Pragmatic Integration of an SRAM Row Cache in Heterogeneous 3-D DRAM Architecture Using TSV

Dong Hyuk Woo, *Member, IEEE*, Nak Hee Seong, and Hsien-Hsin S. Lee, *Senior Member, IEEE*

***Abstract*—As scaling DRAM cells becomes more challenging and energy-efficient DRAM chips are in high demand, the DRAM industry has started to undertake an alternative approach to address these looming issues—that is, to vertically stack DRAM dies with through-silicon-vias (TSVs) using 3-D-IC technology. Furthermore, this emerging integration technology also makes heterogeneous die stacking in one DRAM package possible. Such a heterogeneous DRAM chip provides a unique, promising opportunity for computer architects to contemplate a new memory hierarchy for future system design. In this paper, we study how to design such a heterogeneous DRAM chip for improving both performance and energy efficiency. In particular, we found that, if we want to design an SRAM row cache in a DRAM chip, simple stacking alone cannot address the majority of traditional SRAM row cache design issues. In this paper, to address these issues, we propose a novel floorplan and several architectural techniques that fully exploit the benefits of 3-D stacking technology. Our multi-core simulation results with memory-intensive applications suggest that, by tightly integrating a small row cache with its corresponding DRAM array, we can improve performance by 30% while saving dynamic energy by 31%.**

***Index Terms*—3-D stacking, Cache, dynamic random access memory (DRAM), main memory, through-silicon-via (TSV).**

I. Introduction

WHILE different market segments anticipate their own DRAM product to be optimized for their favored design goals such as low latency, high bandwidth, low power, or high density, meeting all of these requirements is nearly impossible [16]. In reality, the density (or cost) has been one of the most important design goals in conventional DRAM industry mainly because of its extremely competitive market [12], [20]. To survive in such a tough market, vendors have to reduce their manufacturing cost to deliver their products to end users at low cost while staying profitable.

However, the DRAM industry is facing several imminent challenges from the limitation posed by fundamental physics and also from increasing needs by consumers. First of all, the DRAM industry is facing a scaling challenge. As the device feature size keeps miniaturized, the capacitance of the DRAM cell also decreases, at the same time, the junction leakage current drastically increases [18]. In other words, maintaining enough capacitance and reducing leakage current pose a significant challenge for the DRAM industry from scaling the DRAM feature size any further. Aside from such physics limitation, the cost of building a semiconductor fab also increases substantially over technology generations (often called Moore's Second Law [7]), making the competitive DRAM market even tougher.

On the other hand, the need from a DRAM consumer also continues to evolve as the industry rapidly embraces the *cloud computing* paradigm. In particular, the emerging trend of cloud computing drives the DRAM industry to optimize for power efficiency. Cloud service providers typically virtualize their computing resources among their massive number of clients and use middleware to provision resources based on given workloads and their subscribers' priority. The utilization of their servers can easily be saturated during the peak hours. Under such a utility-based computing model, the total service operating cost, including the energy bill for electricity and cooling facility, can be much more significant than the one-time hardware acquisition cost [4]. As a result, the cloud service providers are more willing to pay extra money to buy DRAM modules that significantly save their energy cost over the DRAM's operational lifetime.

In response to such scaling and energy efficiency challenges, the DRAM industry is undertaking novel approaches. One innovative solution is to stack DRAM dies vertically using the emerging 3-D integration technology. By stacking multiple dies without scaling the device size, the DRAM vendors can increase the DRAM density without paying the cost of using a finer lithography technology. For example, stacking multiple DRAM dies with the conventional system in package (SiP) technology has already been commercialized [30]. Furthermore, the DRAM industry expects to have 3-D DRAM chips stacked with through silicon vias (TSVs) [8], [13], [14], [20]. For example, Samsung has demonstrated an 8 Gb 3-D stacked DDR3 DRAM chip that consists of four DRAM layers [13]; in which three layers are slave layers without any I/O related circuit while one layer is a master layer that has shared I/Os. Such sharing is enabled by TSVs that allow high bandwidth, low latency, and low power data communication among layers. As a result, such a TSV-based design can reduce a significant amount of standby and active power compared to an SiP-based design [13]. More recently, Elpida announced an 8 Gb 3-D DDR3 SDRAM stacked with eight DRAM layers and one logic layer dedicated for interface circuits [8], [14]. Implementing DRAM cells and

Manuscript received March 29, 2011; revised September 06, 2011; accepted October 18, 2011. Date of publication December 09, 2011; date of current version December 19, 2012. This work was supported by the Department of Defense and an NSF Grant CCF-0811738. This paper was presented in part as an invited paper in the 54th IEEE International Midwest Symposium on Circuits and Systems, 2011, p. 1–4.

D. H. Woo is with Intel Labs, Intel Corporation, Santa Clara, CA 95054 USA (e-mail: dong.hyuk.woo@intel.com).

N. H. Seong and H.-H. S. Lee are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: nhseong@ece.gatech.edu; leehs@gatech.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVLSI.2011.2176761

interface circuits on separate, heterogeneous layers allows each of them to perform better and to consume less power [8].

Not being satisfied with those benefits, we believe that such a TSV-enabled, heterogeneous 3-D DRAM architecture can provide a unique, promising opportunity for computer architects to contemplate a novel memory architecture for achieving higher performance and better energy efficiency in a cost-effective way. To understand the opportunities and challenges of such a novel, heterogeneous 3-D DRAM architecture, in this paper, we perform a circuit- and architecture-level study with the cost and energy efficiency constraints in mind. The contribution of this paper includes the following.

- We propose a heterogeneous 3-D DRAM chip design that can better exploit spatial locality by tightly integrating a small SRAM cache with its corresponding DRAM array.
- We perform a circuit-level study to evaluate the feasibility of our proposal in terms of area and energy overhead. Especially, we carefully modeled the area and energy overheads of TSVs and found that TSVs do not consume much energy while occupying considerable area when they are integrated with DRAM.
- Although TSVs do not consume much energy, we found that energy consumption of 2-D wires is still a roadblock to realize an SRAM row cache. Thus, we propose a novel DRAM array design called folded bank. With our folded bank design, we can stack DRAM subarrays of one DRAM bank vertically, which allows us to tightly integrate an SRAM row cache while reducing the energy consumption of the 2-D wires.
- To overcome problems discovered from our circuit-level study, we perform an architecture-level study to address the performance and energy efficiency issue.

The rest of this paper is organized as follows. Section II discusses related work, and Section III explains the circuit-level DRAM array architecture that has been largely ignored by computer architects but that is critical in understanding detailed trade-offs in designing a novel DRAM chip. Section IV explores the design space of a TSV-enabled heterogeneous DRAM chip. Section V evaluates different designs proposed in Section IV and Section VI concludes this paper.

II. Related Work

Integrating an SRAM cache into a DRAM chip was studied in the past mainly to address the growing speed disparity in the memory hierarchy [2], [11], [21], [39]. Furthermore, the same concept has even been commercialized into products such as CDRAM, EDRAM, and VCM. However, they were all very expensive, failing to succeed in the mainstream market because both SRAM and DRAM had to be manufactured in the same die. On the contrary, as the DRAM industry faces scaling challenges and looming energy-efficiency issues, they started to turn to the emerging TSV technology to continue to improve the DRAM density by vertically integrating DRAM and logic dies onto one package. This approach improves the cost-effectiveness of a heterogeneous memory hierarchy. This paper explores the design space of heterogeneous 3-D DRAM chips, which present several modern challenges. We studied their circuit-level issues and proposed architectural techniques to minimizing area and energy overhead while improving performance substantially. A recent paper on CPU-DRAM integration [23] demonstrated the benefits of multiple row buffers, but this paper explores circuit-level challenges to implement those row buffers considering the area and power overhead of TSVs. Unlike the previous paper that assumed folded wordlines/bitlines, this paper found that such array architecture can harm the overall density significantly due to the area overhead of TSVs.

On the other hand, as the 3-D integration technology allows heterogeneous stacking onto the same die package, researchers have studied hybrid memory hierarchies that integrate several memory technologies such as SRAM, DRAM, eDRAM, MRAM, STT-RAM, and PCM [5], [23], [24], [31], [35], [36], [38], [40], [41]. However, they are mostly interested in designing hybrid cache architecture integrated into the processor cores [5], [24], [31], [36], [40], [41] or integrating the entire memory hierarchy onto a CPU package [23], [35], [38]. In contrast, this paper explores the design space of a 3-D DRAM chip rather than a 3-D CPU chip. Note that this paper is motivated by a recent observation that our applications have very good spatial locality within a page, but we cannot exploit it very efficiently mainly due to the limited capacity of a row buffer [35]. While they did not suffer from this problem in terms of performance (because they fetched an entire page to the L2 cache), their solution is applicable only when the bandwidth is abundant (when whole memory is stacked on top of a processor). Furthermore, it still spends a lot of energy in on-chip wires. On the other hand, for high-end systems where stacking whole system memory can be challenging, our proposal addresses this problem by placing a small SRAM row cache underneath its corresponding DRAM array in a DRAM chip.

