Towards Energy-Proportional Datacenter Memory with Mobile DRAM

Krishna T. Malladi† Frank A. Nothaft† Karthika Periyathambi†
Benjamin C. Lee‡ Christos Kozyrakis† Mark Horowitz†

†Electrical Engineering, Stanford University
{ktej, fnothaft, karthipd, kozyraki, horowitz}@stanford.edu

‡Electrical and Computer Engineering, Duke University
benjamin.c.lee@duke.edu

ABSTRACT

To increase datacenter energy efficiency, we need memory systems that keep pace with processor efficiency gains. Currently, servers use DDR3 memory, which is designed for high bandwidth but not for energy proportionality. A system using 20% of the peak DDR3 bandwidth consumes 2.3× the energy per bit compared to the energy consumed by a system with fully utilized memory bandwidth. Nevertheless, many datacenter applications stress memory capacity and latency but not memory bandwidth. In response, we architect server memory systems using mobile DRAM devices, trading peak bandwidth for lower energy consumption per bit and more efficient idle modes. We demonstrate 3-5× lower memory power, better proportionality, and negligible performance penalties for datacenter workloads.

1. INTRODUCTION

Energy efficiency and proportionality are major challenges in modern datacenters because they impact cost and scalability [3]. Recent advances have eliminated most inefficiencies in power delivery and cooling; new datacenters have a Power Usage Effectiveness (PUE) around 1.10. The main focus for efficiency is now the tens of thousands of servers. Processor energy efficiency and proportionality have improved significantly over the years, benefiting from low-power circuits, dynamic voltage-frequency scaling, and power gating for unused cores. The use of simpler cores, heterogeneous cores and specialized accelerators have the potential to further improve efficiency [30, 40].

Unfortunately, efficiency and proportionality of server memory systems have improved at a much slower pace. Memory accounts for more than 25% of datacenter energy [15, 27, 29] and this percentage will increase as applications demand larger memory capacities for virtualized multi-cores and memory-based caching and storage [42, 36]. All commercial servers use double data rate (DDR) DRAM technology, which is optimized for bandwidth but not necessarily for energy efficiency. To support channel bandwidths of more than 10GB/s, high-speed DDR3 interfaces consume significant energy, even when memory is idle but in an active power mode (i.e., "active-idle"). This results in energy-disproportional memory systems. When bandwidth utilization is 20%, the energy cost per bit is 2.3× the cost under full bandwidth utilization.

With such costs, DDR3 is highly inefficient for applications that stress capacity and latency but have modest bandwidth requirements (§2,§3). For example, web search packs large indices in memory but uses a small fraction of DDR3 bandwidth [23]. Social networking uses thousands of servers for distributed memory caching, but network bandwidth (e.g., 10Gb/s) limits memory bandwidth utilization [42].

To address the disproportionality of existing server memory systems, we turn to a technology originally designed for mobile platforms: LPDDR2. Mobile-class memory addresses the energy efficiency challenges of server-class memory by forgoing more expensive interface circuitry. LPDDR2 provides the same capacity per chip as DDR3 and similar access latency at lower peak bandwidth. However, without on-die-termination and delay-locked loops, mobile-class memory is susceptible to inter-symbol interference, which poses significant design challenges for high capacity memory systems. And with wider chips, we face challenges in error correction. We address these challenges, presenting a new architecture for LPDDR2 that is viable for servers and leads to energy proportionality without significant performance penalties for datacenter workloads.

Our work makes the following contributions:

- **Emerging Applications Analysis:** We find that large datacenter workloads, from web search, social media to analytics need high memory capacity but under-utilize bandwidth (§2).
- **DRAM Energy Efficiency:** For such emerging workloads, we find that DDR3 idle and termination power lead to a highly energy-disproportional memory system, measured in energy per bit transferred (§3). To address this specific DDR3 limitation, we turn to mobile-class memory, which eliminates termination and reduces idle power (§4).
- **Architecting LPDDR2 for Servers:** We present new architectures for memory channels that include new package and module designs. Our LPDDR2 architecture provides DDR3-competitive capacity and good signal integrity despite a lack of on-die termination and delay-locked loops. With our architecture, we reduce main memory power by 3-5× without significant performance penalties (§5–6).
- **Vertically Integrated Evaluation:** In addition to evaluating memory efficiency, we examine the implications for processor cache capacity by analyzing average memory access energy (§7). Once main memory is energy proportional, the energy inefficiency of large, shared caches is evident. We also assess implications for datacenter capacity by analyzing total cost of ownership (§8). Using energy-efficient main memory increases datacenter capability at the same total cost.

2. BACKGROUND

Memory and Server Organization. Memory systems organize chips hierarchically (Figure 1). A DRAM chip is internally divided

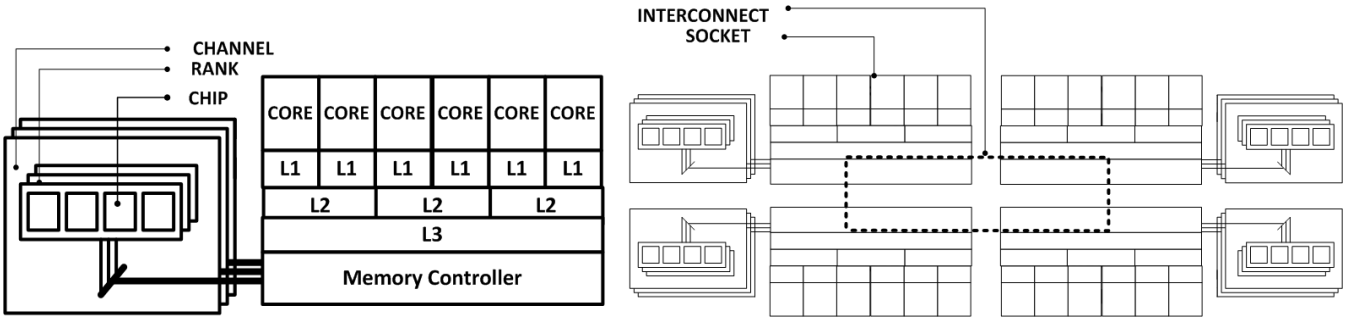


Figure 1: Hierarchical DRAM organization in quad-socket platforms. Interconnected sockets each support 2-4 memory channels. Each channel is populated with 1-4 slots that can hold Dual Inline Memory Modules (DIMMs) as dense as 4GB.

into multiple banks, which share I/O pins. Each chip has 4, 8 or 16 data pins, and N = 16, 8 or 4 chips are placed in parallel to create a 64b interface to a channel respectively. These N chips form a rank. Multiple ranks share a channel that interfaces to a controller. Each controller can manage several channels. Signal integrity limits the number of ranks on a channel to four or less.

Modern servers use multiple processor sockets, each with integrated memory controllers. For example, Intel-Nehalem and AMD-Barcelona processors have tri- and dual-channel DDR3 controllers per socket respectively. Cores issue requests to any controller via QuickPath or HyperTransport interconnects. These platforms provision a requisite amount of memory capacity by populating DIMM slots on all channels. A typical dual-channel four-socket Barcelona in datacenters, filling two DIMM slots per channel each with two ranks comprising of 1Gb, DDR3-1600, x8 DRAMs chips will give 32 GB total capacity and 102 GB/s peak bandwidth.

Datacenter Memory Requirements. Balancing processor computation and memory bandwidth is a recurring optimization in systems design. Clearly, introducing more cores increases memory traffic although the absolute traffic level depends on the application [41]. Transaction applications such as TPC-C, TPC-H, and SAP, perform few operations per data item, require up to 75GB/s of memory bandwidth [53], and need the high bandwidth of DDR3.

However, emerging datacenter applications exhibit different capacity and bandwidth demands, which do not match the system balance in existing server organizations. These applications stress memory capacity and latency but not bandwidth. Such applications include web search, MapReduce data analytics, and distributed memory caching for social media. In web search, each server's web index is sized to fit in memory to achieve short query latencies. Microsoft Bing uses 92% of a server's memory capacity for web indices but utilizes a tiny fraction of memory bandwidth [40]. Search threads are bound by memory latency as their data transfers from the index are short and have no locality.

Microsoft Cosmos, a framework for large-scale data analytics that is similar to MapReduce and Hadoop, under-utilizes memory bandwidth since analyses are often either compute-bound or limited by network bandwidth in a distributed storage system. Under stress testing, Bing and Cosmos servers reach 67-97% processor utilization but only 2-6% memory bandwidth utilization [23]. Google Content Analyzer, BigTable, and web search similarly require substantial memory capacity but modest bandwidth. On a dual-socket Intel Clovertown server, these applications have last-level cache miss rates of less than 10K/msec, which translate into 0.6GB/s of memory bandwidth (less than 3% of peak) [46].

Another important datacenter workload with low bandwidth requirements is social networking (e.g., Facebook, Google+, Twitter, etc). To mitigate disk latencies, Facebook uses thousands of servers to run memcached, an in-memory, distributed object storage with 28TB of capacity. This caches 75% of all non-media data and serves complex user requests in reasonable deadlines [42]. Projects like RAMCloud take this approach one step further, replacing the in-memory cache with a distributed, in-memory filesystem for uniformly fast data accesses [36]. Server memory for such frameworks needs high capacity but sustains low bandwidth as memory traffic is limited by network bandwidth. The peak bandwidth of a 10 Gbps Ethernet adapter and a DDR3-1600 memory system are two orders of magnitude apart.

