Energy Efficient Phase Change Memory Based Main Memory for Future High Performance Systems

Abstract—**Phase Change Memory (PCM) has recently attracted a lot of attention as a scalable alternative to DRAM for main memory systems. As the need for high-density memory increases, DRAM has proven to be less attractive from the point of view of scaling and energy consumption. PCM-only memories suffer from latency issues, high write energy, and the problem of limited write endurance. Research in this domain has focused mainly on using various hybrid memory configurations to address these shortcomings. A commodity DRAM module as a cache for PCM memory has emerged as a potential solution. But this method requires use of a separate memory controller and is unable to achieve better performance than a DRAM-only based memory at low energy.**

We propose a PCM based main memory system design using a small, embedded DRAM (eDRAM) cache to replace the row buffers in the PCM memory chip. This reduces the high latencies of PCM and the energy consumption of the main memory system. Our methodology also eliminates the need for separate memory controllers. Through simulation results, we show competitive performance by reducing average memory access time of a slow PCM based memory and significant energy reductions against a DDR3 commodity DRAM memory system of similar storage size. Our proposed system is highly energy efficient and provides 35.02%improvement in EDP over a DRAM-only system. Our system consumes less energy than the state-of-the-art PCM hybrid system using a commodity DRAM cache.

Keywords-Memory Controller, DRAM, PCM, Energy

I. INTRODUCTION

In the multi-core era, it is essential to be able to have a large and fast main memory for running multi-threaded, multi-programmed workloads efficiently. To this end, traditional DRAM memories have become less attractive as scaling becomes a problem and energy efficiency gets worse. Phase Change Memory (PCM) [5][6][9] has gained increasing popularity as a potential solution. An important advantage afforded by PCM is its high capacity, which has proven useful in the reduction of page faults.

For PCM to replace DRAM as the primary main memory technology, significant architectural changes must be made to PCM based main memory designs. PCM suffers from long latencies and limited write endurance [1][5][11]. Even though read latency for PCM is comparable to DRAM, writes are expensive in terms of delay and energy consumption. Overcoming these problems with a design that can help reduce the relatively long PCM latencies and exploit the advantages offered by PCM would be an energy-efficient solution for high performance memory systems.

We propose the design of intelligently mapped small cache structures per individual PCM chip to act as high-speed row buffers. These row buffers use embedded DRAM (eDRAM) technology [13][20]. These eDRAM cells are split into very small internal banks within the chip. This not only decreases the response time of the small memory arrays, it also greatly reduces read and write energy for these cells compared to large commodity DRAM and PCM arrays. Small eDRAM arrays have been shown to be advantageous in consuming less power, having faster response times, and consuming less read and write energy [20]. As data locality in these row buffers increases, the average latency of a PCM-eDRAM hybrid system is lower than a DRAM-only system.

Our work makes the following contributions.

1) Despite being a hybrid system, we eliminate the need for multiple memory controllers and we require no changes to the standardized, accepted signaling protocols.

2) We show that this methodology not only reduces the average access time of a PCM based memory system, but also has potential to perform better than a similar capacity, state of the art, DRAM-only system.

3) Our proposed system is highly energy efficient and performs 35.02% better EDP than the baseline DRAM-only system, and it consumes less energy than current state-of-the-art hybrid systems.

II. PCM TECHNOLOGY

The ever-increasing need for higher main memory capacity has driven the search for a memory technology that is scalable, denser, and faster with every generation while preferably consuming less energy. One potential solution, Phase Change Memory (PCM), is a form of resistive memory, which has gained a lot of interest due to its scalability. However, PCM's high write latency remains a concern.

PCM is a type of non-volatile memory that uses the unique behavior of chalcogenide ($Ge_2Sb_2Te_5$ or *GST*) glass for storing data bits. The state of this material can be altered between an amorphous and a crystalline state by application of a current pulse, which changes its resistance. This resistance determines the data value stored by a PCM cell. As seen in Figure 1, the two states of operation of PCM memory are defined as *SET* (crystalline or low resistance) and *RESET* (amorphous or high resistance).

The chalcogenide material can be *RESET* by a short and high current pulse. This high current pulse heats the element and then abruptly stops the current flow to quench the heating causing the element to freeze in amorphous form. A *SET* operation requires a moderate current pulse of longer duration. The *SET* current pulse slowly ramps down the current, allowing a slow cooling of the material, which induces the crystal growth. Typically the crystalline state (low resistance) is considered to be a logical 1, and the amorphous state (high resistance) is considered a logical 0. Reading a PCM cell consists of passing a current through it to detect the resistance value. Compared to writes, reads are much faster and comparable to standard DRAM latencies. Writes, especially *SET* operations, are much longer than reads and also result in higher energy consumption. Finally, similar to FLASH technology PCM cells have a limited write lifetime (~10^9-10^{12} writes) [9].

