A Comprehensive Memory Modeling Tool and its Application to the Design and Analysis of Future Memory Hierarchies

Shyamkumar Thoziyoor†, Jung Ho Ahn‡, Matteo Monchiero‡, Jay B. Brockman†, Norman P. Jouppi‡
†University of Notre Dame, ‡Hewlett-Packard Labs

Abstract

In this paper we introduce CACTI-D, a significant enhancement of CACTI 5.0. CACTI-D adds support for modeling of commodity DRAM technology and support for main memory DRAM chip organization. CACTI-D enables modeling of the complete memory hierarchy with consistent models all the way from SRAM based L1 caches through main memory DRAMs on DIMMs.

We illustrate the potential applicability of CACTI-D in the design and analysis of future memory hierarchies by carrying out a last level cache study for a multicore multithreaded architecture at the 32nm technology node. In this study we use CACTI-D to model all components of the memory hierarchy including L1, L2, last level SRAM, logic-process based DRAM or commodity DRAM L3 caches, and main memory DRAM chips. We carry out architectural simulation using benchmarks with large data sets and present results of their execution time, breakdown of power in the memory hierarchy, and system energy-delay product for the different system configurations. We find that commodity DRAM technology is most attractive for stacked last level caches, with significantly lower energy-delay products.

1 Introduction

Future processor chip implementations are expected to be composed of 3D die stacks that could include multiple dies fabricated in diverse fabrication technologies. One design possibility which is particularly attractive is the idea of stacking cache or memory on top of a die that contains multiple CPU cores [6, 16]. Future many-core processors are expected to have large memory bandwidth requirements and stacking cache or memory directly on top of the cores is one possible way of providing the required bandwidth.

There are several options available for stacked memory. Traditionally, SRAM has been the memory technology of choice to implement caches and other memory structures inside a processor die. This is because the processing of SRAM does not add any steps to a logic process and SRAM has been the practical memory technology that could meet access time and random cycle time requirements. Recently however, DRAM has also found its way into various applications. The main compute chip inside the Blue Gene/L supercomputer uses embedded DRAM [11]. Embedded DRAM has also been used in the graphics synthesizer unit of Sony's PlayStation2 [27]. The main reason for the use of embedded DRAM is its much smaller 1T1C cell (compared to 6T SRAM cell) which consumes less area and leakage power. Also, the trend of worsening wire delay with respect to device delay has meant that the smaller cell size of DRAM has a large impact in reducing the propagation delay over its interconnect, which can be significant in large RAMs [21]. The embedded DRAM technology that has been used in commercial applications so far has been logic process based (LP-DRAM), which is the same process used to fabricate the processor or logic die. In contrast, commodity DRAM (COMM-DRAM) used in main memory chips is fabricated in its own commodity DRAM fabrication process. Compared to COMM-DRAM technology, LP-DRAM technology typically has faster transistors and circuitry but worse cell density and retention time [38]. These different technology characteristics translate into different area, delay, and energy properties for LP-DRAM and COMM-DRAM memories with identical functional specifications.

CACTI [34, 40] is a tool that has become widely used in the computer architecture community by architects either directly for modeling of SRAM based caches and plain memories, or indirectly through other tools such as Wattch [7]. CACTI 5.0 [35] was a major revision of CACTI overcoming several limitations of earlier versions. First, the technology models of earlier versions were largely based on linear scaling of a 0.8μm process. However, as scaling has progressed to nanometer dimensions, scaling has become much less linear. To better model this, the base technology modeling in CACTI 5.0 was changed from simple linear scaling of the original CACTI 0.8μm technology to models based on the ITRS roadmap. Second, a major enhancement in CACTI 5.0 was the support added for DRAM circuits along with SRAM. Logic process based DRAM (LP-DRAM) technology has been gaining in popularity, and CACTI 5.0 provides support for modeling embedded memories and caches built on LP-DRAM technology.

1063-6897/08 $25.00 © 2008 IEEE
DOI 10.1109/ISCA.2008.16

In this paper we introduce a version of CACTI that we call CACTI-D that enhances CACTI 5.0. Caches implemented by stacking COMM-DRAM parts on top of processor die have recently been of research interest [6, 16]. CACTI-D supports both LP-DRAM and COMM-DRAM along with the more traditional SRAM. Thus, modeling of the complete memory hierarchy with consistent models all the way from SRAM based L1 caches through main memory DRAMs on DIMMs becomes possible. We illustrate the potential applicability of CACTI-D in the design and analysis of future memory hierarchies by carrying out a last level cache study for a 32nm multicore multithreaded architecture.

2 CACTI-D

A detailed description of all aspects of CACTI-D including modeling of circuits, area, delay, dynamic energy, and leakage power are beyond the scope of this paper, but can be found in the CACTI 5.1 technical report [34]. In the rest of this section we focus on describing key aspects of CACTI-D related to memory array organization with particular emphasis on modeling of main memory organization, device and interconnect technology modeling, DRAM modeling, solution optimization process, and model validation.

2.1 Main Memory DRAM Organization

In CACTI-D we added modeling support for capturing the properties of a main memory DRAM chip organization. We specify the number of banks (N_{banks}) equal to the number of banks in the main memory DRAM chip being modeled; most main memory DRAM chips today have 8 banks. Main memory DRAM chips in DIMMs typically have 4 or 8 data output pins. We model the *internal prefetch width* which determines the number of bits prefetched internally inside the DRAM core. Because of their limited number of data pins, main memory DRAM chips usually operate in burst mode with a *burst length* of 4 or 8. The burst length determines the effective number of data bits that are accessed by a READ or WRITE command. We incorporate the concept of burst length into the CACTI-D array organization, and modify the energy model accordingly. In CACTI-D, we also incorporate the concept of page size by constraining the total number of sense amplifiers in a subbank to be equal to the page size. As shown later in Section 2.5, even with such relatively simple changes we are able to model the area, timing, and power specifications of a main memory DRAM chip with reasonable accuracy.

2.2 Technology Modeling

Unlike versions of CACTI before 5, with a technology model largely based on linear scaling of a 0.8μm technology, CACTI-D uses technology projections from the ITRS [32] for device data and projections by Ron Ho [10, 31] for wire data. CACTI-D includes technology data for four ITRS technology nodes – 90, 65, 45, and 32 nm – which cover years 2004 to 2013 of the ITRS.

2.2.1 Devices

We include data for the three device types that the ITRS defines – High Performance (HP), Low Standby Power (LSTP), and Low Operating Power (LOP). The HP transistors are state-of-the-art fast transistors with short gate lengths, thin gate oxides, low V_{th} and low V_{DD} whose CV/I delay is targeted to improve by 17% every year. As a consequence of their high on-currents, these transistors tend to be very leaky. The LSTP transistors on the other hand are transistors with longer gate lengths, thicker gate oxides, higher V_{th}, and higher V_{DD}. The gate-lengths of the LSTP transistors lag the HP transistors by 4 years. The LSTP transistors trade off high on-currents for maintenance of an almost constant low leakage of 10pA across the technology nodes. The LOP transistors lie in between the HP and LSTP transistors in terms of performance. They use the lowest V_{DD} to control the operating power and their gate-lengths lag those of HP transistors by 2 years. The CV/I of the LSTP and LOP transistors improves by about 14% every year. In addition to data for these ITRS device types, it is also possible for the user to add device data obtained from other sources. For example, we also add long-channel variations of the HP transistors which trade off transistor speed for reduction in leakage.