3. EXPERIMENTAL METHODOLOGY

We use an x86_64 execution-driven simulator based on a Pin front-end [28]. We use 8 out-of-order (OOO) cores at 3 GHz matched with Intel's Nehalem microarchitecture. Each core has a private 8-way, 32-KB L1 data cache, an 8-way 256KB L2 cache. All the cores share a 16-way associative 16MB L3 cache. Using Nehalem model, the L1, L2, L3 latencies are set to 1, 7 and 27 cycles respectively. An integrated memory controller models multiple channels and standardized DRAM devices. We use a closed-page policy, typical in multi-cores with low page locality[1]. Ranks use fast-exit precharge power-down mode.

We study web search, which has a large memory footprint and modest bandwidth requirements (§2) by deploying Nutch, an open-source, Java-based web crawler and search engine. First, we index Wikipedia pages to produce a 30GB dataset, which is distributed across servers' memories. We trace search engine memory activity as the 500 most common Wikipedia queries arrive at the server's maximum sustainable query throughput.

We evaluate distributed memory caching, which has a large memory footprint and modest memory bandwidth due to network constraints. We deploy Memcached, which is an open-source framework for distributed key-value stores in RAM. Hash functions distribute data and load across Memcached servers. Memory is broken into slabs of varying sizes and we consider 100B slabs (*Memcached.A*) and 10KB slabs (*Memcached.B*). To exercise the cache, we use a zipf distribution (parameter = 0.6) that models a long tail and reflects Facebook popularity distributions [44].

We also evaluate more diverse workloads that might run in virtualized, elastic clouds. We use *SPECjbb2005 (jbb)* with 12 warehouses each with 25MB of transaction data and *SPECpower_ssj2008 (power)* at the calibrated maximum sustainable transaction rate. *SPECweb2005 (web)* benchmarks a banking web server that uses Apache Tomcat. Finally, we consider 8-

	Multi-Programmed	Multi-Threaded		Datacenter
Cache Fill B/W	**SPEC CPU 2006**	**SPEC OpenMP**	**PARSEC**	
Low	416.gamess, 447.dealII, 453.povray, 458.sjeng, 464.h264ref, 465.tonto, 481.wrf	ammp, equake	freqmine swaptions	Memcached, Websearch SPECweb
Medium	400.perlbench, 401.bzip2, 403.gcc, 434.zeusmp 435.gromacs, 436.cactusADM, 445.gobmk, 454.calculix, 456.hmmer, 473.astar	apsi, fma3d wupwise	blackscholes, fluidanimate streamcluster	SPECjbb, SPECPower
High	433.milc, 437.leslie3d, 450.soplex, 459.GemsFDTD, 462.libquantum, 470.lbm, 471.omnetpp, 482.sphinx3, 483.xalancbmk	applu, art mgrid, swim	canneal	

Table 1: Low, medium, high bandwidth applications, estimated from last-level cache miss rates.

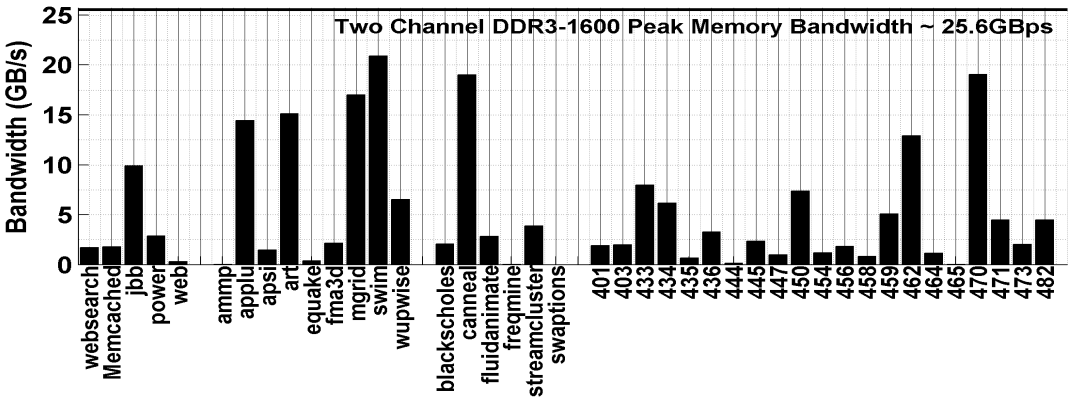


Figure 2: Application memory bandwidth demand estimated from last-level cache miss rates with a fixed memory latency. Actual sustained bandwidth is determined by DRAM internal timings and applications' memory level parallelism (MLP).

way multi-threaded SPEC OMP2001 and PARSEC benchmarks as well as 8-way multi-programmed combinations of SPEC CPU2006 benchmarks with each core running one copy/thread. (Table 1).

We follow the methodology used in prior memory studies [1, 48, 22, 8, 19]. We match the number of application threads or processes to the number of cores. We fast-forward 10 to 20 billion instructions to skip warm-up and initialization stages and focus on memory behavior in steady state, which is consistent with prior architectural evaluations of enterprise workloads [53]. To model the distribution of activity across channels, ranks, and banks, we emulate virtual to physical address translation.

3.1 Workload Validation

We compare the bandwidth requirements of our workloads, as shown in Figure 2, against independently reported measurements.

Nutch. Production search engines (e.g., Microsoft Bing) require less than 6% of peak memory bandwidth [23]. Commercial servers have tri-channel DDR3 with peak bandwidth of 32-39 GB/s. At 6% of peak, search would require 1.9-2.3 GB/s. We use VTune to characterize Nutch on indexed Wikipedia; it consumes 752 MB/s per thread on a Xeon X5670 system and 180 MB/s per thread on an Atom Diamondville. The former measurement suggests that our simulated 8-core system might scale Nutch bandwidth demand to 6.0 GB/s. But contention for shared multiprocessor resources reduces memory demand. Our Nutch simulations indicate 1.8 GB/s of utilized bandwidth, which is consistent with real system measurements for both Bing and Nutch.

SPECjbb, power, web. *Jbb* requires 6 GB/s on a server with four quad-core Intel Core-2 Duo processors [45]. In our environment, *jbb* requires about 10 GB/s. *Power* calibrates transaction rates to the platform's capabilities, which leads to differences in bandwidth demand on each system. On an IBM JS12 blade with a dual-core Power6, *power* uses 30% of peak bandwidth [13], which is 6 GB/s since Power6 has a dual-channel DDR3-1333 system with peak bandwidth of 21 GB/s. Our own VTune measurements for *power* on a four-core Intel Xeon E5507 show 1.2 GB/s. In our simulations, *power* requires 2.5 GB/s. Finally, *web* exhibits low L2 miss rates and does not exercise memory in Simics full-system simulation of a SPARC V9 system [4]. Our *web* simulations show a requirement of 0.3GB/s.

SPEC-OMP, SPEC-CPU, PARSEC. Our multi-threaded applications require up to 21GB/s and the multiprogrammed workloads require up to 24GB/s of main memory bandwidth. These numbers are consistent with similar workload deployments in prior studies [1, 22, 8, 48]. While there are some differences in the specific bandwidth numbers, the cross-validation indicates that we correctly identify applications with low, medium, and high bandwidth demand, which stress memory.

3.2 Memory Bandwidth Demand

Architects choose the channel count to match either the memory capacity or the bandwidth demanded by applications of interest. We start with a dual-channel 16GB DDR3-1600 system using 2Gb parts (Table 2). If higher capacity is needed, system architects may deploy more channels. However, adding channels will also increase a socket's memory bandwidth. For applications with low to medium bandwidth demand, adding channels allow us to reduce per channel bandwidth and improve energy-efficiency without affecting performance.

To trade peak bandwidth for energy efficiency, we need to understand application sensitivity to bandwidth. For the dual-channel DDR3-1600 baseline, [1] Figure 3 characterizes application performance penalties as channel frequency are scaled down for various processor scenarios. Confirming prior studies, most datacenter

[1]DDR3-1600 operates at 800MHz in a double data rate manner.