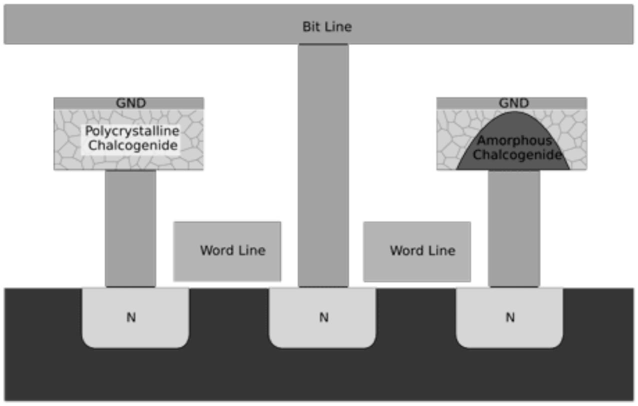

Figure 1: Two PCM Cells in Crystalline (SET) and Amorphous (RESET) States

Unlike capacitive storage in DRAM cells, the phase change material in PCM scales with each technology node and requires less volume. [9][17] Also as the volume of the material keeps reducing, less current is needed to program the material. This makes future generations of PCM memory faster, smaller and more energy efficient. Another advantage of resistive memories is that they consume very little leakage power. PCM provides a non-volatile storage mechanism amenable to process scaling. This will allow PCM to scale down for the next several generations of processing technology and have no physical limits until at least the 20nm technology node.

The most significant difference between PCM technology and SRAM/DRAM is the memory cells. A PCM cell is typically a "1T1R – 1 Transistor, 1 Resistor" structure, while SRAM cell is a conventional "6T – 6 Transistor" structure and DRAM cell is usually a "1T1C – 1 Transistor, 1 Capacitor" structure. The difference of cell structures directly leads to different cell sizes. The SRAM and the embedded DRAM cells have areas of 120−150F^2 and 19−26 $F^{2,}$ respectively, and the commodity DRAM cell area is about 6−8 F^2. The PCM cell has an area ranging from 22.68 F^2 to 9.60 F^2 , but PCM is expected to have multi-level cells, which make it a high capacity memory. [5] The PCM cell area is constrained by two factors: (a) the size of chalcogenide-based phase-change materials (*GST*s) and (b) the size of the selector device that could be a MOSFET, a BJT, or a diode [9].

The size of *GST* determines the minimum required programming current, which further decides the size of the selector device. For the scaling rule of *GST*, Pirovano et. al. [17] reported a detailed scaling analysis using a physics-based electro-thermal model of a cell verified by measurements conducted on sample devices. Their study shows that the *RESET* current can be scaled downward by scaling the contact area of the *GST*. A generic scaling rule with constant voltage implies that a smaller *GST* size is usually preferred because it can lead to a lower requirement on the programming current amplitude.

III. MOTIVATION

Current trends in architecture suggest that the number of cores on chip will steadily rise for the next few years. Without adequate scaling in the memory system, the memory system gap will widen. Multi-threaded and multi-programmed workloads on a single chip increase the demand on the memory system. Therefore an ever-increasing size of main memory is necessary. PCM offers a potential solution to DRAM since it has a comparable read latency and provides better scaling opportunities. Additionally DRAM is more expensive than similarly sized PCM in terms of power dissipation. One potential solution in the form of a commodity DRAM cache has been studied by Qureshi et. al. [1]. This memory architecture aims to reduce the response time of a PCM based main memory through a small DRAM cache and improve the memory lifetime by filtering out unneeded writes. But as seen in Figure 2, it has a limit on the response time of a commodity DRAM module. Additionally, at the memory interface, this requires use of two separate controllers for DRAM and PCM to manage the differing and variable latencies. By leveraging a faster yet smaller cache that can be integrated within a PCM memory chip, the response time of a PCM based memory can fall below that of a DRAM only based system, thus improving upon the response time limitation of a cached PCM system with commodity DRAM cache.

Figure 2 shows that with a faster cache (e.g. eDRAM), the average access latency a PCM based memory can be reduced below that of a DRAM only system, if sufficient data locality is observed. DRAM memory research has resulted in many techniques to exploit row buffer locality at the main memory. Recent research in the area of memory access latency improvement has suggested that multi-threaded, multi-programmed workloads stress the row buffers far beyond their physical capacity [12]. The row buffer size at the main memory is an important parameter in memory performance as it is able to keep an entire OS page open for future accesses or sometimes multiple pages are kept active across different banks. This allows for faster access if the locality is high. Multi-programmed workloads cause these accesses to have reduced locality at the main memory, thus causing row buffer conflicts and high latencies. Many scheduling schemes have been proposed to increase row buffer hits, but this problem is only expected to get worse. In the case of a slower main memory consisting of PCM cells this problem will worsen.

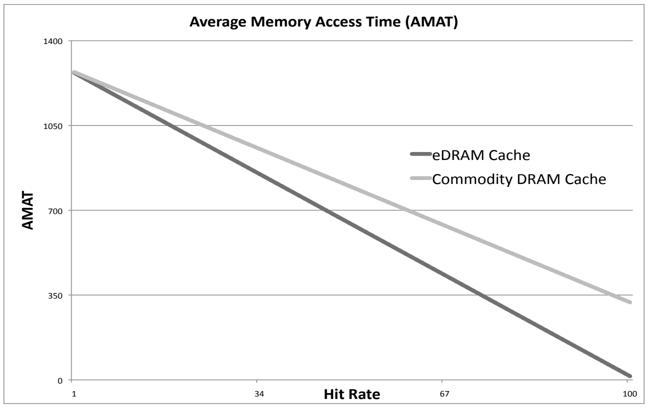

Figure 2: Average Memory Access Time (AMAT) Comparison

A fast cache structure will be able to store these large, frequently accessed pages or sub-pages for future accesses. In this cache, frequently and recently accessed pages from multiple programs can be stored and retrieved at faster speed. This mechanism will be able to exploit the limited spatial and temporal locality observed in the main memory access pattern.

