Microbank: Architecting Through-Silicon Interposer-Based Main Memory Systems

Young Hoon Son[†] Seongil O[†] Hyunggyun Yang[‡] Daejin Jung[†] Jung Ho Ahn[†]
John Kim[§] Jangwoo Kim[‡] Jae W. Lee[¶]
[†]Seoul National University [‡]POSTECH
{yhson96, swdfish, haidj, gajh}@snu.ac.kr {psyjs037, jangwoo}@postech.ac.kr
[§]KAIST [¶]Sungkyunkwan University
jjk12@kaist.edu jaewlee@skku.edu

***Abstract*—Through-Silicon Interposer (TSI) has recently been proposed to provide high memory bandwidth and improve energy efficiency of the main memory system. However, the impact of TSI on main memory system architecture has not been well explored. While TSI improves the I/O energy efficiency, we show that it results in an *unbalanced* memory system design in terms of energy efficiency as the core DRAM dominates overall energy consumption. To balance and enhance the energy efficiency of a TSI-based memory system, we propose μbank, a novel DRAM device organization in which each bank is partitioned into multiple smaller banks (or μbanks) that operate independently like conventional banks with minimal area overhead. The μbank organization significantly increases the amount of bank-level parallelism to improve the performance and energy efficiency of the TSI-based memory system. The massive number of μbanks reduces bank conflicts, hence simplifying the memory system design. We evaluated a sophisticated prediction-based DRAM page-management policy, which can improve performance by up to 20.5% in a conventional memory system without μbanks. However, a μbank-based design does not require such a complex page-management policy and a simple open-page policy is often sufficient – achieving within 5% of a perfect predictor. Our proposed μbank-based memory system improves the IPC and system energy-delay product by 1.62× and 4.80×, respectively, for memory-intensive SPEC 2006 benchmarks on average, over the baseline DDR3-based memory system.**

I. Introduction

Modern microprocessors have adopted highly threaded multi-core designs to achieve energy-efficient performance with an increasing number of transistors on a die [18]. Applications have also evolved to execute multiple tasks to effectively utilize the increasing number of cores. However, the performance potential of a multi-core processor is realized only when the memory system can satisfy the increasing capacity and bandwidth requirements. While the increased transistor density can help increase memory capacity, it is much more challenging to scale memory bandwidth cost-effectively.

To address the increasing demands for processor-to-memory bandwidth, either the number of pins or the data rate of each pin must be increased. Modern multi-core processors have integrated an increasing number of memory controllers on a die [15], [31], [35] to increase the bandwidth by adding more pins, and leveraged DRAM or buffering devices running at a higher clock rate [12]. However, neither the pin transfer rates nor the number of pins can continue to scale easily. Boosting the pin transfer rates degrades the energy efficiency and signal integrity. Adding more pins increases the package area, which in turn increases the fabrication cost. Furthermore, assuming the energy per bit of inter-package data transfer is relatively constant, the power consumed in the memory system increases linearly to the number of pins, which makes memory channels consume a significant portion of the total package power.

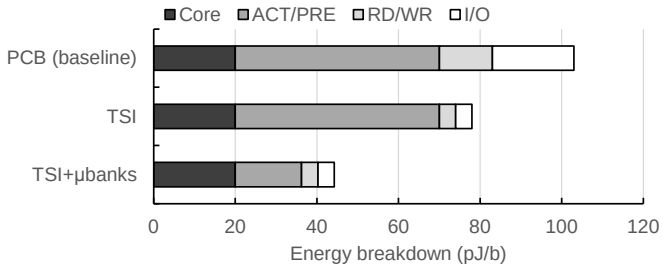


Fig. 1: Energy breakdown of the conventional PCB-based, TSI-based, and proposed μbank-based memory system (detailed modeling is described in Section III-B).

To achieve high memory bandwidth without increasing the die area or off-package access energy, Through-Silicon Interposer (TSI)-based packaging provides an attractive alternative solution [20], [59] by combining two emerging technologies: 3D-stacked memory and interposer-based die integration. The interposer-based die integration connects a processor die and memory dies using in-package metal wires and Through-Silicon Vias (TSVs), while memory dies are stacked vertically using intra-die metal wires and inter-die TSVs. By using a low impedance and high bandwidth intra-package communication medium, TSI-based packaging is considered as a promising technology providing high memory bandwidth, high energy efficiency, and decent scalability in capacity. Processors with TSI-based integration (also known as 2.5D form factor) are expected to be introduced into the market before full 3D stacked processors without silicon interposers because of their advantages in cost and yield [13], [36].

Although there has been a significant amount of research done on TSV-based 3D integration, there have been limited architectural studies on TSI-based integration. In this work, we explore the impact of TSI technology on the design of the main memory system architecture. Compared with a baseline DDR3 main memory system where the components are connected through printed circuit boards (PCBs), the use of TSI in

SC14, November 16-21, 2014, New Orleans
978-1-4799-5500-8/14/$31.00

the memory system reduces the inter-package data transfer (I/O) power as shown in Figure 1. However, with a reduction in the I/O power dissipation, the energy consumed in the main memory system is "unbalanced" as the memory core energy consumption (e.g., activate/precharge energy) begins to dominate the overall energy consumption. To address this, we propose *μbank*, a novel DRAM device architecture customized for TSI-based main memory systems. The μbank DRAM partitions each bank both horizontally and vertically into a large number of small banks (or μbanks) to activate fewer bits per activation, which corresponds to reducing the size of a DRAM page or row. The μbanks operate independently like conventional banks, and the partitioning granularity is chosen to minimize the area (cost) overhead while improving both the performance and energy efficiency.

While a larger number of μbanks significantly increases bank-level parallelism and improves energy efficiency, the effectiveness of prior approaches to memory system design needs to be re-examined. Because there are much fewer bank conflicts due to the larger number of open rows, a complex page-management policy is not necessary and hence μbank simplifies the DRAM controller design. Our evaluation shows that a sophisticated prediction-based page-management policy provides significant performance improvement on a conventional main memory system but a simple open-page policy [50] achieves comparable performance with μbanks. We also revisit the appropriate granularity of address interleaving and show that DRAM row interleaving outperforms cache-line interleaving because inter-thread interference mostly disappears with the massive number of banks.

In summary, this paper makes the following contributions:

- We explore the impact of TSI on memory-system architecture and analyze the challenges in realizing the full potential of the increased bandwidth and capacity offered by the technology.
- We propose μbank, a novel DRAM device organization for TSI-based memory systems to reduce the DRAM row activate and precharge energy while increasing bank-level parallelism.
- We explore the impact of μbanks on the DRAM page-management policy and show how a complex policy is not needed. We evaluate a novel prediction-based DRAM page-management scheme which improves performance by up to 20.5% for a system without μbanks; however, the performance improvement over a simple open-page policy is limited to 3.9% with μbanks .
- Compared to a baseline system with DDR3-based processor-memory interfaces, our TSI-based μbank system improves performance (IPC) by 1.62× and energy-delay product by 4.80× on average for a third of the SPEC CPU2006 benchmarks with high main memory bandwidth demands.

II. DRAM Background

Main-memory DRAM organization has evolved to improve throughput per device (die or package) while lowering random access latencies under a tight budget constraint in die area and fabrication complexity. A DRAM cell, which stores a bit of data, consists of an access transistor and a capacitor. DRAM

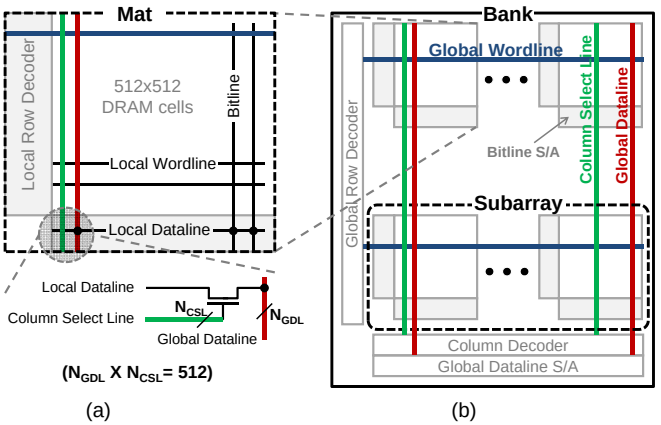


Fig. 2: Conventional DRAM organization. (a) A two-dimensional array called mat is a unit of storage structure and (b) the mats compose again a two-dimensional structure called bank where all the mats share a global row decoder and a column decoder (S/A = Sense Amplifiers).

cells are arranged in a two-dimensional array such that all the cells in a row are controlled by a wordline (WL) and all the cells in a column share a datapath bitline (BL).

This wordline and bitline sharing improves area efficiency but also increases the capacitance and resistance of the datapath and control wires so that single-level drivers and sense amplifiers become inadequate, especially for modern multi-Gbit DRAM devices. Therefore, hierarchical structures [58] are leveraged in the datapath and control wires so that a long wordline is divided into multiple sub-wordlines, each having a dedicated driver and spanning a single mat. A mat is typically composed of 512×512 DRAM cells in a modern DRAM device; it is the building block for the DRAM structure as shown in Figure 2(a). The sub-wordlines within a mat are referred to as local wordlines. Similarly, bitlines refer to the datapath wires within a mat while the wires that are perpendicular to the wordlines and traversing an entire array to transfer data are called global datalines (Figure 2(b)). Local datalines, which are parallel with wordlines, connect bitlines and global datalines. A modern DRAM device has several of these arrays called banks. Datapath wires that encompass all the banks within a device are called inter-bank datalines.

