, and
- we generalize our results and present a performance and energy *heat map* to assess what is the potential impact of emerging memory technologies on performance and energy efficiency.

The rest of the paper is organized as follows: Section II presents the characteristics of some emerging memory technologies and covers related work. Section III discusses the methodology and the models used for this study. Section V presents and discusses the results. Section VI concludes the paper with our conclusions and future work.

II. Background/Related work of Hybrid Memory

Recently emerging memory technologies have garnered a lot of attention, following the wide adoption of flash NAND memory, as potential alternatives to DRAM. This section describes the characteristics of some of the emerging technologies that we evaluate in this paper, and provides an overview of related work.

A. Characteristics of future memory technologies

DDR DRAM technology is challenged by the difficulties in performance, scaling, and energy efficiency, while SRAM scaling is constrained by increasing leakage power. Recently, various emerging memory technologies have begun to address some of these challenges.

Volatile memory technologies, such as Embedded DRAM (eDRAM) [5] and Hybrid Memory Cube (HMC) [6] address the performance and energy inefficiency of DDR DRAM technology. eDRAM is DRAM embedded on the processor chip. Hybrid Memory Cube is a three-dimensional architecture that uses through-silicon vias to effectively reduce the distance traveled by signals, increasing the density of the memory and significantly increasing the performance achieved.

A wealth of non-volatile memory technologies have emerged as storage class memories to replace hard disk drives. Due to the fact that their performance is approaching that of DRAM and their ability for larger capacity makes several of these technologies (e.g. NAND Flash, STTRAM, PCM, FeRAM, ReRAM [8]) potential alternatives to DRAM in future large scale systems.

NAND Flash memory is the most mature and widely adopted type of non-volatile memory [9]. However, flash memory is organized for bank read and writes and its low performance and high energy consumption, especially for write operations, make it a storage–class solution but not viable for main memory.

Phase-Change memory (PCM) technology uses chalcogenide alloy layer that change phase between crystalline (low-resistance) and amorphous (high resistance) [10-15]. The phase change is achieved by heat induced via high current (slowly or abruptly cut-off for a transition to crystalline or amorphous, respectively). PCM has very high capacity but is characterized by asymmetric performance with large latency and high energy consumption for write operations. PCM also suffers from low endurance, and though this may be compensated by wear leveling, it does incur some overhead that adds variability in performance.

Spin-Torque-Transfer (STT) memory is a type of magnetic RAM that uses a "spin-valve" to achieve two distinct resistance states in a magnetic tunnel junction [16, 17]. The change of state is achieved by the spin transfer torque effect of polarized current. Despite having asymmetric performance, with write latencies quite larger than DRAM, STTRAM has good performance, lower energy requirements than most non-volatile memories, and high endurance.

Ferro-Electric memory (FeRAM) functions by storing a charge in a ferroelectric layer [18-20]. FeRAM is similar to DRAM in terms of performance and combines high endurance with non-volatility. FeRAM may require higher energy per operation than DRAM.

Some of the emerging non-volatile memories have limited endurance (memory cells wear out with updates), exhibit asymmetric read and write performance, and may require high currents, especially for write operations. Nevertheless, these are opportunities to address shortcomings of DRAM improving on performance, capacity, energy efficiency, with an additional benefit of non-volatility. We do not consider other less mature technologies, although our study is general enough to be applicable and provide insight to the impact of other technologies with different performance and energy characteristics.

B. Hybrid memory design

Memory systems that integrate DRAM with other technologies have the potential to fill the gaps in DRAM's performance and energy efficiency.

Previous work on non-volatile memories focused on storage-class devices connected via SATA or PCIe interfaces, and flash memory has been used as fast scratch storage in HPC production systems [21, 22]. Flash has also been considered as a fast swap device to effectively extend the size of physical memory [23].

With the emergence of faster non-volatile memory, the focus shifted to future systems and to the role of NVM as an integral part of the main memory, or as fast checkpoint memory [24]. Others have focused on the architecture and implementation of the device and the performance on basic

benchmarks [10-14, 16, 25]. STT-RAM has also been investigated as potential last level cache [26].

To address the asymmetric performance and energy characteristics of NVM, pervious work explored the potential of carefully placing data on DRAM or NVM in hybrid memory system [27], comparing different policies for migrating data between the different modules [28], and finding data objects compatibility with the different modules by comparing different policies for migrating data between the different modules [29]. In contrast, we start by evaluating the potential of hybrid designs and ad-hoc placement policies assuming the existence of an oracle capable of statically partitioning the virtual address space, and at the same time we compare hybrid systems to a simpler hierarchical design that does not need partitioning policies.

With respect to previous work, our work goes beyond the evaluation of a single technology and attempts to evaluate, compare, and combine different technologies using the same methodology. More importantly, we attempt to generalize our results to provide insight on the impact of performance and energy costs as these technologies mature and others emerge.

Finally, we consider several data-intensive workloads to stress the system and the capacity aspects, using much larger memory footprints than in previous studies [29].