2.3 DRAM Modeling

In our modeling of DRAM, we leverage the similarity that exists in the peripheral and global support circuitry of SRAM and DRAM and model only their essential differences. We assume a folded array organization [15] for the DRAM subarray and keep the rest of the array organization identical to that of SRAM. We use the same methodology for SRAM and DRAM for sizing of peripheral support circuitry such as decoders and drivers, and global support circuitry such as repeaters and tristate drivers. We use an analytical gate area model which makes the areas of gates sensitive to transistor sizing so that when transistor sizes change, gate areas also change. The gate area model also considers context-sensitive pitch-matching constraints for circuits such as wordline drivers and sense amplifiers, so transistors of these circuits may get folded and their areas calculated based on the pitch they need to satisfy. This feature is useful in capturing differences in area caused due to different pitch-matching constraints of SRAM and DRAM. By having a common framework that in general places SRAM and

Characteristic	SRAM	LP-DRAM	COMM-DRAM
Cell area (F = Feature-size)	$146F^2$	$30F^2$	$6F^2$
Memory cell device type	ITRS HP/Long-channel	Intermediate oxide based	Conventional oxide based
Peripheral/Global circuitry device type	ITRS HP/Long-channel	ITRS HP/Long-channel	ITRS LSTP
Bitline interconnect	Copper	Copper	Tungsten
Back-end-of-line interconnect	Copper	Copper	Copper
Memory Cell V_{DD} (V)	0.9	1.0	1.0
DRAM storage capacitance (fF)	N/A	20	30
Boosted wordline voltage V_{PP} (V)	N/A	1.5	2.6
Refresh period (ms)	N/A	0.12	64

Table 1: Key characteristics of SRAM, LP-DRAM, and COMM-DRAM technologies.

DRAM on an equal footing and emphasizes only their essential differences, we are able to compare relative tradeoffs involving SRAM and DRAM.

For LP-DRAM, we obtain technology data by extrapolating published data of 180/130/90/65nm LP-DRAM processes [12, 38]. For COMM-DRAM, we obtain technology data by using projections from the ITRS and other sources [3, 23, 24]. Table 1 shows key characteristics of SRAM, LP-DRAM, and COMM-DRAM technologies. Specific parameter values are from our projections for 32nm. We assume that SRAM cells use long-channel ITRS HP transistors, similar to that employed in the 65nm Intel Xeon L3 cache [8]. We assume that COMM-DRAM cells employ conventional thick oxide based transistors, and LP-DRAM cells employ intermediate oxide based transistors similar to that described in [38]. For the peripheral and global support circuitry of the RAMs, we assume that SRAMs and LP-DRAMs use ITRS long-channel HP transistors while COMM-DRAMs use LSTP transistors [23]. Next we describe the essential features of the DRAM circuits in our area, delay, and energy models.

2.3.1 Cell

The most essential difference between SRAM and DRAM is in their storage cell. While SRAM typically uses a 6T cell, DRAM typically uses a 1T1C cell. Table 1 shows the areas we have assumed for SRAM, LP-DRAM, and COMM-DRAM cells. The LP-DRAM cells presented in [38] for four different technology nodes – 180/130/90/65nm – have areas in the range of 19–$26F^2$. In contrast, a typical SRAM cell would have an area of about 120–$150F^2$. COMM-DRAM cells occupy the least area, with typical values in the range of 6–$8F^2$. Table 1 also shows storage capacitance values for LP-DRAM and COMM-DRAM technologies. These values are chosen by assuming that storage capacitance does not change much with technology scaling as storage V_{DD} decreases in order to meet signal-to-noise and retention time targets.

2.3.2 Destructive Readout and Writeback

When data is read out from a DRAM cell, the charge stored in the cell gets destroyed because of charge redistribution between the cell and its capacitive bitline, and there is a need for data to be written back into the cell. Also, after a writeback the bitlines need to be restored to their precharged value. These writeback and restore operations take time and increase the random cycle time of a DRAM array. We include these operations in the CACTI-D delay and energy models.

2.3.3 Refresh

In a DRAM cell, charge leaks out from the capacitor because of various leakage components, so a DRAM cell needs to be refreshed periodically. In CACTI-D, we incorporate refresh power in our DRAM power model. As shown in Table 1, the LP-DRAMs that we have modeled have much lower refresh period (0.12ms) than the COMM-DRAMs (64ms), which means that the LP-DRAM cells have to be refreshed much more often than the COMM-DRAM cells. This is because the LP-DRAM cells employ transistors with oxides that are much thinner than those of the COMM-DRAM cells.

2.3.4 DRAM Operational Models

The operational models of DRAMs may vary depending on whether they are employed in main memory chips or whether they are embedded or stacked. The operational models have an impact on the DRAM timing interface and timing model. In CACTI-D, we include timing interfaces and timing model equations suitable for the different possible models of operation.

Main Memory Operational Model Main memory COMM-DRAM chips are operated using ACTIVATE, READ, WRITE, and PRECHARGE commands.[1] ACTIVATE latches data into a DRAM page. After read or write

[1]It is important to note that in reality the operation of a main memory chip is more complicated than our simplified description here.

operations to this page are carried out, the page can be closed; data in the sense amplifiers are written back into the DRAM cells and the bitlines are restored to their precharged state. Since activating or precharging a page takes time and energy, it is beneficial to decrease the number of activate and precharge operations in accessing a sequence of data requests in a DRAM. Once a page is activated to serve a request, the page can be kept open with the hope that future near-term requests will also target the same page (the *open* page policy) [29]. However, keeping a page open causes more energy leakage over time and additional precharge latency when a subsequent request heads to a different page. The alternative policy, the *closed* page policy, proactively closes pages unless it is already known that the next request hits the same page. If requests are sparse in time or do not hit the page often in sequence, this policy can be more attractive. The page policy of the main memory DRAM chips can be appropriately chosen by the architect depending on the target application.

An important operational principle employed in main memory DRAM chips is the concept of multibank interleaving. The random cycle time (t_{RC}) of main memory DRAM chips tends to be quite large (typically around 50ns). But consecutive accesses to different banks need not be limited by the random cycle time, so main memory DRAM chips also include a specification of multibank interleave cycle time, t_{RRD}. A typical value for would be 7.5ns, so with multibank interleaving the throughput from a main memory DRAM chip can be significantly increased.

