Combining Memory and a Controller with Photonics through 3D-Stacking to Enable Scalable and Energy-Efficient Systems

Aniruddha N. Udipi† Naveen Muralimanohar‡ Rajeev Balsubramonian†
Al Davis† Norman P. Jouppi‡

†University of Utah
Salt Lake City, UT
{udipi, rajeev, ald}@cs.utah.edu

‡HP Labs
Palo Alto, CA
{naveen.muralimanohar, norm.jouppi}@hp.com

ABSTRACT

It is well-known that memory latency, energy, capacity, bandwidth, and scalability will be critical bottlenecks in future large-scale systems. This paper addresses these problems, focusing on the interface between the compute cores and memory, comprising the physical interconnect and the memory access protocol. For the physical interconnect, we study the prudent use of emerging silicon-photonic technology to reduce energy consumption and improve capacity scaling. We conclude that photonics are effective primarily to improve socket-edge bandwidth by breaking the pin barrier, and for use on heavily utilized links. For the access protocol, we propose a novel packet based interface that relinquishes most of the tight control that the memory controller holds in current systems and allows the memory modules to be more autonomous, improving flexibility and interoperability. The key enabler here is the introduction of a 3D-stacked *interface die* that allows both these optimizations *without modifying commodity memory dies*. The interface die handles all conversion between optics and electronics, as well as all low-level memory device control functionality. Communication beyond the interface die is fully electrical, with TSVs between dies and low-swing wires on-die. We show that such an approach results in substantially lowered energy consumption, reduced latency, better scalability to large capacities, and better support for heterogeneity and interoperability.

Categories and Subject Descriptors:
B.3 [**Memory Structures**]: Semiconductor Memories; B.4.3 [**Input/Output and Data Communications**]: Interconnections (subsystems) – *Fiber optics*; B.7.1 [**Integrated Circuits**]: Types and Design Styles – *Memory technologies*

General Terms: Design, Performance

Keywords: Photonics, 3D Stacking, Communication Protocols, DRAM

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
ISCA'11, June 4–8, 2011, San Jose, California, USA.
Copyright 2011 ACM 978-1-4503-0472-6/11/06 ...$10.00.

1. INTRODUCTION

It is clear that the primary challenge to scaling computing systems into the exascale realm is the efficient supply of large amounts of data to hundreds or thousands of compute cores. Increasing socket, core, and thread counts, combined with large datasets in modern scientific and commercial workloads will exert severe capacity and bandwidth pressure on the memory system. It will also likely be severely power-limited, since it already accounts for up to a fourth of total system power in large systems [9, 31]. There also appears to be a clear trend towards heterogeneous systems that incorporate both DRAM and Non-Volatile Memory (NVM) [15, 38]. The memory interface will thus be operating under challenging conditions, and should be designed carefully in order to provide low latency, high bandwidth, low power, and extreme scalability. We study in detail two major components of the memory interface - the physical interconnect and the access protocol - and outline a memory node suitable for future exascale systems.

The biggest impediment to building an efficient interconnect architecture is the fundamental nature of high-speed off-chip electrical interconnects. Neither pin-count nor per-pin bandwidth is projected to scale [22], limiting the number of channels connected to the processor. Per-channel capacity is also limited, since high-speed buses can only support a small number of DIMMs due to signal integrity and power issues [2]. Silicon photonics promise to alleviate many of these problems. They can provide tremendous bandwidth through high frequency operation and Dense Wavelength Division Multiplexing (DWDM). More importantly, the photonic waveguide can directly couple to optical I/Os on-chip without conversion into the electrical domain, unconstrained by the overheads or limitations of electrical pins. Their pitch is also small enough that multiple waveguides can be run in parallel to further increase bandwidth. While silicon photonics inherently have better properties than off-chip electrical interconnects, careful design is still required to fully exploit their potential while being energy-efficient. Also, the use of photonics should ideally not be invasive to the design of commodity memory dies.

With an efficient interconnect in place providing high capacity and bandwidth, the next big impediment to scalability of the memory interface is the tight control held by the memory controller over every micro-operation of the memory system. Changing this paradigm by relegating low-level

functionality to the memory modules themselves will help streamline the overall operation of the memory subsystem.

In this paper, we introduce a novel main memory architecture that makes two synergistic contributions: (i) it improves performance, power, capacity, and cost characteristics by leveraging optics to overcome the pin barrier *without impacting the design of high-volume memory chips*, and (ii) it improves flexibility, interoperability, and scalability by shifting low-level memory controller functionality to the memory modules and adopting a simple interface for communication between the processor and memory.

The key to both contributions lies in the recent emergence of 3D-stacked memory chips, a trend that is likely to see mainstream adoption, and is already on the technology roadmap of major memory vendors for imminent release [18, 26]. Based on these projections, the traditional memory dies on a DIMM will likely be replaced by a set of 3D stacks of memory chips. In such systems, we propose the addition of a novel *interface die* that is stacked with the memory dies and built in a CMOS logic process. Optical waveguides only penetrate this interface die. All conversion between the optical and electrical domains happens here, allowing the memory chips to be oblivious of the off-chip interconnect technology and not requiring the memory industry to invest in special optics-compliant dies. There is little need for optics within the 3D stack because of the availability of low-latency, high-bandwidth, and low-energy through-silicon vias (TSVs) [45] for inter-die communication, and energy efficient low-swing wires for intra-die communication. The additional die space on the interface die can further be used to implement the typical low-level functionalities of the memory controller. The on-processor memory controller now simply dispatches an address in a request packet and eventually receives a data response packet in a speculatively reserved time slot, instead of conventionally issuing ACT, CAS, PRE, $WAKEUP$, $SLEEP$, $REFRESH$, etc., while worrying about many timing constraints [24]. In order to avoid overloading the term, *memory controller* refers to the processor side controller for the rest of the paper. The equivalent logic on the interface die will be called the *stack controller*. Such an architecture is compelling for the following reasons:

- **Memory System Cost:** There is virtually no impact on the area and cost of the memory dies themselves. The same dies can be used with either large-scale photonic memory systems or small-scale single-user/handheld conventional electrical memory systems, eliminating the need for custom photonic dies. Similarly, the interface die will only need minor modifications to be customized for use with different kinds of memory (DRAM, PCM, STT-RAM, etc.). This will help increase manufacturing volumes for the interface die. These are important considerations for the cost-sensitive memory industry.
- **Supporting Heterogeneity:** It is likely that future memory systems will be a combination of different kinds of memory, including DRAM, PCM, STT-RAM, etc. [15, 38]. Each will have its own timing constraints; for instance, DRAM itself has as many as 18 parameters such as t_{FAW}, t_{RRD}, t_{WTR}, t_{RTP}, t_{WTW}, t_{RTRS}, etc., which have to be accounted for during scheduling. Each will also have its own maintenance requirements (wear-leveling, drift management [7], etc.) and specialized coding requirements [37]. All these factors significantly increase complexity. In the current paradigm, all of these will have to be hard coded into the memory controller, requiring explicit processor support every time a new memory technology is introduced, or even when new features are introduced. Putting all this functionality on the interface die allows different types of DIMMs to co-exist on the same memory channel, especially important when, for example, DRAM and PCM are used as different levels of the memory hierarchy.
- **Flexibility and Interoperability:** The current tight integration of device characteristics, interconnect characteristics, and memory controller operation severely restricts interoperability. The memory market today consists of scores of DIMM variants, with a range of speeds, protocols, reliability levels, buffering, and physical profiles, with little cross compatibility. This affects both users and vendors - users are faced with a bewildering array of purchase choices, and vendors have to deal with stock-keeping and compatibility validation issues for a large number of products. The use of a packet-based interface abstracts away these low-level details, and allows the DIMM and the processor to be more independent, with greater plug and play capabilities across different product lines, or even different memory technologies. It also opens the door to innovation and value-addition within the memory modules (in stark contrast to today's conventional DIMMs) while presenting a consistent interface to the rest of the system.
- **Reduced Energy Consumption:** The coupler between the off-chip fiber waveguide and on-chip silicon waveguide is a large contributor to photonic power loss along the light path. The interface die allows this cost to be shared by all memory dies in the stack, reducing the total input laser power required for a given capacity. Similarly, the static energy costs of operating photonic components (such as ring tuning heater power, more details in Section 2.2) are well amortized over several accesses, since they are shared by all memory dies in a stack, and can be kept well-utilized. This results in a substantial energy reduction compared to state-of-the-art designs that take photonics well into each memory die.
- **Efficient Ring Tuning:** A recent study [36] showed that the ring tuning heater power (again, more details follow in Section 2.2) can be reduced if the rings can be aggregated. Unlike prior work, the use of the interface die allows us to co-locate all rings and reduce heating costs.
- **Improved Performance:** The aggregation of rings on the interface die means that a given cache line transfer can take advantage of all the available optical bandwidth, yielding lower transfer latencies. This is especially important with low-energy DRAM chips [43] where entire cache lines are localized to a single subarray in a single chip. Spreading a given number of rings around the chip would effectively reduce the bandwidth available *for a particular cache line*, increasing serialization and queueing delays.