We observed the memory access pattern of a representative benchmark from NAS OMP benchmark suite. Figure 3 shows the percentage of memory accesses for four 1KB sub-pages within a 4KB page. These accesses to the main memory show locality within a given page. Some of the more frequently accessed pages show uniform access across all sub-pages within a given page. But these accesses are distributed in time. Hence a cache structure is required which will be able to keep these lines active longer than a row buffer would in a conventional memory. Also in a multi-threaded, multi-programmed environment, many such lines will need to be maintained at the same time.

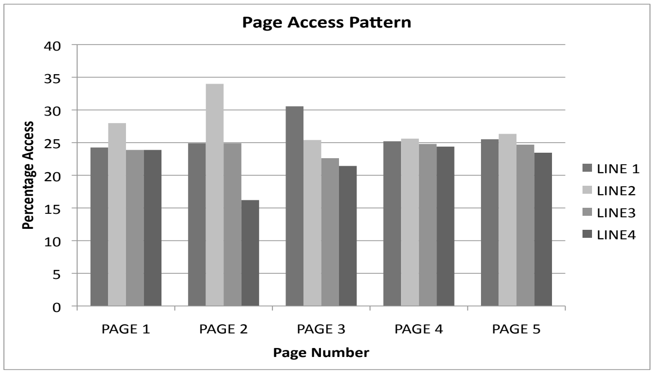

Figure 3: Access Pattern for 1KB Sub-pages (Lines) Within an OS Page

It is therefore evident that if we maintain pages at a sub-page granularity of 1KB in a faster cache, we can ensure a large number of hits to this cache. This will help us reduce the long PCM latencies and make it comparable to a DRAM only system or in some cases even less. We choose line-sizes of 1KB – 2KB as smaller line sizes reduce array response time and read energy. Also further reduction is not possible as it reduces locality within these lines, making the caches less useful.

IV. Hybrid Architecture

Internally, the PCM chip is made up of multiple bank arrays. This architecture is similar to current memory technologies, and the multiple bank structure is expected to continue with future generation of PCM memories also. Our hybrid PCM-eDRAM architecture consists of PCM memory chips with an integrated eDRAM cache as shown in figure 4.

Just as in a commodity memory module (eg. DDR3), a PCM memory module rank is also made up of 8 devices (chips). Each of these 8 devices can be x8 or x16, which refers to the output data width from each chip. As shown in figure 5 (a), together these chips form a rank, which is connected to the data bus. Each of these chips is internally divided into smaller bank arrays. Whenever an access request is made to the memory, the same single bank in every chip in a rank is activated simultaneously to form a larger logical bank across all devices. Each of these devices outputs 8 bits or 16 bits of data to collectively form a 64 bit or 128 bit data bus respectively. In a similar fashion the integrated cache in every chip is also divided into per-bank caches. Thus in a multi-rank memory system our cache organization is logically split into per-rank per-bank (PRPB) caches.