To access data in a DRAM device, a series of commands are applied. First, the row decoder decodes the address that is issued along with an activate command (ACT) and drives a particular global wordline. Each mat has a local row decoder that combines the signals from global wordlines and drives up to one local wordline. Therefore, a row of mats specified by the address activates the target local wordlines. All the transistors controlled by the local wordlines become open and share charges with the corresponding bitlines. A bitline sense amplifier is connected to each bitline because the capacitance of a bitline is much higher than that of a cell [47]. Therefore, a small voltage difference developed by the charge sharing must be amplified. These bitline sense amplifiers latch data and then restore those values back to the open (connected) cells because DRAM cells are passive.

Next, the column decoder drives the column select lines, which are parallel with global datalines, specified by one or

more read commands (RDs) following the activate command after a minimum time of tRCD. The column select lines choose one or more bitlines per mat to move data from the bitline sense amplifiers to the edge of the mat through local datalines. Each local dataline is connected to a global dataline that traverses the entire column of the bank. The global dataline also has a sense amplifier to reduce data transfer time because it is lengthy (spans a few millimeters) and connected to dozens of local datalines, one per row of mats called a subarray, in modern devices. Once the data transfer is over, a precharge command (PRE) precharges the datapath wires to make them ready for the following transfers, which takes tRP. A write (WR) process is the same as a read except that the data transfer path is reversed. Note that these DRAM commands expose a portion of bitline sense amplifiers as row buffers so that they can be exploited in scheduling commands to the DRAM devices. When a device does not have enough bandwidth or capacity, multiple devices are grouped and operate in tandem forming a rank. Ranks are connected through a bus and controlled by a memory controller. The datapath width of a dual in-line memory module (DIMM), which hosts few ranks, is 64 bits.

Modern DRAM devices have multiple banks [34] to improve the data transfer rate on random accesses with little spatial locality. It takes up to a few nanoseconds for a DRAM rank to transfer a cache line, whose size is typically around 64 bytes (5ns on a DDR3-1600 DIMM [51]). However, the time to activate a row, restore it, and precharge the datapath wires in a device, which is called the cycle time (tRC), is still around 50ns [51]. Unless an activated row is used for dozens of cache-line transfers, the datapath out of the device would be greatly under-utilized. In order to alleviate this problem, mats in a device are grouped into multiple banks, each of which operates independently except that all banks in a channel share command and datapath I/O pads that are used to communicate to the outside of the DRAM die.

III. Through-Silicon Interposer (TSI)

In this section, we describe the Through-Silicon Interposer (TSI) technology. We then quantify the energy and performance benefits of the TSI technology when applied to the main memory system, compared with the conventional DIMM-based memory system.

A. TSI Technology Overview

The bandwidth density and energy consumption per data transaction of inter-package communication through PCBs have improved very slowly compared to those of on-die computation and communication. An integrated circuit die, encapsulated in a package, is placed above and connected to a polymer-based substrate through a wire bonding or flip-chip process (Figure 3(a)). The substrate typically has an array of balls or pins whose pitch is around a millimeter and has not decreased substantially over time [1]. This is in contrast to the transistor pitch, which has improved much more rapidly following Moore's Law. Therefore, the number of pins per package has increased at a much slower rate compared with the computational capabilities, and system designers often rely on increasing the data transfer rate per I/O pin to alleviate the bandwidth pressure from on-die transistors.

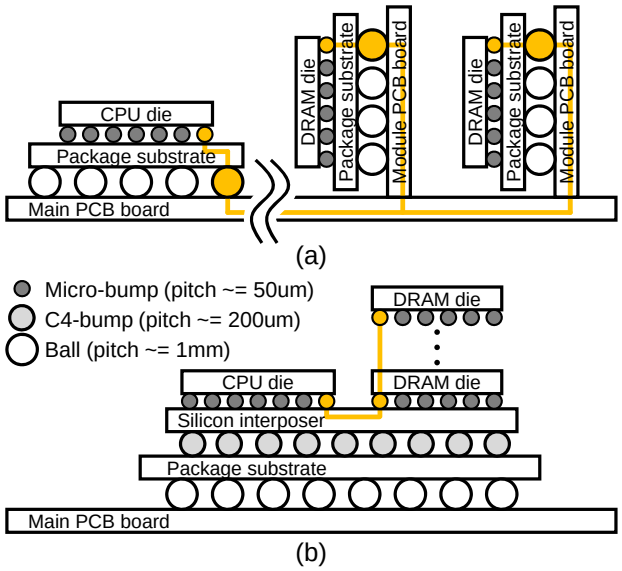


Fig. 3: Packaging technologies. (a) A conventional ball-grid-array-based system and (b) a through-silicon interposer-based system utilizing silicon interposer and through-silicon vias (TSVs).

An impedance mismatch caused by bulky substrate pads and balls as well as multiple wire stubs attached to the inter-die communication channel over PCBs results in the reflection of transmission waves, and hence poor signal integrity. Sophisticated impedance matching, such as on-die termination (ODT) and large drivers, is needed to deliver signals at a rate of multi-Gb/s per pin. If more than two pads are connected to an inter-die channel to increase the main memory capacity, the signal integrity of the channel gets degraded, leading to higher energy consumption per bit and even limiting the maximum data transfer rate. For example, DDR3 interfaces [51] for servers and laptops support multiple ranks per memory channel but consume 20pJ/b. GDDR5 interfaces [28] for graphics support high bandwidth per pad (up to 10 Gbps) but only allow point-to-point connections. LPDDR2 [27] interfaces for mobile systems have neither ODT nor delay-locked loops (DLLs), lowering the data transfer energy, but the data transfer rate per pin is heavily affected by the signal integrity and hardly surpasses one Gb/s. Therefore, bandwidth, energy efficiency, and capacity conflict with each other in the conventional inter-die processor-memory interfaces.

Through-Silicon Interposer (TSI) technology [20], [59] can address the bandwidth density (i.e., bandwidth per cross sectional distance, measured in Gbps/mm) and energy efficiency issues of the interconnects in processor-memory interfaces. A silicon interposer replaces a conventional polymer-based substrate and is located between a die and a substrate (Figure 3(b)). An interposer can be created with a method similar to fabricating a silicon die, but the process is simpler because it only has metal interconnect layers, vias, and pads; thus, it leads to lower costs. Multiple dies are attached to an interposer through *micro-bumps*, whose pitch is smaller than 50μm and an order of magnitude smaller than the ball (pin) pitch of the package. Even if micro-bumps were to be used between a die and a conventional substrate, the ball pitch of the package that

TABLE I: DRAM energy and timing parameters.

Energy Parameter	Value	
I/O energy (DDR3-PCB)	20pJ/b	
I/O energy (LPDDR-TSI)	4pJ/b	
RD or WR energy without I/O (DDR3-PCB)	13pJ/b	
RD or WR energy without I/O (LPDDR-TSI)	4pJ/b	
ACT+PRE energy (8KB DRAM page)	30nJ	
Timing Parameter	**Symbol**	**Value**
Activate to read delay	tRCD	14ns
Read to first data delay (DDR3)	tAA	14ns
Read to first data delay (TSI)	tAA	12ns
Activate to precharge delay	tRAS	35ns
Precharge command period	tRP	14ns

contains the die and the substrate limits the benefits of the micro-bumps. In contrast, the wire pitch of a silicon interposer can be as small as the pitch of the top-level metal layers of the silicon dies; therefore, it is possible to have thousands of inter-die communication channels using only one silicon interposer metal layer. Escape routing [14] can be employed to resolve the pitch mismatch issue between the micro-bumps (few tens of microns) and the silicon-interposer wires (few microns [59]).

The channels over TSIs have a much better signal integrity than those over PCBs because the micro-bumps are smaller than the balls, and there are fewer wire stubs between the pads. With more channels between the dies, the data transfer rates per channel can be lowered while still providing high bandwidth, hence reducing the complexity of the transceivers and the I/O energy consumption per bit. Through Silicon Vias (TSVs)[1] can be used to stack multiple dies, particularly low-power DRAM dies, which effectively resolves the capacity problem in the main memory system.

B. The Energy Efficiency and Latency Impact of the TSI Technology on Main Memory Systems

To quantify the performance and energy efficiency impact of the TSI technology on processor-memory interfaces, we modeled inter-die I/Os by modifying CACTI-3DD [11]. To estimate the energy, area, and latency of a main memory DRAM device, we assumed a 28nm process, PTM low-power model [63] for the wires, and 3 metal layers. The minimal wire pitch of the global wordlines and datalines was conservatively assumed as 0.5μm [58] to reduce the inter-line interference; the pitch of the micro-bumps was 50μm, and that of the interposer wires was 5μm [37]. The capacity of a DRAM die was 8Gb, and the size of the baseline die was 80mm^2. The size of a DRAM page or row per rank was 8KB.

Table I lists the modeled DRAM energy and timing values. In a DDR3 interface [51], which is the baseline that we assume in this work, the dominant portion of the read or write energy is the inter-die I/O energy, which is 20pJ/b [44], [54]. The energy to move data between bitline sense amplifiers and DRAM-side transceivers, which include local, global, and inter-bank datalines, is 13pJ/b. A naive way to apply the TSI technology to the DDR3 interface is to vertically stack DRAM dies in a rank without modifying its physical layer. This can greatly improve the aggregate memory bandwidth of a processor because TSI eliminates the pin-count constraint. The energy efficiency, however, is only modestly improved because the DDR3 physical layer still has ODTs and DLLs that draw considerable power.