While the change to the memory interface is significant, we believe that such a massive change is inevitable given the convergence of disruptive technologies such as silicon photonics, NVM, 3D, etc., and the pressing need for scalability. In fact, an analogy can be drawn between our proposal and the SATA interface for storage devices. Early storage controllers issued every micro-command required to read data out of hard disk drives, such as the angle of rotation required, track movement required, etc. This was eventually

moved into the drives themselves, and the processor now only sees the higher-level SATA interface. This helped with the relatively painless adoption of flash-based SSDs, where an old hard disk drive could simply be swapped out for a new flash drive, despite a complete internal technology revamp and different operational requirements.

2. BACKGROUND AND RELATED WORK

For the rest of the paper, we use DRAM as an evaluation vehicle. However, the general design philosophy applies equally to emerging non-volatile memory, and to heterogeneous systems comprising both DRAM and NVM.

2.1 Low Energy DRAM Chips

Conventional DRAM systems have striped a single cache line across multiple large arrays and multiple DRAM chips, boosting storage density and the available pin bandwidth for the data transfer. It also leads to the overfetch of data into large row buffers. If the access stream has locality, subsequent requests can be serviced quickly by the row buffer. However, the move to multi-core has destroyed locality within memory access streams, rapidly diminishing the utility of this locality-optimized model. Consequently, the energy cost of overfetch has become a major impediment to realizing exascale DRAM systems [29, 42]. It is therefore expected that we will move to memory design styles where a cache line is serviced from a single array (or a small number of arrays) and the notion of a row buffer can be eliminated. A number of recent papers have suggested designs that fit this style [10, 16, 32, 43, 44]. We adopt a similar baseline DRAM chip design in our work. For the rest of the paper, we will assume that a cache line request is serviced by exactly one subarray (referred to as Single Subarray Access – SSA [43]). After read accesses, the bits are transferred over an *on-chip interconnect* to the I/O pins, and then to the processor. An important feature of this design is that the elimination of the row buffer results in fewer variations in access time (row buffer hits vs. misses, for example).

Henceforth, we refer to a single layer of memory as a *DRAM die*. A set of DRAM dies connected by TSVs form a *DRAM stack*. Each DRAM die on each stack is logically independent, and can be addressed and accessed individually. Sets of DRAM stacks may be physically organized into DIMMs, both to hold some common components like the power splitter (details follow), and for convenience as Stock Keeping Units (SKUs).

2.2 Photonic Interconnect Basics

While several implementations of integrated silicon photonics are currently under study [8, 11, 34, 41, 46], we focus on *micro-ring resonator* based technology, which uses "rings" to modulate and demodulate light. We refer the reader to prior work [23, 34, 44] for more details, and subsequently we only discuss issues directly impacting our study.

DWDM: A key advantage of this technology is the ability to independently modulate, transmit, and detect light with different wavelengths on a single channel through Dense Wave Division Multiplexing (DWDM), providing high bandwidth density. This is achieved by using a multitude of *wavelength selective* ring resonators to form a *ring bank* or a *photonic stop*. Although the number of wavelengths that can be supported is limited by coupling losses between rings (which increase with tighter wavelength spacing), prior studies have shown that DWDM with up to 67 wavelengths is achievable with small rings [46]. Combined with fast modulators capable of operating at up to 5 GHz (10 Gbps with dual-rate), a single waveguide is capable of providing roughly 80 GB/s of bandwidth on a 64-wavelength bus.

Static Power: Photonic modulating rings are sized to resonate at a specific wavelength at fabrication time. A key requirement for the implementation of DWDM is to keep the various ring resonators perfectly tuned to their desired wavelength at all times. However, this wavelength is highly temperature dependent, and tends to drift during operation as temperatures vary. To compensate for this drift, the rings need to be constantly heated, a process called *trimming*. Without trimming, not only will those specific detuned wavelengths be affected, but those rings may drift close to adjacent wavelengths, potentially causing interference and data loss. Note that this trimming has to be on *continuously*, independent of usage of that wavelength or even that die. This introduces a constant, non-trivial static power overhead. Similarly, the receivers have to stay on for the entire duration of operation, adding yet another static power source. It is therefore important to keep utilization high in photonic systems.

Laser Power: The laser power is a function of total photonic loss on the entire *light-path* from source to receiver; some pre-specified amount of photonic power, depending on the sensitivity of the receiver, has to finally reach the photodetector after all losses for reliable detection. Typical sources of loss include the on-chip silicon waveguide, off-chip fiber, couplers to go between on- and off-chip waveguides, the modulating rings, etc. With respect to the rings, it is important to note that at any given modulation point, *near rings* (idle rings tuned to the same wavelength or immediately adjacent wavelength at other modulation points on the light path) introduce two orders of magnitude more loss than *far rings* (all other rings). From the perspective of photonic loss, therefore, it is possible to have many hundreds of far rings coupled to a single waveguide, although such an organization may suffer from high trimming power overheads.

2.3 Target System and Methodology

We target the requirements of a future exascale computing system, while considering maturity and scaling projections for nanophotonic interconnect technology. An NSF-sponsored blue-ribbon panel [6] estimated that a single terascale node in a 2017 exascale system would have to support a capacity of 1 TB and a bandwidth of 1 TB/s. Assuming a theoretical bandwidth of 80 GB/s per waveguide and accounting for some inefficiency in achieved throughput, we provision 16 waveguides to reach the 1 TB/s goal. Each of these is considered an independent "channel", with a memory capacity of 64 GB to reach a total capacity of 1 TB. Each channel is physically organized as 32 dies of 2 GB each, with 16 independent banks per die. We assume a 3D stack depth of 8 DRAM dies [26], requiring four stacks per channel. We use the term DIMM to refer to these four stacks. For the rest of the discussion and evaluation, we will only focus on the design of a single channel.

We use an in-house cycle-accurate DRAM simulator for performance studies. We faithfully model the memory controller queue, address/command bus, data read bus, data write bus, bank contention, and refresh. We use a close-page

Parameter	Value
Technology	Yr 2017, 22nm, 5 GHz
DRAM Chip	16 banks, 60 sq. mm
DRAM Capacity	32 Chips for a total of 64 GB per channel
Bank Access Latency	7 ns
On Chip Ntk Latency	5 ns
Queue Size	64 entries each for RD and WR

(a) General parameters

Component	Power/Energy
Read Laser	1.22 mW
Write Laser	0.49 mW
Addr/Cmd Laser	0.24 mW
Modulator	47 fJ/bit
Receiver	440 μW/bit
Thermal Tuning	250 μW/ring
Full Swing Wire	9.7E-14 J/mm
Low Swing Wire	1.8E-14 J/mm
TSV	7.8E-14 J/bit [45]

(b) Energy parameters

Table 1: Methodology and energy parameters

row-buffer policy, as is the norm today for industry standard servers from all major vendors in an effort to improve performance in typical workload mixes with low locality [5, 25, 35]. We use a simple FCFS scheduling policy. We adopt address mapping policies from Jacob et al. [24]. We use measured or estimated energy and loss numbers for the photonic components based on published work [19, 34, 44, 46] and device-physics work currently in progress at HP Labs. Energy and latency numbers for the on-chip electronic interconnect, and DRAM latency and area characteristics are obtained from a modified version of Cacti 6.5 [1]. Simulation and energy parameters are listed in Table 1. Loss parameters are listed in Table 2.