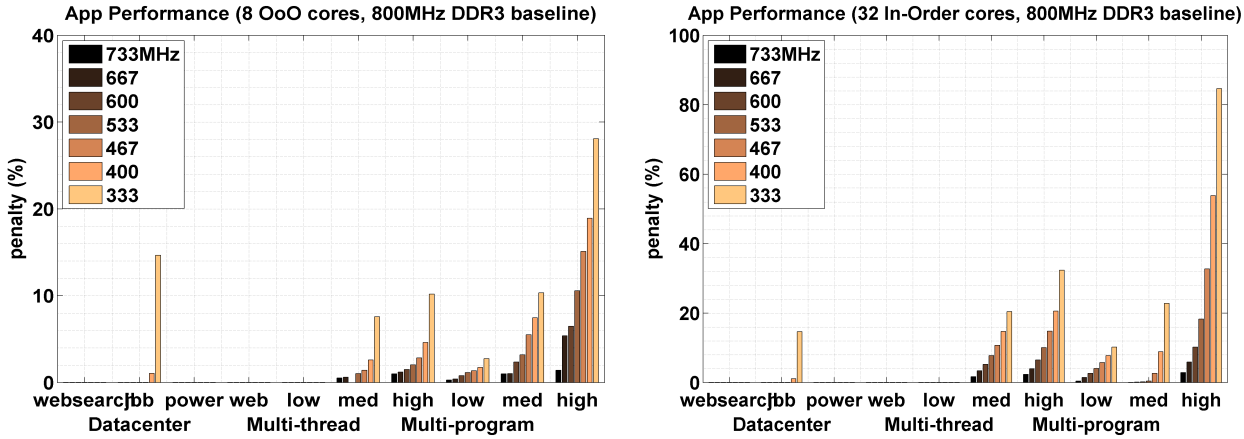


Figure 3: Application bandwidth sensitivity measured by performance penalties for various channel frequencies relative to DDR3-1600. Shown for Out-of-order/In-order cores with 2 channels. Groups refer to Table 1. Note change in Y-axis scale.

workloads do not fully exploit peak bandwidth [4, 13, 23]. Only SPECjbb incurs a 15% penalty and only after bandwidth has been throttled by 60%. Similar tradeoffs are observed for out-of-order and in-order core systems. Four channels further reduce already modest penalties, indicating that high-capacity, multi-channel systems over-provision bandwidth.

4. IMPROVING MEMORY EFFICIENCY

DDR3 Inefficiency. DDR3 dissipates approximately 1-1.5W/GB on average [35], and approximately 2.5W/GB at peak [14]. Our experiments produce numbers in these ranges. Variations in activate, read, and write power reflect differences in channel activity. The cost of over-provisioned DDR3 bandwidth is primarily the power dissipated when idle, which often dominates the total.[2] This power is required to keep high-speed interfaces active. Despite using per rank power-down modes, idle and termination power is nearly 40% of the total power in an active memory system. The latter includes ranks' static power during the active period for a different rank on the same channel. These overheads mean that DDR3 technology is most efficient at high utilization, where these overheads can be amortized over many data transfers. At lower utilization (i.e., few transfers), energy proportionality is poor. This analysis is detailed in Figure 9(a) and Figure 10(a).

LVDDR Limitations. Because static power overheads from high-performance interfaces are the problem, simply scaling voltage and frequency often have a negative impact. For example, LVDDR3 operates at 1.35V and link-speeds of 400, 533, and 800MHz. However, LVDDR3 termination power per pin is 20% higher than that of DDR3, which reflects the challenge of ensuring signal integrity at lower operating voltages. At low channel utilization, such static power increases the energy per bit transferred and harms energy proportionality. LVDDR3-800 is the best low-power alternative to DDR3, over LVDDR3-1600 at low utlizations. However, for our workloads, LVDDR3-800 dissipates an average of 11.8W of which 68% is static power and independent of frequency (Figure 9(a) and Figure 10(a)). Thus, we must address high static power in memory interfaces.

LPDDR2 Opportunities. In contrast to server DRAM, memory designed for mobile platforms has been optimized for low energy

[2]64ms DRAM refresh periods are too infrequent to impact power.

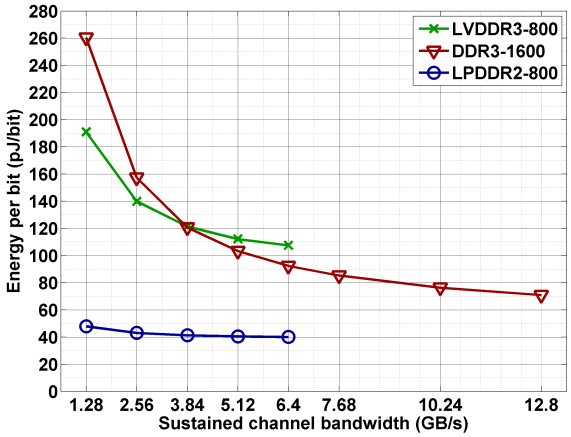


Figure 4: Energy per bit (mW/Gbps = pJ/bit) with varying channel utilization. Assumes four ranks per channel and 3:1 read to write ratio. DDR3 is x8-1600 (Peak BW = 12.8GB/s). LPDDR2 is x16-800 (Peak BW = 6.4GB/s).

costs per bit transferred. Examples include LPDDR2 and mobile XDR. By eliminating expensive delay-locked loops (DLLs) and on-die termination (ODT) from the chip interface, mobile memory addresses the largest source of inefficiency in server memory: idle and termination power. However, without DLLs and ODT, LPDDR2 complicates architecting server systems for capacity and signal integrity. DDR3's link speed (and channel bandwidth) is twice that of LPDDR2.

LPDDR2 chips have more I/O pins since individual memory chips, not modules, are deployed in mobile platforms. Wider LPDDR2 can use fewer chips to supply the same number of bits for a channel, which may improve power efficiency [52, 7, 10, 54] but it complicates error correction (§5.4). We quantify LPDDR2 efficiency and then address these challenges (§5).

On the other hand, mobile and server memories share the same memory core architecture and, thus, have similar timing parameters. Table 2 presents timing parameters for a column access (t_{CAS}), for a row access (t_{RAS}), and for the length of a read cycle (t_{RC}) [20]. LPDDR2's lower voltages may increase latencies but these differ-

ences are small.

While LPDDR2 active standby and powerdown are significantly lower than those of DDR3, LPDDR2 has higher read-write currents compared to the latest DDR3 revision. We calculate memory power [32] and divide by memory bandwidth to determine energy per bit transferred. Figure 4 illustrates energy proportionality, plotting energy per bit as a function of bandwidth utilization.

DDR3-1600 has a peak bandwidth of 1.6 Gbps per pin. However, these pins use a lot of energy. At high channel utilization, interface energy is amortized over more transferred bits, reducing energy per bit. With 100% channel utilization, DDR3 requires 70 pJ/bit; 30% and 10% of this energy is background and termination, respectively. These I/O overheads are incurred even when the chip is idle but in active power mode. Such active-idle power is particularly evident under more typical 20% utilization where background and termination energy is amortized over less work. At low sustained channel bandwidth, say 1.28GBps, energy per bit increases by 3.7× to 260 pJ/bit. While LVDDR3-800's energy per bit is better than that of DDR3-1600 at these low utilizations by 1.4×, it is still high at 190 pJ/bit since it halves bandwidth in exchange for modest power savings.

In contrast, mobile memory is energy proportional; energy per bit is flat as utilization varies. At peak and typical utilization, LPDDR2 consumes 40 and 50pJ/bit, respectively. Compared to DDR3 and LVDDR3 at low utilization (e.g., 20%), LPDDR2 sees a 4-5× energy reduction in exchange for 2× lower peak bandwidth. For applications with modest bandwidth demands, this is a good trade-off.

5. ARCHITECTING MOBILE DRAMS

While §4 shows the potential efficiency of LPDDR2 in servers, we must address several significant challenges posed by its power-efficient interfaces. First, non-terminated LPDDR2 memory chips increase vulnerability to inter-symbol interference, which complicates the design of high-capacity memory systems [43]. Second, wide LPDDR2 chip interfaces may increase error correction costs.

To address these issues, we present a new channel architecture using commodity LPDDR2 devices. We combine board design with stacked dies to obtain DDR3-competitive capacities with good signal integrity. To further increase capacity, we present a new module architecture for LPDDR2, which draws lessons from registered/buffered DDR3. We can also scale channel and socket counts [2, 11]. Finally, we describe error correction for this system.

5.1 Channel Architecture

LPDDR2 devices are edge-bonded, making them suitable for capacity stacking; mass manufacturing four-stacked LPDDR2 dies is already viable [33]. Figure 5(a) shows a Micron LPDDR2 dual-rank package with four LPDDR2 dies, the basic block of our architecture. However, its capacity is limited to 1GB/channel and is not acceptable for high-capacity server memory. To increase capacity, we can architect a given capacity with some combination of channels and packages per channel. With fewer channels and more packages per channel, the system becomes less expensive but performance and signal integrity suffer.

Alternatively, we propose a new architecture called Dual-Line Packages (DLPs). As shown in Figure 5, DLPs have LPDDR2 packages on both sides of a board. We stripe four ranks across multiple packages with two chips from each package. Since two devices share a Chip Select (CS) internally, we place them in the same rank. The two sets of x16 bond-wires from a package are multiplexed with the mirroring wires from the package on the opposite side of the board through an on-board via. Since each edge on the via is x16, four such traces form the 64 bit output DQ bus.

As each package has only two devices from the same rank, only two dies can have active column operations at a given time in the package. Other active-idle dies dissipate very little power and ensure thermal constraints are satisfied. With four ranks, each with 2Gb x16 devices, we obtain a total capacity of 4GB on channel, which is DDR3-competitive. As we apply memory-memory stacking only and not memory-processor stacking, we avoid the thermal and pin challenges of the latter.

