PicoServer: Using 3D Stacking Technology To Build Energy Efficient Servers

16

TAEHO KGIL
University of Michigan, Intel
ALI SAIDI
University of Michigan
NATHAN BINKERT
HP Labs
STEVE REINHARDT
University of Michigan, AMD
KRISZTIAN FLAUTNER
ARM
and
TREVOR MUDGE
University of Michigan

This article extends our prior work to show that a straightforward use of 3D stacking technology enables the design of compact energy-efficient servers. Our proposed architecture, called PicoServer, employs 3D technology to bond one die containing several simple, slow processing cores to multiple memory dies sufficient for a primary memory. The multiple memory dies are composed of DRAM. This use of 3D stacks readily facilitates wide low-latency buses between processors and memory. These remove the need for an L2 cache allowing its area to be re-allocated to additional simple cores. The additional cores allow the clock frequency to be lowered without impairing throughput. Lower clock frequency means that thermal constraints, a concern with 3D stacking, are easily satisfied. We extend our original analysis on PicoServer to include: (1) a wider set of server workloads, (2) the impact of multithreading, and (3) the on-chip DRAM architecture and system memory usage. PicoServer is intentionally simple, requiring only the simplest form of 3D technology where die are

This work is supported in part by the National Science Foundation, Intel and ARM Ltd.
Authors' addresses: T. Kgil (corresponding author), A. Saidi, Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109; email: taeho.kgil@intel.com; N. Binkert, HP Labs, 3000 Hanover St., Palo Alto, CA 94304-1185; S. Reinhardt, Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109; K. Flautner, ARM, Ltd., 110 Fulbourn Rd., Cambridge, UK CB1 9NJ; T. Mudge, Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109.
 DOI 10.1145/1412587.1412589 http://doi.acm.org/10.1145/1412587.1412589

stacked on top of one another. Our intent is to minimize risk of introducing a new technology (3D) to implement a class of low-cost, low-power compact server architectures.

Categories and Subject Descriptors: C.1.4 [**Processor Architectures**]: Parallel Architectures; C.5.5 [**Computer System Implementation**]: Servers

General Terms: Performance, Design, Experimentation

Additional Key Words and Phrases: Low power, Tier-1/2/3 server, 3D stacking technology, chip multiprocessor, full-system simulation

ACM Reference Format:

Kgil, T., Saidi, A., Binkert, N., Reinhardt, S., Flautner, K., and Mudge, T. 2008. PicoServer: Using 3D stacking technology to build energy efficient servers. ACM J. Emerg. Technol. Comput. Syst. 4, 4, Article 16 (October 2008), 34 pages. DOI = 10.1145/1412587.1412589 http://doi.acm.org/10.1145/1412587.1412589

1. INTRODUCTION

3D stacking technology enables new chip multiprocessor (CMP) architectures that significantly improve energy efficiency. Our proposed architecture, PicoServer, employs 3D technology to bond one die containing several simple, slow processor cores to multiple DRAM dies that form the primary memory. In addition, 3D stacking enables a memory processor interconnect that is both very high bandwidth and low latency. As a result, the need for complex cache hierarchies is reduced. We show that the die area normally spent on an L2 cache is better spent on additional processor cores. Having additional cores means that they can be run slower without affecting throughput. Slower cores also allow us to reduce power dissipation and with it thermal constraints, a potential roadblock to 3D stacking. The resulting system is ideally suited to throughput applications such as servers. Our proposed architecture is intentionally simple and requires only the simplest form of 3D technology where die are stacked on top of one another. Our intent is to minimize the risk of realizing a class of low-cost, low-power compact server architectures.

Internet service providers like AOL, Yahoo, and Google require large numbers of Web servers to satisfy customer needs. Server farms based on off-the-shelf general-purpose processors are unnecessarily power hungry, require expensive cooling systems, and occupy a large space. It has been shown that 25% of the operating costs for these "server farms" can be directly or indirectly attributed to power consumption [Mudge 2001]. This figure has the potential to grow rapidly along with the continuing growth in Web services. Employing PicoServers can significantly lower power consumption and space requirements.

Server applications handle events on a per-client basis, which are independent and display high levels of thread-level parallelism. This high level of parallelism makes them ill suited for traditional monolithic processors. CMPs built from multiple simple cores can take advantage of this thread-level parallelism to run at a much lower frequency while maintaining a similar level of throughput and thus dissipating less power. By combining them with 3D stacking we will show that it is possible to cut power requirements further. 3D stacking enables the following key improvements.

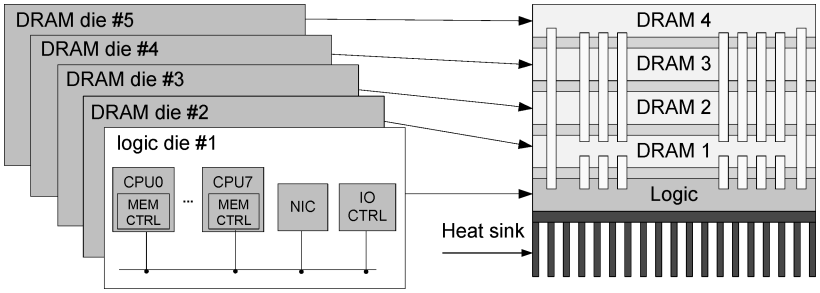


Fig. 1. A diagram depicting the PicoServer: A CMP architecture connected to DRAM using 3D stacking technology with an on-chip network interface controller (NIC) to provide low-latency, high-bandwidth networking.