We use Simics [4] to execute the PARSEC [12] benchmarks ("sim-large" dataset) and Stream [3] on an 8 core machine with private 16 KB L1s per core and a shared 256 KB L2 cache. Every simulator run consists of 5 million DRAM accesses, and we report average memory stall cycles and energy consumption per cache line access. In order to analyze a futuristic memory system with a large physical address space and heavy traffic with high throughput, however, most of these benchmark workloads are unsuitable since they have small data sets and relatively low bandwidth requirements. As a result, most of our analysis uses a synthetic traffic generator with a tunable injection rate and tunable fraction of write operations feeding the DRAM simulator. Prior work [10, 43] has shown that memory access patterns are becoming increasingly random and will likely become more so with increasing socket, core and thread counts. All of our synthetic analyses therefore use a random address pattern. A new request is injected into the channel once every C cycles on average, under a Bernoulli process to ensure that traffic is bursty [17]. With 64-byte cache lines, a 64-bit channel, and dual data rate, every data transfer takes 4 cycles. To prevent saturation, therefore, $C = 4$ is the maximum allowable request rate. This rate corresponds to the target of 1 TB/s described in the previous subsection. We show results for C ranging between 3 (saturation) and 10 cycles, stress-testing the channel to the maximum extent possible.

2.4 Related Work

Optically connected memory has only recently received the attention of the research community. Hadke et al. proposed the OCDIMM architecture [20], using photonic interconnects to replace the electrical channels in an FB-DIMM system, resulting in better performance, energy and scalability characteristics than electrical interconnects. However, that design did not fully exploit two major advantages of silicon nanophotonics - large bandwidth density through DWDM and the ability to break the pin barrier. We address both these concerns in our design. Corona [44] considered the system level implications of photonic technology and briefly explored optically connected 3D memory stacks, but did not analyze in detail the performance or energy tradeoffs involved in such designs. It also did not propose changes to the memory access protocol. Beamer et al. proposed the specially designed PIDRAM chip [10], with aggressive photonic penetration well into the DRAM chip. We consider this design in detail and provide comparisons in Section 3.2.

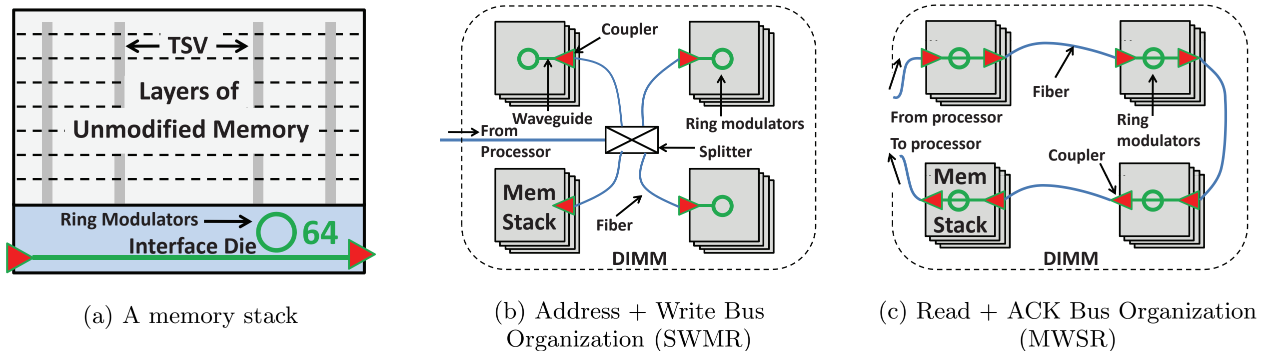
Several papers have explored leveraging photonic interconnects, especially for on-chip communication [14, 27, 28, 47], which has an entirely different set of tradeoffs. 3D stacked DRAM designs have also been explored by several papers [13, 32, 45]. However, these study DRAM stacked directly on top of processing cores, thus dealing with much smaller scale memories, with a different set of constraints than ours, including capacity, thermals, etc.

3. PHOTONICALLY CONNECTED 3D-STACKED MEMORY

Silicon photonics have the potential to provide substantial energy and performance advantages. However, careful consideration of various factors, including the presence of low-energy and low-swing interconnects for on-die communication, the presence of low-energy TSVs in a 3D stack, the trimming power reduction when photonic rings are aggregated, number of stacks on the channel, etc. is required to ensure an energy- and performance-optimal organization.

Based on this, we propose a novel memory architecture that uses photonics to break the pin barrier on a memory stack, but isolates all photonic components to a dedicated 3D stacked die called the *interface die.* The address and data waveguides only penetrate this die, where conversion between the photonic and electrical domains occurs; all intra-stack communication is electrical, either on TSVs or low-swing wires. Read data, for instance, is read out of the array, travels laterally on its local die via conventional electrical interconnects to the nearest TSV, where it travels vertically to the interface die. Here, it is converted to the photonic domain, and shipped off to the processor. Write data, as expected, would follow the same path, but backwards. An example memory stack is shown in Figure 1(a).

We first outline the basic organization and architecture of the system in Section 3.1. We then substantiate our design point with detailed energy analysis in Section 3.2. Performance analysis follows, first in Section 3.3 and then in Section 4 after we describe our access protocol. Finally, Section 3.4 studies the thermal impact of our design.



(a) A memory stack

(b) Address + Write Bus Organization (SWMR)

(c) Read + ACK Bus Organization (MWSR)

Figure 1: Memory channel organization

3.1 Proposed Memory System Architecture

Each memory channel consists of four logical buses - a 16-bit address/command bus, a 32-bit write-data bus, a 64-bit read-data bus, and a 3-bit acknowledgement bus. Since each waveguide can support up to 67 bits through DWDM, these four buses can be combined into two physical fibers, one outbound (address/command + write-data), and one inbound (read-data + acknowledgement). Note that the wavelengths used for each of these are exclusive, and the buses are completely independent, though physically on the same fiber.

Layout of Ring Resonators: We define a *photonic stop* as a set of wavelength-selective rings. The read-data bus, for example, has 64 rings per stop. A DRAM die can contain one or more photonic stops. PS denotes the number of photonic stops. Multiple stops would be distributed symmetrically on the DRAM die, similar to nodes in a hypothetical H-tree network. If two photonic stops are present, each services subarrays in half of the chip; if four stops are present, each services a quarter of the chip, and so forth. Communication between the DRAM arrays and the nearest photonic stop is fully electrical.

Address/Command Bus: In the interest of simplicity, we assume that the address and associated command signals are broadcast to all stacks in the channel. Photonic broadcasts are complicated since every ring coupled to the waveguide has to be fine-tuned to not simply remove all light during demodulation, but only remove a fraction of the total optical power, allowing the rest of the photons to pass downstream. A simpler approach to building a single-writer, multiple-reader (SWMR) photonic interconnect is to use a Y-splitter to divide the incoming laser power among multiple fibers, as shown in Figure 1(b). While it introduces some insertion loss, only one splitter is used on each DIMM, making the overhead tolerable. Modulated light reaches every photonic stop in the channel, where it is converted to the electrical domain before being decoded. The traffic on the address/command bus is fixed for a given channel bandwidth (cache line requests per unit time). Accordingly, we provision a small 16-bit bus (fast and pipelined) to keep utilization high and energy consumption low.

Write-Data Bus: Since write traffic is typically lower than reads (about 25% of all traffic in our benchmarks), we provision only half as much peak bandwidth as for reads. With a separate dedicated data bus for writes, we avoid contention and turnaround issues with respect to read operations.

Read-Data Bus: The *read-data* bus is a multiple writer, single reader (MWSR) system, implemented with a daisy chain as shown in Figure 1(c). Light originates at the processor, travels on optical fiber between memory stacks and silicon waveguides on-chip, and arrives back at detectors on the processor. It passes by, and is coupled to, several photonic stops in the channel along the way - these can be distributed among the many stacks on the channel. One of the photonic stops (as determined by some form of arbitration - see Section 4) modulates the light to carry the appropriate cache line data, and this is detected at the processor.

Acknowledgement Bus: ACK/NACK signals are required for our novel access protocol (Section 4), for which we provision a 3-bit MWSR bus.

Photonic Power Loss: As the number of DRAM stacks on the channel increases, or the number of rings on each stack increases, there is an increase in the total loss, increasing the input laser power required. With a small number of DRAM stacks per channel (a maximum of four in this work), however, total loss can be contained to about 13 dB, as shown in Table 2. As we scale beyond this, solutions like the *guided photonic bus* have been shown to be effective in actively guiding power to only the target DRAM stack or die [10]. This will reduce the loss for a given source-destination pair in the system, providing further scalability in spite of the daisy-chained nature of the architecture.