Embedded DRAM Operational Models One possible way of operating embedded DRAMs is to operate them like main memory DRAM chips using ACTIVATE, READ, WRITE, and PRECHARGE commands. If the target application can benefit from the spatial locality that an open page can offer, then operating an embedded DRAM with an open page policy can offer delay and energy benefits. Another possibility is to operate an embedded DRAM with a vanilla SRAM-like interface [26]. In this case, like in an SRAM, the DRAM is operated using just READ and WRITE commands, and activate and precharge operations are not visible to the embedded DRAM user. A READ (or WRITE) command is associated with both row and column addresses and causes data to be latched into sense amplifiers and then read out. In case of a READ or WRITE, the precharge operation can proceed soon after data is latched in the sense amplifiers which can help in lowering the random cycle time. The flip side of using a vanilla SRAM-like interface is that delay and energy benefits associated with *page access* may be lost. A vanilla SRAM-like interface can still work well for applications without much spatial locality exploitable by page access. Multibank interleaving can improve the vanilla SRAM-like interface, the same as in main memory DRAM. When an embedded DRAM has multiple banks that share the address and data bus, adding a multibank interleave cycle time to its timing interface can improve its throughput just like in a main memory DRAM chip. In CACTI-D, we extend the concept of multibank interleaving to an embedded DRAM with a single bank by considering *multisubbank interleaving*. In CACTI-D a bank is composed of multiple subbanks that share the same address and data bus, so it is possible to interleave accesses to different subbanks of a bank. With pipelining in the access path, the multisubbank interleave cycle time can be improved even further.

2.3.5 DRAM Timing Model

When an embedded DRAM is operated using a vanilla SRAM-like interface, its timing specifications are also similar to that of an embedded SRAM, thus the timing specifications of interest are the access time and random cycle time. In this case, the access time and random cycle time are computed similar to earlier versions of CACTI. In addition to these metrics, we also compute a multisubbank interleave cycle time which can be utilized to take advantage of multisubbank interleaving. For embedded DRAMs operated using a main memory-like interface, we compute timing specifications similar to that of a main memory COMM-DRAM chip, albeit with simplifications. In our main memory timing model we add equations for computation of activation delay, CAS latency, precharge delay, and random cycle time.

2.4 Solution Optimization Methodology

In CACTI 5, we follow a different approach from previous versions of CACTI in finding the optimal solution given an input cache or memory specification. Our new approach allows users to exercise more control on the area, delay, and power of the final solution. The optimization is carried out in the following steps: first, we find all solutions with area within a certain percentage (user-supplied value) of the area of the solution with best area efficiency. We refer to this area constraint as `max_area_constraint`. Next, from this reduced set of solutions that satisfy the `max_area_constraint`, we find all solutions with access time within a certain percentage of the best access time solution. We refer to this access time constraint as `max_acctime_constraint`. To this subset of solutions, we apply an optimization function composed of normalized, weighted combinations of dynamic energy, leakage power, random cycle time, and multisubbank interleave cycle time. In addition there are other internal variables inside CACTI-D that help guide sizing of circuits advanced users can control. One such variable is `max_repeater_delay_constraint` which controls how much worse the delay of repeaters is allowed to get compared to the best delay repeater solution — with this, limited

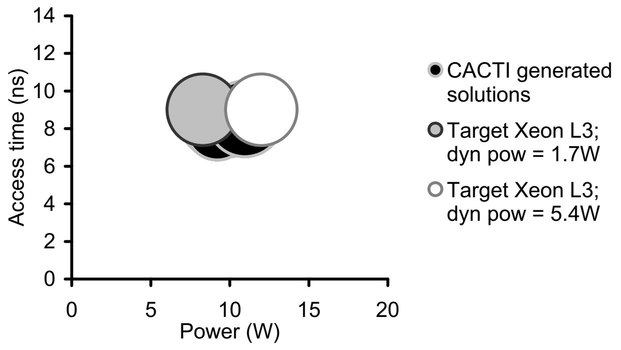


Figure 1: Access time, area and power of the 65nm Xeon L3 cache and of solutions generated by CACTI-D. Two bubbles for the Xeon cache correspond to two quoted values of dynamic power. An activity factor of 1.0 is assumed while computing dynamic power for the CACTI-D generated solutions.

energy savings are possible at the expense of delay. Such an optimization process allows exploration of the solution space in a controlled manner and to arrive at a solution with user-desired characteristics.

2.5 Validation

In order to validate the SRAM modeling of CACTI-D, we compare its projections against published data of a 65nm 16MB Intel Xeon L3 cache [8] and a 90nm 4MB Sun SPARC L2 cache [22]. We choose input specifications for the caches and device/interconnect assumptions based on the published data. In order to explore area, delay, and power tradeoffs that may be possible in the cache design, we vary optimization variables such as `max_area_constraint`, `max_acctime_constraint`, and `max_repeater_delay_constraint` that were described in Section 2.4 within reasonable bounds. In order to make a fair comparison of power for the Xeon L3 cache, we model the impact of sleep transistors, which cut in half the leakage of all mats that are not activated during an access.

Complete details of the validation exercise are described in the CACTI 5.1 tech report [34]; here we simply summarize the results of the validation. Figure 1 shows a bubble chart to illustrate the results of validation for the 65nm Xeon L3. Bubbles are shown corresponding to the target Xeon L3 and various CACTI-D solutions. The area of the bubble corresponds to area of the cache. There are two target bubbles for the Xeon L3 and this corresponds to two quoted dynamic power numbers [8, 37] which we attribute to different activity factors caused by different applications. For both the 65nm Xeon L3 and a similar analysis performed for the 90nm SPARC L2 (not shown due to space limitations), the best access time solution produced by CACTI-D has an average error of about 20% across the dimensions of access time, area, and power. Given the generic nature of CACTI-D, and the variation of specific fabrication processes from the ITRS roadmap, we consider this error to be reasonable.

Metric	Actual value	CACTI-D error
Area efficiency	56%	-6.2%
Activation delay (t_{RCD}) (ns)	13.1	4.5%
CAS latency (ns)	13.1	-5.8%
Row cycle time (t_{RC}) (ns)	52.5	-8.2%
ACTIVATE energy (nJ)	3.1	-25.2%
READ energy (nJ)	1.6	-32.2%
WRITE energy (nJ)	1.8	-33%
Refresh power (mW)	3.5	29%

Table 2: Results of CACTI-D DRAM model validation with respect to a 78nm Micron 1Gb DDR3-1066 x8 DRAM.

In order to validate our DRAM modeling, we compare projections produced by CACTI-D against timing and power data of a 78nm Micron 1Gb DDR3-1066 x8 DRAM device [1]. For timing, we refer to the datasheet of the DRAM device and for power we use the DDR3 Micron power calculator [2]. While the area efficiency of the actual device is not known, we assume its area efficiency to be 56% based on the area efficiency value specified in the ITRS [32] for a $6F^2$ cell based DRAM. Table 2 shows the timing and power data obtained for the 78nm Micron device and the error values with respect to the chosen CACTI-D solution. Multiple CACTI-D solutions are feasible with varied area/delay/power tradeoffs; because of the premium on price per bit of commodity DRAM we select one with high area efficiency. The ACTIVATE, READ, and WRITE energies for the Micron device have been computed by appropriate specification of system usage conditions in the Micron power calculator, and then using the values of power components and average delay between commands that it generates. Note that ACTIVATE energy includes energy due to both activation and precharging. It can be seen that there is good agreement in all metrics with an average error of 16%.

3 Stacked Last Level Cache (LLC) Study

We illustrate the utility of CACTI-D by applying it to the design and architectural evaluation of stacked last level SRAM, LP-DRAM, or COMM-DRAM based caches at the 32nm node for a multicore multithreaded processor architecture.