III. METHODOLOGY

In this section we describe the memory hierarchy designs that we used in the evaluation, and provide details on the motivation for these designs. We explain the simulation framework we developed to collect data movement statistics for the different designs, and the performance models that we employ to gauge their impact.

A. Design Space

In this work we explored four main memory designs with multiple configurations. We use an Intel Xeon processor (Sandy Bridge architecture) with 64B cache lines, a 32KB L1 (8-way associative), a 256KB L2 (8-way associative), and a 20MB L3 (20-way associative) as the reference system and memory configuration. We assumed DRAM to be large enough to contain the memory footprint of each individual benchmark.

For all the designs, the same L1, L2, and L3 cache configurations are used. The four designs used for this work are 4-Level Cache, NVM as Main Memory, NVM+DRAM, and 4- Level Cache as NVM. The details of these designs are as follows:

***4-Level Cache (4LC)*:** this design uses eDRAM and Hybrid Memory Cube (HMC) as Last Level Cache (LLC), with HMC being off-chip. Missed references in the LLC are simply directed towards DRAM. Both eDRAM and HMC are expected to be integrated in the hardware design and to be managed by the on-chip logic and the memory controller. Their presence is entirely transparent to the system software and the application. The DRAM is assumed to be large enough to handle the memory requirements of the application. This design offers an opportunity to use better performing and more efficient LLC before DRAM, by employing a technology that is denser than on chip SRAM, and that is faster than DRAM.

***NVM-as-Main-Memory (NMM)*:** this design uses NVM as main memory and DRAM as a cache. This design aims to decrease DRAM size and hence reduce refresh energy. In addition, by employing DRAM as a cache, a significant portion of NVM memory accesses are filtered to limit the negative impact on performance and dynamic energy consumption of typical NVM technologies. Also in this case, since DRAM is an off-chip cache, it is transparent to the software. For this design we consider PCM, STTRAM, and FeRAM as the NVM technology options.

***NVM+DRAM (NDM)*:** To take into account the characteristics of NVM technologies, such as asymmetric performance and energy required for write operations, this design uses both NVM and DRAM as a partitioned main memory in which data objects are placed where they best fit. This design attempts to determine if an application can take advantage of the differences between NVM and DRAM, and leverage the capacity and low static power of NVM while minimizing its impact on performance and dynamic energy.

4LCNVM: The designs in 4LC and NMM each offer opportunities to decrease latency and increase capacity respectively. To combine those benefits we evaluate a system with no DRAM, but rather an eDRAM/HMC cache followed by an NVM main memory.

In each of the hybrid memory hierarchy designs that we presented above, with the exception of the NDM design, we use emerging memory technology integrated in the memory sub-system, and that as such it is employed transparently to the software stack, and in particular to the applications. For the NDM design, the underlying assumption is that a compiler-based and runtime approach can define the address range partitioning to enforce the desired partitioning. Our study does not propose any specific solution but, as an oracle, explores the potential benefit of the design for an optimal partitioning.

Table 1: Characteristics of different memory technologies

Memory Technology	Read delay (ns)	Write delay (ns)	Read energy (pJ/bit)	Write energy (pJ/bit)
RAM	10	10	10	10
PCM	21	100	12.4	210.3
STTRAM	35	35	58.5	67.7
FeRAM	40	65	12.4	210
eDRAM	4.4	4.4	3.11	3.09
HMC	0.18	0.18	0.48	10.48

To model the performance and energy consumption of the proposed designs, we rely on published characterization parameters for the relevant memory technologies. For caches, DRAM and eDRAM, we acquired parameters from CACTI [30], a memory modeling tool. The HMC characteristics were obtained from experimental data obtained on a prototype [6].

The characteristics of PCM and STTRAM are obtained from the 2013 ITRS report [8], whereas characteristics of FeRAM were obtained from published literature [18]. The characteristics of all memory technologies used in our study are summarized in Table 1.

B. Simulation

In order to model the performance and energy of a given application on each of our designs, we need the data movement statistics (e.g. hit/miss rates, loads/stores, etc.). In order to capture these statistics for a target design we developed a simulation framework based on PEBIL [31], a binary instrumentation tool that automatically instruments all the memory references of an application and captures its memory address stream. The address stream is then fed to a cache simulator for the target design with the output being the cache statistics of the target design (e.g. hits/misses & loads/stores to each level of memory). The raw address stream of an HPC application can be unmanageable. By processing the address stream during the execution, our framework avoids the need to store and process full memory trace offline and results in significant space and time cost reduction.

In order to model the effects of the asymmetric performance in non-volatile memories, we extended our simulation framework to differentiate between loads and stores. In the initial design, the simulator was only able to count cache references (hits and misses); our extensions added the ability to also track memory references due to dirty cache lines evictions. All the memory references are fed to the cache simulator for the target system that differentiates between loads and stores and keeps track of dirty cache lines. At the last level of cache, simple evictions are essentially ignored, whereas in the case of dirty cache lines such evictions cause a write back to the main memory. Assuming a write-back policy, dirty cache lines eventually make their way to the main memory and count as write operations; every other access to fetch a cache lines is counted as a read operation. Hence, with the current cache simulation framework, we can simulate memory hierarchies to obtain hits and misses and loads and stores at each existing level for our proposed designs.