A better way to exploit TSI is to replace the DDR3 interface with the LPDDR (low-power DDR) interface [27], [61]. The shorter physical distance in LPDDR obviates the need for ODTs and DLLs and significantly lowers the I/O and read/write energy. One issue with LPDDR is the lower per-pin transfer rate, but it can be overcome by increasing the number of pins exploiting TSI. Another issue with LPDDR is that the datapath is not delay-locked and jitter can become more problematic, especially across dies [39]. Therefore, we assume each die constitutes a rank, instead of using multiple dies. By applying the TSI technology and exploiting the LPDDR interface, the inter-die I/O energy efficiency improves substantially to be only 4pJ/b.[2] We assume that a CPU-side pad and 8 DRAM-side pads constitute an inter-die channel, where high-speed serial links are not effective solutions.

However, the reduced I/O energy consumption leads to an "unbalanced" main memory design in terms of energy efficiency as the non-I/O portion (e.g., activate/precharge energy) begins to dominate the overall energy consumption. Modern performance-energy balanced cores need a few hundred pico joules (pJ) per operation [7], [53]. For example, a dual-issue out-of-order core, modeled by McPAT [40] (details in Section VI-A), consumes 200pJ/op in 22nm. Assuming 20 memory accesses per kilo-instructions (MAPKI) and a cache line size of 64B, each operation incurs $64 \times 8 \times 20/1000 = 10.24$ bits of data transfers from the main memory on average. Using the conventional interface, it translates to 200pJ/op, which is on a par with the core energy consumption. By utilizing the TSI-based interface, only 40pJ is needed instead, which is much more energy efficient. Therefore, the improved energy efficiency of inter-die I/Os makes the activate and precharge energy more prominent, with their reduction becoming a key design challenge.

The impact of TSI on DRAM access latency is not as significant as the access energy. The internal structure of DRAM devices is mostly unaffected, and the latency of the inter-die channel is not high even in conventional interfaces (e.g., 170ps per inch [14]). However, a lower transfer rate per channel reduces the access latency because fewer serialization and deserialization steps are needed [10], [26]. The following section explores the customization of main memory system design to better exploit the opportunities opened by the TSI technology.

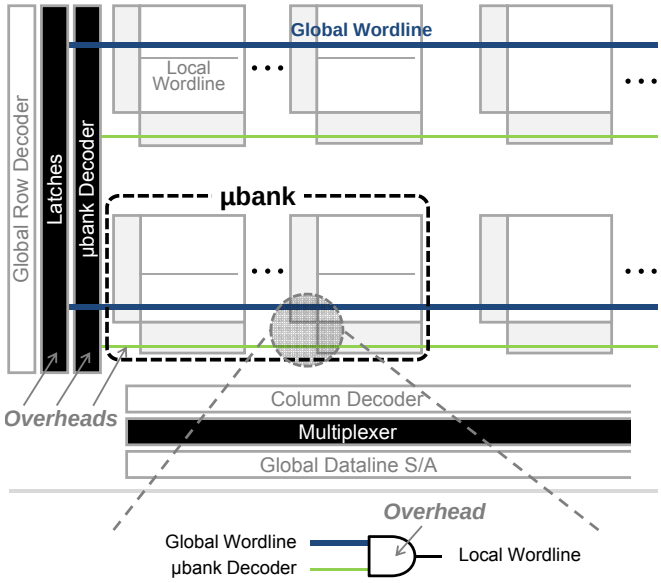
IV. μBANK: A DRAM DEVICE ORGANIZATION FOR TSI-BASED MAIN MEMORY SYSTEMS

A. Motivation for μbank

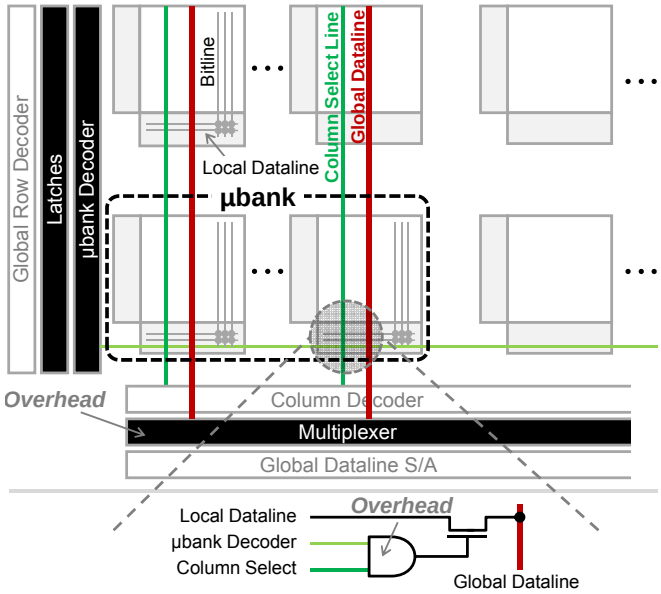
Energy to transfer a cache line through the processor-memory interface decreases substantially with the reduced I/O energy from the TSI technology. As a result, other

[1] We do not assume fine-pitch TSVs [1] in this work because fine-pitch TSVs still have high packaging cost and yield issues [48]. Instead, we assume coarse-pitch TSVs with the same pitch as micro-bumps.

[2] 1pJ/b or lower values have been reported before, especially using high-speed serial links [21], [45]. However, these high-speed serial links assume point-to-point connections and consume most of power statically because clock-data recovery circuitry is always on regardless of actual data transfers.



(a) μbank row and column decoders select latches to update.



(b) μbank row decoder and column select pick data to read and write.

Fig. 4: μbank organization and operations. Compared to the baseline organization, μbank row/column decoders are added per bank. Latches are added to hold currently active wordline per μbank.

components of DRAM power consumption, including static power (e.g. DLL and charge pumps), refresh operations, and activate/precharge operations, represent a substantially higher fraction of total DRAM power. In particular, the energy to precharge the DRAM datapath and activate a row becomes 15× higher than the energy to read a cache line through inter-die channels, as listed in Table I.[3] In this work, we assume that a cache line is a unit of the main memory data transfers and is 64B.

[3]This is a problem called memory *overfetching* [4], which is due to capacity mismatch between a 64B cache line and a 8KB or 16KB DRAM row that is typical in modern DRAM ranks [51].

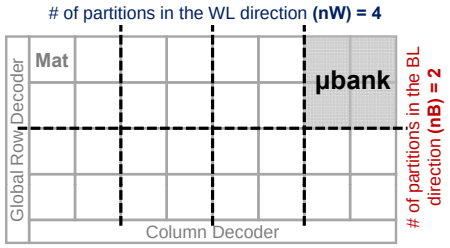
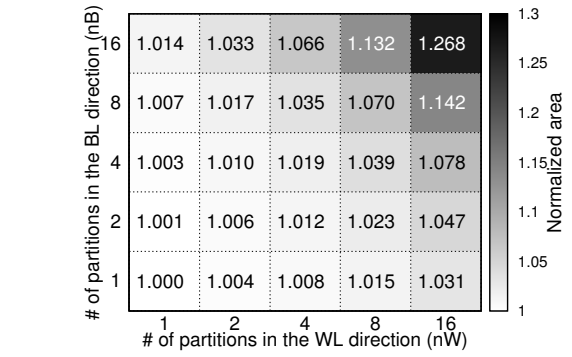


Fig. 5: An example of a μbank design with (nW, nB) = (4, 2). nW is the number of partitions in the wordline direction whereas nB is the number of partitions in the bitline direction.

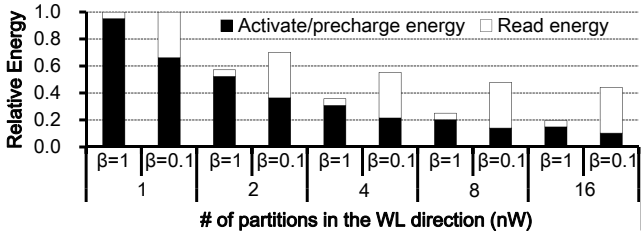
One way to save the activate/precharge energy is to reduce the size of a DRAM row. Because the sub-wordline size of a mat is also 64 bytes (512 bits), the energy overhead from activate/precharge can be minimized by configuring a single mat to provide data for an entire cache line (referred to as a single subarray (SSA) configuration [57]). However, this results in significant DRAM die-area overhead because too many local datalines (i.e., 512) are needed per mat, which need to be routed in parallel with the wordlines. Because the number of metal layers is limited to a few due to extremely tight cost constraints in DRAMs (we assume 3 layers in this paper), the local datalines cannot be routed on top of the wordlines. In addition, the pitch of these datalines is greater (0.5μm [58]) than the width or height of a DRAM cell in order to lower the resistance and capacitance. Therefore, the area of a DRAM die is increased by 3.8× with the SSA configuration compared to the reference DRAM die configuration in Section III-B and thus, makes this approach infeasible. To reduce the area overhead, we can activate multiple mats for a data transfer, which decreases the number of bits transferred *per* mat, hence fewer local datalines are needed.