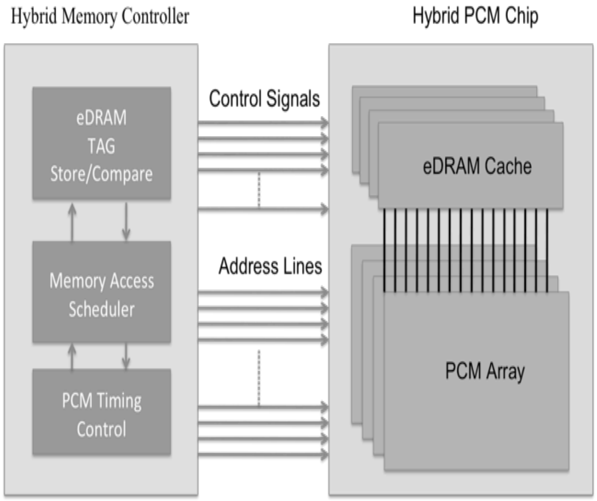

Figure 4: Hybrid Array Architecture with Unified Memory Controller

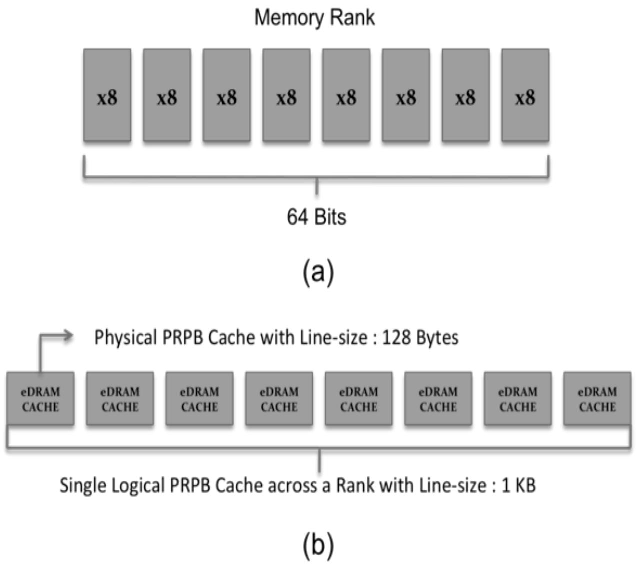

Figure 5: Memory Rank Organization

Whenever an access request is made to this hybrid memory system and data requested is found to be present in the caches, the same PRPB cache across all devices is activated to form one large logical cache. Therefore a logical cache, which has a line-size of 1KB is physically distributed across 8 devices in a rank. This results in smaller individual caches having a line-size of 128B as shown in figure 5 (b). Assuming a last level cache (LLC) line-size of 128B, for every access request to main memory, each cache would provide 16B from its selected line. When a main memory bank is selected, each device in a rank, contributes 8 bits from its bank, on every clock edge, to form a 64 bit data bus transfer. Sixteen such burst transfers would constitute a cache line transfer of 128B.

The cache tag directory is maintained at the memory controller. At every memory access request, a cache look-up is performed to check for presence of data in the PRPB cache. If data is present, an access request is treated similarly to an open-row access request in a conventional memory module, as shown in figure 6. An additional control signal line from the memory controller to PCM module is used to signal the PCM device to supply data from the cache instead of performing a PCM bank array access. A similarly integrated cached-DRAM chip has been previously proposed by Hidaka et. al [10].

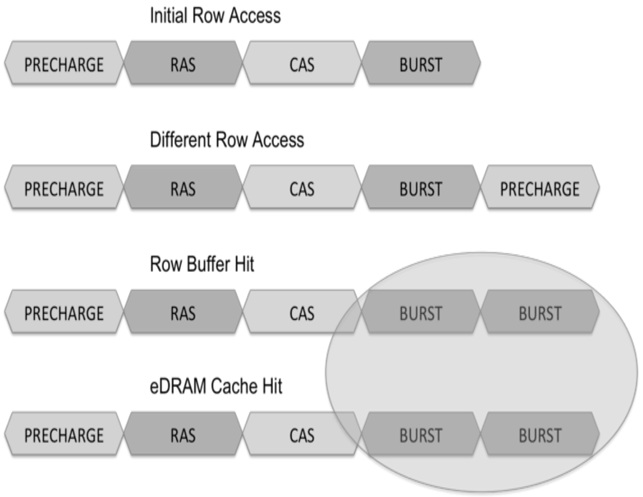

Figure 6: Hybrid Memory Signaling

We focus on three advantages offered by having a fast eDRAM cache integrated with the PCM memory. Firstly, PCM is expected to be a bridging gap between the hard disk and the main memory, which requires a separate independent memory controller between main memory and PCM. Commodity DRAM has been a popular choice for a PCM based hybrid memory. In order to make the rest of the system oblivious to the presence of two memories, the commodity DRAM has been proposed as a cache for a large capacity PCM storage. Our first goal is to reduce this complexity and provide a fast and integrated hybrid memory that uses a single controller. Whenever a new access request is made to the hybrid system, the memory controller looks up the cache tag array for a hit. On a hit in the cache, this access is considered as a row buffer hit as per standard memory access protocols. In a standard JEDEC compliant main memory controller, access requests are scheduled at fixed timing intervals based on memory timing parameters, which are available to the controller [7]. By classifying eDRAM cache hits as row buffer hits and using appropriate access time information, we easily blend this design into the current signaling schemes.

Our second advantage comes from having very small arrays of eDRAM. Though faster memory comes at a cost, these arrays have been shown to be very fast due to their smaller sizes with speed comparable to SRAM and power consumption slightly higher than DRAM [20], [13]. Instead of using a large unified cache, we split our cache memory into PRPB caches, which are further split into even smaller arrays by virtue of being integrated into every PCM chip. This greatly reduces their response time. Read and write energy is therefore much lower than activating a larger DRAM array. Since only one bank in a chip can be active and connected to the I/O ports at any given instant, we do not need separate decoders and look-up overhead for all bank caches within a chip, they are able to share the same circuitry.

Thirdly, we propose the use of an additional signal line that informs the PCM chip to interpret the address bus information as cache decode and way select logic, rather than activate a bank, in the event of a hit. Since the cache size is much smaller than the PCM array size, some lines in the address bus can be multiplexed as shown in Figure 7, to select a particular way in the cache. This eliminates the need for look-up in the device as a look-up has already been performed at the main memory controller. Such an integrated structure will help PCM memory modules become more practical and easily compatible with standard memory controllers with very few changes.