C. Performance Modelling

We now describe our performance and energy models that combine technology specific characterization parameters and application specific data movement statistics to evaluate each of the proposed memory hierarchy designs. Performance is estimated by comparing the wall clock time measured on the reference system, to the estimated wall clock time of a given configuration. Equation (1) illustrates how the runtime of the target design is determined by scaling the runtime of the reference system (T_{ref}) by the ratio of the average memory access time (AMAT) of the proposed hierarchy ($AMAT_{design}$) to that of the reference ($AMAT_{ref}$).

1) $$T_{design} = T_{ref} * (\frac{AMAT_{design}}{AMAT_{ref}})$$

In order to calculate the AMAT of both the reference and design system the cache statistics from the data movement simulator built on top of PEBIL along with the access times in Table 1 were used in Equation (2). In order to calculate the AMAT for an application the number of loads and stores to each level of the hierarchy (Li) is captured in the simulation framework for a given design. This count (e.g. $Stores_{Li}$) multiplied by the access time for that level (e.g. $StoresAccessTime_{Li}$) is the time spent accessing data from that level. By summing this time for each level where N represents the number of level for both loads and stores we get the total access time. To calculate the average we just divide this time by the total number of references.

2) $$AMAT \frac{\sum_{Li=1}^{N}\{(LoadsAccessTime_{Li} * Loads_{Li}) + (StoresAccessTime_{Li} * Stores_{Li})\}}{Total\ Number\ of\ References}$$

In modeling the energy consumed for a particular design the sum of dynamic and static energy for all the levels in the memory hierarchy is calculated. Dynamic energy is the product of the energy for a load or store and the number of loads and stores for each level of the memory hierarchy for a level, as shown in equation (3)). Static energy is estimated as product of time (T) and static power of the memory hierarchy illustrated in equation (4)). The static power is calculated as sum of static power of each level of cache and the refresh power of DRAM/eDRAM in the memory hierarchy. The static power of the caches were obtained from CACTI and background (e.g. static) power of DRAM was obtained from [32]. We assume that the NVM memory technologies do not have any static power. The static/refresh power used is shown in Table 1.

3) $$DynamicEnergy = \sum_{Li=1}^{N}\{((LoadsAccessEnergy_{Li} * Loads_{Li}) + (StoresAccessEnergy_{Li} * Stores_{Li})\}$$

4) $$StaticEnergy = T * StaticPower$$

In order to compare different designs that achieve comparable improvements we use Energy delay product (EDP). EDP for an application is defined as product of energy consumed (e.g. Dynamic Energy + Static Energy) multiplied by time taken for the application (T_{ref}), and represents the overall gain by taking both performance and energy into account. For example, two configurations would be equivalent in terms of EDP if one is faster but uses a proportionally higher amount of energy.

IV. Experimental Setup

In order to evaluate our designs we first created a set of configurations within each design to explore. We also selected a set of applications and benchmarks to represent HPC and data intensive workloads.

A. Design Configurations

For this work there were four designs defined above as 4LC, NMM, NDM, and 4LCNVM. Within each design we

defined a configuration space to investigate. 4LC and 4LCNVM configuration space is described in Table 2. For these designs, we explored configurations with changes in the eDRAM capacity and the page size.

Table 2: eDRAM /HMC configurations (capacity per core)

Design name	eDRAM capacity (MB)	Page size(B)
EH1	16	64
EH2	16	128
EH3	16	256
EH4	16	512
EH5	16	1024
EH6	16	2048
EH7	8	2048
EH8	8	2048

For the NMM design we used a series of configurations with changes in the DRAM capacity and the page size. Table 3 details these DRAM configurations. For the NDM design we explored a DRAM of size 512MB.

Table 3: NMM configurations (capacity per core)

Design Name	**DRAM-capacity (MB)**	**Page-size (KB)**
N1	128	4
N2	256	4
N3	512	4
N4	512	2
N5	512	1
N6	512	0.512
N7	512	0.256
N8	512	0.128
N9	512	0.064

B. Workload

To explore the designs and their configurations we developed a test workload comprised of applications and benchmarks to represent an HPC and data intensive workload. The benchmarks were chosen from NPB [33] and CORAL [34] benchmarks. The NAS Parallel Benchmarks (NPB) is a collection of kernels and pseudo-applications that represent computation and data movement in computational fluid dynamics workloads. The NPB benchmarks used were CG—conjugate gradient solver with irregular memory access and communication and two pseudo applications BT -- Block Tri-diagonal solver and SP-- Scalar Penta diagonal solver. All of them are class –D workloads, which have memory footprint of 0.8-2GB per core. CORAL is a suite of benchmarks that represent DOE workloads, and comprises of benchmarks of scalable scientific, throughput, data centric workloads. From CORAL suite we have selected a) AMG2013, parallel algebraic multigrid solver for linear systems arising from problem on unstructured grids, which involves updating points of the grid according to a fixed pattern; b) Graph500-- a scalable breadth-first on undirected Kronecker graphs as a kernel to represent graph algorithm performance; c) Hash-- a data-centric benchmark which is used to evaluate the performance of the architecture integer operations, specifically for hashing, and for memory-intensive genomics applications. We have also used bioinformatics application Velvet [35]. For all iterative benchmarks, we have reduced the number of iterations to keep the simulation time within reasonable limits.