Component	Unit	Qty SWMR	Qty MWSR
Si Waveguide	0.3 dB/cm	3.25	6.5
Fiber	0.4 dB/Km	0.01	0.01
Coupler	1.0 dB [39]	2	10
4-way splitter	8.0 dB	1	0
Near ring loss	0.17 dB	0	6
Far ring loss	0.001 dB	48	262
Total Loss	dB	11.03	13.24
Laser Efficiency 32% Receiver Sensitivity 1.08 A/W Receiver Current Swing 20 μA			

Table 2: Photonic parameters and losses in a 4-stack channel, with 1 stop per stack

3.2 Energy Considerations

The proposed design helps reduce energy across several fronts, compared to state-of-the-art photonic DRAMs:

(i) As described previously, photonic components consume non-trivial amounts of power for trimming and other analog operations, independent of bus activity. The key take-away for our design is that photonic resources cannot be over-provisioned without a significant energy impact. It is

essential to provide the minimal photonic resources required to exploit the benefits of the technology, and run this at as high a utilization rate as possible to amortize the static energy costs over many useful transfers per unit time. On the other hand, electrical interconnects can be over-provisioned because their static energy consumption is relatively low. An efficient design should therefore ideally use photonics for heavily utilized, shared channels, while more localized communication with relatively low utilization is best handled electrically. The interface die helps do exactly this, providing substantial static energy savings by eliminating the need to have photonic rings on every die, while still breaking the pin barrier.

(ii) The use of low-swing wiring [21] for on-die traversal provides dual benefit. First, it reduces the dynamic energy consumption of electrical wires by 4-8X, saving electrical communication energy. Second, the reduced electrical communication costs allow the optimal design to have fewer photonic stops overall, reducing the static energy costs of running these stops.

(iii) Having all photonic components on a single die reduces the number of fiber-waveguide couplers (which can be relatively lossy components) required by a factor of 8 (for an 8 memory die stack), helping reduce total power loss. Low-cost TSVs [45] are used instead for inter-die communication. This helps reduce the input laser power required.

(iv) Recent work [36] has shown that trimming power is dependent on the total area being heated, and aggregating rings on the interface die helps reduce some of this energy.

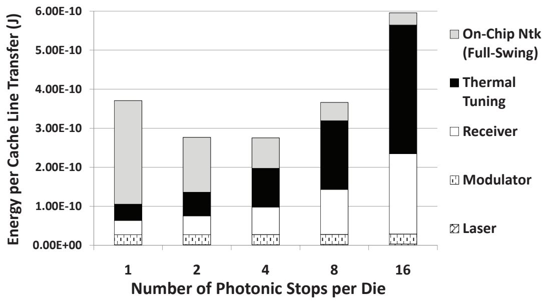
Energy Calculation Methodology: The energy consumption of the channel consists of 3 components - read-data bus, write-data bus, and address/command bus. We report total energy as the sum of the average read and write energy per cache line, weighted by the fraction of requests that are reads or writes, and the address/command bus energy. For most of our energy analysis, we aggressively assume the peak bus utilization of about 75% (50% for writes), just below saturation. This shows photonics in the best light possible, allowing maximum amortization of its static energy. Our device energy parameters are listed in Table 1. For our initial graphs, we assume that the trimming power scales up linearly with the number of rings. In a subsequent graph, we also consider insight from a recent paper [36] that shows a reduction in trimming power when rings are aggregated.

Prior Work: We start by evaluating a *single die* system, *i.e.*, the channel is attached to just one DRAM die, with no 3D stacking or daisy-chaining, similar to Beamer et al.'s state-of-the-art PIDRAM design [10]. The entire cache line is fetched from subarrays connected to one of the photonic stops. Figure 2(a) shows the average energy consumption per cache line transfer in such a system, with a varying number of photonic stops, PS. We see two conflicting effects as PS increases. First, there is a reduction in the electrical energy spent in transferring data between the array and the photonic stop since the distance to the nearest stop is lowered. On the other hand, there are now many more rings on the die, and since only one stop can actively be transmitting data at any given time, there is increased static power overhead, increasing the average photonic energy per useful data transfer. These two factors balance out at 4 stops, a trend similar to the one described in [10]. Note that the actual balance point is dependent on several system parameter assumptions like the die area, static power overhead, etc. In general, however, this argues for specialized memory with photonics going well into the die, and rings spread throughout the die.

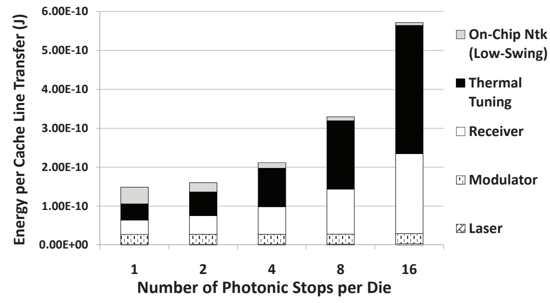
Using Low Swing Wiring: As Figure 2(b) shows, the trend starts to shift when the on-chip communication is made more efficient through the use of low-swing wires, since this emphasizes the overhead of increased photonic resources. In this configuration, a single-stop design is energy-optimal. In absolute terms, the best low-swing configuration (1 stop) has an energy consumption 46% lower than the best full-swing configuration (4 stops), clearly arguing for minimal photonic use with more efficient on-chip communication.

Basic 3D Extension: The analysis presented so far was for a single-die DRAM channel, but to allow capacity scaling, each channel will likely support more than a single die. A first-step towards scaling capacity is 3D stacking of DRAM dies [18, 26]. The straightforward approach to building photonic 3D DRAM would be to simply stack the photonic DRAM dies. The optical signal would now have to traverse dies at multiple levels, traveling on waveguides on each die, with some form of vertical coupling (perhaps free space optics) to go between dies. While the increased loss can be handled by hierarchical power guiding [10], there are now even more rings idling away while exactly one stop on exactly one of the dies in the stack is actually modulating light with useful data. For example, with an 8-deep DRAM stack, and 4 stops per die, there are a total of 2048 rings in the channel, with only 64 operational at any given time. This further emphasizes the importance of the photonic components' static power over the dynamic energy in electrical wires. The energy - PS curve shifts accordingly, and a system with a single photonic stop per die becomes energy-optimal, with *both* low-swing and full swing electrical interconnect assumptions (we don't graph this due to space constraints). Note that this design still has 8X more rings (one stop on each of the eight dies) than can be usefully employed at a time, idling away static power due to under-utilization. The most efficient 3D configuration consumes 2.4X the energy of a single die channel, for 8X capacity gain. This is clearly inefficient.

Proposed Single-Stack Design: We next consider our novel model with a single stack of DRAM chips. Photonic components are only placed on the interface die and TSVs are used for inter-die communication. Now that we only have one photonic die in the DRAM channel, the decision on the number of photonic stops on the die is almost exactly the same as the *single-die* system discussed previously, except that the non-photonic energy now includes TSVs (this vertical communication came virtually for free with photonics due to their distance-independent energy consumption). The energy-optimal design point continues to be either 4 photonic stops with full-swing wires or 1 stop with low-swing wires. This final design point (single photonic die, 1 stop per die, low-swing wires + TSVs) consumes 48% less energy than the best simplistic 3D design (8 photonic dies, 1 stop per die, low-swing wiring + free vertical communication), due to the reduced number of photonic stops, despite the additional TSV energy. There is no loss in performance between the fully-optical 3D stack and our proposed design since the TSVs can handle very high bandwidths [45] and

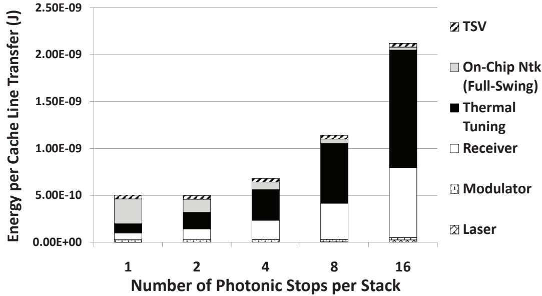


(a) With full-swing on-chip wires

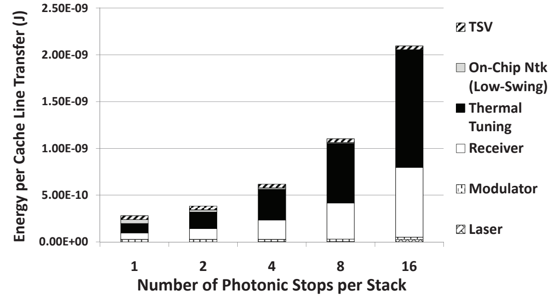


(b) With low-swing on-chip wires

Figure 2: Determination of optimal PS in a *single-die* channel (1 DRAM die, no stacking or daisy-chaining)