3.1 System Architecture

Figure 2 shows a block diagram of our assumed system architecture. The processor chip is assumed to be composed

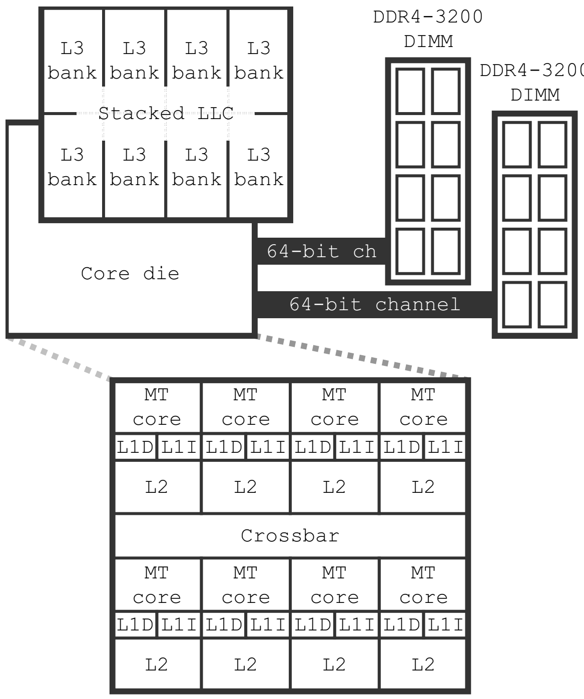


Figure 2: System architecture for LLC study.

of a 2-layer die stack with the top die implementing the shared last level (L3) cache. The bottom die is composed of 8 multithreaded 32nm Niagara-like cores with one 4-way SIMD FPU per core. Each core is assumed to have 32KB 8-way set-associative private SRAM-based L1 instruction and data caches and a 1MB 8-way set-associative private SRAM-based unified L2 cache. The main memory subsystem is assumed to be composed of two channels with each channel connected to a single-ranked 8GB DIMM that is made up of 8Gb DDR4-3200 devices which are also assumed to be fabricated on 32nm technology.

We assume that the LLC is composed of 8 banks connected to the 8 L2 banks of the core die through a crossbar implemented in the core die. For the 3D connections, we assume face-to-face through-silicon via technology similar to that described in [28] which has sub-FO4 communications delays. We compute the area occupied by the bottom die by scaling the area of Niagara [17] core components to 32nm and using CACTI-D to compute the area of the L1 and L2 caches. The area available per LLC bank is fixed to be 1/8th of the bottom core area and this comes out to be 6.2mm^2.

3.2 Application Benchmarks

SPEC benchmarks fit in caches much smaller than the 192MB caches we study, and many other benchmarks have had their data sets limited to reduce simulation running time. In order to exercise our very large caches, we use a subset of NAS Parallel Benchmark (NPB) [14] applications in our chip multiprocessor system LLC study. We scaled the version 3.2.1 NPB applications to match our simulated system. We skipped the initialization phase of each benchmark, and then started timing simulation and data collection. We executed 10 billion instructions after initialization. OpenMP versions of NPB applications were used to distribute the workload across 32 hardware threads. The benchmarks were compiled with the Intel C and Fortran compilers (ver. 10.1).

3.3 Simulation Methodology

HP Labs' COTSon simulator [9] based on AMD's SimNow infrastructure is used for performance evaluation with timing simulation modified for our target system. A full-system emulator executes each multithreaded NPB application on top of the guest OS (Linux), generating instruction sequences with supplementary information like thread ID and synchronization primitives annotated. The timing simulator takes these instruction sequences, distributes them into simulated cores by thread mapping, and executes them following the constraints imposed by synchronization primitives such as locks and barriers. Four hardware threads are executed concurrently in a simulated core and each thread is modeled to execute a floating point arithmetic instruction every cycle (modeling SIMD execution), and all other instructions every 4 cycles on average with up to 1 memory request generated to the L1 cache per cycle. A MESI protocol is used for cache coherency.

3.4 Operational Considerations for Embedded/Stacked DRAM Cache

In Section 2.3.4 we discussed the different operational models and options associated with operating an embedded or stacked DRAM. An embedded/stacked DRAM can either be operated using a main memory-like interface or it may be operated using an SRAM-like interface with an optional multibank or multisubbank interleaving cycle time added to its timing interface in order to improve throughput. When the embedded/stacked DRAM is employed as cache there are certain additional considerations related to cache access that influence the choice of DRAM operational model. In a DRAM cache, the access granularity is a cache line (less than 1Kb), which is much smaller than a DRAM page size (the LP-DRAM and COMM-DRAM caches in our LLC study have page sizes of either 8Kb or 16Kb). For a normal access mode where both tags and data are accessed concurrently, it is natural to map a cache set to a DRAM page (Figure 3(a)) since all the cache lines in a set will be fetched during a cache access, exploiting locality within a page. However for a sequential access cache, where data is accessed only after the tag lookup in order to save energy [35], map-

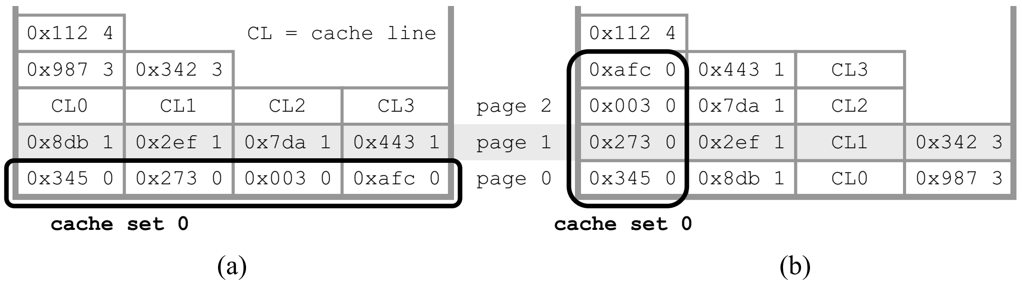


Figure 3: Mapping between cache lines and DRAM pages: a cache set is either (a) mapped to a page, or (b) striped across pages. The numbers within a cache line are the cache tag followed by the set number.

ping cache sets to DRAM pages does not provide locality since the specific cache line to access in a cache set is determined during the tag lookup phase, and it is unlikely that another cache line in the set will be accessed before a line in another set is accessed. Another possible mapping is to stripe cache sets across DRAM pages (Figure 3(b)). In this case, each DRAM page contains the same way in sequential cache sets. However since a cache set is fully associative, addresses in a DRAM page would not usually be sequential even when cache sets are sequential.

So for both mapping scenarios, an open page policy is not likely to be beneficial because the page hit ratio between successive memory requests is very low. When the size of a DRAM page is larger than a cache set, mapping multiple cache sets into a DRAM page improves spatial locality of a DRAM cache for sequential accesses, however its performance advantage is limited by the random access nature of last-level caches which receive interleaved requests across multiple simultaneously executed threads. Thus for our study, we operate DRAM caches using an SRAM-like interface with multisubbank interleaving and map multiple cache sets to DRAM pages.