5.2 Signal Integrity (SI)

To demonstrate feasibility, we analyze signal integrity. Lumping DRAM dies together into a single package moves the impedance discontinuity problem to the end of the link. Each DRAM I/O pin presents a load on the bus, causing a small impedance discontinuity and a reflection. Lumping these loads on the end of the link makes the discontinuities and reflections larger. As load increases, we will reach a point where the bus no longer functions. We simulate SPICE models of the PCB transmission lines and the bond wire inductance and the ESD/pad/driver capacitance associated with the DRAM package/pin (Figure 6). We model few energy losses, which is generally a worst-case situation for reflections.

In Figure 6, the data eye diagrams for write data (controller to DRAM) and read data (DRAM to controller), clearly show open eyes in both directions, which means this type of communication is possible. Write data is much cleaner than read data which is expected. Consider link interfaces at the transmitter (i.e., memory controller) and receiver (i.e., DRAM). During writes, the link is terminated at the transmitter end, so the quality of the impedance near the receiver is not that critical. Discontinuities at the receiver will cause reflections, but these reflections travel back to the terminated transmitter and are absorbed: none of these new reflections will be able to reflect again and make it back to the receiver. Thus, even though the large load affects impedance at the receiver, the received signal for write data is good.

Read data looks worse since now the termination is on the transmitting DRAM, and the reflected data at the cleaner controller side is re-reflected back to the DRAM where it gets reflected back to the controller. Fortunately by placing the DRAMs close together, their reflections still leave a signal with good noise margins. The CA bus operates at DQ speeds but is more heavily loaded. However, CA communication is only in one direction (controller to DRAM), somewhat alleviating signal integrity challenges. Thus, our simulations indicate the feasibility of grouping four chips into a package and placing four packages onto a shared channel, despite using non-terminated LPDDR2 dies.

5.3 System Capacity Strategies

The signal integrity analysis shows that stacked modules support 4GB per channel using 2Gb x16 LPDDR2 chips. To further increase system capacity, we could add more channels. While some applications would benefit from the parallelism of more channels (Figure 3), many emerging applications do not. Additional channels also introduce complexity in controllers [2] and we cannot tune capacity/bandwidth ratios since adding a channel simultaneously increases both capacity and bandwidth. This bandwidth overprovisioning is less expensive for energy-proportional LPDDR2 than for DDR3 but it still requires pins and chip area. To solve this problem, we increase LPDDR2 channel capacity with buffers.

Technology Parameter	DDR2 [31]	DDR3 [32, 35]	LVDDR3 [32, 35]	LPDDR [20, 39]	LPDDR2 [37, 39]
Operating Voltage	1.8V	1.5V	1.35V	1.8V	1.2V
Operating Frequency	400MHz	800MHz	400MHz	200MHz	400MHz
Typical Device Width (pins)	4	8	8	16	16
Peak Channel Bandwidth (sequential)	6.4GBps	12.8GBps	6.4GBps	3.2GBps	6.4GBps
Dynamic					
Timing (CAS, RAS, RC)	12, 40, 55ns	15, 38, 50ns	15, 38, 50ns	12, 40, 54ns	15, 42, 57ns
Active Current (read, write)	160, 160mA	180, 185mA	125, 130mA	130, 130mA	210, 175mA
Energy per bit (peak, typical)	111, 266mW/Gbps	70, 160 mW/Gbps	110, 190 mW/Gbps	110, 140 mW/Gbps	40, 50 mW/Gbps
Static					
Idle current (power-down, standby)	50, 70mA	35, 45mA	22, 32mA	3.6, 20mA	1.6, 23mA
Min power-down period	84ns	90ns	90ns	20ns	20ns
Slow Powerdown Exit latency	20ns	24ns	24ns	7.5ns	7.5ns

Table 2: Memory technology comparison showing key latency and energy parameters for 2Gb parts.

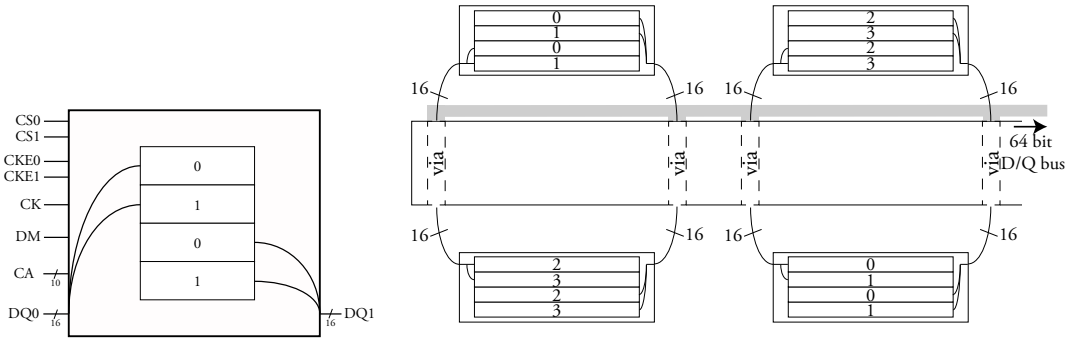


Figure 5: (a) LPDDR2 package with four x16 2Gb devices. (b) Dual Line Package (DLP) architecture with four packages per channel. Each chip select (CS) signal is shared by two devices. Each of the two sets of 16 pins are multiplexed by two devices. Constituent chips of the same rank are indicated by the same rank number (e.g., 0 to 3).

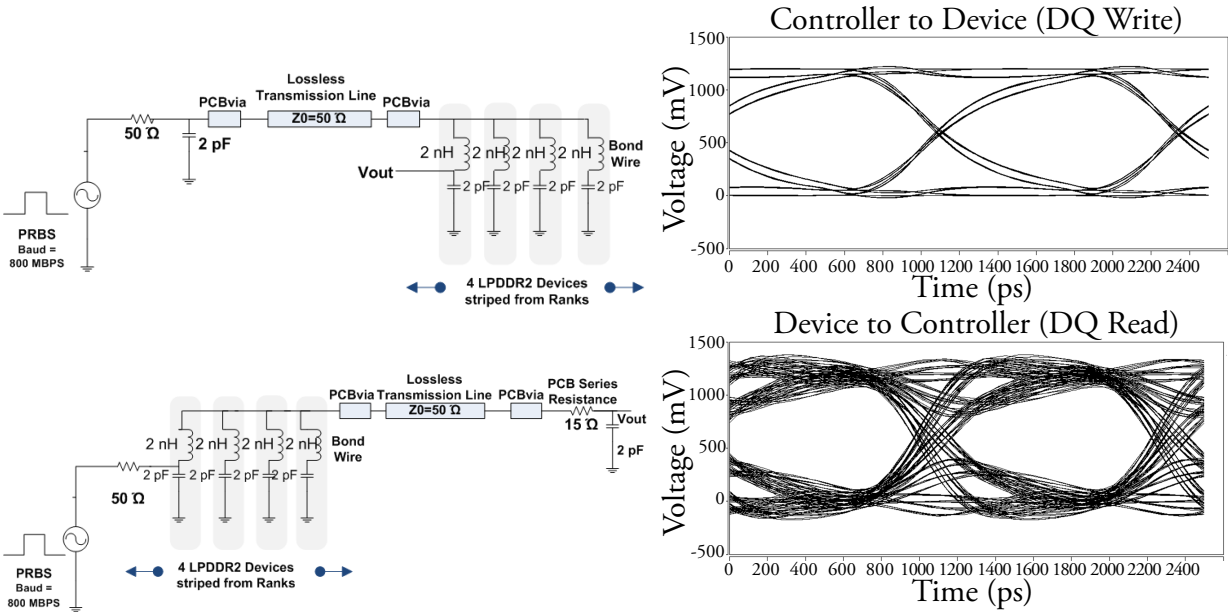


Figure 6: LPDDR2 signal integrity SPICE simulations with four packages per channel, demonstrating open eyes for controller-memory and memory-controller links. Signal integrity demonstrates feasibility of building the proposed capacity.

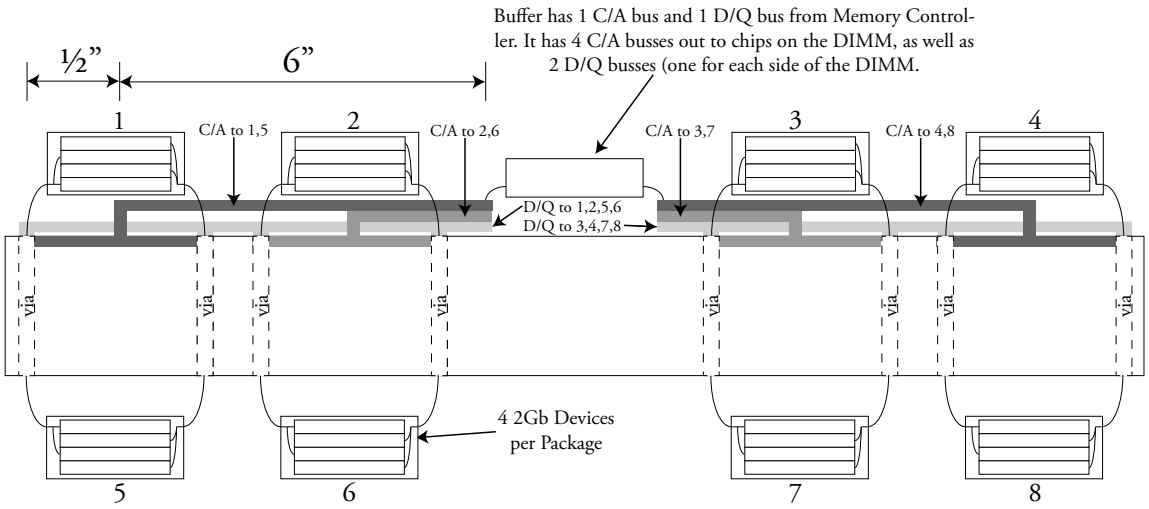


Figure 7: Channel architecture with load-reduced buffer that allow multiple LPDDR2 packages. Double DQ bus to provide point-point buffer-to-device connections. First line communicates with packages 1, 2, 5, 6. Each via has 16 bits to DQ that are multiplexed by two devices from the same package. Quadruple CA bus, reducing load to two packages per CA line.