III. DRAM Architecture

Before detailing our proposals, in this section, we explain DRAM terminology and its array architecture for readers to better understand the design trade-offs of our proposal. Fig. 1(a) shows an example of a *DRAM module* consisting of one or more *ranks*.[1] A rank is a set of DRAM chips that respond to a single DRAM command in lockstep. As a result, the aggregate pin data width of these chips is equivalent to the data width of a DRAM data bus. In this example, nine chips each of which outputs eight bits (often called $\times 8$ or 8 DQs) form a bus with 64 data bits and eight error correction bits. The reason why a data bus is spread across multiple chips is to reduce the cost of implementing a large number of data pins on each chip. As a result of such a spread data bus, upon a single row open command, each of these chips opens its own portion of a single DRAM row in parallel. In this example, if a DRAM row is as large as 8 kB, each $\times 8$ chip whose "physical" row size is 1 kB opens its own 1 kB row.

Each chip consists of multiple *banks* as shown in Fig. 1(b). A bank is an independent memory array that has an independent set of a row decoder, sense amplifiers, I/O gating circuits, and a column decoder while multiple banks in a chip share off-chip interface-related circuits. Such independence allows a memory

[1]Fig. 1(a) shows a simple example of a DRAM module that only contains one rank on one side of the module. Some modern DRAM modules accommodate up to two ranks on each side.

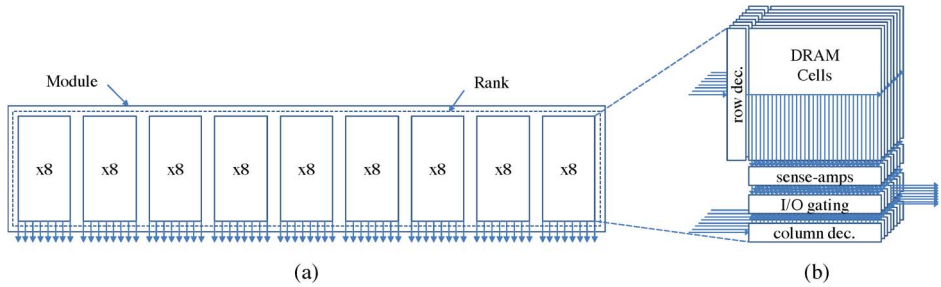

Fig. 1. DRAM architecture. (a) One DRAM module and (b) multiple banks in a DRAM chip.

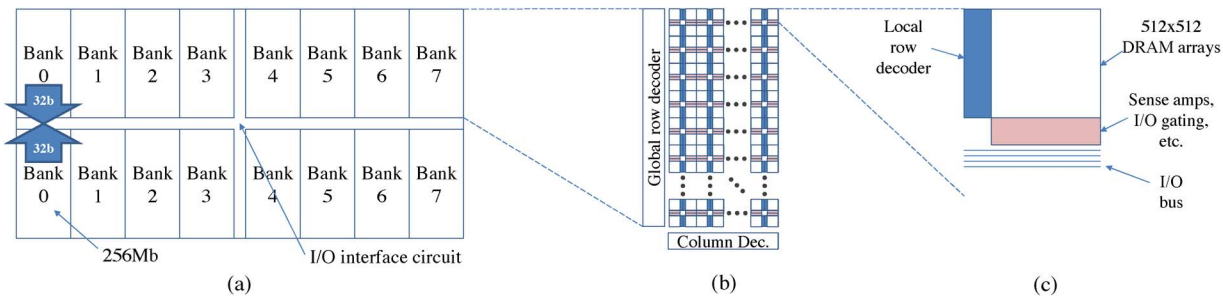

Fig. 2. DRAM subarray architecture. (a) One DRAM chip, (b) a 256 Mb array block, and (c) one subarray.

controller to exploit bank-level parallelism for achieving better performance. In the context of an entire rank, one "logical" bank is spread across eight "physical" banks (eight chips).

While DRAM vendors provide such an abstract DRAM array architecture to computer architects through a DRAM standard, their internal implementation has evolved over time to address many implementation challenges. One example of such an internal structure is shown in Fig. 2(a). The figure represents an 8-bank $\times 8$ 4 Gb DDR3 chip. As shown in the figure, those eight banks are linearly laid out horizontally while each bank is split into two 256 Mb half-banks vertically [25], [26], [37]. These 16 half-banks form four quadrants separated by periphery circuits. Also note that each of the two half-banks of one bank provides 32-bit data to the interface circuits forming a 64-bit global I/O bus. The reason why a 64-bit bus is needed in a $\times 8$ chip is to support the minimum burst length of eight by the DDR3 standard.

As shown in Fig. 2(b), each 256 Mb block is constructed by replicating a smaller array called subarray [3], [12], [15], [25]. Such a hierarchical design is better than a plain design for its higher signal-to-noise ratio, lower power consumption, and higher speed [12]. As shown in the figure, these subarrays share one global row decoder, which activates a subset of subarrays based on one part of a given row address. They also share a column decoder that provides global column select signals to each subarray. Each subarray is a 256 kb DRAM array with 512 wordlines and 512 bitlines [3], [25]. The size of a subarray does not rapidly change across different DRAM generations [3]. Rather, DRAM designers have been implementing more subarrays as the manufacturing process evolves. As shown in Fig. 2(c), each subarray has its own local row decoder, which decodes the other part of a row address to activate a local row. At the end of bitlines, each subarray has a set of sense amplifiers, which also behave like a row buffer by latching sensed signals. A subset of the latched signals is selected by column select signals (generated by the column decoder) to drive an I/O bus so that those data can reach the periphery circuits [see Fig. 2(a)]. Note that the data width of the I/O bus is much narrower than the data width of the sense amplifiers. In the case of our example, a $\times 8$ 4 Gb DDR3 DRAM chip, this bus is only 32-bit wide while one row of each 256 Mb half-bank consists of 4096 sense amplifiers (i.e., one "physical" row of each bank is 8 kb).

IV. Designing TSV-Enabled Heterogeneous DRAM Chips

Unlike a conventional, planar DRAM architecture described in the previous section, small, fast, and short TSVs allow one to integrate multiple DRAM dies vertically providing high bandwidth, low latency, and low power interconnection among dies. Furthermore, such a multiple die design allows industry to accommodate one logic die inside a DRAM chip as shown in Elpida's demonstration [8], [14] where one logic layer is dedicated for off-chip interface circuits. However, dedicating only interface logic in the logic layer is likely to under-utilize the die space because, regardless of the small size of the interface logic, the logic layer should be considerably large enough to accommodate the same number of I/O pins as a conventional DRAM chip. Furthermore, if the industry opts for wafer-to-wafer bonding, the logic die should have the same size of the other DRAM dies. Thus, we believe that there will be much space available for implementing other enhancement circuits in this logic layer to further improve the performance and energy efficiency of a DRAM chip without paying too much additional cost. Note that Elpida also expects such heterogeneous integration to provide more interesting functionality in the future [8].

Clearly, such a heterogeneous 3-D DRAM chip will enable many possible, interesting designs in the future. In this paper, we investigate a heterogeneous memory architecture as the first

step to exploit the opportunities for performance and energy. The motivation of our heterogeneous memory architecture is as follows. According to a recent study [35], although applications may have very good spatial locality, a conventional DRAM architecture cannot exploit such spatial locality because each bank relies only on a single row buffer. In other words, the row buffer is not large enough to capture such spatial locality due to conflicting row misses. As such, the current DRAM architecture will end up repeatedly opening the same row that was recently closed, making such operations not only redundant but also highly inefficient. Moreover, such conflict gets even more serious as the number of cores on a single die grows in the future [33]. Unlike a conventional single-core processor in which a single row per memory bank is good enough to capture the memory access pattern for a single application, multi-core processors will have many programs running concurrently sharing the same set of row buffers. Clearly, it is very likely that one application that runs on one core will close a row opened by another application running on another core before the latter application can fully utilize the row buffer.

Obviously, such redundant DRAM row open operations also consume DRAM energy considerably. To address the problem of such wasted energy, we need an associative SRAM cache in a DRAM chip to keep several active DRAM rows. Although such technique was considered in a conventional DRAM design, it failed to succeed in commodity DRAM that we will discuss in Section IV-A. We will then demonstrate how the feasibility of a row cache will change with 3-D-IC technology by studying the circuit-level design issues with detailed DRAM and TSV models. We will also address several circuit-level design challenges with a few low-cost architectural solutions.

A. Understanding Design Challenges of an SRAM Cache in a DRAM Process

First of all, we study why an SRAM cache in a conventional DRAM process was not successful. Such an SRAM cache design in a conventional DRAM process was studied by several prior literature and was even realized in ill-fated products that attempted to bridge the speed gap between DRAM and SRAM [2], [11], [21], [39]. All of them have demonstrated fairly impressive performance improvement. However, none of these techniques was successfully realized into mainstream products due to the following challenges.