4 Results

4.1 CACTI-D Cache Projections

We use CACTI-D to obtain projections for all levels of the memory hierarchy including L1, L2, and L3 caches, and main memory DRAM chips. With the optimization methodology of CACTI-D, for a given memory and cache input specification it is possible to obtain various feasible solutions that exhibit tradeoffs in area, access time, cycle times, dynamic energy, and leakage power. We apply optimizations and selected appropriate configurations for each level. Table 3 shows the values of key metrics of the caches and main memory used in our architectural study. For this study, we fix the clock frequency of the processor cores to 2GHz based on the access time of the 32KB L1 instruction and data cache.

For the 8-banked L3 cache, we consider 5 design options: 24MB (SRAM), 48MB and 72MB (LP-DRAM), and 96MB and 192MB (COMM-DRAM). For each DRAM technology, we choose one solution that is optimized for capacity (config C) and another one that uses smaller mats with better energy and delay properties (config ED) and the same associativity as the SRAM L3. We limit the clock frequency relative to the core for each cache configuration by limiting the maximum number of pipeline stages inside any of the caches to 6. Note that the access times shown in Table 3 are just for cache access and do not include communication or control delays. It can be seen that the access time and random cycle time of the 6MB LP-DRAM L3 bank is the same as that of the 3MB SRAM L3 bank. The access times of the COMM-DRAM configurations are about 3 times higher than that of comparable LP-DRAM configurations but still much smaller than that of the main memory DRAM chip because of their smaller capacity[2]. By sacrificing capacity, the 48MB LP-DRAM and the 96MB COMM-DRAM are able to have lower dynamic energy per read access than the 72MB LP-DRAM and the 192MB COMM-DRAM respectively. Because of the large access granularity of a main memory DRAM access and because 8 chips have to be accessed in parallel, the dynamic energy of a read access is much higher for main memory than for a L3 cache. Leakage of both LP-DRAM L3s (2.0W and 2.1W) is less than that of the SRAM L3 (3.6W) even though we have modeled an aggressive leakage control mechanism for the SRAM L3 similar to that of the 65nm Xeon [8].

We compute miss penalties for use inside the architectural simulator by considering cache hit times, tag array access times, and communication and control delays involved in transfer of information between cache levels. While computing miss penalty on an L2 miss we consider the communication delay through the crossbar between L2 and L3. Inside CACTI-D we incorporate a model [39] for the delay and energy consumed in a crossbar. We compute the length

[2]The access time shown for the main memory DRAM chip is the sum of t_{RCD} and CAS latency.

Parameter	L1	L2	L3					Main Memory
			SRAM	LP DRAM		COMM DRAM		DRAM chip
				ED	C	ED	C	
Capacity	32KB	1MB	24MB	48MB	72MB	96MB	192MB	8Gb
Number of banks	1	1	8	8	8	8	8	8
Number of subbanks	1	4	4	32	16	64	32	64
Associativity	8	8	12	12	18	12	24	N/A
Clock frequency w.r.t. CPU	1	1	1	1	1	1/3	1/4	1/16
Access time (CPU cycles)	2	3	5	5	7	16	21	61
Random cycle time (CPU cycles)	1	1	1	1	3	5	10	98
Area (mm^2)	0.17	2.0	6.2	5.7	6.0	4.8	6.2	115
Area efficiency (%)	25	67	64	36	51	30	47	46
Standby/Leakage power (W)	0.009	0.157	3.6	2.0	2.1	0.015	0.026	0.091
Refresh power (W)	0	0	0	0.3	0.12	0.00018	0.001	0.009
Dynamic read energy per cache line (nJ)	0.07	0.27	0.54	0.54	0.59	0.6	0.92	14.2

Table 3: Projections of key properties of the various caches and main memory chip at the 32nm node.

of the interconnect route between L2 and L3 by measuring the dimensions of the 8x8 crossbar from the Niagara2 die photo and then scaling to 32nm.

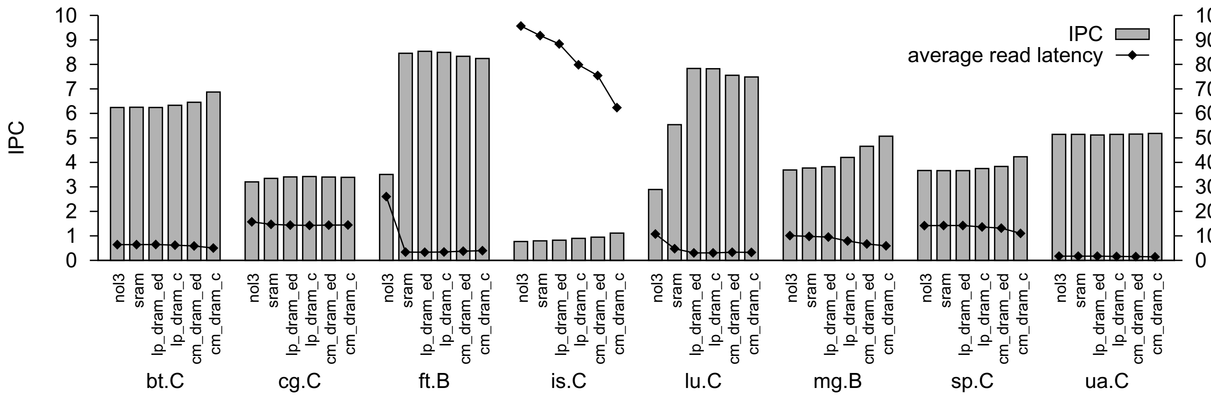
4.2 IPC

Figure 4(a) shows the instructions per cycle (IPC) and the average read latency results for our applications on 6 different cache configurations. Because threads are executed in order, correlation between the IPC and the average read latency is high. We can group the applications by their memory access and working set characteristics. First, on ft.B and lu.C the majority of working sets not fitting in the L2 cache can be located within the L3 cache, so configurations with L3 caches achieve much higher IPCs and lower average read latencies. On both applications (especially lu.C), the SRAM L3 cache is not big enough, so this causes higher miss rates compared to the LP-DRAM L3 caches leading to lower performance. The COMM-DRAM L3 caches don't gain a performance advantage over the LP-DRAM L3 caches for these applications since their L3 miss rates are low (less than 9%) and the latency and the bandwidth of the COMM-DRAM caches are worse than those of the LP-DRAM caches. Second, the majority of working sets not fitting in the L2 cache are bigger than the size of L3 caches in bt.C, is.C, mg.B, and sp.C. However since there is locality while accessing these datasets on these applications, bigger L3 caches decrease the number of main memory requests and improve the performance of the applications. The amount of performance improvement on different cache configurations is the function of 1) the frequency of the L3 accesses per instruction, and 2) the sensitivity of L3 misses over L3 capacity. The performance of the last set of applications, ua.C and cg.C, is not very sensitive to the size of the L3 caches. In ua.C, it is because the frequency of the L3 accesses are very low, so that there are few cache misses regardless of the cache configurations. In cg.C, working sets not fitting in the L2 cache do not experience locality, so all L3 caches fail to filter the memory requests to main memory.