(a) With full-swing on-chip wires



(b) With low-swing on-chip wires

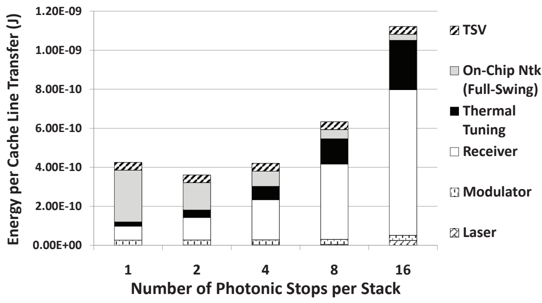
Figure 3: Determination of optimal PS in a daisy-chained *four-stack (32 die)* channel

the channel still gets its provisioned bandwidth. There is also no capacity loss, and no need for modification of the memory dies. This argues that despite photonic energy being virtually distance-independent, the static energy costs mean that photonic penetration should be minimized.

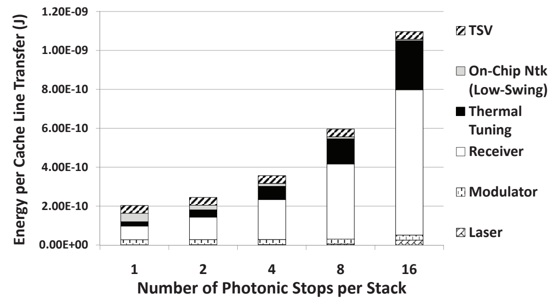
Proposed Daisy-Chained Multi-Stack Design: Finally, we explore a more realistic, expanded channel, with four stacks daisy-chained together on the same waveguide, each stack comprising 8 memory dies [26]. The operation of the channel is similar to the single stack design, except there are now more stops coupled to the waveguide. As before, only one of these can actively modulate the data waveguide at any given time, and this is handled by some form of arbitration, which we describe in Section 4. Such a design provides two major benefits: first, there is increased capacity on a single channel, a critical metric in servers for increased scalability. Second, it provides a large number of independent banks that can operate in parallel to keep the shared photonic channel well utilized, improving the photonic energy characteristics, as well as improving overall performance. However, to enable a workload-independent comparison, we do not quantify this benefit, instead assuming a fixed utilization figure for all the designs under study.

We repeat our photonic stop study in a system with four DRAM stacks (a total of 32 dies) on a single waveguide, the likely use case for future optically connected memory. As seen in Figure 3, once again, the static energy of the photonic components dominates. It is clear that as the number of dies connected to the channel rises, it is more energy efficient to use a smaller number of photonic stops per stack. This supports our hypothesis that photonics should only be used to overcome the pin barrier and cheap electrical interconnects should be used within a stack. This design point (4 stacks, 1 stop per stack, with low-swing interconnects and TSVs between dies) still consumes 23% less energy than the best fully-optical 3D extension of state of the art photonic DRAMs (1 stack, 8 stops per stack, with low-swing interconnects and photonic vertical communication, no TSV energy), while providing 4X capacity, and the potential for improved performance and further reduced energy consumption due to the increased number of banks.

Sensitivity to Ring Trimming Power: Recent work has indicated that the trimming power may be less dependent on the absolute number of rings, and more correlated with the total die area, ambient temperature, and other factors [36]. However, that analysis was for large dies (300-400 mm^2) with several thousand rings spread across the die, unlike our 60 mm^2 DRAM dies with a few hundred rings at most, concentrated in "stops". In light of these differences, we don't expect our trimming power to be independent of ring count, as claimed in that paper. We developed a simple thermal current flow model with rings laid out in a 2D array and a heater in the center of that area. In steady state, the power that the heater needs to supply equals the total heat that gets dissipated from the array, which is proportional to the *perimeter* of the array. There is negligible heat flow in the +Z and -Z directions since a thermal insulator is expected to be present to prevent heat loss. Based on this model, we arrive at an aggressive estimate for ring trimming power of about 50μW/ring for a 64-ring stop (a fifth of our original assumption). Figure 4 shows an energy analysis for our final multi-stack design point based on this estimate. As expected, reducing the trimming power de-emphasizes the photonic energy component, though other static com-

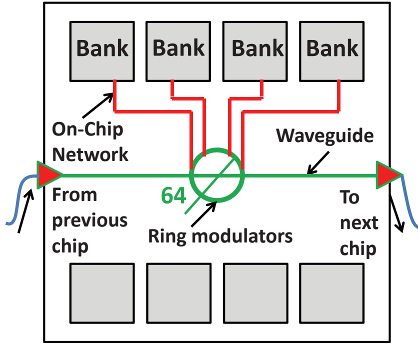


(a) With full-swing on-chip wires

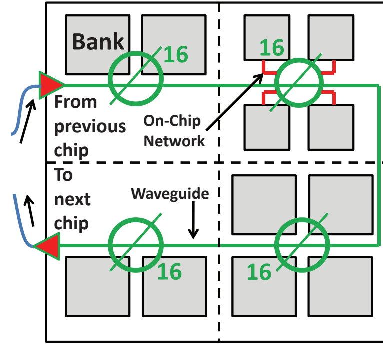


(b) With low-swing on-chip wires

Figure 4: Energy analysis with an aggressive trimming power assumption - 50μW/ring



(a) Rings concentrated in one stop



(b) Rings distributed as "mini-stops"

Figure 5: Utilizing a given ring budget on a die

ponents such as the analog receiver circuitry now begin to show up more dominantly. As a result, the optimal point may shift towards having more photonic stops. However, we continue to see that a single photonic stop is optimal when using low-swing wires.

Impact of Traffic: As we reduce injection rate from one in 4 cycles to one in 10, utilization falls from 75% for reads/50% for writes to 30% for reads/20% for writes. The static photonic energy has to be amortized over fewer useful transfers, increasing energy consumption per cache line by as much as 88%, arguing for a design that maximizes utilization.

Summary of Energy Analysis: The analysis presented above shows that several factors influence the appropriate application of silicon photonics, including on-die electrical communication efficiency, channel organization, 3D die stacking, traffic, utilization, etc. In the expected usage scenario in a terascale memory node for future exascale systems, there isn't a need to take photonics well into memory chips, instead using them only to break the pin barrier.

The proposed single-stack design with an interface die reduces energy consumption by 48% with no loss in capacity or performance, compared to a fully photonic 3D stack, while avoiding the need for disruptive memory die changes. The daisy-chained design provides at least 23% energy reduction compared to a fully photonic 3D stack, with 4X capacity, and potential for performance improvements and further energy reduction due to increased bank count.

3.3 Utilizing a Given Ring Budget

Once we have defined the "photonic power budget", *i.e.* the number of photonic stops on a memory stack, there are two ways to organize the rings within a photonic stop, as shown in Figure 5. The *concentrated architecture* provides the entire bandwidth to a single request, significantly reducing serialization and queuing latencies. However, it may require a relatively large on-chip electrical traversal to go from the arrays to the stop, perhaps increasing energy consumption. The *distributed architecture* splits the total available bandwidth among several "mini" stops (through mutually exclusive wavelengths), simultaneously servicing multiple requests. Since arrays are closer to photonic stops, electrical traversal is reduced, but individual accesses suffer from increased serialization and queueing delays. Going from 1 stop to 2, 4, and 8 "mini-stops", net energy reduced by 7%, 10%, and 12%. Corresponding latency increases were 10%, 30%, and 67%, which may also affect system-level power consumption. We believe that the modest energy saving is not worth the significant loss in performance.

Note that we could, instead, stripe a cache line across multiple subarrays so that each mini-stop and its corresponding subarrays can handle a portion of the cache line - there would then be no impact on performance in moving from a centralized to a distributed organization. However, assuming that the subarray size is fixed, the extent of overfetch increases as more subarrays are involved in servicing a cache line, so we do not consider this method of operation. Based on this analysis, we believe that rings should be concentrated into one centralized photonic stop when attempting to optimize both performance and energy.