- First, implementing a row cache requires long, high-bandwidth wires, which consume significant energy. (Note that such energy inefficiency was largely ignored in the past studies [21], [39].) In particular, because sense amplifiers of DRAM are distributed across a chip [see Fig. 2(b)], if one wants to implement a separate row cache in a planar DRAM die, a row-wide, global data bus between each set of the sense amplifiers and a separate row cache will be needed. Such a wide bus is claimed to be feasible in previous architectural studies [21], [39] because everything is on-chip, but it may require more metal layers. More importantly, it will consume a significant amount of energy. Such a high bus energy cost may be canceled out if an application has very good spatial locality so a lot of DRAM lookups are eliminated by the row cache, but it may also consume more energy if there is poor spatial locality [35]. Alternatively, we can implement a set-associative SRAM row cache right next to one row of sense amplifiers. Such a tightly integrated sense amplifier and row cache pair is helpful in reducing the wire complexity of a chip, but it necessitates replicating the set-associative SRAM row cache many times over a chip (redundantly).
- Second, unlike a logic process, which has one or two poly layers with many metal layers, a typical DRAM process has more poly layers with two metal layers [12], [19]. Only very recently, the state-of-the-art DRAM process started to use three metal layers [25]. In other words, with only two or three metal layers, the DRAM industry used them to implement all wires such as wordlines, bitlines, column select signals, local I/O lines, and global I/O lines. On the other hand, a typical high-performance 6 T SRAM cell itself requires two to three metal layers (Vdd and ground wires, wordlines, bitlines, and interconnection between transistors). Thus, to reasonably implement small SRAM cells in a DRAM die, we may need more masks than those in a conventional DRAM process. Furthermore, even with three metal layers, an SRAM cell (around $140F^2$ where F is a feature size) occupies much larger space than a DRAM cell (6 to $8F^2$) [1]. Due to area overhead, to redundantly place an SRAM row cache right next to each sense amplifier array is extremely expensive.
- Last, a DRAM process is typically optimized for reducing the leakage current of a storage capacitor. This goal is typically achieved by increasing the threshold voltage through substrate biasing [19]. Due to such higher threshold voltage, an SRAM cell implemented on a DRAM die is much slower unless it is specially manufactured.

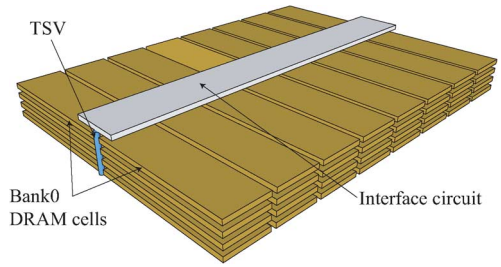

Fig. 3. Baseline 3-D chip (flipped).

B. Designing a Tightly Integrated SRAM and DRAM Stack with TSVs

1) Baseline: Recently, the emerging technology of 3-D die stacking and TSV has renewed the feasibility of integrating an SRAM row cache within a DRAM chip, which was not practical in terms of cost as discussed in the previous section. In this work, we re-investigate the new design issues and study how computer architects can overcome these prior unresolved challenges with a novel TSV-enabled heterogeneous 3-D DRAM chip. To perform this study, we first define our baseline 3-D DRAM design that consists of four DRAM dies and one logic die as shown in Fig. 3. Here, we assume that each DRAM

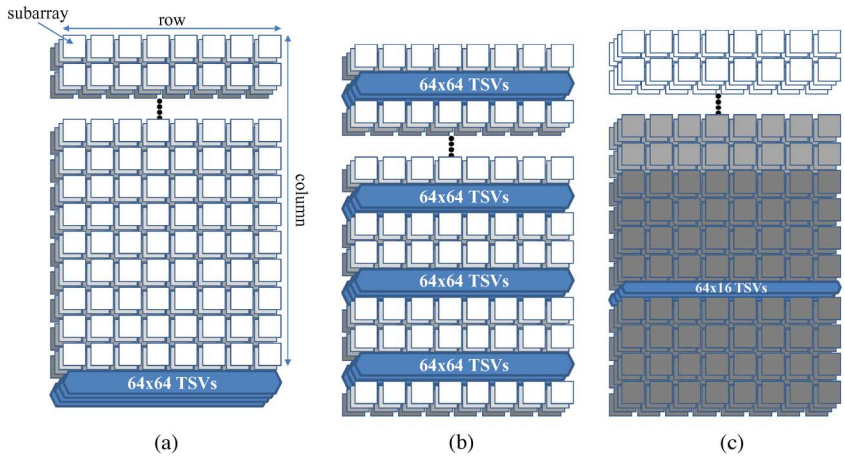

Fig. 4. Different 3-D design of four half-banks (not to scale; different colors mean different banks). (a) Naïve wide TSV Bu; (b) tightly integrated TSV bus; (c) folded bank.

die implements eight banks of DDR3 DRAM, similar to Samsung's implementation [13]. On the other hand, we assume a dedicated interface layer similar to Elpida's stack [14]. These DRAM layers and the interface layer are connected through a TSV bus located in the middle of a chip. In this baseline model, the data width of our baseline TSV bus is 64-bit same as our 2-D baseline [see Fig. 2(a)]. Note that we are not modeling our baseline with a more aggressive 3-D DRAM model such as folded wordline/bitline architecture from Tezzaron. Recently, based on Tezzaron's brief announcement, a few recent computer architecture papers assumed that we can fold wordlines or bitlines of each DRAM array into multiple dies, which results in significantly reduced DRAM access latency. However, we do not assume such a folded wordline/bitline architecture because we believe that it will harm the density of DRAM cells significantly. For example, if we want to fold bitlines into multiple layers connecting those pieces of the bitline with TSVs, the pitch of bitlines should be as large as the pitch of TSVs. Typically, the pitch of bitlines are extremely small (tens of nm) while that of TSVs are a few order of magnitude bigger (a few μm). If the bitline pitch should be extended to align a bitline and its corresponding TSV, DRAM cell density will be significantly reduced not to mention corresponding extended wordline length. To avoid this, one can potentially place a 2-D array of TSVs instead of a linear array of TSVs. However, in this case, we need complicated wiring between bitlines and TSVs, which significantly affects the performance of sense amplifiers.

2) Naïve Wide TSV Bus: Based on such a baseline model, first of all, we try to increase the data width up to the size of a row. To achieve this goal, we place a TSV bus per bank so that four banks stacked vertically can share the TSV bus as shown in Fig. 4(a). (Note that this figure only depicts four 256 Mb half-banks.) Below these four banks, we have an SRAM row cache in the logic layer. With this design, we can easily solve the last two challenges mentioned previously because the SRAM row cache is implemented with a logic process. However, such a naïve design cannot address the first challenge, an increased wire count and its corresponding energy inefficiency. To evaluate such a design (and other designs throughout our paper), we modified CACTI 5 [32]. Here, we did not modify its circuit-level models, but we replaced its H-tree model mainly designed for a cache memory with a bus model as explained in Section III as well as its array architecture model to reflect changes originated from our 3-D design. Here, we assume that DRAM is designed with 32-nm technology, and the TSV pitch is 3.6 μm in year 2013 [1]. (Note that this is a conservative model because we have already fabricated a 3-D stacked many-core processor with the TSV pitch of 2.5 μm in 2010 [10].) According to our modified CACTI, the energy consumption for reading the entire 8 Kb from an open row to the logic die is 63.5 times higher than that for reading 64 bits in our baseline [see Fig. 2(a)]. Although such a wide bus design does not need I/O gating circuits and its corresponding column select signals reducing energy consumption in these parts, it consumes a significant amount of energy in the bus between the sense amplifiers and TSVs.

3) Tightly Integrated TSV Bus: Alternatively, we want to bring the TSV bus closer to the sense amplifiers to solve the energy inefficiency issue. As shown in Fig. 4(b), we try to layout a TSV bus per row of subarrays and evaluate its feasibility. According to our analysis, the width of our subarray that consists of 512 rows and 512 columns is 49.6 μm. If we want to layout the number of TSVs in power-of-two along this subarray, we can place only eight TSVs along the width of one subarray. In other words, if we want to bring the entire 512 bits from this subarray at one DRAM cycle, we need 64 rows of eight TSVs per subarray. Unfortunately, the height of the 64 rows is 230.4 μm while that of one subarray cells is only 32.8 μm. Even if we assume that two subarrays can share these buses, this TSV overhead is prohibitively high. Note that such high overhead is mainly because our DRAM subarray is already very small and we want to align a wide bus with such a small subarray to address the energy inefficiency issue mentioned earlier.

4) Folded Bank: From these two designs, we learn that we have tradeoff between dynamic energy consumption and area overhead. To address this tradeoff, we propose to make subarrays of one bank to share the same set of TSVs similar to our first design but to fold each bank vertically (instead of stacking four banks vertically) as shown in Fig. 4(c). By folding each bank,

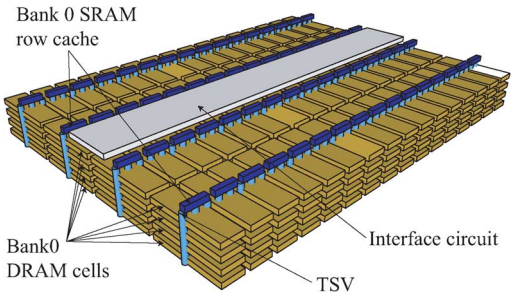

Fig. 5. Final floorplan (flipped).

we can reduce the length of wires between the sense amplifiers and the TSV bus. Here, note that only one layer of a bank actively uses the TSV bus simultaneously because each layer has a different set of rows that belong to the same bank. Also note that this is a scalable design. As the number of DRAM layers increases in the future, we can fold one bank into more layers, which reduces the wire length of each die. In addition to such a folded design, we also placed the TSV bus in the middle of a bank to further reduce the wire length.