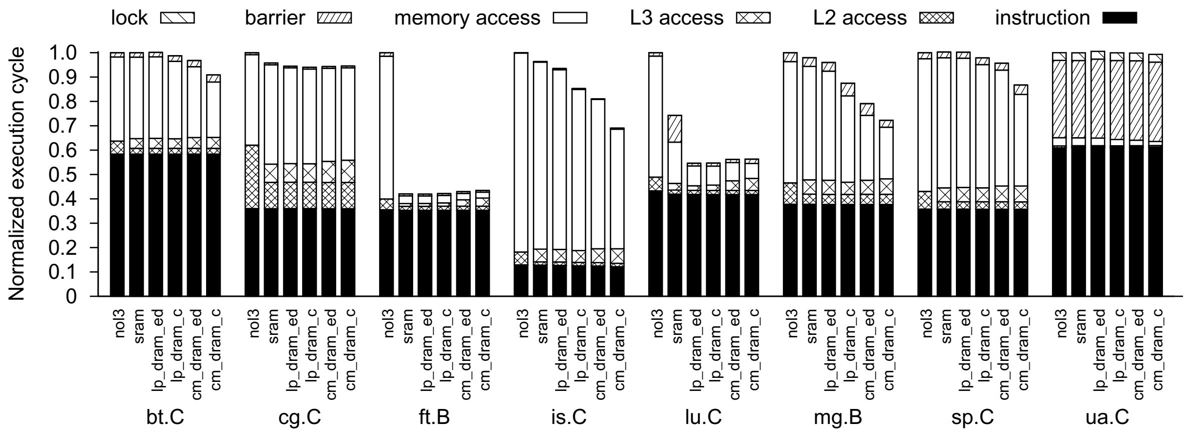
Figure 4(b) shows the normalized execution cycle breakdown of the applications. The execution cycle is broken down into six categories: threads are processing instructions not waiting for memory requests; threads are stalled and L2 caches are serving requests; threads are stalled and L3 caches are serving requests; threads are stalled and main memories are serving requests; threads are idle waiting for all the threads to reach a barrier; threads wait for other threads to release locks. From the Figure 4(b), it can be seen that memory access time occupies the majority of the execution cycles. In all the applications, the introduction of an L3 cache reduces the memory access time. However making the L3 cache larger either saves total execution cycles by reducing the memory access time further, or makes performance worse when the reduction in memory access time cannot justify the latency and bandwidth penalty of the larger L3.

4.3 Memory Hierarchy Power and System Energy-Delay

For each application Figure 5(a) shows the breakdown of power consumed in the memory hierarchy of the different system configurations. We assume a memory bus power of 2mW/Gb/s suitable for the 2013 time-frame. On average, over all applications, the memory hierarchy with no L3 consumes 6.6W, which accounts for 23% of system power. The addition of an LLC has the potential to reduce power by reducing the number of accesses to main memory, thereby saving on bus power and dynamic power consumed in the memory chips. However, the addition of an LLC can increase memory hierarchy power by consuming



(a) IPC and average read latency

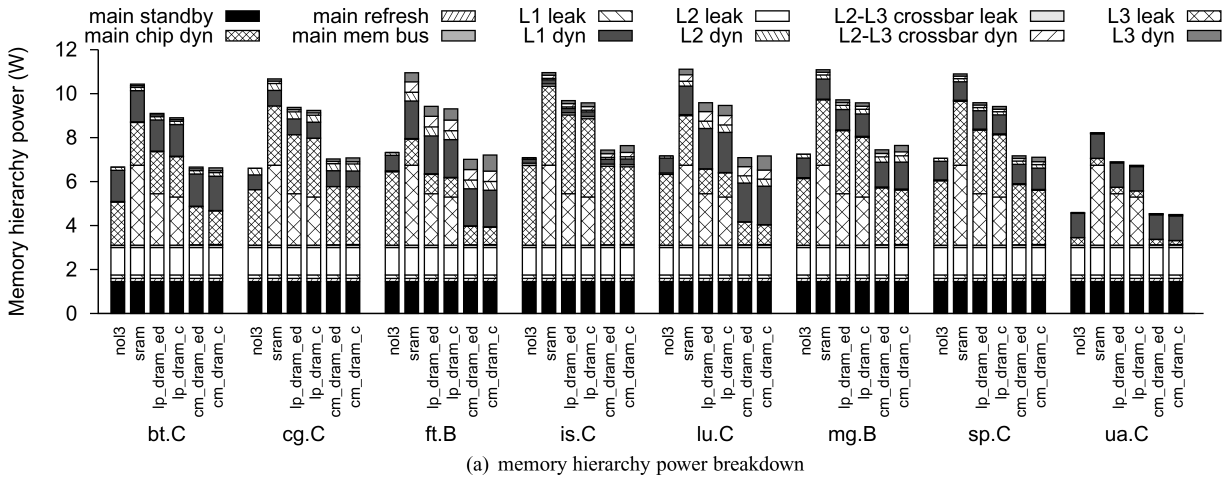


(b) execution cycle breakdown

Figure 4: (a) IPC and average read latency, and (b) normalized execution cycle breakdown of NPB applications on different cache configurations.

leakage and refresh power. For each application, the addition of the SRAM or LP-DRAM L3s results in an increase in the memory hierarchy power because the reduction in main memory and bus power is not enough to overcome the additional leakage of these logic process based caches. On average, the SRAM, LP-DRAM config ED, and LP-DRAM config C increase memory hierarchy power by 58%, 37%, and 35% respectively. The COMM-DRAM L3s perform best by increasing the memory hierarchy power by the least amount. For bt.C, ft.B, lu.C, and ua.C, the COMM-DRAM L3s are successful in reducing the memory hierarchy power slightly. Overall, on average, the 96MB config ED and the 192MB config C COMM-DRAM L3s increase memory hierarchy power by 1.2% and 2.3% respectively. Thus the memory hierarchy power of ED and C configurations are not that different. For either DRAM technology, the lower dynamic energy of config ED does not help much because L3 dynamic power is a relatively small percentage of total memory hierarchy power. Similarly the greater capacity and worse dynamic energy of config C do not hurt much in terms of power because both config C and config ED have comparable leakage power. When combined with the fact that the performance of the benchmarks is not too sensitive to L3 access time and random cycle time, these trends suggest that leakage power should be a primary consideration in the design of L3 caches.

In all system configurations including those with COMM-DRAM L3s, the main power drain in the memory hierarchy is the main memory chips. In the system with the 192MB COMM-DRAM L3, dynamic power in main memory DRAMs on average accounts for 26% of the memory hierarchy power while standby accounts for 22%. Our system architecture included only two DIMMs distributed over two channels. In systems with more memory capacity per channel, standby power in the memory chips can account for an even greater percentage and become a dominant component of system power.