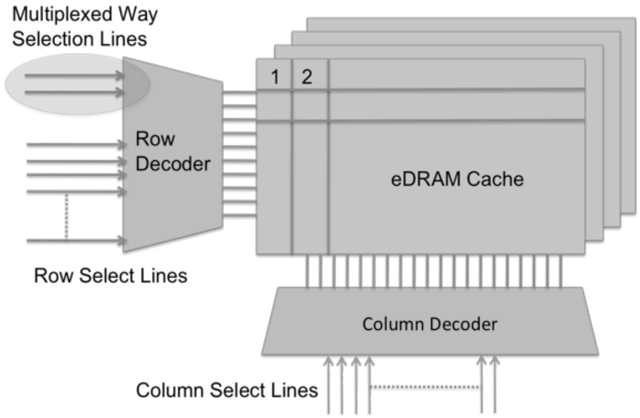

Figure 7: eDRAM Cache Selection

We also change the mapping scheme to distribute sub-pages across different banks in a rank. This reduces contention between the sub-pages within the same page. As observed in Figure 3, sub-pages in a page are accessed almost uniformly. These sub-pages need to be kept active in the fast row buffers simultaneously with minimum contention to avoid a costly bank access to the PCM array. This reduces conflict misses in the eDRAM cache at least among sub-pages from the same page.

The mapping example in Figure 8 shows the effect of mapping on the eDRAM cache. Logically, the eDRAM cache is split across the physical memory as a PRPB cache. We use the rank (R0-R1) select bits and bank (B0-B2) select bits from the memory address to select a PRPB cache. The row (r0-r14) select bits are used to select a cache set within the selected PRPB cache. Each logical cache line-size (sub-page size) is 1KB. Each of these sub-pages, therefore, resides in a different set in a different PRPB cache. This reduces the potential conflict between sub-pages from the same page.

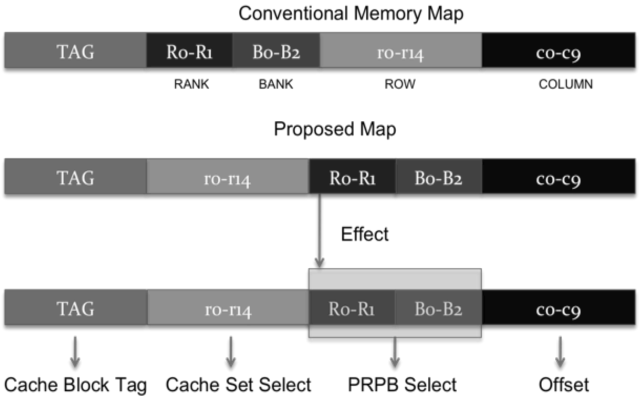

Figure 8: Memory mapping example

V. EXPERIMENTAL SETUP

Power consumption is a major concern in the design of modern systems. Most of the recent research in power management has focused more on dynamic management of CPU power consumption, assuming it to be the most dominant contributor to system power consumption. However, recent studies have shown that in modern systems CPU is no longer the only primary energy consumer. Main memory has become a significant energy consumer, contributing to as much as 30-40% of total consumption on modern server systems. To analyze the benefits of a hybrid PCM memory architecture, it is essential to accurately model its energy consumption behavior.

Testing of our proposed eDRAM/PCM hybrid memory is done via trace driven simulation. Pin [21] is used to capture an 8-threaded trace on a machine with 8 Intel Xeon X5450 running at 3 GHz. A single, interleaved thread trace stores each instructions thread id, opcode, source and destination registers, and the load/store address. These traces are compressed and stored for the NAS Parallel Benchmark Suite [22]. We evaluate a set of multi-programmed workloads, which are listed in Table 1.

Simulations are performed on a full system simulator consisting of sixteen virtual processors. All non-memory instructions are assumed to finish in a fixed time, i.e., register-to-register instruction dependencies are not modeled. For memory instructions, each processor contains 16 MSHRs. The processor stalls when all MSHRs are full. Each cycle, the simulator reads from the Pin generated trace file and is able to fetch at most 4 consecutive instructions per thread. Instruction fetch also stalls if a later interleaved instruction for a thread that has previously fetched this cycle is encountered. This disallows future instructions from a thread to hoist above instruction in a different thread that it may have held a dependency on. In other words, during a cycle each trace is allowed to consume at most one burst of consecutive instructions, with a max burst size of 4 instructions. This creates more realistic trace consumption issue width since it often cannot consume the max issue width across all threads. This also limits the system from being flooded with an unrealistic number of memory instructions.

Table 1 - List of Evaluated Workloads

NAS Parallel Workloads				Evaluated Multiprogrammed Workloads
Name	***Description***	***Input Set***		***Name***
CG	Conjugate Gradient	Class D		CG + IS
IS	Integer Sort	Class D		IS + EP
EP	Embarrassingly Parallel	Class D		LU + MG
LU	LU decomposition	Class D		UA + CG
MG	Multigrid computation	Class D		CP + LU
UA	Unstructured Adaptive Computation	Class D		MG + IS
SP	Pentadiagonal Solver	Class D		CG + LU
				IS + UA

In addition to the multi-threaded simulation, the simulator is also capable of running multi-programmed workloads as well. Two traces are fed to the simulator and each trace is assigned eight processors. Both traces contend for the large shared cache and more memory. In order to keep each application's addresses distinct, an additional bit is added to the addresses for each trace to distinguish its origin. Since we are evaluating only user-level code, each memory request's "physical address" consists of its virtual address plus the bit to indicate the source trace. Although this does not account for a full page table and TLB, this method still provides a reasonable interleaved access stream.