3.4 Impact on Thermals

We used Hotspot 5.0 [40] to perform thermal evaluation of our designs based on peak activity factors from our perfor-

mance simulations (Section 4). The baseline 3D stack with 8 DRAM dies settled at a temperature of 320.15K. Adding a photonic interface die, including a layer of SiO_2 required to build optical components resulted in a slight rise in temperature at the DRAM to 320.21K. The less than 0.1K increase in temperature will have practically no impact on refresh. Another interesting side-effect of the SiO_2 layer is that it helps thermally isolate the rings, reducing the amount of heat that needs to be supplied to keep the rings correctly tuned. Note that prior work on 3D DRAM architectures [13, 32, 45] has assumed DRAM stacked on top of processing cores, making thermals much more of a concern than in our study with only DRAM dies.

4. PACKET-BASED INTERFACE AND STACK CONTROLLERS

We believe that a memory system overhaul should not be limited to the DRAM chip or the channel's physical interconnect, but should also include a redesigned interface protocol. Aided by rapidly advancing technologies such as silicon photonics and 3D stacking, future memory systems can potentially support very large capacities at theoretically very large peak bandwidths. This comes at the cost of increased pressure on the address/command bus, and large amounts of state to manage potentially hundreds of independent banks, with further complexity to handle heterogeneous systems. A new interface is required to overcome these bottlenecks, and efficiently translate theoretical peak bandwidth to actual sustained throughput, and is a vital component in designing a scalable interface for terascale memory. The introduction of an interface die on the memory stack opens up the interesting possibility of shifting some memory controller functionality to the interface die, streamlining the operation of the memory system. This section describes such an interface in more detail.

4.1 Interface design

We start by making the memory more autonomous – the memory controller deals with it at a much higher level than existing protocols. We propose moving all low-level device management to the interface die in the memory stacks. Essentially, the *only* information the memory module requires is the address of the block being requested and the kind of operation - read or write. This is sent to the memory pool in a *request packet*, over the *address channel* (Figure 1(b)). The memory modules then decode this packet, and the appropriate module fetches the requested block. This is returned to the memory controller via a *data packet* over the shared *data channel* (Figure 1(c)). Using such a shared resource typically involves arbitration, leading to non-trivial performance penalties. In this particular case, however, we are presented with a special scenario where every transfer on the data bus occurs due to an explicit request by a single entity – the memory controller. Coupled with the fact that close-page memory organizations offer more deterministic access latencies, time slots on the data bus can be reserved apriori, obviating the need for the memory modules to go through an arbitration stage before every data transfer. Unlike conventional DDR systems, there isn't a slew of commands from the controller to handle every micro-operation (bank activation, column select, precharge, etc.).

Basic Operation: As discussed previously, sustaining high bandwidth is a fundamental focus of the redesigned interface

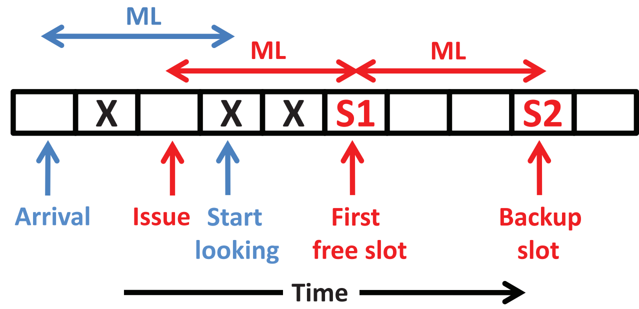


Figure 6: Proposed slot reservation scheme

for both performance and power reasons. We therefore make the availability of data I/O at the processor-side memory controller on the return path the minimum unit of arbitration. Every incoming request (for one cache line) is placed in a queue of finite length on the processor chip and arbitrates for a *slot* to use the data I/O for the eventual response. The width of the time slot is determined by the size of the cache line and the width of the data path. With 64-byte cache lines and a 64-bit data channel, for example, one request can ideally be serviced every 4 cycles (assuming a fully pipelined data bus, with dual data rate transmission), yielding a slot that is 4 cycles wide. The position of the reserved slot is determined by the total "memory latency" (ML cycles, say) - comprising the command transfer, bank access, on-chip network access and data return latency. Common timing constraints such as t_{RAS}, t_{RCD}, t_{CAS}, etc. are subsumed under ML and handled by the stack controller. The earliest available slot that is at least ML away is reserved for the request (see Figure 6). The request is then issued *just-in-time*, *i.e.*, the address transfer, bank access, on-chip network traversal, etc. all occur timed in such a way that the data can use the processor I/O at exactly the reserved slot; the resources for these preceding operations are automatically reserved, eliminating the need for separate arbitration. There also isn't a need for an accompanying tag to help match the response to the corresponding request. Similarly, the DIMM is also aware that it must provide its data response a fixed latency after it receives the request – so the request need not carry a deadline with it. The memory controller and the interface die can exchange information such as the value of ML for each memory module, regardless of the technology used, through the use of control registers during boot-time.

Handling Conflicts: Clearly, the above design works well in the expected usage scenario - when the latency within the memory is consistent and predictable, and where bank conflicts and timing constraint violations are uncommon because of the large number of banks. However, conflicts can occasionally happen. For example, in the interface just described, a new *slot* is alloted every 4 cycles. Even if two requests go to the same DIMM, they can still be completed with a 4-cycle offset because of bank-level parallelism on the DIMM. But if two requests go to the same bank, the second request cannot start until after the first has completed. There might also be other constraints that hold the access up, such as refresh, scrubbing, low-power wake-up, t_{FAW}, etc. These prevent the request from having its data ready at its expected deadline. The memory controller on the processor die could not have anticipated this because, by design, it no longer deals with the minutiae of per-bank state.

To handle such situations, we introduce the following novel mechanism. In addition to the data return slot described above (called Slot 1, say), each request speculatively reserves a second slot, called Slot 2 (see Figure 6). Slot 1 is chosen to be at least ML away in time from the cycle of issue, including the *uncontended* bank access time, exactly as before. If, for whatever reason, the bank was busy at that time, a negative acknowledgement (NACK) is returned by the memory module in Slot 1 (on the acknowledgement bus), and the actual data follows in Slot 2. This wastes Slot 1, reducing effective bandwidth, but this scenario is infrequent enough to have minimal impact. Note that Slot 2 has to be placed far enough away that if data is indeed returned in Slot 1 (the common case), the speculative reservation can be cleared in time for a subsequent request to use it. The gap between Slot 1 and Slot 2 has to therefore be at least ML. This results in some additional latency for Slot 2 returns, but ensures that it is not wasted in the common case. Again, Slot 2 need not be explicitly conveyed with the request packet, and is some previously agreed upon (through control registers) time from Slot 1. The scheme therefore still works without the overheads of tags.

Other Scenarios: A contrived example scenario that is yet to be considered is when a long series of requests are all targeted at the same bank. In this case, neither Slot 1 nor Slot 2 will be able to return data beyond a certain number of requests since accesses will essentially have to be spaced apart by a time equal to the bank access time (and not the 4 cycle slot). In such a situation, NACKs are returned in both Slot 1 and Slot 2, and the request has to be retried at a future time. The memory controller backs off for a fixed time, and then arbitrates for a slot and issues the request. This process can potentially be repeated multiple times, consuming resources in the form of queue occupation, wasted return slots, and address re-transmission energy, but is infrequent enough to not impact overall performance.

Handling Writes: As in most modern memory systems, we assume that writes are buffered at the memory controller, prioritizing read requests. When the write queue fills up to a pre-determined level, a burst of write requests is issued. This helps avoid frequent "turnaround" delays in the on-stack interconnect (note that there is no bus turnaround since we use a dedicated write bus). It also reduces the frequency of non-uniformity in access time that might arise due to DRAM timing constraints if writes and reads are interleaved arbitrarily [24]. Write data is sent out over a relatively narrow data bus over multiple cycles. If the bank is free, the write operation is performed. If not, a NACK is sent back (on a pre-reserved slot on the ack-bus) and the request is retried, wasting energy, but such situations are, again, infrequent. Our evaluations show that writes had to be refused in only about 5% of requests, and are almost always completed with the first retry. Alternatively, such requests could be buffered on the memory module, and NACKed only if that buffer is full. We do not consider this optimization in this work.