Buffers. To hold the number of channels constant and increase channel capacity, we need to scale either module capacity or the number of modules per channel. The former may be difficult due to packaging constraints. For this reason, we increase the number of modules per channel. But simply adding modules will overload the shared channel and degrade signal integrity.

Following the example of Load-Reduced (LR) DDR3 DIMMs, we propose buffered LPDDR2 DIMMs that buffer and re-time both DQ and CA buses to ensure signal integrity. The buffer is positioned between the channel and DRAM chips. Chips and buffers communicate via point-to-point links to ensure reliable communication even as capacity increases, reducing channel load and permitting a larger number of stacked chips. Such buffers can double LPDDR2 per channel at modest latency and pin cost.

Figure 7 illustrates our new LPDDR2 module architecture. The buffer has 64 DQ and 14 CA pins on the input side. These pins need to be duplicated to provide point-point links to the multiple LPDDR2 packages on the board. To double channel capacity from 4 to 8GB without compromising signal integrity, the output DQ bus needs to be replicated $2\times$ while the CA bus needs to be replicated $4\times$ due to its higher load. We use our new DLPs with packages on both sides of the board. The capacity can now be scaled by having multiple DLPs per channel since the buffer isolates the load on the controller and enables multiple buffers per channel. Figure 8 shows the SI simulations of the proposed 8GB architecture with 8 packages per channel. The signal and timing margins are all sufficiently high for the controller-buffer and buffer-chip lines that allow for reliable communication in the relevant directions.

The proposed load-reduced buffers' static power overhead is small since termination is not required for LPDDR2. However, PLLs for clock re-timing incur a small active power cost. Registered DDR3 modules are very common in servers and LRDIMMs are increasingly becoming popular. We envision that our architecture for buffered LPDDR2 DLPs will be just as attractive, but with excellent energy-proportionality.

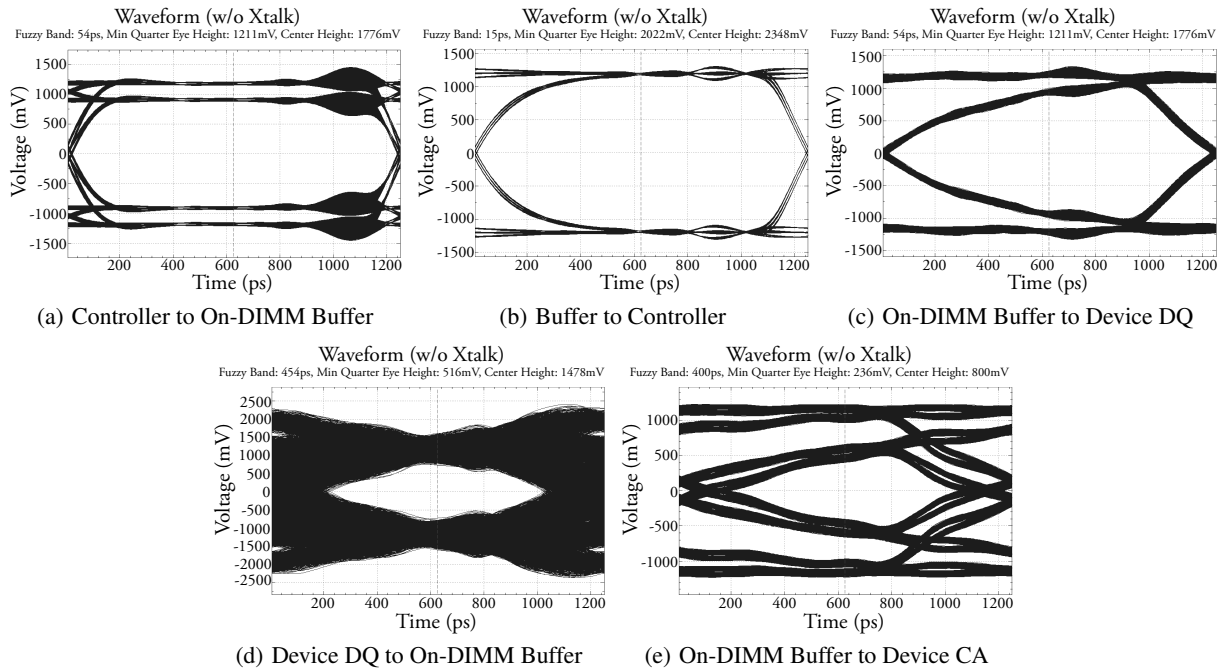
5.4 Error Correction Strategies

Memory system designers have used Error Correcting Codes (ECC) with Single Error Correction Double Error Detection (SECDED) [48, 52] to detect and correct memory errors. These techniques were originally developed for DRAMs with single bit outputs, where SECDED would protect from chip failures as well, and used 8 parity bits to protect a 64bit data word. This 72bit interface has become a standard, and much work has been done on protecting wider parts within this form factor. As the width of the DRAM increased, Single Symbol Correct, Double Symbol Detect codes were used, that could correct a multi-bit symbol, rather than a single bit [5]. Using the B-adjacency algorithm, b bits can be protected through the use of two b-bit wide symbols [18]. While this code uses more parity bits, it can be implemented in the same overhead by protecting larger blocks of memory; percentage overhead decreases as block size increases. These types of algorithms have been used to protect DDR3 x4 devices using a block size of 128 bits, and have been extended to protect x8 devices.

The wide x16 LPDDR2 interface causes two challenges for ECC. The first is due to quantization: 72 bits is not divisible by 16 and we cannot naturally provide data using the standard memory interface. The second is the larger number of bits needed to protect against a loss of 16 bits. These challenges are similar to those in architectures that access fewer chips for energy-proportionality [1, 48, 54], and we leverage some of their techniques to address these issues.

There are four ways to deal with the quantization issue: make x18 LPDDR2 DRAM, increase the overhead and create an 80/64 module, create a 144/128 bit channel, or embed the ECC data into the memory space. While the first solution seems simple, it would generate a different DRAM part type, which is expensive. Using x16 parts, we can either use one to protect four or eight other DRAMs. In the former, we double the parity overhead, which increases the cost of the memory by 12.5% but makes protecting from chip-kill much easier. In the latter, we merge two channels into a single channel with twice the bandwidth and larger block transfers.

The last option is to maintain a 64 bit interface and embed the ECC into the memory space instead of sitting in disparate devices



(a) Controller to On-DIMM Buffer (b) Buffer to Controller (c) On-DIMM Buffer to Device DQ

(d) Device DQ to On-DIMM Buffer (e) On-DIMM Buffer to Device CA

Figure 8: Signal Integrity analysis for Load-Reduce Buffers to increase channel capacity using LPDDR2 Package Modules. Figure 8(a) to Figure 8(e) show open eyes (for reliable communication) using On board buffer.

[54, 52]. The memory controller performs two accesses, one for data and one for ECC. Embedded ECC does not require dedicated chips for parity and is energy-efficient. However, memory system capacity falls as ECC is embedded in the data space and memory controller complexity increases as it must map ECC words. This basic idea is quite flexible, and can support multi-tiered error correction [52], where the OS can determine which pages in memory only need error detection (i.e., clean pages) and which need protection and correction. Furthermore, since errors are infrequent, the correction data will be accessed infrequently, reducing overheads. This approach is important for modern web applications that map large, slowly changing datasets in their memories.

Thus, while LPDDR2 device width causes some challenges for ECC, there are many ways these challenges can be addressed. In our view, it seems likely that some form of multi-tiered error correction will be implemented in these systems, since they are likely to support chip-kill with modest overhead.