—*High-Bandwidth Buses between DRAM and L1 Caches that Support Multiple Cores: Thousands of Low-Latency Connections with Marginal Area Overhead between Dies are Possible.* Since the interconnect buses are on chip, we are able to implement wide buses with a relatively lower power budget compared to interchip implementations.

—*Modification in the Memory Hierarchy due to the Integration of Large-Capacity On-Chip DRAM.* It is possible to remove the L2 cache and replace it with more processing cores. The access latency for the on-chip DRAM [1] is also reduced because address multiplexing and off-chip I/O pad drivers [Matick and Schuster 2005] are not required.

—*Overall Reduction in System Power, Primarily due to the Reduction in Core Clock Frequency.* The benefits of 3D stacking stated in items 1 and 2 allow us to integrate more cores clocked at a modest frequency (in our work 500–1000MHz) on chip while providing high throughput. Reduced core clock frequency allows their architecture to be simplified; for example, by using shorter pipelines with reduced forwarding logic.

The potential drawback of 3D stacking, now that the technology has been shown feasible, is thermal containment. However, this is not a limitation for the type of simple, low-power cores that we are proposing for the PicoServer, as we show in Section 4.5. In fact, the ITRS projections of Table II predict that systems consuming just a few watts do not even require a heat sink.

The general architecture of a PicoServer is shown in Figure 1. For the purposes of this study we assume a stack of 5∼9 dies. The connections are by vias that run perpendicular to the dies. The dimensions for a 3D interconnect via varies from $1\sim3\mu m$ with a separation of $1\sim6\mu m$. Current commercial offerings can support 1,000,000 vias per cm^2 [Gupta et al. 2004]. This is far more than we need for PicoServer. These function as interconnect and thermal pipes. For our studies, we assume that the logic-based components (i.e., the microprocessor cores, the network interface controllers (NICs), and peripherals) are on the bottom layer and conventional capacity-oriented DRAMs occupy the

[1]We will refer to die that are stacked on the main processor die as "on-chip" because they form a 3D chip.

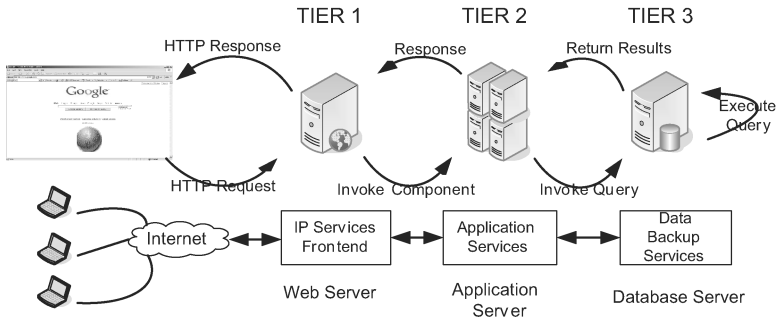


Fig. 2. A typical 3-tier server architecture: tier 1, Web server; tier 2, application server; tier 3, database server.

remaining layers. To understand the design space and potential benefits of this new technology, we explored the trade-offs of different bus widths, numbers of cores, frequencies, and memory hierarchies in our simulations. We found bus widths of 1024 bits with a latency of 2 clock cycles at 250MHz to be reasonable in our architecture. In addition, we aim for a reasonable area budget, constraining the die size area to be below 80mm^2 at 90nm process technology. Our 12-core PicoServer configuration, which occupies the largest die area, is conservatively estimated at approximately 80mm^2. The die areas for our 4- and 8-core PicoServer configurations are, respectively, 40mm^2 and 60mm^2.

We also extend our analysis on PicoServer. Specifically, our additional analysis examines three points.

—*Analysis on Additional Server Workloads.* We expand our scope of server workloads to all tiers in a server farm. We show performance and energy efficiency for tier-1, -2, and -3 workloads in the server space (see Figure 2).

—*Impact of Multithreading.* We show the impact of multithreading on servers leveraging 3D technology. Overall, multithreading improves throughput, but only to a limited extent when considering the area efficiency. This is because 3D technology improves the overall latency to memory, which reduces the benefits of multithreading.

—*Detailed Analysis on System Memory Usage and On-Chip DRAM Architecture.* We provide a detailed breakdown of system memory usage. Based on the usage behavior, we describe the role of on-chip DRAM. We also describe how the on-chip DRAM architecture changes using 3D technology. 3D technology enables us to implement heavily banked system memory architectures that consume less power.