Realistic data intensive workloads are memory bound and often problems that do not scale and typically run with a large memory footprint. To ensure that the memory systems for the designs were properly exercised and mimicked data intensive workloads we used problem sizes resulting in a large memory footprint. In addition, these workloads are best suited to understand the benefits of NVM in future many-core systems, where the per-core memory capacity will be limited, and NVM can provide greater capacity than DRAM.

Table 4 details the runtime commands, memory footprint/core, and execution time on the reference system of each benchmark and application.

Table 4: Characteristics of the benchmark

Suite	Benchmarks	Footprint /Core (GB)	Time (s)	Inputs
NPB	BT	1.69	36.0	Class: D
NPB	LU	0.8		Class: C
CORAL	Graph500	4	157.0	"-s 22 -e 4"
CORAL	Hashing-2	4	389.6	"-m 30M -n 50K"
CORAL	AMG2013	3	156.3	"-r 72 72 72 -P 1 1 1 -pooldist 1"
CORAL	CG	1.5	54.8	Class: D
Application	Velvet	4	116.5	Default

V. RESULTS

We evaluated the different memory hierarchy configurations (shown in Table 2 and Table 3) on all of the benchmarks. The key parameters which were investigated were page-size and DRAM capacity and their impact on run time and total energy savings/overhead. Energy Delay Product has been used to compare the configurations experimented in a given design.

In Figure 1 and Figure 2 we show the runtime and energy consumed by applications in NMM design, respectively normalized to the base case that has 3 on chip SRAM caches followed by a DRAM big enough to support necessary memory footprint. For the initial configuration N1, we observe time overhead of 5% and total energy overhead of 12%. Increase in DRAM capacity results in increase in hit rate, which causes decrease in total access time (~2%) and dynamic energy (~10%) but increase in static energy (~5%) because of increase in DRAM capacity. The decrease in page-size causes less contiguous data to be present/more data to be fetched, as needed. The impact of decreasing the page size causes an increase in access time (~2.5%), and decreases the dynamic energy (~4%) —since less bits will be accessed, but increases the static energy (~1%) since static energy is also proportional to the access time. However, when the page size is decreased from 4KB to 2KB, the access time and hence the leakage energy decreases, which suggests that the data access pattern is more suited to 2KB pages (i.e., the amount of contiguous

data brought in by 4KB is not utilized efficiently). Among the configurations tested, N5 has least time overhead while N6 has most energy savings. Both N5 and N6 have DRAM size of 512MB, but page size is 1024B for N5 and 512B for N6; a t5he larger page size seem to favor performance but the smaller page size favors energy efficiency. However, if we consider EDP, N6 is more efficient than N5.

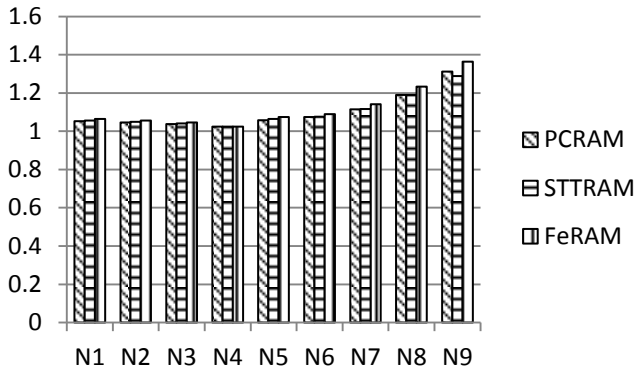

Figure 1: Average of normalized run time of all benchmarks for NMM

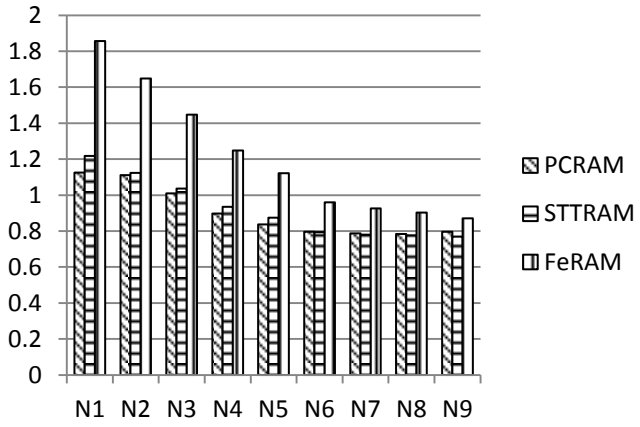

Figure 2: Average of normalized energy of different benchmarks for NMM

Similar experiments were made with capacity and page size of eDRAM/HMC for 4LC and 4LCNVM. The time and energy savings/overhead for 4LC is shown in Figure 3 and Figure 4 respectively. We observe that the run time decreases by approximately 2% and using a page-size comparable with the cache line size results in large energy savings (~17%). Increasing the page size results in an increase of dynamic and hence total energy consumption while the time taken fluctuates within a band of 2% from that of base configuration and hence leakage energy follows a similar pattern.