Handling Heterogeneity: With the new interface, the memory controller only needs knowledge of the access time ML of the memory modules to reserve an appropriate slot, which can be saved in control registers on setup, much like existing DRAM systems. The rest of the operation is completely independent of the underlying memory technology, enabling the support of heterogeneous systems.

4.2 Impact and Results

4.2.1 General Benefits

The proposed packet-based scheme reduces complexity in the memory interface, freeing the memory controller from having to deal with mundane tasks like timing parameter control, refresh, sleep modes, etc. Abstracting away these low-level details simplifies the use of heterogeneous memory systems, since the memory controller only needs a limited set of parameters from each module to operate. With this ability, data movement between these modules becomes more efficient, eliminating the overhead of multiple hops between distinct memory controllers, reducing both hop latency and bandwidth usage. This will be very useful if the two modules are at different levels of the memory hierarchy, requiring frequent data transfers. It also improves inter-operability, allowing plug-and-play style operation of different kinds of memory modules without explicitly requiring memory controller redesign and support. It also offers memory module vendors more flexibility and the ability to innovate, allowing them to customize their DIMMs, unconstrained as long as their interface die supports the new features. It also simplifies the arbitration process, requiring only a single point of arbitration – the data bus slots.

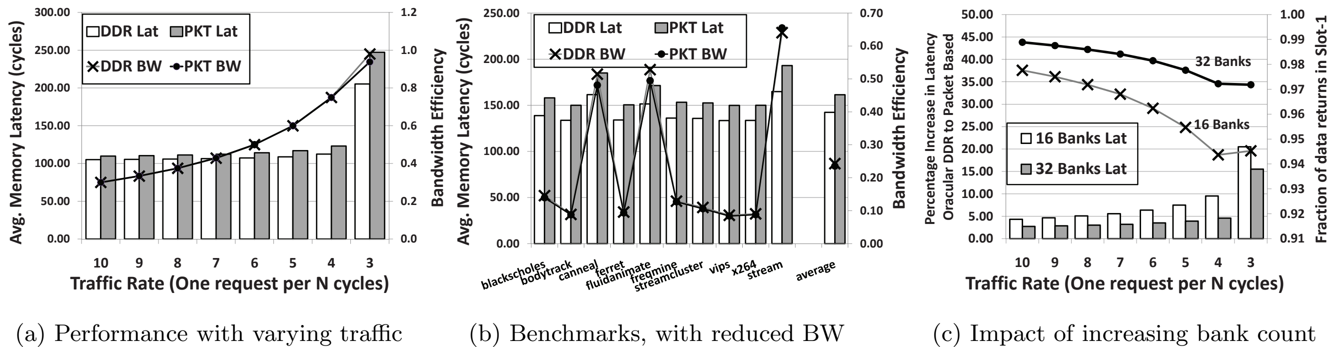
4.2.2 Address/Command Bus Energy Savings

In current low-level DDR interfaces with a capacity of 64 GB and 64 byte cache lines, each transaction consists of 36 address bits, 6 command bits (RAS and CAS; we assume an implicit precharge), and 512 data bits, for a total of 554 bits. Aggressive power-down schemes have been shown to be both essential [33], and easily achievable [43] in large-scale systems to keep idle power in check. With a large number of independent banks/arrays (as in our target system), potentially *every* request would therefore also involve a *Wake-up*, and a *Sleep* command, adding a further 58 bits [1]. Refresh, scrubbing, or other maintenance commands also add traffic on the bus in conventional systems, but these are harder to quantify. Every request would therefore transmit 600+ bits on the channel. A self-managing packet-based system such as ours, on the other hand, would only need to transmit the address and data, totalling to about 550 bits. This translates to a non-trivial energy saving of about 10% in the bit transport energy on every transaction.

4.2.3 Memory Controller Complexity

Memory controller complexity is expected to grow substantially as the number of banks increases and heterogeneity becomes more common. In our proposed model, the memory controller simply buffers requests, tracks the next available data slot, and issues the request accordingly. Thus, the logic that handles timing constraints and maintenance operations can be taken out of the memory controller, saving valuable processor die area. This is significant since future multi-core processors are expected to have multiple memory controllers, growing as the square root of the number of processing clusters [30].

[1] Assuming a power-down granularity of 1024-bytes, the total capacity consists of 64M power-down "domains", which need 26 address bits. Adding 3 command-bits, and multiplying by 2 for "sleep" and "wake-up", we arrive at 58-bits. This can vary based on sleep granularity.



(a) Performance with varying traffic (b) Benchmarks, with reduced BW (c) Impact of increasing bank count

Figure 7: Comparing oracular DDRx and unscheduled packet-based protocols

The stack controllers on the interface dies in the memory must now implement this logic, introducing some overhead due to replication in multi-stack channels. However, the interface die is already being employed to simplify photonic integration and has sparse functionality on it, providing enough dead silicon to introduce some logic functionality without severely impacting area or cost.

4.2.4 Protocol Performance Characteristics

The new interface's lack of global system knowledge may sometimes lead to sub-optimal scheduling decisions, resulting in a minor negative performance impact compared to a traditional low-level DDRx[2] standard interface. It is speculative and relies on a coarse grained approach to resource management, with a simple retry mechanism as a safety net. Mis-scheduled requests may also suffer an added latency penalty beyond that required for resources to become free. For example, a request that could not be serviced in Slot-1, but could potentially have been serviced say 2 cycles later, is now made to wait until Slot-2. These effects may also have a combined impact on the bandwidth. The DDRx system on the other hand is oracular in nature and the memory controller is fully aware of the entire memory system. However, our results show that even with these disadvantages, the simple packet-based interface reaches performance levels very close to the oracular DDRx interface.

Even under peak load, less than 5% of requests use Slot-2, and only about 0.5% need to back-off and retry, ensuring that the overall performance loss is small. Figure 7(a) compares the packet interface and the oracular DDRx interface under varying load, for the final design point described in Section 3.2. There is practically no difference in achieved bandwidth (the line graphs show this as a fraction of total provisioned bandwidth), and even in the heavily loaded case, there is less than 0.01% degradation. The latency characteristics are dependent on the load. At a traffic rate of one in 10 cycles, the latency penalty is just over 4%, rising to about 9% at a peak load of one in 4 cycles. With lower traffic, there are fewer possibilities of the conflicts that raise access latency in the packet-based system.

Both latency and bandwidth trends with real benchmarks are very similar to those with synthetic traffic, as shown in Figure 7(b). Since these applications have relatively low bandwidth requirements, we show results for a scaled down channel with a provisioned bandwidth of 10 GB/s (an 8-bit wide bus), and even this is not fully utilized.

Sensitivity Analyses: The impact of changing the channel organization (number of stacks, number of dies per stack, etc.) is primarily due to the change in bank count, which determines how frequent conflicts are. Figure 7(c) shows the impact of increasing the number of banks per DRAM chip in the system. The bar graph shows the percentage increase in latency moving from an oracular DDRx protocol to the unscheduled packet-based protocol. As expected, increasing bank count reduces the negative performance impact. At peak load, the latency impact reduces from 9% with 16 banks to just 4.5% with 32 banks. At low loads, increasing bank count bridges the gap between the packet-based and oracular implementations from 4.3% to just 2.7%. The line graphs correspondingly show the fraction of accesses that only use 'Slot-1' in the packet-protocol, i.e. cases where there was no resource conflict. This metric improves from about 94% to 97% as the bank count is increased, even at peak load. Current DRAM parts already have 8 banks, and it is very likely that this will rise to 16 or 32 in the future, making the performance losses negligible. Another sensitivity analysis we carried out was to artificially make the traffic extra-bursty. While both DDR and the packet-based protocol suffered latency increases, the performance difference between the two only strayed marginally from the standard Bernoulli injection case.