6. EVALUATING EFFICIENCY

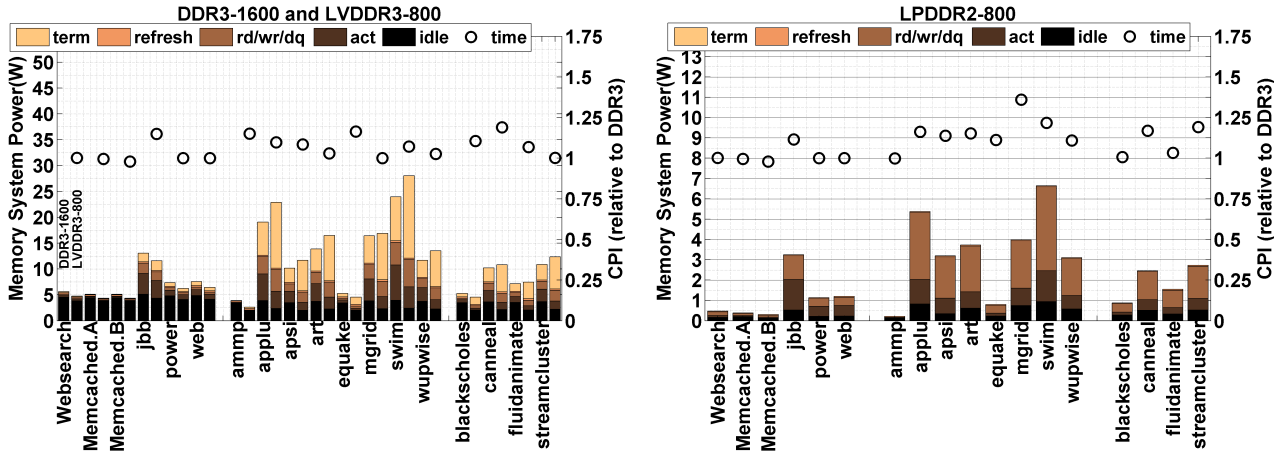
LPDDR2 is far more energy-proportional than DDR3 (§4) and this section shows that LPDDR2 is also far more energy-efficient as it reduces memory power with modest performance penalties.

Power. Using DLPs, we architect a 16GB LPDDR2 system with two channels per processor socket. Relative to DDR3 of the same capacity, LPDDR2 reduces memory system power by 3-5× for our applications (Figure 9(a), Figure 10(a) versus Figure 9(b), Figure 10(b)). Strikingly, LPDDR2 total power is often less than DDR3 idle power. LPDDR2 also benefits from the fast power-down modes that further reduce background power and that, unlike DDR3, do not require slow DLL re-calibration. In addition to these lower idle overheads, LPDDR2 dynamic power also falls due to lower activate current and voltage and fewer chips per access, which reduce the cost of activates, reads, and writes relative to DDR3.

Datacenter benchmarks, such as websearch and memcached, show significant power reductions due to their low bandwidth requirements. Power falls from approximately 5W to well below 1W, a 5-6× reduction. Other applications, with diverse memory behavior, also save memory power. Multiprogrammed benchmarks show a power reduction in the range of 3.3× for *482.sphinx3* to 16.8× for *465.tonto*. The savings are proportional to bandwidth, with an average power reduction of 4.4× (Figure 2). Significant contributors to this reduction are very low active-idle states and a much better power-down state. Active power is also lower because wider LPDDR2 devices means fewer chips are accessed per cache line. Similarly, multithreaded benchmarks show an average power reduction of 4.3× in a range of 3.3× for *art* to 18.8× for *ammp*.

Performance. Most applications realize significant power savings with modest performance penalties. Figure 10(b), Figure 9(b) show application time on an LPDDR2 memory system normalized to that on a DDR3 memory system. For instance, the performance of datacenter workloads like *websearch*, *Memcached* and *SPECJbb*, *SPECpower*, and *SPECweb* is barely affected when DDR3 is replaced by LPDDR2.

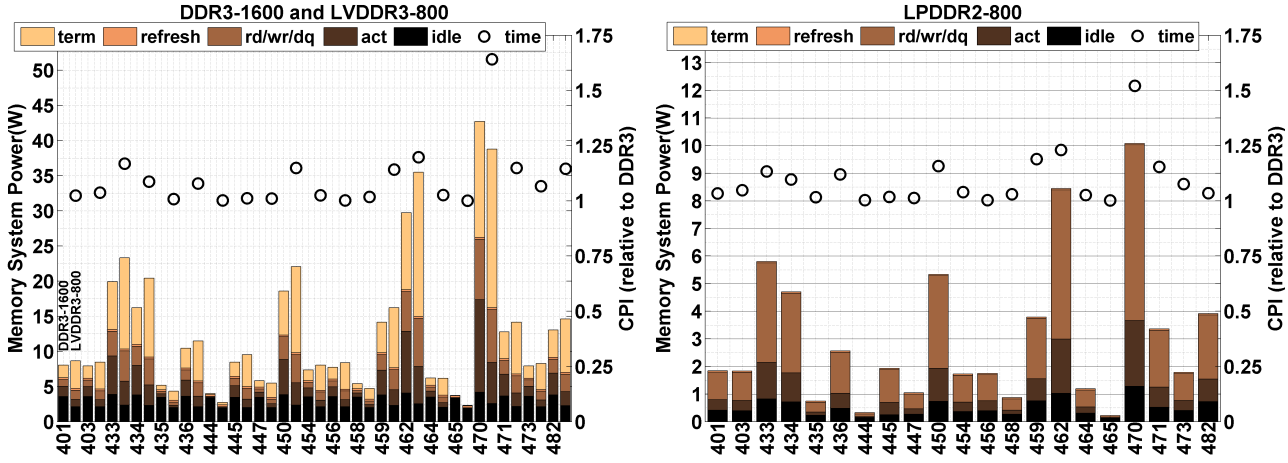
For conventional workloads (*SPEC-CPU*) multiprogrammed mixes, and *PARSEC, SPEC-OMP*, the performance impact varies from nearly zero for a majority of workloads to a worst case of 1.55x. Even when an application is stalled waiting for memory, it may not have sufficient memory-level parallelism (MLP) to benefit from additional sequential memory bandwidth. For example, *art* is as a memory-bound application with low MLP. However, other memory-bound workloads (e.g., *swim*) are impacted by the introduction of LPDDR2. Note that any performance penalties arise entirely from reduced bandwidth and not latency, which is unchanged when adopting LPDDR2 over DDR3. The applications affected by the lower LPDDR2 bandwidth generally have bursty cache misses that introduce channel contention.



(a) DDR3-1600 and LVDDR3-800 power, performance. Circles show LVDDR-800 performance penalties.

(b) LPDDR2-800 power, performance. See change in y-axis scale.

Figure 9: Datacenter, Multithreaded workloads with 16GB. Idle, termination power are significant in DDR3.



(a) DDR3-1600 and LVDDR3-800 power, performance.

(b) LPDDR2-800 power, performance. See change in y-axis scale.

Figure 10: Multiprogrammed workloads with 16GB. x-axis label numbers correspond to Table 1.

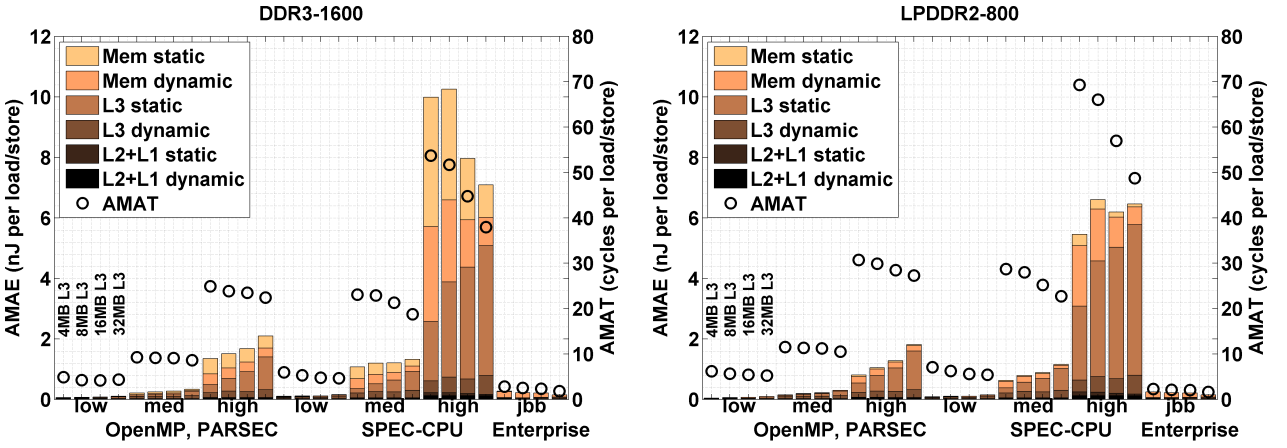


Figure 11: DDR3 and LPDDR2 average memory access energy (AMAE) in nJ per memory instruction and average memory access time (AMAT) in cycles per memory instruction. Shown for 4, 8, 6, and 32MB L3 cache sizes.