Furthermore, we carefully calculated the wire complexity so that our design does not need more metal layers than our baseline. In our baseline ($\times 8$ DRAM) design, eight subarrays of a half-bank [see Fig. 2(a)] form a half row (4 Kb) while we fetch 32 bit data from the half-bank. In other words, we fetch 4-bit data from each subarray, which has 512 columns. Consequently, the required number of column select signals is $128 (= 512/4)$. On the other hands, to allow massive data transfer, our design uses 128 wires between the sense amplifier and the TSV bus. As a result, we need four column select signals between the column decoder and each subarray. By using this design, we can equalize the number of wires between the baseline and our design. However, due to such a narrower bus design, the TSV bus width of each half-bank is now 1 kb (128 wires from each subarray). This reduced bus bandwidth forces us to fetch one half row (4 kb) over four cycles. This circuit-level limitation necessitates an architectural technique, which will be detailed later. As a result of such optimization, we are able to design a well-balanced stacked DRAM layers. The detailed results and analysis will be discussed in Section V.

Our final floorplan based on such a folded bank architecture is shown in Fig. 5. As shown in the figure, we align each bank of our SRAM row cache with its corresponding DRAM bank. By placing an SRAM bank right next to TSVs, we minimize the energy consumed in transferring an entire row to an SRAM bank. Note that if an application does not consume entire row data, the energy consumed to transfer those unused bits to the SRAM row cache is completely wasted because our baseline does not bring those data to the interface circuits at all. This is why we want to minimize the wire length of the bus between the sense amplifiers and the SRAM row cache.

On the other hand, such floorplan necessitates long, cross-chip communication between an SRAM bank and the interface circuits. Note that this long bus is comparable to the bus between the sense amplifiers and the interface circuits in our baseline. In spite of such a long bus between the SRAM row cache and the interface logic, we opted for such design because the bus between the SRAM bank and the interface circuits is just 64 bit, which does not consume much energy and is anyway used by demand requests.

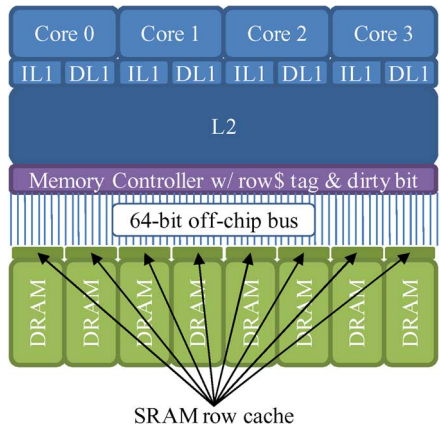

Fig. 6. New memory hierarchy.

C. Heterogeneous Memory Management

With such heterogeneous DRAM chips, in this subsection, we will study how to manage them efficiently. Our new memory hierarchy is shown in Fig. 6. In which, each DRAM chip has a very small SRAM row cache implemented with a CMOS logic process. The interconnection between the SRAM row cache and DRAM cells is based on high-bandwidth TSVs that provide entire row data across four memory bus cycles. On the other hand, the SRAM row cache is connected to a memory controller in the CPU package through a conventional 64-bit off-chip bus. Note that, if we could bring one entire row closer to the CPU, e.g., to the L2 cache, we could better exploit its spatial locality without suffering from the off-chip delay. However, such an approach can also seriously degrade system performance due to limited pin count and the trailing-edge effect of large transfer over a conventional memory interface [35]. Such an aggressive memory hierarchy can be feasible only if the system memory is stacked atop of a CPU, which could be an issue for certain high-end products due to poor thermal conductivity when too many layers are stacked. This is the major reason of having an SRAM row cache as a more reasonable solution to achieving high performance and energy efficiency for high-end products.

In such a new memory hierarchy, we need a new memory controller that understands such heterogeneity and manages cache eviction intelligently to improve performance and energy efficiency. Interestingly, we can achieve this goal easily with a simple memory scheduling policy, first-ready first-come-first-serve (FR-FCFS) [28], [29]. If we place a tag array for the SRAM row cache in the memory controller [21], the FR-FCFS scheduling policy can easily detect row cache hits locally and schedule memory commands.

For memory scheduling, we need a new set of commands. In the case of row cache hits, our memory controller sends a new command that notifies the DRAM chips to look up their

[2]Note that, while our tag array is fully-associative, its corresponding SRAM row cache is a directly-mapped data array that is indexed by the index address provided by the memory controller.

own SRAM row cache. For this new command, we need a new control pin, index address strobe (IAS), in addition to the conventional row address strobe (RAS), and column address strobe (CAS) pins. In the case of cache hits, we enable IAS and send cache index bits[2] immediately followed by CAS signals and column addresses to read or write data in the SRAM row cache. In the case of cache misses, the memory controller needs to manage cache eviction as well. If a row to be evicted is clean, the memory controller can simply send IAS and RAS along with a row address to be cached. In order to perform an SRAM cache fill operation over four cycles as mentioned previously, we use the burst length of four to fetch an entire row serially and effectively. On the other hand, if a row to be evicted is dirty, the memory controller has two choices. One obvious solution is to evict a dirty row first and then to open a new row later. This is a conventional way to open a new row when we have another row already opened. On the other hand, in such a heterogeneous DRAM chip, we can potentially move a dirty cached row to a small, single-entry write-back buffer temporarily, open a new row, and then write-back the dirty row later. The cost of this write-buffer in terms of area is also small given our additional CMOS logic layer. In this paper, we call this write-back policy delayed write-back, and we will evaluate performance and energy efficiency of the delayed write-back design later. For such delayed write-back, the memory controller sends IAS, RAS, and CAS altogether to notify a chip to evict a row to the write-back buffer and open a new row. The delayed write-back will be initiated with IAS and CAS later with the row address of the dirty row. As a result of these new command modes and the multiplexed address scheme, we only need one extra pin in the DRAM chips. Once a new row is opened, the memory controller precharges bitlines immediately similar to the operations in a conventional closed-row policy.

V. Evaluation

A. Circuit-Level Evaluation

For circuit-level modeling, we modified the DRAM model of CACTI 5 [32] as explained in Section IV-B. With our modified CACTI, we modeled area, delay, and dynamic energy of DRAM. We also modeled the latency and power consumption of TSVs similar to a recent study [17]. Note that the current version of CACTI does not have an accurate DRAM leakage model, so we failed to model it. However, we believe that our proposal can reduce leakage energy consumption as our proposed memory architecture significantly improves the overall performance (will be shown later) with very minimal extra hardware that will contribute little leakage current compared to our baseline's large DRAM arrays. In the later part of this section, we will perform a conservative study only with the leakage energy overhead for our new SRAM caches.

First of all, we evaluate the area overhead of our scheme. In this evaluation, we have included the area overhead of a grid of power delivery TSVs [9] for both the baseline and our proposal. Note that, unless otherwise mentioned, we have placed an array of clustered TSVs per grid point where the cross section area of clustered TSVs is $20\ \mu\text{m} \times 20\ \mu\text{m}$ and the grid size is $436\ \mu\text{m} \times 331\ \mu\text{m}$, which is the area occupied by 8×8 subarray. Such density is similar to a recent study proposed for reducing power noice of many-tier 3-D systems [9]. From this evaluation, we found that the area overhead of our proposal (see Fig. 5) (in DRAM layers) is 6%.

TABLE I
tRCD Breakdown

	Baseline	Folded bank
inter-bank address bus	20%	30%
intra-bank address bus	39%	9%
row decoder / wordline	17%	17%
bitline / sense amplifier	24%	24%
total	100%	82%

TABLE II
tCL Breakdown

	Baseline	Folded bank
inter-bank address bus	14%	24%
column select	30%	9%
I/O gating / output driver	14%	2%
intra-bank data-out bus	28%	5%
inter-bank data-out bus	14%	0%
total	100%	40%

To understand the difference, we performed an in-depth analysis and found that the TSV area accounts for most of this overhead. Other than the area occupied by TSVs, we observed minor differences in various components such as the output drivers and column select signal related wires and circuits. These differences are negligibly small compared to the TSV area overhead.

Despite our slightly larger die, we found that our proposed design can actually decrease the access latency of DRAM. In particular, we found that the row-to-column delay (tRCD) and the row precharge delay (tRP) are reduced by 18% and 14%, respectively. Such reduced delay is found to be the result of the reduced wire length within a bank due to the folded bank architecture. This effect is well represented in Table I. As shown in the table, our folded bank architecture suffers from longer inter-bank bus latency because our new floorplan (see Fig. 5) now has 16×4 half-banks instead of 8×2 half-banks of our baseline (see Fig. 3), which makes the worst-case inter-bank bus wire longer. However, intra-bank bus latency is found to be reduced significantly because one half-bank is folded across four layers.

On the other hand, the column access strobe latency (tCL) was significantly reduced. As shown in Table II, the latency of the inter-bank address bus increases, but the latencies of the column select bus and the intra-bank data-out bus decrease due to our new floorplan. Such trend is similar to tRCD. However, one interesting result is that the inter-bank data-bus delay of our new design is zero. This is because our SRAM cache is located right next to the TSV bus storing the entire row data. Note that we suffer this latency when we read data from our SRAM cache.