5. CONCLUSIONS

In this paper, we attempt to improve the modern memory interface by studying two of its major components - the physical interconnect and the communication protocol. As regards the physical interconnect, the primary limitation to scalability is the poor scaling of electrical interconnects. Photonic technology, when used prudently, has the potential to address this, by providing fast, high-bandwidth communication within a reasonable power envelope. However, the energy- and performance-optimal photonic design is strongly influenced by several factors, including the total number of memory stacks on the channel, the on-die wiring technology, traffic on the channel, etc. We show that the expected use case for large-scale systems argues against using photonics inside the memory dies, instead using them only to break the pin barrier. As regards the access protocol, the primary limitation to scalability is the low-level control paradigm of the aging DDR family interface. We argue that the memory modules need to be made more autonomous, with low-level device management being handled locally, and the memory controller only performing global scheduling via

[2]DDR here refers to the general family of conventional, low-level, JEDEC-based memory interfaces, and not necessarily a particular Dual Data Rate signalling standard.

a novel packet-based interface. Based on these arguments, we propose a novel memory architecture that exploits 3D die stacking technology to introduce an *interface die* that serves two purposes. First, it aggregates all photonic components on a separate layer, using them more efficiently to reduce energy consumption and enable daisy-chaining to provide larger capacities. Our final design provides at least 23% energy reduction compared to fully-optical 3D extensions of state-of-the-art photonic DRAMs, with 4X capacity, and the potential for performance improvements and further energy reduction due to increased bank counts. Another important advantage is that it allows the memory dies to be interface-agnostic, minimizing disruption to memory manufacturing. A single memory product can be used for both photonic and electrical systems. Second, it houses a stack controller that handles low-level device management including timing, maintenance, coding, etc., freeing the memory controller from having to deal with these details. This helps streamline the memory interface, simplify the memory controller, enable heterogeneity, and allow innovation in the DIMMs while presenting a standard interface to the processor. Thus, we propose a novel design for a terascale memory node capable of operating in an exascale environment with low energy consumption, high bandwidth, low latency, improved scalability, and reduced complexity.

6. ACKNOWLEDGMENTS

This work was supported in parts by NSF grants CCF-0811249, CCF-0916436, NSF CAREER award CCF-0545959, HP, Intel, SRC grant 1847.001, and the University of Utah.

7. REFERENCES

[1] CACTI: An Integrated Cache and Memory Access Time, Cycle Time, Area, Leakage, and Dynamic Power Model. http://www.hpl.hp.com/research/cacti/.

[2] Fully-Buffered DIMM Technology in HP ProLiant Servers - Technology Brief. http://www.hp.com.

[3] STREAM - Sustainable Memory Bandwidth in High Performance Computers. http://www.cs.virginia.edu/stream/.

[4] Virtutech Simics Full System Simulator. http://www.virtutech.com.

[5] N. Aggarwal et al. Power Efficient DRAM Speculation. In *Proceedings of HPCA*, 2008.

[6] D. E. Atkins et al. Report of the NSF Blue-Ribbon Advisory Panel on Cyberinfrastructure. Technical report, National Science Foundation, 2003.

[7] M. Awasthi et al. Handling PCM Resistance Drift with Device, Circuit, Architecture, and System Solutions. In *Non-Volatile Memories Workshop*, 2011.

[8] R. Barbieri et al. Design and Construction of the High-Speed Optoelectronic Memory System Demonstrator. *Applied Opt.*, 2008.

[9] L. Barroso and U. Holzle. *The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines.* Morgan & Claypool, 2009.

[10] S. Beamer et al. Re-Architecting DRAM Memory Systems with Monolithically Integrated Silicon Photonics. In *Proceedings of ISCA*, 2010.

[11] R. G. Beausoleil et al. Nanoelectronic and Nanophotonic Interconnect. *Proceedings of IEEE*, 2008.

[12] C. Benia et al. The PARSEC Benchmark Suite: Characterization and Architectural Implications. Technical report, Princeton University, 2008.

[13] B. Black et al. Die Stacking (3D) Microarchitecture. In *Proceedings of MICRO*, December 2006.

[14] M. J. Cianchetti, J. C. Kerekes, and D. H. Albonesi. Phastlane: A Rapid Transit Optical Routing Network. In *Proceedings of ISCA*, 2009.

[15] J. Condit et al. Better I/O Through Byte-Addressable, Persistent Memory. In *Proceedings of SOSP*, 2009.

[16] E. Cooper-Balis and B. Jacob. Fine-Grained Activation for Power Reduction in DRAM. *IEEE Micro*, May/June 2010.

[17] W. Dally and B. Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann, 1st edition, 2003.

[18] Elpida Memory, Inc. News Release: Elpida Completes Development of Cu-TSV (Through Silicon Via) Multi-Layer 8-Gigabit DRAM. http://www.elpida.com/pdfs/pr/2009-08-27e.pdf.

[19] K. Fang et al. Mode-locked Silicon Evanescent Lasers. *Optics Express*, September 2007.

[20] A. Hadke et al. OCDIMM: Scaling the DRAM Memory Wall Using WDM based Optical Interconnects. In *Proceedings of HOTI*, 2008.

[21] R. Ho. *On-Chip Wires: Scaling and Efficiency*. PhD thesis, Stanford University, August 2003.

[22] ITRS. International Technology Roadmap for Semiconductors, 2008 Update.

[23] J. Ahn et al. Devices and architectures for photonic chip-scale integration. *Applied Physics A: Materials Science and Processing*, 95, 2009.

[24] B. Jacob, S. W. Ng, and D. T. Wang. *Memory Systems - Cache, DRAM, Disk*. Elsevier, 2008.

[25] John Carter, IBM Power Aware Systems. *Personal Correspondence*, 2011.

[26] U. Kang et al. 8Gb 3D DDR DRAM Using Through-Silicon-Via Technology. In *Proceedings of ISSCC*, 2009.

[27] N. Kirman et al. Leveraging Optical Technology in Future Bus-Based Chip Multiprocessors. In *Proceedings of MICRO*, 2006.

[28] N. Kirman and J. F. Martinez. An Efficient All-Optical On-Chip Interconnect Based on Oblivious Routing. In *Proceedings of ASPLOS*, 2010.

[29] P. Kogge(Editor). *ExaScale Computing Study: Technology Challenges in Achieving Exascale Systems.* Defense Advanced Research Projects Agency (DARPA), 2008.

[30] S. Li et al. McPAT: An Integrated Power, Area, and Timing Modeling Framework for Multicore and Manycore Architectures. In *Proceedings of MICRO*, 2009.

[31] K. Lim et al. Disaggregated Memory for Expansion and Sharing in Blade Servers. In *Proceedings of ISCA*, 2009.

[32] G. Loh. 3D-Stacked Memory Architectures for Multi-Core Processors. In *Proceedings of ISCA*, 2008.

[33] D. Meisner, B. Gold, and T. Wenisch. PowerNap: Eliminating Server Idle Power. In *Proceedings of ASPLOS*, 2009.

[34] D. A. B. Miller. Device Requirements for Optical Interconnects to Silicon Chips. *Proceedings of IEEE Special Issue on Silicon Photonics*, 2009.

[35] C. Natarajan, B. Christenson, and F. Briggs. A Study of Performance Impact of Memory Controller Features in Multi-Processor Environment. In *Proceedings of WMPI*, 2004.

[36] C. Nitta, M. Farrens, and V. Akella. Addressing System-Level Trimming Issues in On-Chip Nanophotonic Networks. In *Proceedings of HPCA*, 2011.

[37] E. Ordentlich et al. Coding for Limiting Current in Memristor Crossbar Memories. In *Non-Volatile Memories Workshop*, 2011.

[38] M. Qureshi, V. Srinivasan, and J. Rivers. Scalable High Performance Main Memory System Using Phase-Change Memory Technology. In *Proceedings of ISCA*, 2009.

[39] Raymond G. Beausoleil, HP Labs. *Personal Correspondence*, 2010.

[40] K. Skadron et al. Temperature-Aware Microarchitecture. In *Proceedings of ISCA*, 2003.

[41] N. Streibl et al. Digital Optics. *Proceedings of IEEE*, 1989.

[42] J. Torrellas. Architectures for Extreme-Scale Computing. *IEEE Computer*, November 2009.

[43] A. N. Udipi et al. Rethinking DRAM Design and Organization for Energy-Constrained Multi-Cores. In *Proceedings of ISCA*, 2010.

[44] D. Vantrease et al. Corona: System Implications of Emerging Nanophotonic Technology. In *Proceedings of ISCA*, 2008.

[45] D. H. Woo et al. An Optimized 3D-Stacked Memory Architecture by Exploiting Excessive, High-Density TSV Bandwidth. In *Proceedings of HPCA*, 2010.

[46] Q. Xu et al. Micrometre-Scale Silicon Electro-Optic Modulator. *Nature*, 435:325–327, May 2005.

[47] J. Xue et al. An Intra-Chip Free-Space Optical Interconnect. In *Proceedings of ISCA*, 2010.