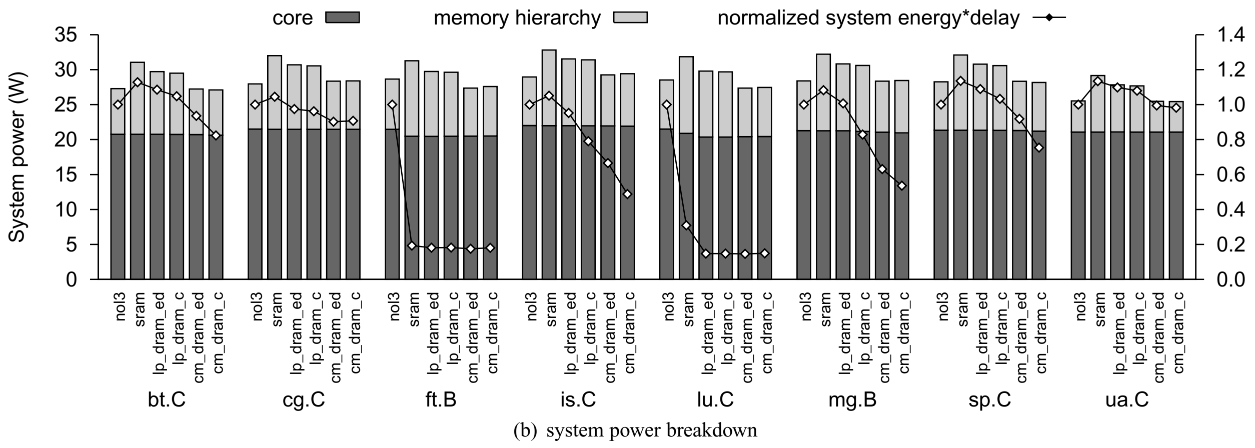


Figure 5: (a) Memory hierarchy power, (b) system power breakdown and system energy-delay product

For each application and for each system configuration, Figure 5(b) shows breakdown of system power into core power and memory hierarchy power, and system energy-delay product normalized with respect to the system with no L3. We compute core power by scaling the power of the 90nm Niagara (63W) to 32nm. We scale the power by assuming linear scaling for capacitance, increase in CPU clock from 1.2GHz to 2GHz, reduction in VDD from 1.2V to 0.9V, and 40% of power to be leakage power. We also adjust core power suitably to account for the fact that the processor die that we have considered has 8 4-way SIMD FPUs per core whereas the 90nm Niagara had just 1 FPU per chip. With these assumptions, we compute total core power of the bottom die to be 22.3W.

Because of the high contribution of standby and leakage power in both memory hierarchy power and core power, when the addition of an L3 results in reduction of execution time, it is also beneficial for system energy-delay. Because of its high leakage power, the SRAM L3 is able to improve system energy-delay for only 4 applications even though it improves the execution time of 7. The LP-DRAM L3s do better than the SRAM L3, but the COMM-DRAM L3s perform best. On average, the 96MB COMM-DRAM L3 improves system energy-delay by 33% while the 192MB improves it by 40%.

We also studied the temperature increase due to stacking of L3 die of different technologies by using HotSpot [33]. The maximum power density is with the stacked SRAM L3, but since we have modeled long-channel HP transistors and aggressive leakage control mechanisms, the maximum power consumed per bank is only about 450mW. The maximum observed temperature difference between the different technologies was less than 1.5K.

5 Related Work

Apart from CACTI [35,40], there are a number of SRAM and cache modeling efforts in the literature. The effort by Amrutur and Horowitz [4, 5] has influenced subsequent SRAM modeling efforts including CACTI-D. CACTI-D follows [5] in the use of the method of logical effort for sizing decoders. A major contribution of eCACTI [20] was the incorporation of a leakage power model into CACTI; CACTI 4 and CACTI 5 both borrow the leakage power modeling methodology of eCACTI. Rodriguez and Jacob [30] incorporated SRAM data for 90/65/45/32nm technologies based on predictive models, and modeled the impact of pipelining overhead in caches. Zeng, Rose, and Guttman developed PRACTICS [41] which also added a DRAM delay model whose access time was validated against a 250nm NEC embedded DRAM design. Liang, Turgay, and Brooks [19] made use of a mixture of analytical and empirical methods in order to improve the accuracy of SRAM power models. CACTI-D adds to these existing efforts by providing a new technology modeling foundation as well as support for DRAM technologies and operational models, making it possible to model the complete memory hierarchy with consistent models all the way from SRAM based L1 caches through main memory DRAMs on DIMMs. CACTI 6.0 [25] extended CACTI 5.0 by adding support for modeling of NUCA (Non Uniform Cache Access) architecture and interconnect alternatives. CACTI-D development proceeded concurrently with CACTI 6.0.

Black et al. [6] described a stacked last level cache study in which they carried out architectural evaluation of stacked COMM-DRAM and SRAM L2 caches and showed the architectural and thermal benefits of incorporating a stacked COMM-DRAM L2. Our last level cache (LLC) study is similar in concept to that carried out by Black et al., with several differences. First, we use CACTI-D to model LP-DRAM caches in addition to SRAM and COMM-DRAM caches. Second, with the CACTI-D framework we are able to study area, access time, bandwidth, and energy tradeoffs of a large set of organizations, capacities, and policies for the three technologies. Third, with CACTI-D we are able to provide a detailed energy breakdown of power consumed in the memory hierarchy.

Numerous other studies have also been carried out in the area of 3D die stacked architectures. 3DCacti [36] extended CACTI to consider different design possibilities of partitioning a memory array for multilayer 3D implementation. Puttaswamy and Loh [28] also carried out a study that looked at alternative partitioning strategies. Kgil et al. evaluated the performance of *PicoServer* [16], a 3D multicore architecture that assumed connection of slower, simpler, low power cores directly to main memory in a stacked die. Li et al. [18] studied memory networking and data management considerations for a 3D architecture. Jaleel, Mattina, and Jacob [13] carried out a last level cache study that showed the architectural benefits of having a shared last level cache for parallel bioinformatics workloads with a large degree of data sharing. In their study, they used miss rates as the evaluation metric and did not look at the impact of latency.

6 Conclusions

We have introduced CACTI-D, a comprehensive memory modeling tool that supports modeling of logic process based DRAM (LP-DRAM) and commodity DRAM (COMM-DRAM) technologies in addition to SRAM. With CACTI-D, uniform modeling of the complete memory hierarchy all the way from SRAM based L1 caches through main memory DRAMs on DIMMs is now possible.

We illustrated the utility of CACTI-D by carrying out a study of last level cache (LLC) tradeoffs involving SRAM, LP-DRAM, and COMM-DRAM L3 caches for a multicore multithreaded architecture. We used CACTI-D to model the area, access time, dynamic read and write energy per access, and standby and leakage power of all memory components of our study including L1, L2, and L3 caches, and main memory DRAMs. We carried out an architectural study to evaluate the different system configurations through simulation of benchmarks with large data set sizes. On average, execution time of applications was reduced by the addition of any of the considered L3s, but the 96MB and 192MB COMM-DRAM L3s achieved the best reduction of 39% and 43% respectively. The COMM-DRAM LLCs also performed best in terms of system power and system energy-delay product. On average, the 96MB improved system energy-delay product by 33% while the 192MB improved it by 40%. For each application, the logic process based SRAM and LP-DRAM based caches resulted in an increase in memory hierarchy power and system power because of their higher leakage. On average, the LP-DRAM L3s performed better than the SRAM L3 in all metrics. Finally, the high percentage of main memory system power we observed due to standby power suggests that appropriate use of DRAM power-down modes, combined with supporting operating system policies, may significantly reduce main memory power.