To increase the number of active rows without exploding the die area, we can exploit the bitline sense amplifiers within each mat to retain data. Note that, in conventional DRAM devices, the number of active rows is the same as the number of banks, and that increasing the number of banks incurs a high area overhead because global-dataline sense amplifiers and row/column decoders are bulky [38]. The overhead in exploiting the abundant bitline sense amplifiers is to add latches between row predecoders (global row decoders) and local row decoders to specify the row for the reads or writes [33] (Figure 4(a)). Further increasing the number of bitline sense amplifiers by decreasing the number of sub-wordlines per mat would incur a much higher area overhead because the size of a sense amplifier and related circuitry is an order of magnitude larger than a DRAM cell [54]. Thus, we group some number of physically adjacent mats and refer to them as a *μbank*, as shown in Figure 4. A μbank consists of a two-dimensional array of mats where only a row of mats can be activated. The number of active rows in a bank is equal to the number of μbanks. The additional μbank row and column decoders are used to identify the specific latches to be updated, and the signal from the μbank row decoder is again combined with the column select signals to specify the μbank used for the data transfers (Figure 4(b)).

The total number of μbanks within a bank is determined by the number of mats grouped together in both the wordline



(a) Relative area



(b) Relative energy

Fig. 6: The relative main memory DRAM area and energy while varying the number of partitions in the BL and WL directions. All the values are normalized to those of one μbank per bank values, respectively.

and bitline dimensions. We define nW as the number of bank partitions in the wordline direction, and nB as the number of bank partitions in the bitline or global dataline direction. Thus, the total number of μbanks per bank is $nW \times nB$. If nW = nB = 1, μbank is equivalent to a bank, which corresponds to the baseline in our evaluation. For example, for a bank consisting of 32 mats (Figure 5), it can be divided into four partitions in the wordline direction (nW = 4) and two partitions in the bitline direction (nB = 2) – thus, there are 8 μbanks per bank, each of which consists of 4 mats.

B. μbank Overhead

The area and energy overhead of μbanks in a TSI-based main memory system are shown in Figure 6. We assume that an 8Gb DRAM die has 16 banks and 2 channels, where each channel serves 8 banks. The channel bandwidth is 16GB/s so that a 64B cache line can be transferred every 4ns, which determines the internal clock frequency of the DRAM mats to be 250MHz. Each bank, whose size is 512Mb, consists of 2,048 512×512 mats, and is laid out as a 64×32 array. Figure 6(a) shows the area overhead of μbank. The x-axis is the number of partitions in the WL direction (nW), and the y-axis is the number of partitions in the BL direction (nB). The area overhead is normalized to nW = nB = 1, which has one μbank per bank.

Because of the added latches, the DRAM die area increases as the number of μbanks per bank increases. When nW = 1, 128 mats (2 rows of mats per bank to latch a 8KB row) are activated together, and each mat provides 4 bits of data out of the 512 activated cells within each mat per read or write to access 64B of data. For this configuration, routing 128 column select lines per mat incurs a noticeable area overhead because the pitch of a column select wire is greater than that of a DRAM cell [58]. Instead, we configure a column select line to choose 8 bits of bitlines and place a multiplexer between the 2 global datalines and the 1 global-dataline sense amplifier. As nW increases, while the number of global-dataline sense amplifiers stays unchanged, the total number of global datalines in a bank increases as the number of global datalines per μbank is fixed to the column width. Again, multiplexers are placed to select the right set of global datalines for the global-dataline sense amplifiers. Meanwhile, the number of column select lines, which share the same metal layer with the global datalines, decreases for a reduced number of columns per μbank. Therefore, compared to a baseline of nW = 1, the sum of the global datalines and the column select lines per bank does not increase as we increase the number of partitions in the wordline direction until 16. If we partition a bank into 16 pieces in both directions (nW = nB = 16), there is a 26.8% area overhead. However, for most of the other μbank configurations (when $nW \times nB < 64$), the area overhead is under 5%.

In addition to the area overhead, we quantify the energy overhead in Figure 6(b) and show the relative energy consumption of the DRAM configurations per read[4] when the ratios of activate commands to read/write commands (β) are 1.0 and 0.1, respectively. If $\beta = 1$, it means that there is an activate command for every read/write command. As β decreases, the overhead of the activate/precharge operations is amortized over multiple read/write commands. We normalize the energy consumption of each configuration to that of a single-μbank configuration for both values of β. As the number of μbanks per bank increases, more latches dissipate power, but their impact on the overall energy is negligible because the DRAM cells still account for a dominant portion of the bank area and power. The energy consumption per read is much more sensitive to the number of mats involved per activate command, where more μbanks in the wordline direction (nW) reduce the activate/precharge power, hence the energy per read. This is more prominent when β is high (i.e., low access locality).

V. Revisiting DRAM Page Management Policies

As the previous section focused on μbank-based memory systems tailored to the TSI technology, this section re-evaluates the effectiveness of conventional page-management schemes and then proposes a new scheme to exploit the larger number of banks available in the new memory systems.

Conventional memory controllers [16], [50] hold all pending memory requests in a queue and generate proper DRAM commands to service the requests, while obeying various timing constraints. The memory controllers can also schedule the requests and apply different page management policies to improve the per-bank row hit rates. For example, when the controller generates RD or WR commands to a specific bank, it can check the queue to find future memory requests that are also targeted to the bank. As long as the queue is not empty, the controller can make an effective decision of closing the row or keeping it open. The two most basic page-management policies either keep the page open (activated) expecting for the next access's row hit (i.e., open-page policy) or close it immediately

[4]Energy per write is similar to energy per read and not shown here due to limited space.

expecting for a row miss (i.e., close-page policy) [50]. The more sophisticated, adaptive policies include minimalist-open policy [32] to close a row after observing a small number of row hits, and reinforcement learning (RL) approaches [24], [43] to adapt the scheduling policy based on the access history in the past. However, if the queue is empty, the controller must manage the page speculatively.

In fact, there exist two factors that make it difficult to employ these conventional memory scheduling policies to μbank-based memory systems, which require future memory requests available in the request queue (i.e., pending requests.) First, a memory request stream is now distributed over a larger number of banks in μbank-based memory systems compared with conventional memory systems, hence decreasing the average number of pending requests per bank. Second, the higher channel bandwidth provided by μbank-based memory systems further reduces the average queue occupancy. As a result, the request queues in μbank-based memory systems are very likely to fail in providing the information of future memory requests to the memory controller so that it cannot make an effective decision to manage the pages.

Therefore, for μbank-based memory systems, we devise a *prediction-based* page management scheme to adapt between close-page and open-page policies, based on the history of past memory requests. In this way, the new page management scheme can make effective page-management decisions without examining future memory requests in the queue. Our example design is based on a standard 2-bit bimodal branch predictor, which tracks the prediction results with either open or close (instead of taken or not taken) for each bank. The 2-bit predictor utilizes four states (00: strongly open, 01: open, 10: close, 11: strongly close), in which the two open states result in "predict open" page policies and the two close states result in "predict close" page policies. Depending on the accuracy of the previous prediction, the state is changed accordingly, similar to conventional branch predictors. However, there exist several differences between conventional branch predictors and the page-management policy predictor, which can affect the effectiveness of the prediction differently. For example, even though conventional branch predictors resolve branches in several clock cycles, the page-management predictor may take much longer (e.g., several milliseconds.) In addition, address aliasing is much less of an issue in the page-management predictor because the number of DRAM pages is much smaller than the program's address space.

VI. Evaluation

A. Experimental Setup

We simulated a chip-multiprocessor system with multiple memory channels to evaluate the system-level impact of the μbank-based main memory systems exploiting the Through-Silicon-Interposer (TSI) technology (Figure 7). We assumed a system with 64 out-of-order cores, each running at 2GHz. Each core issues and commits up to two instructions per cycle, has a 32-entry reorder buffer, and has separate L1 I/D caches. Four cores share an L2 cache. We set the size and associativity of each L1 cache and L2 cache to 16KB and four, and 2MB and 16, respectively. Each cache is partitioned into four banks, and the line size of all the caches is 64B. The system uses a MESI cache-coherency protocol, and a reverse directory is

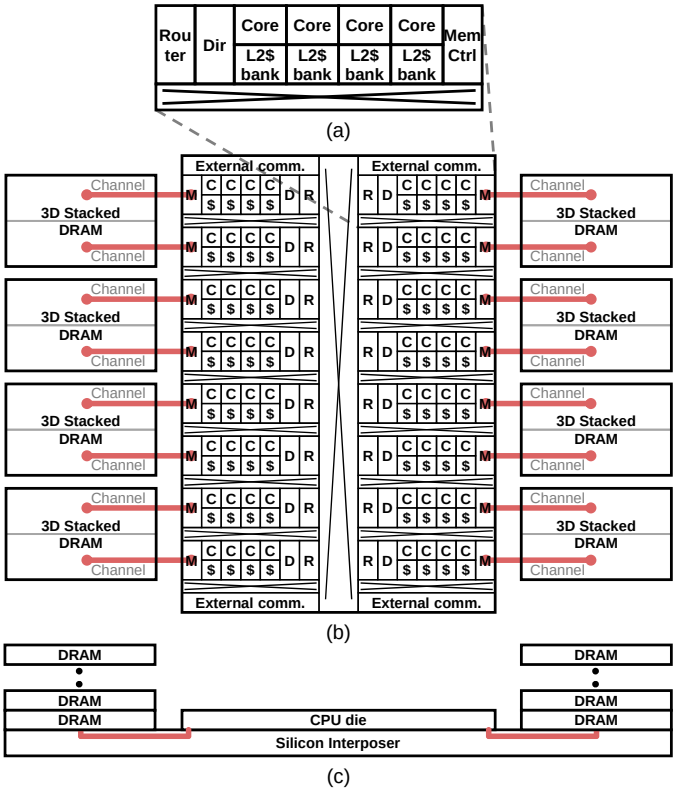


Fig. 7: A 64-core chip-multiprocessor system with 16 clusters. (a) Each cluster has four cores (C), an L2 cache ($), a directory unit (D), a memory controller (M), and a router (R). A floorplan is shown in (b) and a cross-section view of the system is shown in (c).