The time and energy savings/overhead for 4LCNVM is shown in Figure 5 and Figure 6, respectively. We observe that a page-size comparable with line size of last level cache results in significant energy savings (~57%). Increases page size result in increases in dynamic and hence total energy consumption. Increasing page size has a behavior similar to the same change in 4LC design. In both 4LC and 4LCNVM, the configuration with the least overhead in time and most saving in energy is EH1. EH1 has eDRAM/HMC size of 16MB and page size of 64B.

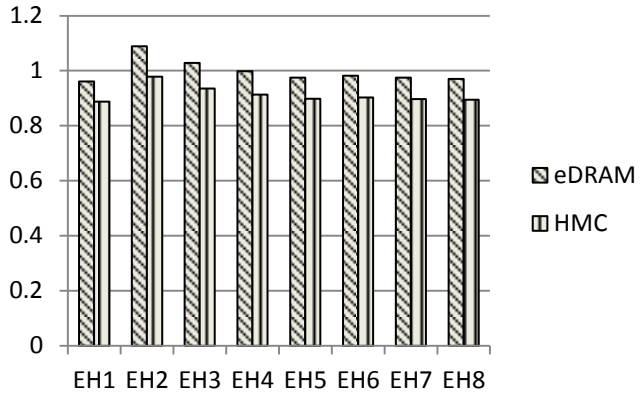

Figure 3: Average of normalized run time of different benchmarks for 4LC

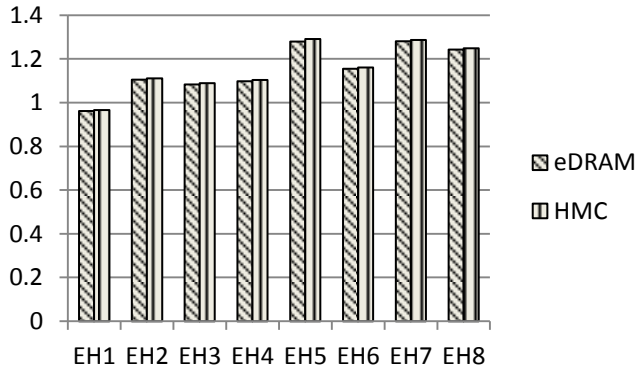

Figure 4: Average of normalized total energy of different benchmarks for 4LC

For the NDM design, the data placement is determined by identifying, in the application, a contiguous range of addresses that accounts for the bulk of the memory references. We have identified address ranges referenced by different basic blocks, and then merged ranges close to each other. Typically we found 2 or 3 address ranges in each workload. Then, in order to evaluate merit in splitting the address space between different memory technologies, we placed an address range to NVM at a time, and the rest to DRAM. Among the permutations tested, only few of the address ranges associated with NVM were frequently accessed; the best performance of these permutations is shown in Figure 7 and Figure 8. In rest of the permutations the memory accesses were concentrated in DRAM and hence the performance of the memory hierarchy is similar to that of base case and is not included in the figure.