	Xeon+DDR3 8-cores		Atom+DDR3 16-cores		Xeon+LPDDR2 8-cores		Atom+LPDDR2 16-cores	
	Cost ($)	Power (W)	Cost ($)	Power (W)	Cost ($)	Power (W)	Cost ($)	Power (W)
Processor (2-socket)	760	125	360	25	760	125	360	25
Motherboard	200	30	1340	3	200	30	1340	3
Network Interface	0	5	0	5	0	5	0	5
Memory (32GB/2-sockets)	600	40	600	40	775	10	775	10
Storage (HDD)	100	10	100	10	100	10	100	10
Total	**1660**	**210**	**2400**	**83**	**1835**	**180**	**2575**	**53**
No. Servers ($\times 10^3$, in 15MW)	70		180		83		283	
TCO ($ per server per month)	$86.4		$86.4		$86.4		$86.4	
Capability	1.0		2.5		1.2		4.0	

Table 3: Define TCO-neutral price with Xeon+DDR3 baseline ($86). Atoms use TCO-neutral price for custom board ($1340). Atom+LPDDR2 shows TCO-neutral price for mobile memory ($775 for 32GB). Capability quantifies data center throughput normalized against Xeon+DDR3.

Mobile-class LPDDR2-800 performance effects are similar to those in server-class LVDDR3-800. LVDDR3 lowers the voltage from 1.5 to 1.35V and also reduces channel frequency. Lowering the voltage reduces memory core power, but does not address the inefficiencies of interface circuitry. Lowering the channel frequency also reduces power, but does not address static power dissipated by interface circuitry. For the same performance penalty, (e.g., iso-penalty LVDDR3-800 and LPDDR2-800), LVDDR3 saves much less power. We cannot perform an iso-savings comparison of performance because LVDDR3 cannot match LPDDR2's power savings.

7. PROCESSOR CACHE INTERACTIONS

Memory performance and power efficiency impact processor cache architecture. To understand these effects, we introduce *Average Memory Access Energy (AMAE)*, which is analogous to the conventional Average Memory Access Time (AMAT). AMAE provides a way to evaluate new memory technologies by combining the effects of dynamic and static energy in both processor caches and main memory. Quantifying the average number of Joules per memory instruction, $\texttt{AMAE}_{\texttt{L(i)}} = \texttt{Ed}_{\texttt{L(i)}} + \texttt{Es}_{\texttt{L(i)}} + \texttt{MR}_{\texttt{L(i)}} \times \texttt{AMAE}_{\texttt{L(i+1)}}$.

Dynamic energy for accessing level i in the memory hierarchy is denoted by $\texttt{Ed}_{\texttt{L(i)}}$, the miss rate is denoted by $\texttt{MR}_{\texttt{L(i)}}$. $\texttt{Es}_{\texttt{L(i)}}$ is the total static energy consumed during an application's execution amortized over the number of memory accesses. CACTI estimates cache energy, DRAMSim estimates activate and read-write energy, and Micron power calculators [32] estimate DQ energy.

In conventional wisdom, the high latency and energy of DRAM, motivates larger processor caches that improve average memory access time and energy. By reducing memory activity, caches also increase opportunities for DRAM low-power modes while reducing the likelihood of memory contention. However, larger caches dissipate more static power and are less effective for emerging applications (e.g., web search, memcached). Balancing benefits and costs requires a holistic view with AMAE.

Figure 11 illustrates AMAE for a variety of L3 cache sizes. The trade-offs between dynamic and static energy vary across applications, which are placed along the x-axis in order of decreasing memory intensity. Accessing DDR3 is expensive and larger caches mitigate its cost. AMAT falls with cache size, especially for bandwidth-intensive applications. In contrast, the net change in AMAE from larger caches is modest. The increased energy of larger caches cancels, in large part, reductions in DRAM dynamic energy (due to fewer accesses) and DRAM static energy (shorter execution time).

Consuming less energy, LPDDR2 reduces AMAE when compared to DDR3. Lower LPDDR2 energy also magnifies the impact of static energy as L3 cache sizes increase. For many workloads, increasing the cache size leads to flat or increasing AMAE. Although larger caches reduce execution time, LPDDR2 is energy-proportional and opportunities to further reduce memory static energy are small. On the other hand, larger caches introduce a new problem in cache static energy. For AMAE, it is preferable to pay the high dynamic energy accessing DRAM rather than rather than continuously consume high static energy for a large L3. Most importantly, perhaps, we illustrate an analysis framework and an AMAE metric that allows architects to reason about emerging memory technologies and their interactions with other layers in the cache and memory hierarchy.

8. DATACENTER COST IMPLICATIONS

As servers adopt mobile hardware for efficiency, more of each dollar is spent on computing and less is spent on overheads. On the other hand, at least initially, capital costs for mobile hardware may be higher. Table 3 analyzes these trade-offs, accounting for capital costs in datacenter construction and IT equipment and operating costs from power [12, 40]. The model assumes $0.07/kWh, $200M facility cost, and a 15MW budget. Facility and IT capital costs are amortized over 15 and 3 years, respectively.

TCO-neutral prices. In the early stages of a new technology (e.g., LPDDR2-based servers) when costs are evolving, end-users might more tractably reason about the price they would willingly pay for expected benefits. We define a TCO-neutral price for a component as the price that produces a TCO matching some baseline. Our baseline is Xeon+DDR3 (TCO=$86).

Until now, advances in processor efficiency have out-paced those for memory. Moreover, simpler OOO cores or in-order cores, such as mobile Atoms are being considered for servers. Conventional motherboards are over-provisioned for such cores motivating systems like the one from SeaMicro, which eliminates 90% of motherboard components [30]. By adopting Atoms or other power efficient processors, server power falls from 190 to 63W, a 3× reduction. To realize such efficiency, datacenters might willingly pay for custom boards as long as TCO does not change. Sweeping board prices, TCO is held to $86 if customization costs less than $1340.

As processor efficiency improves, memory becomes an efficiency bottleneck. DDR3 dissipates 4-5× more power than LPDDR2 for applications with moderate memory activity, such as web search. Because mobile memory reduces power costs, datacenters might willingly pay a premium for LPDDR2. Sweeping LPDDR2 prices, we find Atom+LPDDR2 is justified if mobile memory prices are less than $775 per 32GB, a 30% premium over

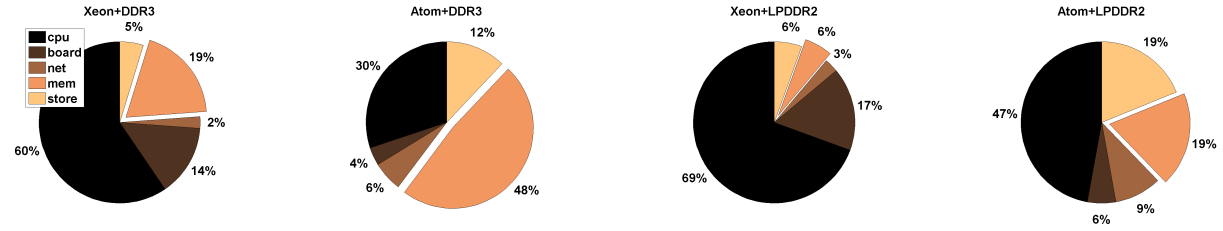


Figure 12: Power breakdown across server components.

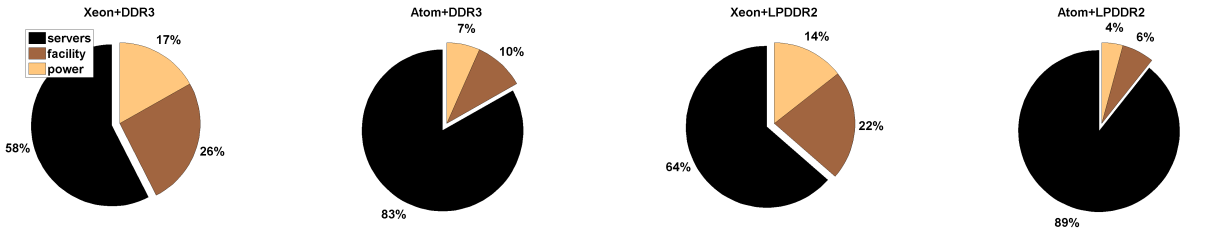


Figure 13: Total cost of ownership breakdown across servers and infrastructure.

DDR3 prices. This analysis is conservative because it precludes TCO increases, which might be justified by additional datacenter capacity enabled by mobile hardware.

Capacity. Mobile processors and memory shift TCO breakdowns (Figure 13). Of each dollar spent, 89% goes to server costs and not infrastructure overheads. In contrast, with Xeon+DDR3 servers, only 58% of costs go to servers. Table 3 presents datacenter capacity normalized to Xeon+DDR3 for web search based on published measurements [40]. Within a 15MW critical load, we can deploy 2.5× more 16-core Atom servers than 8-core Xeon servers, leading to a commensurate capacity increase even when taking into account that an Atom core sustains 0.5× the query throughput of a Xeon core. Atom+LPDDR2 power is even lower and allows a further 1.6× increase in the number of servers. Capacity increases by a cumulative total of 4.0× over Xeon+DDR3.

This analysis assumes no performance penalty from mobile memory. While true for search, which uses less than 10% of DDR3 peak bandwidth [23], other applications may see performance penalties that degrade the 1.6× gain from LPDDR2. For example, if application performance falls by 20% when using mobile memory, the 1.6× increase in servers is offset, in part, by the 0.8× impact on per server capacity. Thus, we illustrate a framework that allows architects to reason about justifiable prices for the benefits of emerging technologies.