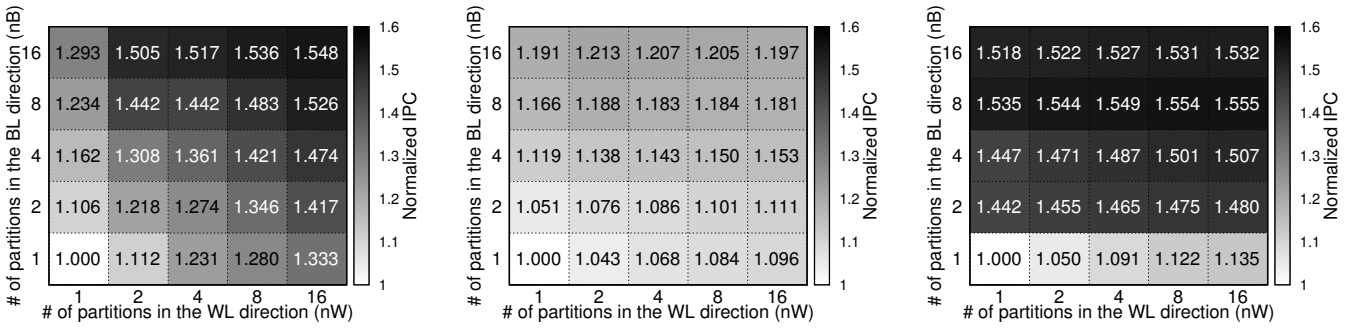
associated with each memory controller. Figure 7(a) shows that each cluster consists of four cores, an L2 cache, a directory unit, a memory controller, and a router.

The system has 16 memory controllers, and each controller has one memory channel, whose bandwidth is 16GB/s excluding the ECC bandwidth. To evaluate single-threaded programs, we populated only one memory controller for the simulated system to stress the main memory bandwidth. Each memory controller has a 32-entry request queue by default, and applies PAR-BS [46] and the open-page [50] policy for memory access scheduling. The main memory capacity is 64GB. We modified McSimA+ [5] to model the μbank-based main memory systems. We used McPAT [40] to model the core/cache power, area, and timing. The power, area, and timing values of the processor-memory interfaces and the main memory DRAM devices were modeled as explained earlier in Section III-B and summarized in Table I.

For the evaluation, we used the SPLASH-2 [60], SPEC CPU2006 [19], PARSEC [9], and TPC-C/H [3] benchmark suites. We used Simpoint [52] to identify the representative phases of the SPEC CPU2006 applications. Per SPEC application, we chose the top-4 slices in weight, each having 100 million instructions. As for SPLASH-2 and PARSEC, we simulated regions of interest and used the datasets listed in [40]. As for server workloads, we used two database workloads (TPC-C/H) from TPC benchmark. We carefully tuned PostgresSQL DB [2] to populate the target system with

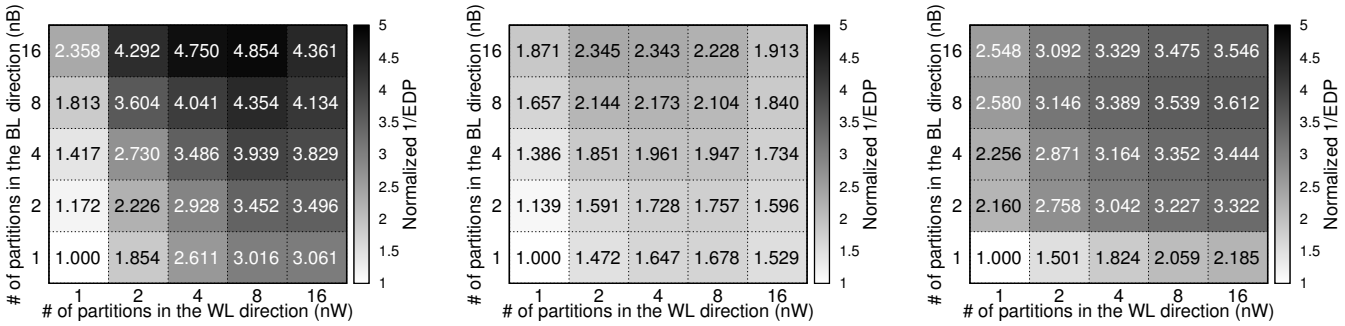
Group	SPEC CPU2006 applications
spec-high	429.mcf, 433.milc, 437.leslie3d, 450.soplex, 459.GemsFDTD, 462.libquantum, 470.lbm, 471.omnetpp, 482.sphinx3
spec-med	403.gcc, 410.bwaves, 434.zeusmp, 436.cactusADM, 458.sjeng, 464.h264ref, 465.tonto, 473.astar, 481.wrf, 483.xalancbmk
sepc-low	400.perlbench, 401.bzip2, 416.gamess, 435.gromacs, 444.namd, 445.gobmk, 447.dealII, 453.povray, 454.calculix, 456.hmmer

TABLE II: We categorized the SPEC CPU2006 applications into 3 groups depending on the number of main memory accesses per kilo instructions (MAPKIs).



(a) 429.mcf (b) spec-high (c) TPC-H

Fig. 8: The relative IPC of (b) 429.mcf, (b) the average of spec-high, and (c) TPC-H. Baseline is (nW, nB) = (1, 1) for each.



(a) 429.mcf (b) spec-high (c) TPC-H

Fig. 9: The relative 1/EDP of (a) 429.mcf, (b) the average of spec-high, and (c) TPC-H. Baseline is (nW, nB) = (1, 1) for each.

the database workloads. We classified the SPEC CPU2006 applications into three groups based on the main memory accesses per kilo-instructions (MAPKI) [19] (Table II). We created two mixtures of multiprogrammed workloads: mix-high from spec-high applications and mix-blend from all three groups. Per mixture, a simulation point is assigned to each core, and the number of populated points is proportional to their weights.

B. The System-level Impact of the μbanks

We first show that dividing a bank in either bitline or wordlines directions improves the performance with diminishing returns as the number of μbanks increases. In general, the bitline-direction partitioning (changing nB) yields higher returns on investment. For these experiments, we ran bandwidth-intensive SPEC CPU2006 applications and database workloads to estimate the performance and energy-efficiency gains on the die area increase with adopting μbanks. Figure 8 shows the relative IPC values of 429.mcf, the average for the spec-high applications, and TPC-H. The unpartitioned configuration ((nW, nB) = (1, 1)) is the baseline. When a global dataline traverses more μbanks (higher nB), the IPC increases because there are more active rows while the row size is unchanged. As a global wordline traverses more μbanks (higher nW, the row size becoming 8KB/nW), IPC steadily increases on low nB values. However, creating many wordline partitions on high nB values can reduce IPC improvements because more active rows become available for higher nW, but each row gets smaller, underutilizing spatial locality in a request stream. On 429.mcf, nB and nW had a similar impact on performance, and (nW, nB) = (16, 16) performs the best, for which the IPC is 54.8% higher than the baseline. TPC-H is more sensitive to nB than nW, and the configuration that gives the highest performance is (nW, nB) = (16, 8).

We observed similar trends in the energy-delay product (EDP) metric. Figure 9 shows the relative 1/EDP of 429.mcf, spec-high, and TPC-H. As we present the reciprocal of the relative EDP, a higher value indicates a better energy efficiency. First, 429.mcf achieves the highest IPC and 1/EDP values in (nW, nB) = (16, 16) and (8, 16) configurations, respectively. However, TPC-H and spec-high achieve the highest IPC and 1/EDP values in (16, 8) and (2, 16) configurations, respectively. These results indicate that the merit of a lower activate/precharge energy for a smaller row size can outweigh

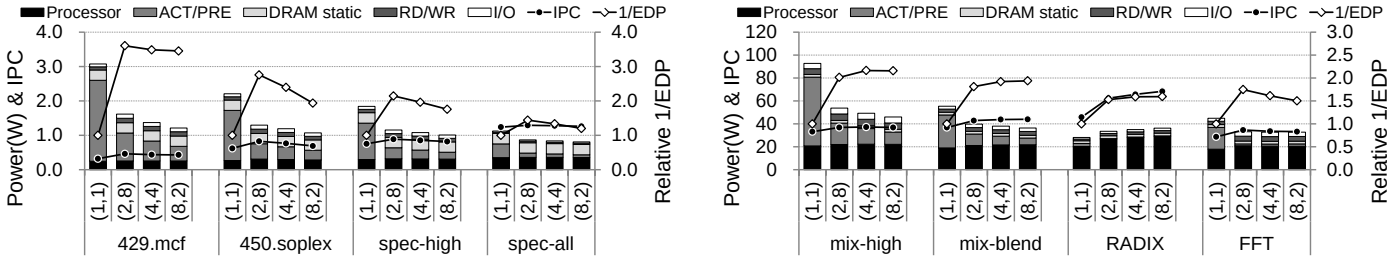


Fig. 10: The relative IPC, 1/EDP, and power breakdown of applications on representative μbank configurations. spec-all stands for the average of all single-threaded SPEC CPU2006 applications. (nW, nB) = (1, 1) is the baseline.