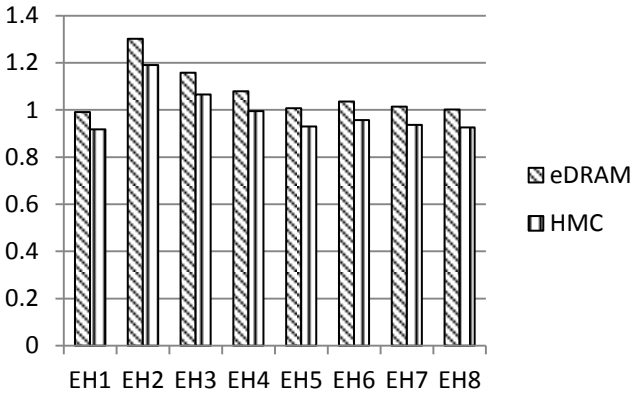

Figure 5: Average of normalized run time of all benchmarks for 4LCNVM

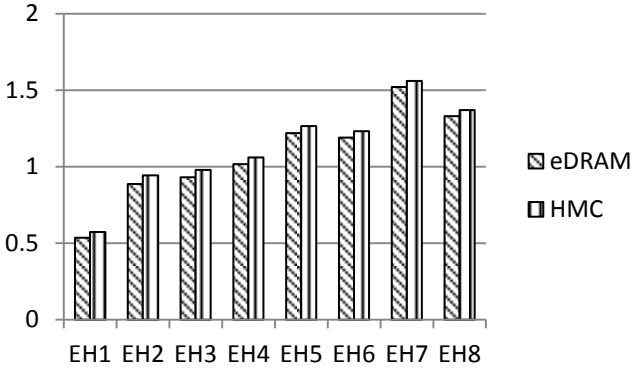

Figure 6: Average of normalized total energy of all benchmarks for 4LCNVM

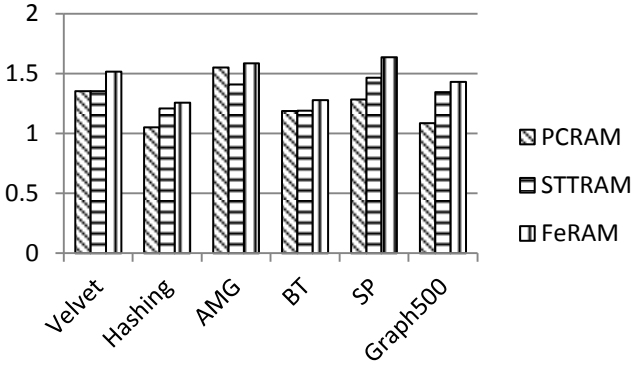

Figure 7: Average of normalized run time of all benchmarks for NDM design

In the NDM design, all the workloads have time overhead in the range, 5 to 63%, across different technologies. In the workloads, Velvet, Hashing, AMG and Graph500 we observe energy savings while in the case of BT, SP there is energy overhead. The former set of workloads has significant static energy as compared to dynamic energy while in the latter set the dynamic and static energy is comparable. When a NVM based device is used, the dynamic energy is likely to increase because of increase in access time with respect to DRAM but static energy is expected to decrease. Hence only the workloads which have relatively large static energy have energy savings in the hybrid design.

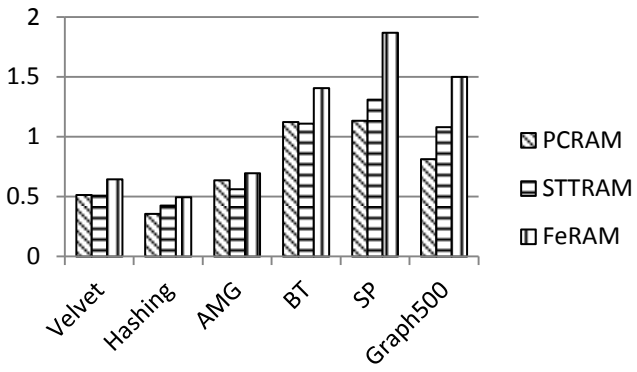

Figure 8: Average of normalized total energy of all benchmarks for NDM design

Figure 9 and Figure 10 visualize the impact of higher latency and energy per operation. Figure 9 is a *heat map* of the average slowdown caused by higher latency, whereas Figure 10 is a *heat map* that shows average energy consumption. The maps are generated using the execution profile of all the benchmarks for the NMM design (512MB DRAM, 512B page size) and scale DRAM latency and energy costs with respect to DRAM.

Through this heat map we can appreciate the impact of read/write latency. We observe that in general read operations dominate, and an increase in read latency has higher impact than an increase in write latency. For example, a 5x increase in read results in 5% runtime penalty, whereas a similar increase in write latency results in only 1% runtime penalty. We also observe that the performance penalty is very limited, with a 17% performance penalty for a 20x increase in both read and write latency.

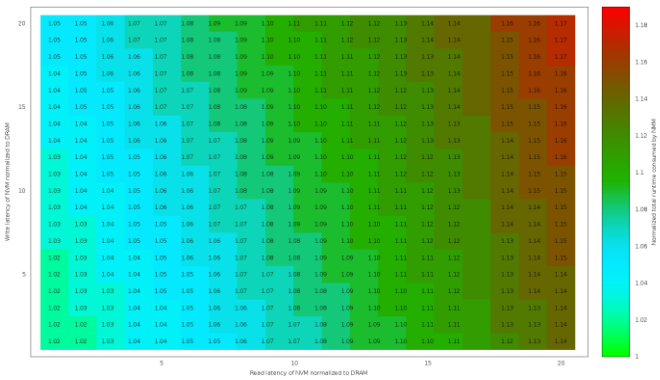

Figure 9 : Heat-map of normalized runtime of NMM as a function of read and write latency

Similar inferences can be made on the impact of read and write energy. Up to a 9x increase in write energy and 2x increase in read energy yields less or the same energy consumption of DRAM. As before, the dominance of read

operations puts more weight on the cost of the read operations. Finally, unlike for performance, small increases in energy cost over DRAM are compensated by the lower static energy of NVM and result in energy savings so there are several energy saving configurations with higher dynamic energy cost per operation than DRAM.

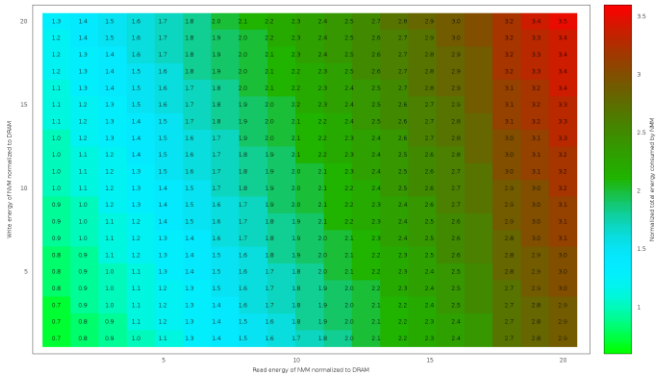

Figure 10: Heat-map of normalized energy consumed by NMM as a function of read and write latency

VI. Conclusions and Future Work

In this paper we present a first step in building a simulation framework to evaluate hybrid memory technology on data intensive applications. Using our simulation framework we modeled different memory technologies and hybrid memory architectures.