The on-chip cache is modeled as an L1/L2 cache hierarchy per processor core. The L1 cache size is 64KB and L2 cache size is 512KB across all cores. The dynamic and leakge power for the caches has been modeled using CACTI [13]. Beyond the cache, requests interact with our detailed memory model.

In order to accurately simulate for energy and memory access latencies, we model our simulator based on accepted standards and various parameters for memory modeling obtained from manufacturer data sheets. For a new technology such as PCM, we survey widely acknowledged published literature to obtain parameters. Our in-house DRAM simulator is based on DRAMsim [7,15] and follows standard JEDEC protocols for DDR3 memory [19]. From [5,9] it is clear that PCM access protocols will not be much different from those of DRAM. Therefore we extend our DRAM simulator by accounting for the fundamental differences between DRAM and PCM technologies, although following similar memory access protocols. To model the timing and power of a state of the art DRAM system, we consider the DDR3 SDRAM based system. The parameters are obtained from the Micron data sheet for x8 1Gb DDR3 SDRAM running at 667MHz. The model assumed for PCM is similar and timing and power characteristics are based on data obtained from [9,11,17]. These parameters are listed in table 2. We use CACTI to model eDRAM caches [13]. To obtain parameters for these caches, they are split into small physical caches, per-chip per bank as shown in figure 4.

Table 2: DRAM and PCM parameters for 1Gb memory chip

	DRAM	**PCM**
Row Read Power	210mW	78mW
Row Write Power	195mW	773mW
Activation Power	75mW	25mW
Leakage Power	90mW	45mW
Refresh Power	4mW	0mW
Row Read Delay	15ns	28ns
Row Write Delay	22ns	150ns
Row Buffer Hit Read/Write Delay	15ns	15ns

VI. RESULTS

We compare our design against a baseline system with 32GB DDR3 commodity DRAM. For our proposed technique, we vary the eDRAM cache size and also the line-size (effectively the row buffer size at the memory). We also use another effective and popular solution employing a 1GB commodity DRAM module as cache, similar to [1]. Our integrated caching methodology and the commodity DRAM cache both have a 32GB PCM second-level. Table 3 lists the memory configurations that we have evaluated.

Table 3: Evaluated Configurations

Our Evaluated Configurations	
32 GB commodity DDR3 DRAM only system	
256M_1024: 256MB eDRAM cache / 1KB line-size	32GB PCM Back-up
256M_2048: 256MB eDRAM cache / 2KB line-size	32GB PCM Back-up
128M_1024: 128MB eDRAM cache / 1KB line-size	32GB PCM Back-up
128M_2048: 128MB eDRAM cache / 2KB line-size	32GB PCM Back-up
1GB commodity DRAM	32GB PCM Back-up

In Figure 9, we observe that the average access time of a PCM based main memory is drastically reduced. Access time comparable to a fast commodity DDR3 system has been realized. Although our primary focus is to provide a low energy speedup solution, we see that some benchmarks show possible speedup beyond DRAM if more eDRAM cache hits can be ensured.

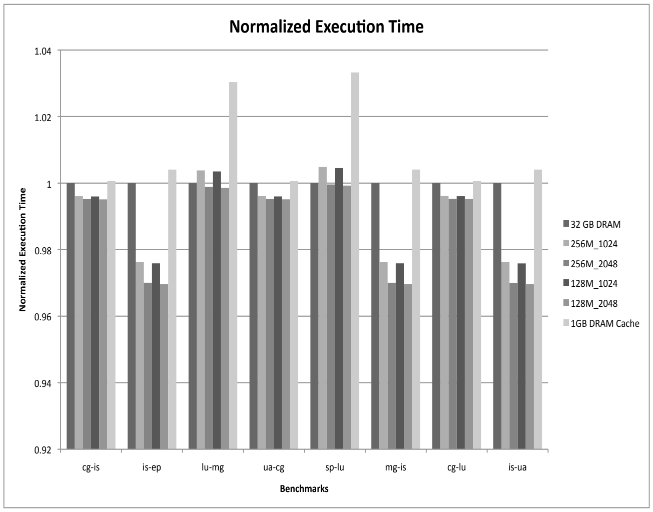

Figure 9: Normalized Execution Time

In Figure 10, significant energy gains are observed compared to the baseline. This is mainly because of the high leakage energy associated with a large capacity DRAM based memory. Compared to a small eDRAM cache, the array read and write latency as well as read/write power is high for DRAM, eventually resulting in higher memory power consumption. We also observe that as we increase our eDRAM cache line-size, performance is slightly improved. This is due to the increased locality in the cache, although having a bigger line size does also increase the read and write energy for this cache, leading to an increase in the overall energy consumption.