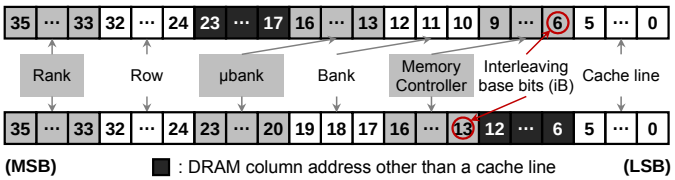


Fig. 11: Two of the possible address interleaving schemes for (nW, nB) = (2, 8)

the overhead of more ACT/PRE commands. Therefore, a balanced approach is necessary in increasing the number of active rows and reducing the size of the rows under a given area constraint.

The configurations with more μbanks is also very effective when we consider the area overhead. Because the DRAM industry is highly sensitive to die area (hence cost), we chose the configurations with an area overhead less than 3% in this experiment, but achieving the most of IPC and EDP benefits. Figure 10 shows the relative IPC and 1/EDP, and power breakdown of single-threaded, multiprogrammed, and multithreaded applications on the representative μbank configurations. The ones with more partitions in the wordline direction dissipate less activate/precharge power. The IPC and EDP values are improved more in memory-bandwidth intensive applications, which have high MAPKI values. In particular, RADIX, a SPLASH-2 multithreaded application, has high MAPKI values and row-hit rates for μbank-based systems, whose IPC value improves by 48.9% on (nW, nB) = (8, 2) configuration.

C. The Impact of Address Interleaving and Prediction-Based Page-Management Schemes on μbanks

More active rows offered by the μbank microarchitecture can alter the optimal address mapping for DRAM accesses, such as the location of the address interleaving base bit (iB). The page management schemes also need to be re-evaluated as discussed in Section V. An example of alternative address interleaving is shown in Figure 11. Existing memory controllers [32], [33] often choose the (micro-)bank number from the low significant bits of memory addresses such that the address interleaving is done at the granularity of one or few cache lines instead of a DRAM row granularity. However, the average row-buffer hit rate for the address interleaving at a DRAM row granularity can increase substantially as the number of active rows increases. Therefore, combined with the open-page policy, a page-granularity interleaving could be more beneficial to μbank than cache line-interleaving.

On the other hand, with fewer active rows, the close-page policy with cache-line interleaving (iB = 6) can perform better than the open-page policy with page-interleaving. To evaluate the impact, we varied iB from 6 to 13 and chose (nW, nB) = (1, 1), (2, 8), (4, 4), and (8, 2) configurations (Figure 12). The baseline configuration is (nW, nB) = (1, 1), open-page policy, and page interleaving (iB = 13). First, with fewer active rows, the difference between the two is not much for both IPC and 1/EDP. This is because the memory access scheduler (we used PAR-BS [46]) detects and restores spatial locality that can be extracted from the request queue of the memory controller. Second, the open-page policy with page-interleaving has greater advantage with the increased number of active rows. For example, with 16 times more available active rows, the open-page policy with page-interleaving clearly outperforms the close-page policy, as high as 17.2% on spec-high for (nW, nB) = (2, 8). This is due to the spatial locality in main memory accesses that was hidden by the intra-thread and inter-thread interference, but is now successfully restored by more active rows.

The prediction-based page-management schemes consistently provide performance gains over fixed management schemes for applications, but only modestly on average. In this experiment, we evaluated three prediction-based page management schemes – local (per bank history bimodal) prediction, global (per thread history bimodal) prediction, and tournament-based prediction schemes. As for the tournament scheme, we applied a bimodal scheme to pick one out of the open, close, local, and global predictors. We treated the open- and close-page management polices as static predictors. We implemented them on top of the default out-of-order scheduler (PAR-BS [46]) to recover access locality to each bank across multiple interleaved request streams.

Our key finding is that the simple, static open-row policy achieves comparable performance with the prediction-based policies, which obviates the needs for complex page-management policies in μbank-based memory systems. Figure 13 shows the relative IPC and 1/EDP values and the predictor hit rates of the prediction schemes for (nW, nB) = (1, 1), (2, 8), and (4, 4). 429.mcf has a lower spatial locality in main memory accesses while canneal has a higher spatial locality than the average of the spec-high applications. Therefore, the close-page policy has a higher prediction hit rate and better performance than the open-page policy on 429.mcf, and vice versa on canneal. Note that the local prediction scheme has a higher hit rate than both static policies, open and close, contributing to the highest hit rate and IPC of the tournament predictor. In the experiments, the global predictor

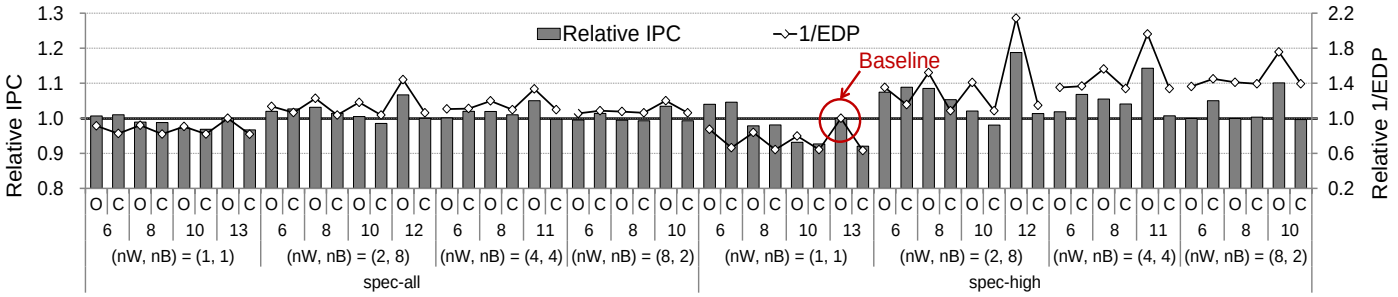


Fig. 12: The relative IPC and 1/EDP values of the average of spec-all and spec-high applications as we vary the page-management policy (close (C) and open (O)) and the interleaving base bits (6 to 13) on the representative μbank configurations.

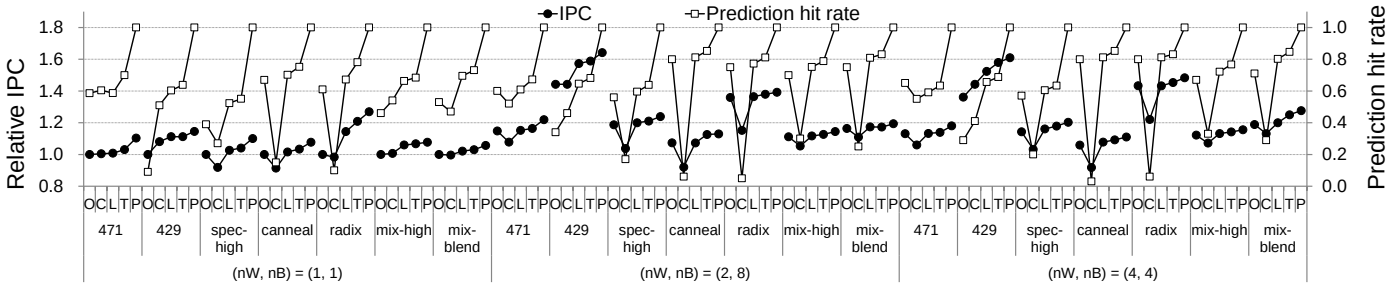


Fig. 13: The relative IPC and the predictor hit rate of alternative page-management schemes on the tested applications. C, O, L, T, and P stand for close, open, local predictor, tournament predictor, and perfect (ideal) predictor, respectively.

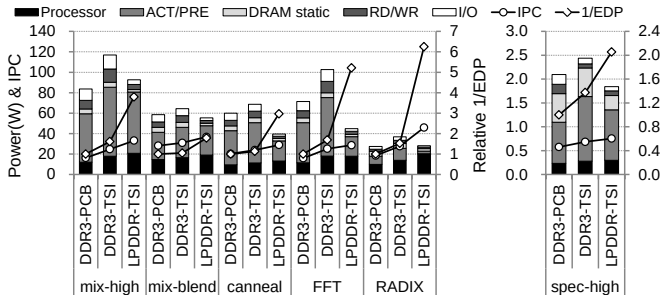


Fig. 14: The IPC, power, and relative EDP of 3 processor-memory interfaces (DDR3-PCB, DDR3-TSI, and LPDDR-TSI) on multiprogrammed, multithreaded, and the average of spec-high workloads.

never performs the best, so it is neither presented in Figure 13 nor considered as a candidate of the tournament predictor. In many applications, the open-page policy outperforms the close-page policy and performs on a par with the tournament predictor scheme because μbanks sufficiently provide many activated rows such that the prediction hit rate of the open-page policy is as high as that of the tournament predictor. The tournament predictor performs better than the open-page policy by 3.9% on average, and up to 11.2% on 429.mcf for (nW, nB) = (2, 8). In the future, for workloads with more complex access patterns, the cost of the tournament predictor may be justified.