9. RELATED WORK

To improve the efficiency of server-class memory, Lim et al. compare various grades and generations of DDR [26, 27]. These technologies provide modest performance and efficiency trade-offs. To increase the dynamic range of these trade-offs, one might reconfigure the voltage and frequency of memory systems [6, 8]. Such an approach allows users to match memory system capabilities to application demands and increase efficiency. However, voltage scaling applies only to the memory core and only in a narrow range defined by the device margins. Frequency scaling applies to the memory channel, reducing power, but often increasing the energy per bit because of the static power component.

A large body of work manages data placement [10, 24, 47, 9, 38, 16] and batches memory requests [7, 17] to increase the length of idle periods, which are necessary to exploit power modes that have long exit latencies. Despite these techniques, DDR power modes often incur performance penalties due to batching and the latency of the critical word is always affected. Although many of these techniques could be applied to LPDDR2, mobile-class power modes have much shorter exit latencies, effectively eliminating the problem these techniques try to solve. With faster transitions between power modes, LPDDR2 provides more opportunities to exploit them or simpler control heuristics to manage them.

Seeking to improve energy proportionality, other researchers work to narrow the access width of a memory operation. Instead of accessing a wide row that spans multiple KB, they access a smaller subset [1, 48, 51, 54]. However, rank subsetting significantly increases the amount of peripheral circuitry in a chip, degrading density [49]. In contrast, LPDDR technologies use the same DDR memory core, achieving comparable densities. LPDDR only changes the interface circuitry, addressing the power overheads in chip I/O, the largest efficiency bottleneck in DRAMs [49].

10. CONCLUSION

Many data center applications under-utilize the memory bandwidth available in modern servers. We presented an alternative design for server memory systems that uses commodity DRAM chips for mobile applications. These memory systems tradeoff peak bandwidth for significant gains in energy efficiency and proportionality. We have shown that LPDDR2 based servers provide high capacity and small performance impact for data center workloads.

11. ACKNOWLEDGMENTS

We sincerely thank David Lo, Jacob Leverich, Daniel Sanchez, Scott Best and Wayne Richardson for their useful feedback and discussion. Krishna Malladi is supported by Benchmark-Capital Stanford Graduate Fellowship. This work is supported in part by NSF grant CCF-1149252.

12. REFERENCES

[1] J. Ahn et al. Future scaling of processor-memory interfaces. In *SC*, 2009.

[2] M. Awasthi et al. Handling the problems and opportunities by multiple on-chip memory controllers. In *PACT*, 2011.

[3] L. Barroso et al. The case for energy proportional computing. *IEEE Computer*, 2007.

[4] A. Bosque et al. Characterization of Apache web server with Specweb2005. In *MEDEA*, 2007.

[5] Bossen et al. b-adjacent error correction. *IBM Journal of Research and Development*, 1970.

[6] H. David, O. Mutlu, et al. Memory power management via dynamic voltage/frequency scaling. In *ICAC*, 2011.

[7] V. Delaluz et al. DRAM energy management using software & hardware directed power mode control. In *HPCA*, 2001.

[8] Q. Deng et al. Memscale: Active low-power modes for main memory. In *ASPLOS*, 2011.

[9] B. Diniz, D. Guedes, and R. Bianchini. Limiting the power consumption of main memory. In *ISCA*, 2007.

[10] X. Fan, C. Ellis, and A. Lebeck. Memory controller policies for DRAM power management. In *ISLPED*, 2001.

[11] B. Ganesh et al. FB DIMM memory architectures: Understanding mechanisms, overheads, & scaling. In *HPCA*, 2007.

[12] J. Hamilton. Cost of power in large-scale data centers. `http://perspectives.mvdirona.com`.

[13] H. Hanson et al. What computer architects need to know about memory throttling. In *WEED*, 2010.

[14] Hewlett-Packard. DDR3 memory technology. Technology brief TC100202TB, Hewlett-Packard, 2010.

[15] U. Hoelzle and L. Barroso. *The Datacenter as a Computer*. Morgan and Claypool, 2009.

[16] H. Huang et al. Improving energy efficiency by making DRAM less randomly accessed. In *ISLPED*, 2005.

[17] I. Hur and C. Lin. A comprehensive approach to DRAM power management. In *HPCA*, 2008.

[18] B. Jacob et al. *Memory Systems: Cache, DRAM, Disk*. Morgan Kaufmann, 2007.

[19] A. Jaleel et al. High performance cache replacement using re-reference interval prediction (RRRIP). In *ISCA*, 2010.

[20] JEDEC. JEDEC standard for LP-DDR2. Standard JESD209-2B, JEDEC, 2010.

[21] B.-H. Jeong et al. A 1.35V 4.3GB/s 1Gb LPDDR2 DRAM with controllable repeater and on-the-fly power-cut scheme for low-power and high-speed mobile application. In *ISSCC*, February 2009.

[22] Kim et al. Thread cluster memory scheduling: Exploiting differences in memory access behavior. In *MICRO*, 2010.

[23] C. Kozyrakis, S. Sankar, et al. Server Engineering Insights for Large-Scale Online Services. *IEEE Micro*, 2010.

[24] A. Lebeck, X. Fan, H. Zeng, , and C. Ellis. Power aware page allocation. In *ASPLOS*, 2000.

[25] S. Li et al. McPaT: An integrated power, area, and timing modeling framework for multicore and manycore architectures. In *MICRO*, 2009.

[26] K. Lim et al. Understanding and designing new server architectures for emerging warehouse-computing environments. In *ISCA*, 2008.

[27] K. Lim et al. Disaggregated memory for expansion and sharing in blade servers. In *ISCA*, 2009.

[28] C. Luk et al. Pin: Building customized program analysis tools with dynamic instrumentation. In *PLDI*, 2005.

[29] D. Meisner, B. Gold, and T. Wensich. PowerNap: Eliminating server idle power. In *ISCA*, 2009.

[30] R. Merritt. Startup SeaMicro packs 512 Intel Atoms in server. *EE Times*, 2010.

[31] Micron. Calculating memory system power for DDR2. Technical Note tn4704, Micron, 2005.

[32] Micron. Calculating memory system power for DDR3. Technical Note TN-41-01, Micron, 2007.

[33] Micron. 152-ball x32 mobile lpddr pop (ti-omap). Data Sheet MT46HxxxMxxLxCG, Micron, 2008.

[34] Micron. Micron 1Gb: x16, x32 Mobile LPDDR SDRAM. Data Sheet MT46H32M32LF-5, Micron, 2010.

[35] Micron. Micron 2Gb: x4, x8, x16 DDR3 SDRAM. Data Sheet MT41J128M16HA-125, Micron, 2010.

[36] J. Ousterhout et al. The case for RAMClouds:scalable high-performance storage entirely in DRAM. *SIGOPS*, 2010.

[37] R. Palmer et al. A 4.3GB/s mobile memory interface with power-efficient bandwidth scaling. In *VLSI*, 2009.

[38] V. Pandey, W. Jiang, Y. Zhou, and R. Bianchini. DMA-aware memory energy management. In *HPCA*, 2006.

[39] Rambus. Mobile XDR memory versus LPDDR2. `http://www.rambus.com/us/technology/solutions/mobile_xdr/mxdr_vs_lpddr2.html`.

[40] V. Reddi et al. Web search using mobile cores: Quantifying and mitigating the price of efficiency. In *ISCA*, 2010.

[41] B. Rogers, A. Krishna, G. Bell, K. Vu, X. Jiang, and Y. Solihin. Scaling the bandwidth wall: Challenges in and avenues for CMP scaling. In *ISCA*, 2009.

[42] P. Saab. Scaling memcached at Facebook. *Facebook Engineering Note*, 2008.

[43] R. Schmitt et al. Signal and power integrity limitations for mobile memory in 3D packaging. *EE Times*, 2010.

[44] N. Sharma, S. Barker, D. Irwin, and P. Shenoy. Blink: Managing server clusters on intermittent power. In *ASPLOS*, 2011.

[45] K. Shiv et al. SPECjvm2008 performance characterization. In *SPEC Benchmark Workshop on Computer Performance Evaluation and Benchmarking*, 2009.

[46] L. Tang et al. The impact of memory subsystem resource sharing on datacenter applications. In *ISCA*, 2011.

[47] M. Tolentino et al. Memory MISER: Improving main memory energy efficiency in servers. *IEEE Trans*, 2009.

[48] A. Udipi et al. Rethinking DRAM design and organization for energy-constrained multi-cores. In *ISCA*, 2010.

[49] T. Vogelsang. Understanding the energy consumption of dynamic random access memories. In *MICRO*, 2010.

[50] D. Wang et al. DRAMSim2: A cycle accurate memory system simulator. *IEEE Comput. Archit. Letters*, 2011.

[51] F. Ware and C. Hampel. Improving power and data efficiency with threaded memory modules. In *ICCD*, 2006.

[52] D. H. Yoon and M. Erez. Virtualized and flexible ecc for main memory. In *ASPLOS*, 2010.

[53] L. Zhao et al. Exploring Large-Scale CMP Architctures using Manysim. *IEEE Micro*, 2007.

[54] H. Zheng et al. Mini-rank: Adaptive DRAM architecture for improving memory power efficiency. In *MICRO*, 2008.