We evaluated four different designs of fast volatile memory (eDRAM/HMC) and dense and non-volatile (PCM/STTRAM/FeRAM) memory. Our NVM designs shows that a simple hierarchy of NVM following DRAM can provide energy saving of as much as 21%, with an overhead of 7% in runtime. Our 4LC design shows that faster technology as fourth level cache before DRAM provides modest savings in runtime and energy. Combining the two further improves the overall energy reduction to as much as 47% without any overhead in runtime. In all these three designs, the memory hierarchy is extended vertically by adding an extra level, integrated into the memory sub-system, and that does not require software support to be utilized.

Finally, we explored a hybrid hierarchy where DRAM and denser NVM form a partitioned address space. We found that for applications with a large memory footprint (and therefore incurring significant DRAM static energy costs) there is a potential average 42% savings in total energy, but at the cost of an average overhead of 25% in runtime, suggesting that the this solution, which is also more complex, is not as promising as the other designs. Further investigation should explore dynamic partitioning, that may change between computation phases, and take access patterns into account.

In our work, we focused on testing the potential of integrating emerging memory technologies in the memory hierarchy. We have not factored in the cost (e.g. total cost of ownership) or wearing, which is typical of NVM.

Future work will continue testing new applications, and focus on improving the modeling validating the results with an emulation platform.

Acknowledgment

This work was supported in part by the National Science Foundation (NSF) under NSF OCI award 0951583 entitled "I/O Modeling EAGER" and Intel Corporation. This work used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number ACI-1053575.

References

[1] W. A. Wulf and S. A. McKee, "Hitting the Memory Wall: Implications of the Obvious," *Computer Architecture News*, vol. 23, pp. 20-24, 1995.

[2] P. Kogge, "ExaScale Computing Study: Technology Challenges in Achieving Exascale Systems," *CSE Dept. Tech. Report TR-2008-13*, 2008.

[3] W. Chairs, R. Stevens, and A. White, "Workshop on Architectures and Technology for Extreme Scale Computing," 2009.

[4] M. Pavlovic, Y. Etsion, and A. Ramirez, "On the memory system requirements of future scientific applications: Four case-studies," in *Workload Characterization (IISWC), 2011 IEEE International Symposium on*, 2011, pp. 159-170.

[5] J. Barth, W. Reohr, P. Parries, *et al.*, "A 500MHz Random Cycle 1.5ns-Latency, SOI Embedded DRAM Macro Featuring a 3T Micro Sense Amplifier," in *Solid-State Circuits Conference, 2007. ISSCC 2007. Digest of Technical Papers. IEEE International*, 2007, pp. 486-617.

[6] J. Jeddeloh and B. Keeth, "Hybrid memory cube new DRAM architecture increases density and performance," in *VLSI Technology (VLSIT), 2012 Symposium on*, 2012, pp. 87-88.

[7] M. H. Kryder and K. Chang Soo, "After Hard Drives-What Comes Next?," *Magnetics, IEEE Transactions on*, vol. 45, pp. 3406-3413, 2009.

[8] ITRS. (2013). *2013 International Technology Roadmap for Semiconductors*. Available: http://www.itrs.net/Links/2013ITRS/Home2013.htm

[9] H. Niijima, "Design of a solid-state file using flash EEPROM," *IBM J. Res. Dev.*, vol. 39, pp. 531-545, 1995.

[10] R. A. Bheda, J. A. Poovey, J. G. Beu, *et al.*, "Energy efficient Phase Change Memory based main memory for future high performance systems," in *Green Computing Conference and Workshops (IGCC), 2011 International*, 2011, pp. 1-8.

[11] B. C. Lee, E. Ipek, O. Mutlu, *et al.*, "Architecting phase change memory as a scalable dram alternative," presented at the Proceedings of the 36th annual international symposium on Computer architecture, Austin, TX, USA, 2009.

[12] M. K. Qureshi, J. Karidis, M. Franceschini, *et al.*, "Enhancing lifetime and security of PCM-based Main

Memory with Start-Gap Wear Leveling," in *Microarchitecture, 2009. MICRO-42. 42nd Annual IEEE/ACM International Symposium on*, 2009, pp. 14-23.

[13] M. K. Qureshi, V. Srinivasan, and J. A. Rivers, "Scalable high performance main memory system using phase-change memory technology," presented at the Proceedings of the 36th annual international symposium on Computer architecture, Austin, TX, USA, 2009.

[14] Z. Wangyuan and L. Tao, "Exploring Phase Change Memory and 3D Die-Stacking for Power/Thermal Friendly, Fast and Durable Memory Architectures," in *Parallel Architectures and Compilation Techniques, 2009. PACT '09. 18th International Conference on*, 2009, pp. 101-112.