D. The Impact of TSI on Processor-Memory Interfaces

To quantify the performance and energy benefits of the TSI-based processor-memory interfaces *without* μbanks, we compared three interfaces: module-based DDR3 connected through PCBs (DDR3-PCB), TSV-based stacked DDR3-type dies connected through a silicon interposer (DDR3-TSI), and TSV-based stacked LPDDR-type dies connected through a silicon interposer (LPDDR-TSI). Figure 14 shows the IPC values, power breakdowns, and relative EDP values of the interfaces on multiprogrammed (mix-high and mix-blend) and multithreaded (FFT, RADIX, and canneal) workloads. For DDR3-PCB, we used eight memory controllers to keep their I/O pin count realistic (around 1,600 pins). For DDR3-TSI, a rank consists of eight DRAM dies. All configurations have the DRAM row size of 8KB. Exploiting TSI improves both the performance and energy efficiency even on conventional DDR3 interfaces, while adopting low-power processor-memory interfaces further saves main memory access energy. For mix-high, DDR3-TSI and LPDDR-TSI achieve 52.5% and 104.3% higher IPC, and 37.8% and 73.7% lower EDP than that of DDR3-PCB, respectively. For LPDDR-TSI, the relative portion of the activate and precharge (ACT/PRE) power out of the total memory power increases to 76.2% for mix-high, and thus, reducing the ACT/PRE power becomes the primary goal of μbank.

VII. Related Work

Die Stacking Technologies: Die stacking technologies have recently received much attention to improve the throughput, latency, and energy efficiency of main memory systems. Virtex-7 FPGAs from Xilinx use stacked silicon interconnect (SSI) technology, which combines multiple FPGA dies using TSIs within a single package [36]. Nvidia has announced that its future Volta GPU will integrate stacked DRAM and a GPU into a single package in a similar way [22]. IBM and Sony have also announced their plans to adopt the TSI technology to scale Moore's Law with Module on Interposer [25] and High-

Bandwidth Memory (HBM) [29]. While TSI-based integration technology is a promising option to provide energy-efficient high-performance main memory, the architectural implications of this technology are much less studied than full 3D stacking technologies [30], [41], [42]. Hybrid Memory Cube (HMC) [49] stacks multiple DRAM dies on top of a logic die and the CPU communicates with the HMCs through high-speed serial links. TSI replaces these links with lower-speed parallel interposer wires. As a result, the HMC has a higher latency and static power and is not necessarily more energy-efficient for the system size being considered (e.g., single-socket system). An HMC-TSI hybrid approach would be an interesting approach because the system size is scaled up, but we leave it as part of future work.

DRAM Microarchitectures and Systems: Researchers have proposed various modifications to the conventional DRAM organization to improve performance and energy efficiency. Kim et al. [33] propose the subarray-level parallelism system that hides the access latency by overlapping multiple requests to the same bank. Gulur et al. [17] replace the existing single large row buffer with multiple sub-row buffers to improve row buffer utilization. μbank introduced in this paper subsumes both designs in that it partitions each bank along both bitlines and wordlines. This work was done in parallel with Half-DRAM [62], which also exploits vertical and horizontal partitioning of the conventional bank structure. However, Half-DRAM applies partitioning in a conventional processor-memory interface and they do not discuss how to exploit the massive number of row buffers. Tiered-Latency DRAM [38], CHARM DRAM [54], and row-buffer decoupling [47] reorganize a DRAM mat to lower the access time for an entire mat or its portion. Although they are introduced in the context of the conventional non-stacking DRAM system, they are complementary and applicable to the μbank devices as well. Alternatively, there are DRAM system-level solutions that need not modify the DRAM device organization. Sudan et al. [56] propose micro-pages which allow chunks from different pages to be co-located in a row buffer to improve both the access time and energy efficiency by reducing the waste of overfetching. Rank subsetting [4], [57], [61], [64] is a technique that utilizes a subset of DRAM chips in a rank to activate fewer DRAM cells per access and improves energy efficiency or reliability. Unlike these system-level techniques, μbank is a device-level solution that either obviates the need for these techniques or complements them.

DRAM Access Scheduling: As discussed in Section V, most of the DRAM controllers that have been proposed [16], [55] make a decision of whether to leave a row buffer open or not after a column access (open-page vs. close-page) by inspecting future requests waiting in the request queue. Kaseridis et al. [32] propose the Minimalist Open scheme, which keeps the row buffer open only for a predetermined time interval (tRC). Piranha [8] takes a similar approach with a different interval ($1\mu s$). To improve long-term adaptivity, both Ipek et al. [24] and Mukundan et al. [43] take a reinforcement learning (RL) approach to optimize the scheduling policy by considering the access history in the past. Although the implementation details of the memory controllers are not available, Intel implements an Adaptive Page Management (APM) Technology [23] while AMD has a prediction mechanism that determines when to deactivate open pages based on the history of the particular page [6]. Our results show that with the large number of banks, a simple open-page policy can be sufficient to simplify memory controller design.

VIII. Conclusion

In this work, we explore the main memory system architecture that exploits the Through-Silicon Interposer (TSI) technology. We show how a conventional DRAM approach results in an unbalanced design because the core DRAM energy consumption dominates the overall energy in a TSI-based main memory system. Therefore, we propose μbank, a novel DRAM device organization where each bank is partitioned into a massive number of microbanks (μbanks) to minimize the activate/precharge energy with a significantly increased amount of bank-level parallelism. We analyzed the performance benefits of μbanks and quantified the overhead. In addition, we show that the massive number of row buffers available due to μbanks simplifies the page-management policies as a simple open-page policy providing similar performance compared to a complex, predictor-based page-management policy.

Acknowledgments

We thank the reviewers for their comments. This material is supported in part by Samsung Electronics, by the Future Semiconductor Device Technology Development Program (10044735) funded by MOTIE and KSRC, by Global Ph.D Fellowship Program through NRF (NRF-2013H1A2A1034174), by the IT R&D program of MSIP/KEIT (10041313, UX-oriented Mobile SW Platform) and by the MSIP under the "IT Consilience Creative Program" (NIPA-2014-H0201-14-1001).

References

[1] "International Technology Roadmap for Semiconductors." [Online]. Available: http://www.itrs.net/models.html

[2] "PostgreSQL." [Online]. Available: http://www.postgresql.org

[3] "TPC Benchmark." [Online]. Available: http://www.tpc.org

[4] J. Ahn, N. P. Jouppi, C. Kozyrakis, J. Leverich, and R. S. Schreiber, "Future Scaling of Processor-Memory Interfaces," in *SC*, Nov 2009.

[5] J. Ahn, S. Li, S. O, and N. P. Jouppi, "McSimA+: A Manycore Simulator with Application-level+ Simulation and Detailed Microarchitecture Modeling," in *ISPASS*, Apr 2013.

[6] AMD, *BIOS and Kernel Developer's Guide (BKDG) for AMD Family 15th Models 30h-3Fh Processors*, Jun 2014.

[7] O. Azizi, A. Mahesri, B. C. Lee, S. J. Patel, and M. Horowitz, "Energy-performance Tradeoffs in Processor Architecture and Circuit Design: a Marginal Cost Analysis," in *ISCA*, Jun 2010.

[8] L. A. Barroso, K. Gharachorloo, R. McNamara, A. Nowatzyk, S. Qadeer, B. Sano, S. Smith, R. Stets, and B. Verghese, "Piranha: A Scalable Architecture Based on Single-Chip Multiprocessing," in *ISCA*, Jun 2000.

[9] C. Bienia, S. Kumar, J. P. Singh, and K. Li, "The PARSEC Benchmark Suite: Characterization and Architectural Implications," in *PACT*, Oct 2008.

[10] D. W. Chang, Y. H. Son, J. Ahn, H. Kim, M. Ahn, M. J. Schulte, and N. S. Kim, "Dynamic Bandwidth Scaling for Embedded DSPs with 3D-stacked DRAM and Wide I/Os," in *ICCAD*, Nov 2013.

[11] K. Chen, S. Li, N. Muralimanohar, J. Ahn, J. B. Brockman, and N. P. Jouppi, "CACTI-3DD: Architecture-level Modeling for 3D Die-stacked DRAM Main Memory," in *DATE*, Mar 2012.

[12] E. Cooper-Balis, P. Rosenfeld, and B. Jacob, "Buffer-On-Board Memory System," in *ISCA*, Jun 2012.

[13] J. R. Cubillo, R. Weerasekera, Z. Z. Oo, E.-X. Liu, B. Conn, S. Bhattacharya, and R. Patti, "Interconnect Design and Analysis for Through Silicon Interposers (TSIs)," in *3DIC*, Jan/Feb 2012.

[14] W. J. Dally and J. W. Poulton, *Digital Systems Engineering*. Cambridge University Press, 1998.

[15] S. Damaraju, V. George, S. Jahagirdar, T. Khondker, R. Milstrey, S. Sarkar, S. Siers, I. Stolero, and A. Subbiah, "A 22nm IA multi-CPU and GPU System-on-Chip," in *ISSCC*, Feb 2012.

[16] E. Ebrahimi, R. Miftakhutdinow, and C. Fallin, "Parallel Application Memory Scheduling," in *MICRO*, Dec 2011.

[17] N. D. Gulur, R. Manikantan, M. Mehendale, and R. Govindarajan, "Multiple Sub-Row Buffers in DRAM: Unlocking Performance and Energy Improvement Opportunities," in *ICS*, Jun 2012.

[18] J. L. Hennessy and D. A. Patterson, *Computer Architecture: A Quantitative Approach*, 5th ed. Morgan Kaufmann Publishers Inc, 2012.

[19] J. L. Henning, "SPEC CPU2006 Memory Footprint," *Computer Architecture News*, vol. 35, no. 1, 2007.

[20] R. Ho, P. Amberg, E. Chang, P. Koka, J. Lexau, G. Li, F. Y. Liu, H. Schwetman, I. Shubin, H. D. Thacker, X. Zheng, J. E. Cunningham, and A. V. Krishnamoorthy, "Silicon Photonic Interconnects for Large-Scale Computer Systems," *Micro, IEEE*, vol. 33, no. 1, Jan/Feb 2013.