Although we have a win in the access latency, our design consumes a significant amount of energy in moving the row data between the sense amplifiers and the SRAM cache. Table III

TABLE III
READ ENERGY BREAKDOWN

	Baseline	Folded bank
inter-bank address bus	8%	13%
column select	58%	1%
I/O gating / output driver	4%	64%
intra-bank data-out bus	25%	741%
inter-bank data-out bus	5%	0%
total	100%	818%

shows the relative energy that we consume when we read 64-bit data in the baseline and an entire row in the folded bank architecture. Note that here we bring one row over four DRAM cycles in the folded bank architecture. Nonetheless, this higher energy consumption of moving rows do not occur frequently in the dynamic scenario as the majority of the accesses will be satisfied by the SRAM cache. We will detail the results later. Similar to the read energy, we found that the write energy of the folded bank architecture is 4.6 times higher than that of the baseline. On the other hand, we found that activation energy increases by 18% mainly due to the inter-bank bus energy in our new floorplan.

So far, we have assumed that TSV pitch is 3.6 μm, which is the ITRS prediction of year 2013 [1]. Furthermore, we studied how these results vary as TSV pitch changes. For this study, we varied the TSV pitch from 2.5 to 10 μm. Note that we have already fabricated a 3-D stacked many-core processor with the TSV pitch of 2.5 μm in 2010 [10]. As shown in Fig. 7(a) and (b), neither latency nor energy is sensitive to the TSV pitch.

However, area overhead is reasonably sensitive to the TSV pitch [see Fig. 7(c)]. As shown in the figure, as the TSV pitch increases from 2.5 to 10.0 μm, area overhead increases from 5% to 14% when we uses a 20 μm $\times$ 20 μm power delivery TSV cluster per power delivery grid point. Note that the reason why it does not monotonically increase is that CACTI explored a large design space to evaluate the multiple design goals such as area, latency, and energy and ended up with different array configurations automatically [32]. (For example, when we used the 8 μm TSV pitch, the degree of two-level sense-amplifier multiplexing was 2/2. On the other hand, when we used 9 μm TSV pitch, it was 4/1.) To further understand this overhead, we also performed a set of studies with a larger power delivery overhead (80 $\times$ 80). Not surprisingly, as the larger power delivery TSVs penalize both the baseline and our proposal, the additional TSV area overhead of our proposal has relatively reduced. As the results show, because DRAM is sensitive to the manufacturing cost, smaller TSV geometry is found to be a critical enabler for future 3-D stacked DRAM designs. In the following section where we evaluate the performance and energy benefit of our proposal, we will use our original TSV pitch of 3.6 μm because latency and energy are turned out to be insensitive to TSV pitch.

B. Architecture-Level Simulation

1) Simulation Framework: In addition to such circuit-level modeling, we also performed architecture-level study using

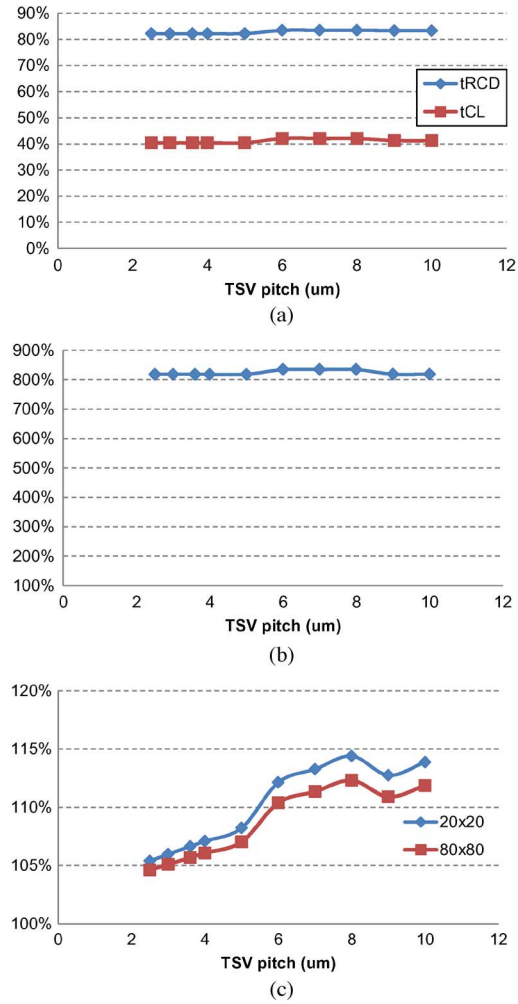

Fig. 7. Sensitivity study for different TSV pitches. (a) Relative tRCD/tCL; (b) relative read energy; (c) relative area.

SESC [27]. We extended the SESC simulator to model detailed memory backend. In our simulation, we modeled a quad-core processor with one memory channel. We know that a topline multi-core processor will have more cores and more memory channels, but simulating such a large number of cores takes days or longer to simulate. Hence, we proportionally scaled down the number of cores and that of memory channels to a configuration that we believe to be reasonable, a quad-core processor with one memory channel. Our quad-core processor has a 4 MB L2 cache and a DDR3-1600 like memory interface. When we model the DRAM latency, we used the estimated latency from our modified CACTI. When we model SRAM latency, we also considered the round-trip time between the interface circuit and one SRAM bank, which accounted for the major portion of the access latency. Note that, compared to the round-trip latency, the SRAM access latency is negligibly small because of its small size. Detailed parameters are listed in Table IV.

Throughout this paper, we simulated multi-programmed workload that consists of four memory-intensive applications from the SPEC2006 benchmark suite. We defined a memory-intensive application as an application whose number of L2 cache misses per thousand instructions (MPKI) is higher than five when it runs on a single-core processor with a 1 MB L2 cache.

TABLE IV
PROCESSOR CONFIGURATIONS

Core	3.0 GHz, 14-stage, OoO, 4-wide fetch/issue/retire out-or-order quad-core processor	
ROB size	192	
Physical register file size	128 (INT) / 128 (FP)	
Branch predictor	Hybrid (16K global / local / meta tables), 2K BTB, 32-entry RAS	
L1 I cache	2-port 2-way, 64B-line, 32KB, LRU, 1-cycle, 8-entry MSHR	
L1 D cache	2-port 4-way, 64B-line, 32KB write-back, LRU, 2-cycle latency, 1-cycle throughput, 8-entry MSHR	
Shared L2 cache	2-port 8-way, 64B-line, 4MB inclusive write-back, LRU, 6-cycle latency, 1-cycle throughput, 8-entry MSHR	
Memory	FR-FCFS scheduling policy, 8B-wide bus, DDR3-1600	
	DRAM	tCL-tRCD-tRP: 7-9-8, tRAS: 35 ns, tWR: 15 ns
	SRAM + DRAM	tCL-tRCD-tRP: 5-4-7, tRAS: 35 ns, tWR: 15 ns, SRAM access latency: 4 bus clk

TABLE V
SIMULATION WORKLOAD

Workload	Applications	Workload	Applications
MIX0	429.mcf, 450.soplex, 462.libquantum, 473.astar	MIX1	433.milc, 459.GemsFDTD, 471.omnetpp, 473.astar
MIX2	433.milc, 450.soplex, 471.omnetpp, 473.astar	MIX3	410.bwaves, 429.mcf, 433.milc, 471.omnetpp
MIX4	433.milc, 462.libquantum, 471.omnetpp, 483.xalancbmk	MIX5	462.libquantum, 471.omnetpp, 482.sphinx3, 483.xalancbmk
MIX6	429.mcf, 459.GemsFDTD, 473.astar, 482.sphinx3	MIX7	410.bwaves, 462.libquantum, 471.omnetpp, 483.xalancbmk
MIX8	429.mcf, 450.soplex, 462.libquantum, 471.omnetpp	MIX9	410.bwaves, 429.mcf, 433.milc, 450.soplex
MIX10	410.bwaves, 429.mcf, 462.libquantum, 471.omnetpp	MIX11	429.mcf, 450.soplex, 462.libquantum, 482.sphinx3

From our profiling runs, we found that 10 SPEC 2006 applications meet this criteria, and we randomly selected 12 sets of four applications using our custom random number generation code. These groups of workload are listed in Table V. Note that the reason why we selected multi-programmed workload is to evaluate the benefit of our proposal in the datacenter computing environment where many virtual machines are sharing a physical multi-core processor maximizing the utilization of a system. As clearly stated in Section I, we believe that energy-efficient DRAM is especially needed in such a highly utilized system.

2) Performance Evaluation: With these sets of workload, we first evaluated the performance improvement achieved with different DRAM chip designs. In this evaluation, we simulated five designs. The first two designs have a conventional DRAM system with an open-row policy and a closed-row policy, respectively. The reason why we simulated the closed-row policy is to compare the closed-row policy against our scheme that can precharge bitlines right after a row is opened. On the other hand, we also evaluated our heterogeneous 3-D DRAM chip design where each bank maintains an SRAM row cache that can hold 8, 16, and 32 rows.[3] The speedup of these different configurations is shown in Fig. 8(a). Here, the speedup is defined as the improved system throughput. Also note that, throughout this paper, all reported relative numbers are normalized to that of a conventional DRAM system with an open-row policy, unless otherwise mentioned. First of all, we found that the open-row policy is still useful compared to the closed-row policy. Such effect is well represented in Fig. 8(b), which shows the hit rate of a row buffer. As shown in this figure, the hit rate of a DRAM system with the open-row policy is 53%, on average. Not surprisingly, as we increase the capacity of our SRAM row cache, the hit rate increases [see Fig. 8(b)] leading to higher performance [see Fig. 8(a)]. For example, a heterogeneous 3-D DRAM chip with a 32-entry SRAM row cache can improve performance by 30%, on average. Such improvement was observed by past studies [6], [11], [21], [39], and we just confirm that a row cache is still useful even if we have multiple cores that compete with the shared row cache space. One interesting observation is that, as we increase the capacity of a row cache, the hit rate was increased by 5% (on average) while its corresponding performance is only improved by 2%. This effect suggests that a small row cache is sufficient in improving the overall system performance.