Acknowledgments

We would like to thank Arun Lokanathan of Micron Technology for helping improve our understanding of commodity DRAM, and Wolfgang Mueller of Qimonda for providing data associated with his 2005 IEDM paper. We thank Ayose Falcon, Paolo Faraboschi, and Daniel Ortega of HP Labs for their help with the simulation framework. We also thank Jacob Leverich of Stanford University and the reviewers of the paper for their constructive comments.

References

[1] Micron DDR3 SDRAM Products. http://www.micron.com/products/dram/ddr3/.

[2] Micron System Power Calculator. http://www.micron.com/support/part_info/powercalc.aspx.

[3] J. Amon, et al. A highly manufacturable deep trench based DRAM cell layout with a planar array device in a 70nm technology. In *IEDM*, 2004.

[4] B. S. Amrutur and M. A. Horowitz. Speed and Power Scaling of SRAM's. *JSSC*, 35(2), Feb 2000.

[5] ——. Fast-Low Power Decoders for RAMs. *JSSC*, 36(10), Oct 2001.

[6] B. Black, et al. Die Stacking (3D) Microarchitecture. In *MICRO 39*, Dec 2006.

[7] D. Brooks, V. Tiwari, and M. Martonosi. Wattch: A framework for architectural-level power analysis and optimizations. In *ISCA*, Jun 2000.

[8] J. Chang, et al. The 65-nm 16-MB Shared On-Die L3 Cache for the Dual-Core Intel Xeon Processor 7100 Series. *JSSC*, 42(4), Apr 2007.

[9] A. Falcon, P. Faraboschi, and D. Ortega. Combining Simulation and Virtualization through Dynamic Sampling. In *ISPASS*, Apr 2007.

[10] R. Ho. *On-chip Wires: Scaling and Efficiency*. Ph.D. thesis, Stanford University, 2003.

[11] S. S. Iyer, et al. Embedded DRAM: Technology platform for the Blue Gene/L chip. *IBM Journal of Research and Development*, 49(2/3), Mar/May 2005.

[12] J. J. Barth, et al. A 500-MHz Multi-Banked Compilable DRAM Macro With Direct Write and Programmable Pipelining. *JSSC*, 40(1), Jan 2005.

[13] A. Jaleel, M. Mattina, and B. Jacob. Last Level Cache (LLC) Performance of Data Mining Workloads On a CMP - A Case Study of Parallel Bioinformatics Workloads. In *HPCA*, Feb 2006.

[14] H. Jin, M. Frumkin, and J. Yan. The OpenMP Implementation of NAS Parallel Benchmarks and Its Performance. Technical Report NAS-99-011, NASA Ames Research Center, 1999.

[15] B. Keeth and R. Baker. *DRAM Circuit Design: A Tutorial*. IEEE Press, 2000.

[16] T. Kgil, et al. PicoServer: Using 3D Stacking Technology to Enable a Compact Energy Efficient Chip Multiprocessor. In *ASPLOS*, Oct 2006.

[17] P. Kongetira, K. Aingaran, and K. Olukotun. Niagara: A 32-Way Multithreaded Sparc Processor. *IEEE Micro*, 25(2), 2005.

[18] F. Li, C. Nicopoulos, T. Richardson, Y. Xie, V. Narayanan, and M. Kandemir. Design and Management of 3D Chip Multiprocessors Using Network-in-Memory. In *ISCA*, Jun 2006.

[19] X. Liang, K. Turgay, and D. Brooks. Architectural Power Models for SRAM and CAM Structures Based on Hybrid Analytical/Empirical Techniques. In *ICCAD*, Nov 2007.

[20] M. Mamidipaka and N. Dutt. eCACTI: An Enhanced Power Estimation Model for On-chip Caches. Technical Report TR-04-28, Center for Embedded Computer Systems, 2004.

[21] R. E. Matick and S. E. Schuster. Logic-based eDRAM: Origins and rationale for use. *IBM Journal of Research and Development*, 49(1), Jan 2005.

[22] H. McIntyre, et al. A 4-MB On-Chip L2 Cache for a 90-nm 1.6-GHz 64-bit Microprocessor. *JSSC*, 40(1), Jan 2005.

[23] W. Mueller, et al. Trench DRAM Technologies for the 50nm Node and Beyond. In *International Symposium on VLSI Technology, Systems, and Applications*, Apr 2006.

[24] ——. Challenges for the DRAM Cell Scaling to 40nm. In *IEDM*, Dec 2005.

[25] N. Muralimanohar, R. Balasubramonian, and N. P. Jouppi. Optimizing NUCA Organizations and Wiring Alternatives for Large Caches with CACTI 6.0. In *MICRO*, Dec 2007.

[26] K. Noh, et al. A 130nm 1.1V 143MHz SRAM-like Embedded DRAM COMPILER with Dual Asymmetric Bit Line Sensing Scheme and Quiet Unselected IO scheme. In *Symposium on VLSI Circuits*, Jun 2004.

[27] M. Oka and M. Suzuoki. Designing and Programming the Emotion Engine. *IEEE Micro*, 19(6), Nov/Dec 1999.

[28] K. Puttaswamy and G. H. Loh. Implementing Caches in a 3D Technology for High Performance Processors. In *ICCD*, Oct 2005.

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

[30] S. Rodriguez and B. Jacob. Energy/Power Breakdown of Pipelined Nanometer Caches (90nm/65nm/45nm/32nm). In *ISPLED*, Oct 2006.

[31] Ron Ho. Tutorial: Dealing with issues in VLSI interconnect scaling. In *ISSCC*, Feb 2007.

[32] Semiconductor Industries Association. International Technology Roadmap for Semiconductors. http://www.itrs.net/, 2006 Update.

[33] K. Skadron, M. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan. Temperature-Aware Microarchitecture. In *ISCA*, Jun 2003.

[34] S. Thoziyoor, N. Muralimanohar, J. Ahn, and N. P. Jouppi. CACTI 5.1. Technical Report HPL-2008-20, HP Labs.

[35] S. Thoziyoor, N. Muralimanohar, and N. P. Jouppi. CACTI 5.0. Technical Report HPL-2007-167, HP Labs.

[36] Y.-F. Tsai, Y. Xie, V. Narayanan, and M. J. Irwin. Three-Dimensional Cache Design Exploration Using 3DCacti. In *ICCD*, Oct 2005.

[37] R. Varada, M. Sriram, K. Chou, and J. Guzzo. Design and Integration Methods for a Multi-threaded Dual Core 65nm Xeon® Processor. In *ICCAD*, Nov 2006.

[38] G. Wang, et al. A $0.168\mu m^2/0.11\mu m^2$ Highly Scalable High Performance embedded DRAM Cell for 90/65-nm Logic Applications. In *Symposium on VLSI Circuits*, Apr 2005.

[39] H. Wang, X. Zhu, L.-S. Peh, and S. Malik. Orion: A Power-Performance Simulator for Interconnection Networks. In *MICRO*, Nov 2002.

[40] S. Wilton and N. P. Jouppi. An Enhanced Access and Cycle Time Model for On-Chip Caches. Technical Report 93/5, DEC WRL, 1994.

[41] A. Zeng, K. Rose, and R. J. Gutmann. Memory Performance Prediction for High-Performance Microprocessors at Deep Submicrometer Technologies. *TCAD*, 25(9), Sep 2006.