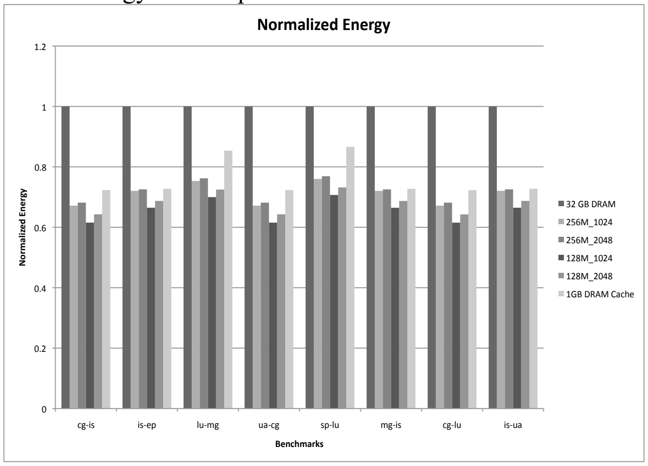

Figure 10: Normalized Energy Consumption

We also notice from Figure 10 the line size has a greater effect on memory performance than capacity. From our results we observe that, as eDRAM cache line-size is increased from 1KB to 2KB, an improvement in performance is observed, which is not the case when cache capacity is increased from 128MB to 256MB. A larger line-size for the eDRAM cache essentially increases the locality at main memory, which has a greater effect on performance. As has been observed in Figure 3, main memory accesses during certain access intervals are made to different sub-pages within the same page. An open row in a commodity DRAM is able to hold 2-4 pages. Although the size of the row buffer is negligibly small compared to the memory size, subsequent hits to this open row are sufficient to improve memory performance. Hence, data locality is more important than capacity in a main memory cache. Even though a larger capacity might help by holding large, sparsely accessed datasets, there is a higher cost in terms of power and area for this benefit. We show that for typical applications, a smaller cache is nearly as efficient and provides a low energy solution because small memory caches save in leakage power as well as read and write energy.

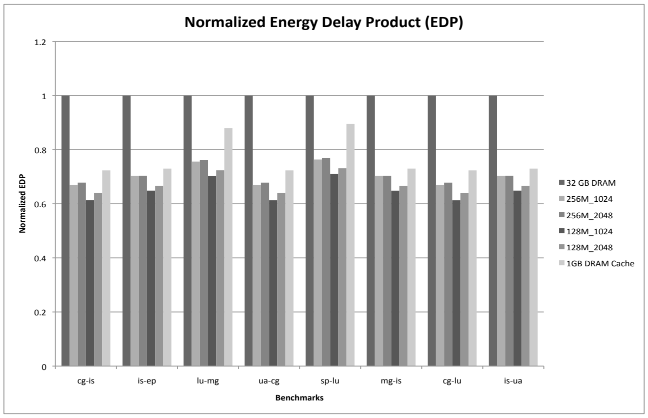

Figure 11: Normalized Energy Delay Product (EDP)

Table 4: Energy and EDP Values for Baseline Configuration

	Energy (nJ)	EDP (nJ-s)
cg-is	1.67E+10	4.97E+09
is-ep	2.80E+10	1.40E+10
lu-mg	1.99E+10	7.00E+09
ua-cg	1.61E+10	4.60E+09
sp-lu	1.81E+10	5.79E+09
mg-is	2.47E+10	1.09E+10
cg-lu	1.51E+10	4.03E+09
is-ua	2.84E+10	1.44E+10

Table 5: Results Summary (Average Energy and EDP Saving)

Configuration	% Energy Saving	% EDP Saving
256M_1024	28.84	29.56
256M_2048	28.09	29.04
128M_1024	34.4	35.02
128M_2048	31.9	32.82
1GB DRAM Cache	24.11	23.3

Table 4 lists the values for Energy and Energy Delay Product (EDP) for the baseline configuration of 32GB commodity DRAM. Energy Delay Product (EDP) is the typical performance metric for energy efficiency as it summarizes the tradeoff between performance and energy. It is an indicator of the cost of speedup. Improved performance typically costs additional energy. For example, faster memories are power hungry, but they perform better in terms of latency. Figure 11 and Table 5 show a 35.02% performance gain over baseline EDP as the eDRAM cache capacity and line size is varied. Additionally, these results also show that the smaller eDRAM approach provides better EDP than the recent use of a commodity DRAM cache, sized at 3% of the PCM [1].

VII. RELATED WORK

Hybrid architectures have been proposed to make PCM usable in low power cache designs for on-chip caches. PCM memory with DRAM buffers has been used to hide the relatively long latencies of PCM. Qureshi et. al. [1] have shown performance gains by using a large PCM memory with a small DRAM buffer. The performance gain reported has mainly come from reduction of page faults against a baseline DRAM memory of 1/4th capacity. Such a system makes use of a commodity DRAM module as cache for PCM memory and aims to bring down the relatively long response time of PCM memory. But even then, the access latencies of the hybrid memory cannot be reduced beyond that of a DRAM only system. This hybrid system also makes use of separate memory controllers for DRAM and PCM subsystems of the main memory.