[21] K. Hu, R. Bai, T. Jiang, C. Ma, A. Ragab, S. Palermo, and P. Y. Chiang, "0.16-0.25 pJ/bit, 8 Gb/s Near-Threshold Serial Link Receiver With Super-Harmonic Injection-Locking," *JSSC*, vol. 47, no. 8, 2012.

[22] J. H. Huang, "Opening Keynote: Pascal Next-Generation GPU," in *GPU Technology Conference*, Mar 2014.

[23] Intel, *Intel® Xeon® Processor 7500 Series Datasheet*, Mar 2010.

[24] E. Ipek, O. Mutlu, J. F. Martínez, and R. Caruana, "Self-Optimizing Memory Controllers: A Reinforcement Learning Approach," in *ISCA*, 2008.

[25] S. S. Iyer, "The Evolution of Dense Embedded Memory in High Performance Logic Technologies," in *IEDM*, Dec 2012.

[26] B. Jacob, S. W. Ng, and D. T. Wang, *Memory Systems: Cache, DRAM, Disk*. Morgan Kaufmann Publishers Inc, 2007.

[27] JEDEC Standard, *Low Power Double Data Rate 2 (LPDDR2)*, 2010.

[28] ——, *GDDR5 SGRAM Datasheet*, 2013.

[29] ——, *High Bandwidth Memory DRAM Datasheet*, 2013.

[30] D. Jevdjic, S. Volos, and B. Falsafi, "Die-Stacked DRAM Caches for Servers: Hit Ratio, Latency, or Bandwidth? Have It All with Footprint Cache," in *ISCA*, Jun 2013.

[31] C. Johnson, D. Allen, J. Brown, S. Vanderwiel, R. Hoover, H. Achilles, C.-Y. Cher, G. May, H. Franke, J. Xenedis, and C. Basso, "A Wire-Speed Power™ processor: 2.3GHz 45nm SOI with 16 cores and 64 threads," in *ISSCC*, Feb 2010.

[32] D. Kaseridis, J. Stuecheli, and L. K. John, "Minimalist Open-page: A DRAM Page-mode Scheduling Policy for the Many-core Era," in *MICRO*, Dec 2011.

[33] Y. Kim, V. Seshadri, D. Lee, J. Liu, and O. Mutlu, "A Case for Exploiting Subarray-Level Parallelism (SALP) in DRAM," in *ISCA*, Jun 2012.

[34] C. Kozyrakis, "Scalable Vector Media-processors for Embedded Systems," Ph.D. dissertation, University of California at Berkeley, 2002.

[35] V. Krishnaswamy, D. Huang, S. Turullols, and J. L. Shin, "Bandwidth and Power Management of Glueless 8-Socket SPARC T5 System," in *ISSCC*, Feb 2013.

[36] S. Lakka, "Xilinx SSI Technology, Concept to Silicon Development Overview," in *Hot Chips Tutorial*, Aug 2012.

[37] D. U. Lee, K. W. Kim, K. W. Kim, H. Kim, J. Y. Kim, Y. J. Park, J. H. Kim, D. S. Kim, H. B. Park, J. W. Shin, J. H. Cho, K. H. Kwon, M. J. Kim, J. Lee, K. W. Park, B. Chung, and S. Hong, "25.2 A 1.2V 8Gb 8-channel 128GB/s high-bandwidth memory (HBM) stacked DRAM with effective microbump I/O test methods using 29nm process and TSV," in *ISSCC*, Feb 2014.

[38] D. Lee, Y. Kim, V. Seshadri, J. Liu, L. Subramanian, and O. Mutlu, "Tiered-Latency DRAM: A Low Latency and Low Cost DRAM Architecture," in *HPCA*, Feb 2013.

[39] H.-W. Lee, S.-B. Lim, J. Song, J.-B. Koo, D.-H. Kwon, J.-H. Kang, Y. Kim, Y.-J. Choi, K. Park, B.-T. Chung, and C. Kim, "A 283.2 μW 800Mb/s/pin DLL-based data self-aligner for Through-Silicon Via (TSV) interface," in *ISSCC*, Feb 2012.

[40] S. Li, J. Ahn, R. D. Strong, J. B. Brockman, D. M. Tullsen, and N. P. Jouppi, "The McPAT Framework for Multicore and Manycore Architectures: Simultaneously Modeling Power, Area, and Timing," *ACM TACO*, vol. 10, no. 1, Apr 2013.

[41] G. H. Loh and M. D. Hill, "Efficiently Enabling Conventional Block Sizes for Very Large Die-stacked DRAM Caches," in *MICRO*, Dec 2011.

[42] N. Madan, L. Zhao, N. Muralimanohar, A. Udipi, R. Balasubramonian, R. Iyer, S. Makineni, and D. Newell, "Optimizing Communication and Capacity in a 3D Stacked Reconfigurable Cache Hierarchy," in *HPCA*, Feb 2009.

[43] J. M. J. F. Martinez, "MORSE: Multi-objective Reconfigurable Self-optimizing Memory Scheduler," in *HPCA*, Feb 2012.

[44] Micron Technology Inc., *Calculating Memory System Power for DDR3*, 2007. [Online]. Available: http://www.micron.com/products/support/power-calc

[45] N. Miura, K. Kasuga, M. Saito, and T. Kuroda, "An 8Tb/s 1pJ/b 0.8mm2/Tb/s QDR Inductive-Coupling Interface Between 65nm CMOS GPU and 0.1μm DRAM," in *ISSCC*, Feb 2010.

[46] O. Mutlu and T. Moscibroda, "Parallelism-Aware Batch Scheduling: Enhancing both Performance and Fairness of Shared DRAM Systems," in *ISCA*, Jun 2008.

[47] S. O, Y. H. Son, N. S. Kim, and J. Ahn, "Row-Buffer Decoupling: A Case for Low-Latency DRAM Microarchitecture," in *ISCA*, Jun 2014.

[48] Y. Ohara, A. Noriki, K. Sakuma, K.-W. Lee, M. Murugesan, J. Bea, F. Yamada, T. Fukushima, T. Tanaka, and M. Koyanagi, "10 μm Fine Pitch Cu/Sn Micro-Bumps for 3-D Super-Chip Stack," in *3DIC*, Sep 2009.

[49] J. T. Pawlowski, "Hybrid Memory Cube," in *Hot Chips*, Aug 2011.

[50] S. Rixner, W. J. Dally, U. J. Kapasi, P. R. Mattson, and J. D. Owens, "Memory Access Scheduling," in *ISCA*, Jun 2000.

[51] Samsung Electronics, *DDR3 SDRAM Datasheet*, 2012.

[52] T. Sherwood, E. Perelman, G. Hamerly, and B. Calder, "Automatically Characterizing Large Scale Program Behavior," in *ASPLOS*, Oct 2002.

[53] T. Singh, J. Bell, and S. Southard, "Jaguar: A Next-Generation Low-power x86-64 Core," in *ISSCC*, Feb 2013.

[54] Y. H. Son, S. O, Y. Ro, J. W. Lee, and J. Ahn, "Reducing Memory Access Latency with Asymmetric DRAM Bank Organizations," in *ISCA*, Jun 2013.

[55] L. Subramanian, V. Seshadri, Y. Kim, B. Jaiyen, and O. Mutulu, "MISE: Providing Performance Predictability and Improving Fairness in Shared Main Memory Systems," in *HPCA*, Feb 2013.

[56] K. Sudan, N. Chatterjee, D. Nellans, M. Awasthi, R. Balasubramonian, and A. Davis, "Micro-pages: increasing DRAM efficiency with locality-aware data placement," in *ASPLOS*, Oct 2010.

[57] A. N. Udipi, N. Muralimanohar, N. Chatterjee, R. Balsubramonian, A. Davis, and N. P. Jouppi, "Rethinking DRAM Design and Organization for Energy-constrained Multi-cores," in *ISCA*, Jun 2010.

[58] T. Vogelsang, "Understanding the Energy Consumption of Dynamic Random Access Memories," in *MICRO*, Dec 2010.

[59] R. Weerasekera, J. R. Cubillo, and G. Katti, "Analysis of Signal Integrity (SI) Robustness in Through-Silicon Interposer (TSI) Interconnects," in *Electronics Packaging Technology Conference*, Dec 2012.

[60] S. C. Woo, M. Ohara, E. Torrie, J. P. Singh, and A. Gupta, "The SPLASH-2 Programs: Characterization and Methodological Considerations," in *ISCA*, Jun 1995.

[61] D. H. Yoon, J. Chang, N. Muralimanohar, and P. Ranganathan, "BOOM: Enabling Mobile Memory Based Low-Power Server DIMMs," in *ISCA*, Jun 2012.

[62] T. Zhang, K. Chen, C. Xu, G. Sun, T. Wang, and Y. Xie, "Half-DRAM: a High-bandwidth and Low-power DRAM Architecture from the Rethinking of Fine-grained Activation," in *ISCA*, Jun 2014.

[63] W. Zhao and Y. Cao, "New Generation of Predictive Technology Model for Sub-45nm Design Exploration," in *ISQED*, Mar 2006.

[64] H. Zheng, J. Lin, Z. Zhang, E. Gorbatov, H. David, and Z. Zhu, "Mini-Rank: Adaptive DRAM Architecture for Improving Memory Power Efficiency," in *MICRO*, Nov 2008.