[15] D. H. Yoon, J. Chang, R. S. Schreiber, *et al.*, "Practical nonvolatile multilevel-cell phase change memory," presented at the Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, Denver, Colorado, 2013.

[16] E. Kultursay, M. Kandemir, A. Sivasubramaniam, *et al.*, "Evaluating STT-RAM as an energy-efficient main memory alternative," in *Performance Analysis of Systems and Software (ISPASS), 2013 IEEE International Symposium on*, 2013, pp. 256-267.

[17] P. Zhou, B. Zhao, J. Yang, *et al.*, "Energy reduction for STT-RAM using early write termination," presented at the Proceedings of the 2009 International Conference on Computer-Aided Design, San Jose, California, 2009.

[18] K. Hoya, D. Takashima, S. Shiratake, *et al.*, "A 64Mb Chain FeRAM with Quad-BL Architecture and 200MB/s Burst Mode," in *Solid-State Circuits Conference, 2006. ISSCC 2006. Digest of Technical Papers. IEEE International*, 2006, pp. 459-466.

[19] H. Shiga, D. Takashima, S. Shiratake, *et al.*, "A 1.6 GB/s DDR2 128 Mb Chain FeRAM With Scalable Octal Bitline and Sensing Schemes," *Solid-State Circuits, IEEE Journal of*, vol. 45, pp. 142-152, 2010.

[20] D. Takashima, H. Shiga, D. Hashimoto, *et al.*, "A scalable shield-bitline-overdrive technique for 1.3V Chain FeRAM," in *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2010 IEEE International*, 2010, pp. 262-263.

[21] P. Cicotti, J. Bennet, S. strande, *et al.*, "Evaluation of I/O technologies on a flash-based I/O sub-system for HPC," presented at the Workshop on Architecture and Systems for Big Data, Galveston Island, TX, 2011.

[22] P. Cicotti, M. Norman, R. Sinkovits, *et al.*, "Gordon: A Novel Architecture for Data Intensive Computing," in *On the road to Exascale Computing: Contemporary Architectures in High Performance Computing*, J. S. Vetter, Ed., ed: Chapman & Hall/CRC Press, 2013.

[23] A. Badam and V. S. Pai, "SSDAlloc: hybrid SSD/RAM memory management made easy," presented at the Proceedings of the 8th USENIX conference on Networked systems design and implementation, Boston, MA, 2011.

[24] S. Kannan, A. Gavrilovska, K. Schwan, *et al.*, "Optimizing Checkpoints Using NVM as Virtual Memory," in *Parallel & Distributed Processing (IPDPS), 2013 IEEE 27th International Symposium on*, 2013, pp. 29-40.

[25] B. Giridhar, M. Cieslak, D. Duggal, *et al.*, "Exploring DRAM organizations for energy-efficient and resilient exascale memories," presented at the Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, Denver, Colorado, 2013.

[26] J. Wang, X. Dong, and Y. Xie, "OAP: an obstruction-aware cache management policy for STT-RAM last-level caches," presented at the Proceedings of the Conference on Design, Automation and Test in Europe, Grenoble, France, 2013.

[27] L. E. Ramos, E. Gorbatov, and R. Bianchini, "Page placement in hybrid memory systems," presented at the Proceedings of the international conference on Supercomputing, Tucson, Arizona, USA, 2011.

[28] M. Pavlovic, N. Puzovic, and A. Ramirez, "Data placement in HPC architectures with heterogeneous off-chip memory," in *Computer Design (ICCD), 2013 IEEE 31st International Conference on*, 2013, pp. 193-200.

[29] L. Dong, J. S. Vetter, G. Marin, *et al.*, "Identifying Opportunities for Byte-Addressable Non-Volatile Memory in Extreme-Scale Scientific Applications," in *Parallel & Distributed Processing Symposium (IPDPS), 2012 IEEE 26th International*, 2012, pp. 945-956.

[30] N. Muralimanohar, R. Balasubramonian, and N. P. Jouppi, "CACTI 6.0: A Tool to Model Large Caches," Published in International Symposium on Microarchitecture2007.

[31] M. Laurenzano, M. Tikir, L. Carrington, *et al.*, "PEBIL: Efficient Static Binary Instrumentation for Linux.," presented at the IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), White Plains, NY, 2010.

[32] Micron. *System Power Calculators.* . Available: http://www.micron.com/products/support/power-calc

[33] D. H. Bailey, E. Barszcz, J. T. Barton, *et al.*, "The NAS parallel benchmarks—summary and preliminary results," presented at the Proceedings of the 1991 ACM/IEEE conference on Supercomputing, Albuquerque, New Mexico, USA, 1991.

[34] O. Ridge, Argonne, and Livermore. *CORAL: Benchmark Codes*. Available: https://asc.llnl.gov/CORAL-benchmarks/

[35] D. R. Zerbino and E. Birney., "Velvet: algorithms for de novo short read assembly using de Bruijn graphs," *Genome Research* vol. 18, pp. 821-829.