In addition to those five configurations, we also evaluated three heterogeneous DRAM chip designs without using the delayed write-back policy as shown in Fig. 9. We found that the delayed write-back policy is useful in improving the overall system performance. For example, it improves the system performance by around 4% for a 32-entry SRAM row cache.

3) Energy Efficiency Evaluation: Moreover, we also evaluated the energy efficiency of our proposed architecture. Fig. 10 represents relative dynamic energy consumed during DRAM array lookup operations (including activation, read, write, and precharge energy). As shown in the figure, we can significantly reduce the energy consumed in DRAM lookup operations. On average, a DRAM chip with a 32-entry SRAM row cache consumes only 35% of energy of our baseline system, a DRAM chip with the open-row policy. This result suggests that, even though our new 3-D DRAM design consumes significantly higher read or write energy (see Section V-A), our SRAM row cache can filter out lots of DRAM lookup operations [see Fig. 8(b)], therefore, the overall DRAM lookup energy is significantly reduced.

However, such reduced DRAM lookup energy does not come for free; we are also spending energy in other additional circuits

[3]In each chip, the capacity of one DRAM bank is 512 Mb while that of a 32-entry row cache is 256 kb. Instead of making the line size of this row cache a row size, we split this cache into multiple banks so that the line size of each cache is just 64 bit, which is the access unit from the interface circuit.

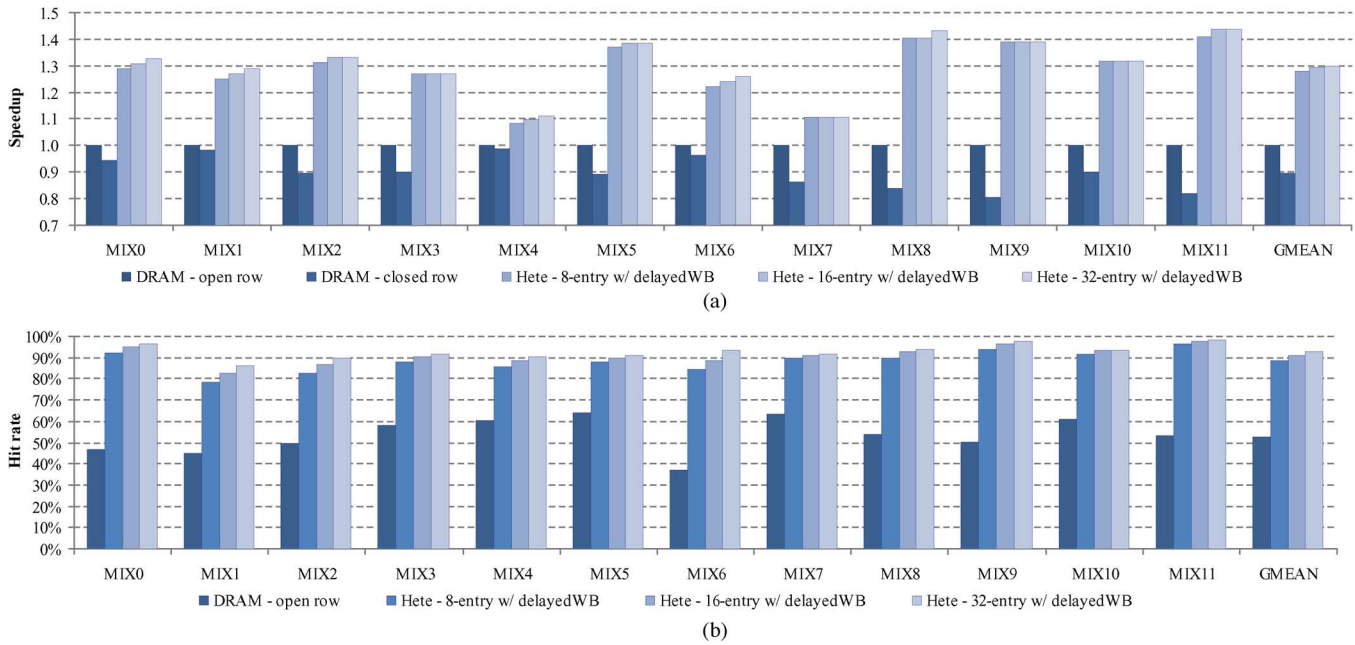

Fig. 8. Speedup and row hit rate. (a) Speedup; (b) row hit rate.

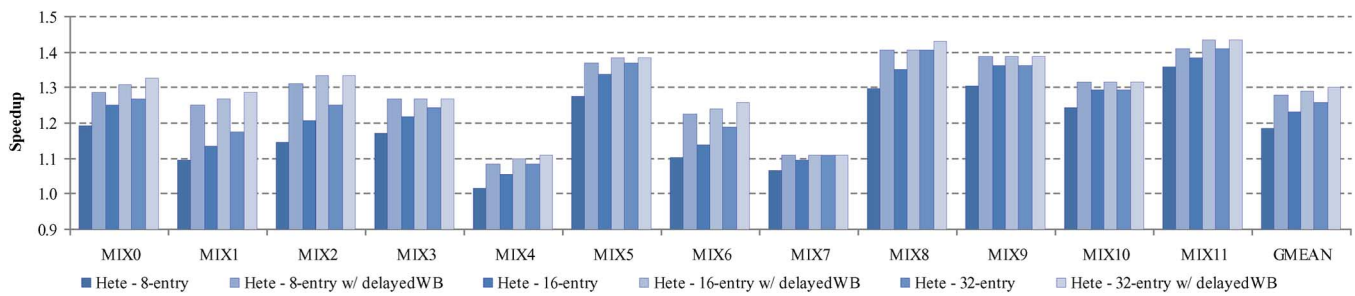

Fig. 9. Effect of the delayed write-back policy.

of our proposed scheme. Thus, to model dynamic energy consumption of the entire chip, we also modeled energy consumption of SRAM lookup operations, TSVs, the delayed write-back buffer, and refresh operations as shown in Fig. 11. This evaluation suggested that a heterogeneous DRAM chip with a 32-entry SRAM row cache can save dynamic energy of a DRAM chip by 31%, on average. Furthermore, we observed that, as we increase the capacity of a row cache, a DRAM chip gets more energy efficient. Unlike performance, which is not significantly improved as we increase the capacity of a row cache, the energy efficiency can be significantly improved. As shown in the figure, as the capacity of the row cache increases from eight entries to 32, the dynamic energy saving increases from 5% to 31%. We believe that such discrepancy between the performance and the energy efficiency is partly originated from our microarchitecture, which already employs several techniques that make the overall performance less sensitive to the memory access latency.

Fig. 12 represents the dynamic energy breakdown of a heterogeneous 3-D DRAM chip with a 32-entry SRAM row cache. Note that 100% represents the dynamic energy consumption of our baseline DRAM chip with the open-row policy. As shown, the TSV energy and the delayed write-back buffer contribute very little energy while DRAM lookup, SRAM lookup, and refresh operations consume significant energy. From this result, we were wondering why a larger SRAM cache is still helpful in improving energy efficiency. To answer this question, we analyzed our energy numbers in detail and made an interesting observation that most of SRAM lookup energy is spent in the on-chip bus between interface circuit and an SRAM bank. Because our SRAM is so small, the SRAM lookup energy accounts for only 2% of SRAM energy consumption. The other 98% was consumed in the address bus and data bus, which are not sensitive to the capacity of our SRAM row cache. From this observation, we concluded that filtering out more DRAM lookup operations with a larger SRAM cache is more helpful. Furthermore, we also found that, if we increase the number of dies from four to eight, for example, we can reduce the wire length of these buses, saving more energy in the DRAM chips. However, we did not evaluate this scheme because having more layers than our baseline may have a undesirable adverse effect to the overall manufacturing cost.