Dhiman et. al. [11] propose a hybrid PCM/DRAM memory which employs a DRAM memory to reduce writes to PCM memory. This work mainly concentrates on using a hybrid system to improve the write endurance of a PCM based memory and also reduce the energy consumption of a main memory system. PCM based memories allow reduced power dissipation and this property has been exploited to design a low power memory system.

In contrast to the previously mentioned works, we have designed a unified memory system with a single memory controller. We have proposed a hybrid PCM module which can be used as a practical DRAM replacement for main memory designs.

VIII. CONCLUSION

In this work, we proposed a novel hybrid PCM system that leveraged eDRAM to create a cache for the PCM memory that replaces the row buffer. This removes the need for multiple memory controllers and provides minor performance gains and large energy improvements. In doing so improve the state-of-the-art PCM hybrid technology. Our scheme improves the attractiveness of PCM as a potential main memory replacement. This provides great opportunity to continue energy-efficient memory scaling into the future.

REFERENCES

[1] Moinuddin K. Qureshi, Vijayalakshmi Srinivasan, Jude A. Rivers, “Scalable high performance main memory system using phase-change memory technology,” in ISCA’09.

[2] Ricardo Gonzalez and Mark Horowitz, “Energy dissipation in general purpose microprocessors,” in IEEE Journal of Solid State Circuits, Vol. 31, No. 9, September 1996.

[3] Mircea R. Stan and Kevin Skadron, “Power aware computing,” published by IEEE Computer Society, December 2003.

[4] Omid Azizi, Aqeel Mahesri, Benjamin C. Lee, Sanjay J. Patel, Mark Horowitz, “Energy-performance tradeoffs in processor architecture and circuit design: A marginal cost analysis,” in ISCA’10

[5] Prasanth Mangalagiri, Aditya Yanamandra, Yuan Xie, N. Vijaykrishnan, Mary Jane Irwin, Karthik Sarpatwari, O. O. Awadel Karim, “A low power phase change memory based hybrid cache architecture,” in GLSVLSI’08.

[6] Xiaoxia Wu, Jian Li, Lixin Zhang, Evan Speight, Ram Rajamony, Yuan Xie, “Hybrid cache architecturewith disparate memory technologies,” in ISCA’09.

[7] Bruce Jacob, Spencer W. Ng, David T. Wang, *Memory Systems – Cache, DRAM, Disk*. Elsevier, 2008

[8] Xiangyu Dong, Norman P. Jouppi, Yuan Xie, “PCRAMsim: System level performance, energy, and area modeling for phase-change RAM,” in ICCAD’09.

[9] Benjamin C. Lee, Engin Ipek, Onur Mutlu, Doug Burger, “Architecting phase change memory as a scalable DRAM alternative,” in ISCA’09.

[10] Hideto Hidaka, Yoshio Matsuda, Mikio Asakura, Kazuyasu Fujishima, “The cache DRAM architecture : A DRAM with an on-chip cache memory,” in IEEE Micro’1990.

[11] Gaurav Dhiman, Raid Ayoub, Tajana Rosing, “PDRAM: A hybrid PRAM and DRAM main memory system,” in DAC’09.

[12] Kshitij Sudan, Niladrish Chatterjee, David Nellans, Manu Awasthi, Rajeev Balasubramonian, Al Davis, “Micro-pages : Increasing DRAM efficiency with locality-aware data placement,” in ASPLOS’10.

[13] Shyamkumar Thoziyoor, Naveen Muralimanohar, Jung Ho Ahn, and Norman P. Jouppi, *CACTI 5.1*, HP Laboratories, Palo Alto.

[14] Micron system power calculator,
http://www.micron.com/support/dram/power_calc.html

[15] David Wang, Brinda Ganesh, Nuengwong Tuaycharoen, Kathleen Baynes, Aamer Jaleel, and Bruce Jacob, “DRAMsim: A memory system simulator,” ACM SIGARCH Computer Architecture New, Vol. 33, No. 4, September 2005.

[16] Benjamin C. Lee, Ping Zhou, Engin Ipek, Onur Mutlu, Jun Yang, Youtao Zhang, Bo Zhao, Doug Burger, “Phase change technology and the future of main memory,” IEEE Micro Top Picks 2010.

[17] Pirovano A, Lacaita A.L., Benvenuti A, Pellizzer F., Hudgens S., Bez, R., “Scaling analysis of phase-change memory technology,” IEDM’03.

[18] Micron Technical Note, Calculating memory system power for DDR3.

[19] JEDEC SDRAM DDR3 standard

[20] IBM Announces Industry's Densest, Fastest On-Chip Dynamic Memory in 32-Nanometer, Silicon-on-Insulator Technology, IBM press release, September 2009.

[21] Vijay Janapa Reddi, Alex Settle, and Daniel A. Connors, Robert S. Cohn, “PIN: A Binary Instrumentation Tool for Computer Architecture Research and Education,” in WCAE’04.

[22] D. Bailey, E. Barszcz, J. Barton, D. Browning, R. Carter, L. Dagum, R. Fatoohi, S. Fineberg, P. Frederickson, T. Lasinski, R. Schreiber, H. Simon, V. Venkatakrishnan, S. Weeratunga, “The NAS Parallel Benchmarks,” RNR Technical Report RNR-94-007, March 1994.