Up to this point, we evaluated every energy overhead induced by our proposal except the leakage energy consumption of the SRAM row cache. Unfortunately, we do not have a concrete leakage energy model for DRAM, thus we cannot perform a fair evaluation. However, we believe that, as we improve the overall

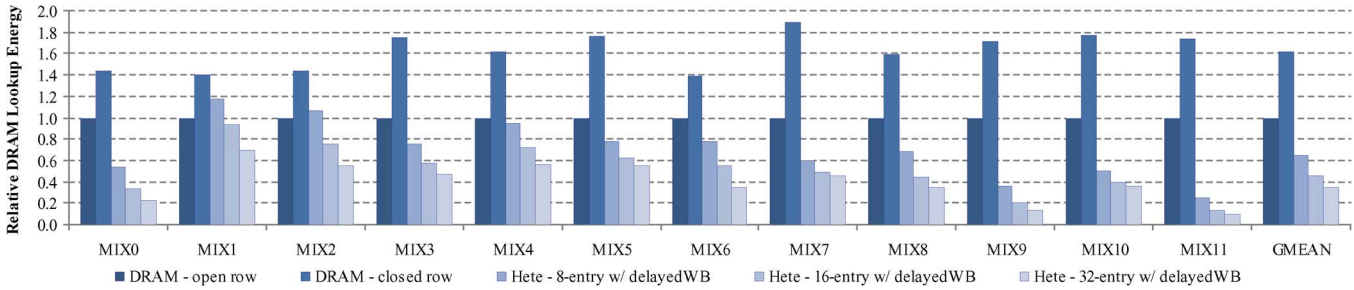

Fig. 10. Relative DRAM lookup energy.

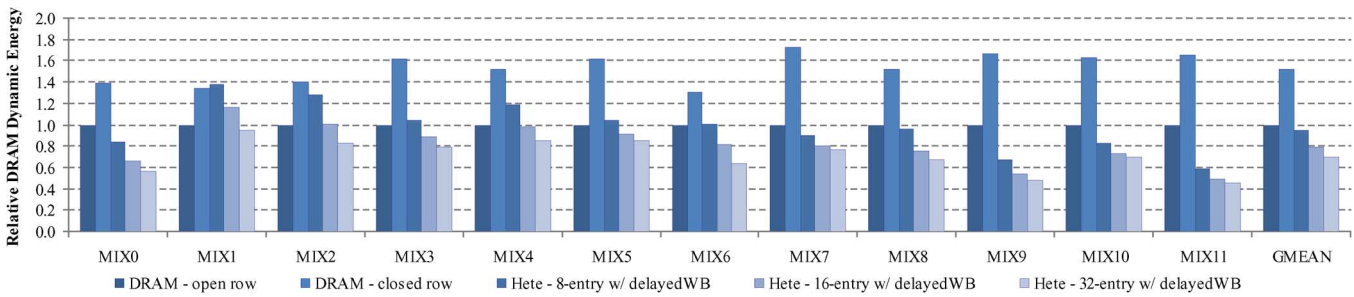

Fig. 11. Relative dynamic energy of an entire DRAM chip.

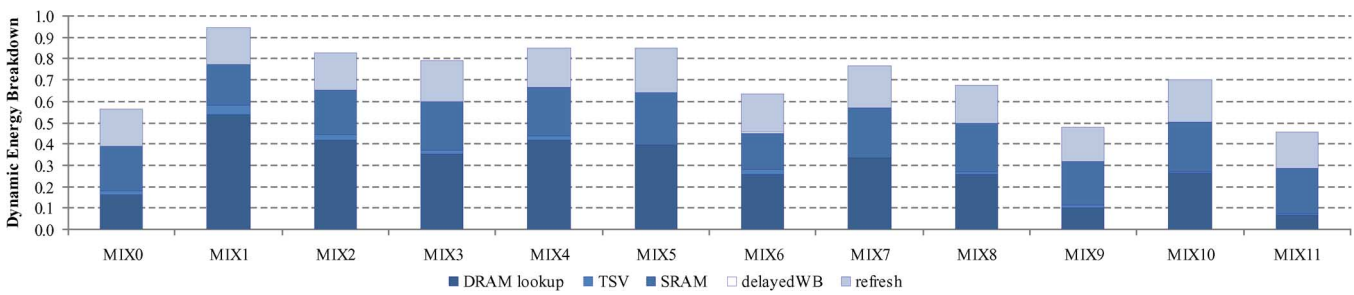

Fig. 12. Dynamic energy breakdown of a heterogeneous DRAM chip (with delayed write-back policy).

system performance, we can save lots of leakage energy not only from DRAM chips but also from all system components including the CPU die [22]. Nevertheless, here we performed a very conservative study by simply adding the leakage overhead of the SRAM cache only. From this conservative study, on average, our heterogeneous 3-D DRAM chips with 8-, 16-, and 32-entry SRAM row caches save energy by 1%, 14%, and 17%, respectively.

VI. Conclusion

As the DRAM industry starts to revolutionize the conventional planar DRAM design with heterogeneous 3-D stacking technology that integrates DRAM and logic on a single die package, it is also a timely moment for computer architects to contemplate about how to take advantage of these new enabling technologies for improving the DRAM architecture and the overall memory hierarchy. In this paper, we proposed a TSV-enabled, energy-efficient SRAM row cache that is tightly integrated with its corresponding 3-D DRAM array. To evaluate our proposal, we studied its feasibility from the circuit perspective as well as the architectural perspective. From our circuit-level study, we found several novel design challenges in dealing with density and energy efficiency issues for our heterogeneous 3-D DRAM architecture. We addressed these issues by proposing a novel floorplan and several architectural techniques. Our evaluation with memory intensive applications shows that well-balanced heterogeneous 3-D DRAM chips can improve system performance by 30% while saving dynamic energy by 31%, on average.

References

[1] "International Technology Roadmap for Semiconductors," 2007. [Online]. Available: http://public.itrs.net

[2] K. Arimoto, M. Asakura, H. Hidaka, Y. Matsuda, and K. Fujishama, "A circuit design of intelligent cache DRAM with automatic write-back capability," *IEEE J. Solid-State Circuits*, vol. 26, no. 4, pp. 560–565, Apr. 1991.

[3] R. Baker, *CMOS: Circuit Design, Layout, and Simulation*. New York: Wiley-IEEE Press, 2007.

[4] L. Barroso, "The price of performance," *Queue*, vol. 3, no. 7, p. 53, 2005.

[5] B. Black, M. Annavaram, N. Brekelbaum, J. DeVale, L. Jiang, G. H. Loh, D. McCaule, P. Morrow, D. W. Nelson, D. Pantuso, P. Reed, J. Rupley, S. Shankar, J. Shen, and C. Webb, "Die stacking (3D) microarchitecture," in *Proc. Int. Symp. Microarch.*, 2006, pp. 469–479.

[6] V. Cuppu, B. Jacob, B. Davis, and T. Mudge, "A performance comparison of contemporary DRAM architectures," in *Proc. Int. Symp. Comput. Arch.*, 1999, p. 0222.

[7] J. Dorsch, "Does Moore's law still hold up?," EDA Vision, Nov. 2001. [Online]. Available: http://www.edavision.com/200111/feature.pdf

[8] Elpida, Tokyo, Japan, "Elpida completes development of Cu-TSV (Through Silicon Via) multi-layer 8-Gigabit DRAM," 2009. [Online]. Available: http://www.elpida.com/pdfs/pr/2009-08-27e.pdf
[9] M. Healy and S. Lim, "Power delivery system architecture for many-tier 3d systems," in *Proc. IEEE 60th Electron. Components Technol. Conf. (ECTC)*, 2010, pp. 1682–1688.
[10] M. B. Healy, K. Athikulwongse, R. Goel, M. M. Hossain, D. H. Kim, Y.-J. Lee, D. L. Lewis, T.-W. Lin, C. Liu, M. Jung, B. Ouellette, M. Pathak, H. Sane, G. Shen, D. H. Woo, X. Zhao, G. H. Loh, H.-H. S. Lee, and S. K. Lim, "Design and analysis of 3d-maps: A many-core 3d processor with stacked memory," in *Proc. IEEE Custom Integr. Circuits Conf. (CICC)*, 2010, pp. 1–4.
[11] H. Hidaka, Y. Matsuda, M. Asakura, and K. Fujishima, "The cache DRAM architecture: A DRAM with an on-chip cache memory," *IEEE Micro*, vol. 10, no. 2, pp. 14–25, 1990.
[12] K. Itoh, *VLSI Memory Chip Design*. New York: Springer, 2001.
[13] U. Kang, H.-J. Chung, S. Heo, S.-H. Ahn, H. Lee, S.-H. Cha, J. Ahn, D. Kwon, J. H. Kim, J.-W. Lee, H.-S. Joo, W.-S. Kim, H.-K. Kim, E.-M. Lee, S.-R. Kim, K.-H. Ma, D.-H. Jang, N.-S. Kim, M.-S. Choi, S.-J. Oh, J.-B. Lee, T.-K. Jung, J.-H. Yoo, and C. Kim, "8 Gb 3-D DDR3 DRAM using through-silicon-via technology," *IEEE J. Solid-State Circuits*, vol. 45, no. 1, pp. 111–119, Jan. 2010.
[14] M. Kawano, S. Uchiyama, Y. Egawa, N. Takahashi, Y. Kurita, K. Soejima, M. Komuro, S. Matsui, K. Shibata, J. Yamada, M. Ishino, H. Ikeda, Y. Saeki, O. Kato, H. Kikuchi, and T. A. Mitsuhashi, "A 3D packaging technology for 4 Gbit stacked DRAM with 3 Gbps data transfer," in *Proc. Int. Electron Devices Meet.*, 2006, pp. 1–4.
[15] B. Keeth and R. Baker, *DRAM Circuit Design: A Tutorial*. Piscataway, NJ: Wiley-IEEE Press, 2000.
[16] K. Kilbuck, "Main memory technology direction," presented at the Microsoft Windows Hardw. Eng. Conf., Los Angeles, CA, 2007.
[17] D. Kim and S. Lim, "Through-silicon-via-aware delay and power prediction model for buffered interconnects in 3D ICs," in *Proc. 12th ACM/IEEE Int. Workshop Syst. Level Interconnect Prediction*, 2010, pp. 25–32.
[18] K. Kim, "Technology for sub-50 nm DRAM and NAND flash manufacturing," in *IEDM Tech. Dig*, 2005, pp. 323–326.
[19] Y.-B. Kim and T. Chen, "Assessing merged DRAM/logic technology," *Integr., VLSI J.*, vol. 27, no. 2, pp. 179–194, 1999.
[20] D. Klein, "The future of memory and storage: Closing the gaps," presented at the Microsoft Windows Hardw. Eng. Conf., Los Angeles, CA, 2007.
[21] R. Koganti and G. Kedem, "WCDRAM: A fully associative integrated cached-DRAM with wide cache lines," presented at the 4th IEEE Workshop Arch. Implementation of High Perform. Commun. Subsyst., Mystic, CT, 1997.
[22] H.-H. S. Lee, J. B. Fryman, A. U. Diril, and Y. S. Dhillon, "The elusive metric for low-power architecture research," presented at the Workshop Complexity-Effective Design, San Diego, CA, 2003.
[23] G. H. Loh, "3D-stacked memory architectures for multi-core processors," in *Proc. Int. Symp. Comput. Arch.*, 2008, pp. 453–464.
[24] G. H. Loh, "Extending the effectiveness of 3D-stacked DRAM caches with an adaptive multi-queue policy," in *Proc. Int. Symp. Microarch.*, 2009, pp. 201–212.
[25] Y. Moon, Y.-H. Cho, H.-B. Lee, B.-H. Jeong, S.-H. Hyun, B.-C. Kim, I.-C. Jeong, S.-Y. Seo, J.-H. Shin, S.-W. Choi, H.-S. Song, J.-H. Choi, K.-H. Kyung, Y.-H. Jun, and K. Kim, "1.2 V 1.6 Gb/s 56 nm 6F2 4 Gb DDR3 SDRAM with hybrid-I/O sense amplifier and segmented sub-array architecture," in *Proc. IEEE Int. Solid-State Circuits Conf.*, 2009, pp. 128–129.
[26] C. Park, H. Chung, Y.-S. Lee, J. Kim, J. Lee, M.-S. Chae, D.-H. Jung, S.-H. Choi, S. Young Seo, T.-S. Park, J.-H. Shin, J.-H. Cho, S. Lee, K.-W. Song, K.-H. Kim, J.-B. Lee, C. Kim, and S.-I. Cho, "A 512-mb DDR3 SDRAM prototype with C IO minimization and self-calibration techniques," *IEEE J. Solid-State Circuits*, vol. 41, no. 4, pp. 831–838, Apr. 2006.
[27] J. Renau, B. Fraguela, J. Tuck, W. Liu, M. Prvulovic, L. Ceze, S. Sarangi, P. Sack, K. Strauss, and P. Montesinos, "SESC Simulator," Jan. 2005 [Online]. Available: http://sesc.sourceforge.net
[28] S. Rixner, "Memory controller optimizations for web servers," in *Proc. Int. Symp. Microarch.*, 2004, pp. 355–366.
[29] S. Rixner, W. Dally, U. Kapasi, P. Mattson, and J. Owens, "Memory access scheduling," in *Proc. Int. Symp. Comput. Arch.*, 2000, pp. 128–138.
[30] Samsung Semiconductor, Hwasung-City, Gyeonggi-Do, Korea, "Package Information," 2010. [Online]. Available: http://www.samsung.com/global/business/semiconductor/support/ PackageInformation/index.html
[31] G. Sun, X. Dong, Y. Xie, J. Li, and Y. Chen, "A novel architecture of the 3D stacked MRAM L2 cache for CMPs," in *Proc. Int. Symp. High Perform. Comput. Arch.*, 2009, pp. 239–249.
[32] S. Thoziyoor, N. Muralimanohar, J. Ahn, and N. Jouppi, "CACTI 5.1," HP Laboratories, Palo Alto, CA, 2008.
[33] A. Udipi, N. Muralimanohar, N. Chatterjee, R. Balasubramonian, and N. Jouppi, "Rethinking DRAM design and organization for energy-constrained multi-cores," in *Proc. Int. Symp. Comput. Arch.*, 2010, pp. 175–186.
[34] D. H. Woo, N. H. Seong, and H.-H. S. Lee, "Heterogeneous die stacking of sram row cache and 3-d dram: An empirical design evaluation," in *Proc. IEEE 54th IEEE Int. Midw. Symp. Circuits Syst. (MWSCAS)*, 2011, pp. 1–4.
[35] D. H. Woo, N. H. Seong, D. L. Lewis, and H.-H. S. Lee, "An optimized 3D-stacked memory architecture by exploiting excessive, high-density TSV bandwidth," in *Proc. Int. Symp. High Perform. Comput. Arch.*, 2010, pp. 429–440.
[36] X. Wu, J. Li, L. Zhang, E. Speight, R. Rajamony, and Y. Xie, "Hybrid cache architecture with disparate memory technologies," in *Proc. Int. Symp. Comput. Arch.*, 2009, pp. 34–45.
[37] C. Yoo, K. Kyung, K. Lim, H. Lee, J. Chai, N. Heo, D. Lee, and C. Kim, "A 1.8-V 700-Mb/s/pin 512-Mb DDR-II SDRAM with on-die termination and off-chip driver calibration," *IEEE J. Solid-State Circuits*, vol. 39, no. 6, pp. 941–951, Jun. 2004.
[38] W. Zhang and T. Li, "Exploring phase change memory and 3D die-stacking for power/thermal friendly, fast and durable memory architectures," in *Proc. Int. Conf. Parall. Arch. Compilation Tech.*, 2009, pp. 101–112.
[39] Z. Zhang, Z. Zhu, and X. Zhang, "Cached DRAM for ILP processor memory access latency reduction," *IEEE Micro*, vol. 21, no. 4, pp. 22–32, Apr. 2001.
[40] B. Zhao, Y. Du, Y. Zhang, and J. Yang, "Variation-tolerant non-uniform 3D cache management in die stacked multicore processor," in *Proc. Int. Symp. Microarch.*, 2009, pp. 222–231.
[41] P. Zhou, B. Zhao, J. Yang, and Y. Zhang, "Energy reduction for STT-RAM using early write termination," in *Proc. Int. Conf. Comput.-Aided Design*, 2009, pp. 264–268.

Dong Hyuk Woo (S'06–M'11) received the B.S. degree in electrical engineering from Seoul National University, Seoul, South Korea, in 2005 and the M.S. and Ph.D. degrees in electrical and computer engineering from the Georgia Institute of Technology, Atlanta, in 2007 and 2010, respectively.

He is currently a Research Scientist with Intel Labs, Intel Corporation, Santa Clara, CA. His current research interests include heterogeneous many-core architecture, general-purpose graphics processing unit, 3-D integration, and emerging memory technologies.

Dr. Woo was a recipient of the Award of Minister of Information and Communication of Republic of Korea in 2003. He has co-authored a paper that was nominated for the Best Paper Award at HPEC-07 and a paper that was selected as IEEE Micro's top picks from the computer architecture conferences of 2010.

Nak Hee Seong received the B.S. and M.S. degrees in computer engineering from Seoul National University, Seoul, South Korea, in 1996 and 1998, respectively, and is currently pursuing the Ph.D. degree in electrical and computer engineering from the Georgia Institute of Technology, Atlanta.

In 2000, he joined the System LSI Division, Samsung Electronics Corporation, South Korea, as a System Architect. His current research interests include emerging memory technologies, main memory scheduling, and 3-D integration.

Mr. Seong was a recipient of Samsung Electronics Global Scholarship. He has co-authored a paper that was selected as IEEE Micros top picks from the Computer Architecture Conferences of 2010.

Hsien-Hsin S. Lee (M'96–SM'07) received the Ph.D. degree in computer science and engineering from the University of Michigan at Ann Arbor.

He is an Associate Professor with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta. His research interests include computer architecture, 3-D IC, energy-efficient computing, and cyber-security. Prior to joining academia in 2002, he was a Senior Processor Architect and a Researcher with Intel Corporation designing Pentium III processor and conducting research for Itanium architecture. Later, he managed the architecture team and developed their future generation StarCore DSP at a joint technology center of Agere Systems and Motorola. He holds 4 U.S. patents.

Dr. Lee's doctoral thesis was awarded the Horace H. Rackham School Distinguished Dissertation Award at the University of Michigan. He was a recipient of the Department of Energy Early CAREER PI Award, the Georgia Tech's ECE Outstanding Jr. Faculty Member Award, the NSF CAREER Award, and an IBM Faculty Award. He has co-authored four conference papers that received the Best Paper Award at MICRO-33, CASES-04, IBM PAC^2, and ANCS-11, and one paper selected in IEEE MICRO Top Picks of Computer Architecture Conferences in 2010. He serves on several editorial boards including the ACM Transactions on Architecture and Code Optimization (TACO) and the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS (TCAD). He is a member of Tau Beta Pi and a senior member of the ACM.