The article is organized as follows. In the next section we provide background for this work by describing an overview of server platforms, 3D stacking technology, and trends in DRAM technology. In Section 3, we outline our methodology for the design-space exploration. In Section 4, we provide more details for the PicoServer architecture and evaluate various PicoServer configurations. In Section 5, we present our results in the PicoServer architecture for server benchmarks and compare our results to conventional architectures that do not

employ 3D stacking. These architectures are CMPs without 3D stacking and conventional high-performance desktop architectures with Pentium 4-like characteristics. A summary and concluding remarks are given in Section 6. Much of the findings we present in this article can also be found in Kgil [2007].

2. BACKGROUND

This section discusses the current state of server platforms, 3D stacking technology, and DRAM technology. We first show how servers are currently deployed in datacenters and analyze the behavior of current server workloads. Next, we explain the state of 3D stacking technology and how it is applied in this article. Finally, we show advances in DRAM technology. We explain current and future trends in DRAM used in the server space.

2.1 Server Platforms

2.1.1 3-*Tier Server Architecture.* Today's datacenters are commonly built following a 3-tier server architecture. Figure 2 shows a 3-tier server farm and how it might handle a request for service. The first tier handles the bulk of the requests from the client. Tier-1 server applications handle events on a per-client basis, which are independent and display high levels of thread-level parallelism. Tier-1 servers handle Web requests and forward requests that require heavier computation or database accesses to tier 2. Tier-2 servers execute user applications that interpret script languages and determine what objects (typically database objects) should be accessed. Tier-2 servers generate database requests to tier-3 servers. Tier-3 servers receive database queries and return the results to tier-2 servers.

For example, when a client request comes in for a Java servlet page, it is first received by the front-end server: tier 1. Tier 1 recognizes a Java servlet page that must be handled and initiates a request to tier 2, typically using remote message interfaces (RMIs). Tier 2 initiates a database query on the tier-3 servers, which in turn generate the results and send the relevant information up the chain, all the way to tier 1. Finally, tier 1 sends the generated content to the client.

Three tier server architectures are commonly deployed in today's server farms, because this allows each level to be optimized for its workload. However, this strategy is not always adopted. Google employs essentially the same machines at each level, because economies of scale and manageability issues can outweigh the advantages. We will show that, apart from the database disk system in the third tier, the generic PicoServer architecture is suitable for all tiers.

2.1.2 *Server Workload Characteristics.* This section describes the individual workload behavior of applications commonly found in server farms. Server workloads display a high degree of thread-level parallelism (TLP), since connection-level parallelism through client connections can be easily translated into thread-level parallelism (TLP). Table I shows the behavior of commercial server workloads. Most of the commercial workloads display high TLP and

Table I. Behavior of Commercial Workloads (adapted from Kunkel et al. [2000])

Attribute	Web99	JBOB (JBB)	TPC-C	SAP 2T	SAP3T DB	TPC-H
Application Category	Web Server	Server Java	OLTP *	ERP†	ERP†	DSS‡
Instruction Level Parallelism	low	low	low	med	low	high
Thread Level Parallelism	high	high	high	high	high	high
Instruction/Data working-set	large	large	large	med	large	large
Data Sharing	low	med	high	med	high	med
I/O Bandwidth	high (network)	low	high (disk)	med (disk)	high (disk)	med (disk)

*OLTP : Online Transaction Processing.
†ERP : Enterprise Resource Planning.
‡DSS : Decision Support System.

low instruction-level parallelism (ILP), with the exception of decision support systems. Conventional general-purpose processors, however, are typically optimized to exploit ILP. These workloads suffer a high cache-miss rate, regularly stalling the machine. This leads to low instructions per cycle (IPC) and poor utilization of processor resources. Our studies have shown that, except for computation-intensive workloads like PHP application servers, video streaming servers, and decision support systems, out-of-order processors have an IPC between $0.21 \sim 0.54$ for typical server workloads (i.e., at best, modest computation loads with an L2 cache of 2MB). These workloads do not perform well because much of the requested data has been recently DMAed from the disk to system memory, invalidating cached data and leading to cache misses. Therefore, we can generally say that single-thread optimized out-of-order processors do not perform well on server workloads. Another interesting property of most server workloads is the significant amount of time spent in kernel code, unlike SPECCPU benchmarks. This kernel code is largely involved in interrupt handling for the NIC or disk driver, packet transmission, network stack processing, and disk cache processing.

Finally, a large portion of requests are centered around the same group of files. These file accesses translate into memory and I/O accesses. Due to the modest computation, memory and I/O latency are critical to high performance. Therefore, disk caching in the system memory plays a critical part in providing sufficient throughput. Without a disk cache, the performance degradation due to the hard disk drive latency would be unacceptable.

To perform well on these classes of workloads, an architecture should naturallly support multiple threads to respond to independent requests from clients. Thus, intuition suggests that a CMP or SMT architecture should be able to better utilize the processor die area.

2.1.3 *Conventional Server Power Breakdown.* Figure 3 shows the power breakdown of a server platform available today. This server uses a chip multiprocessor implemented with many simple in-order cores to reduce power consumption. The power breakdown shows that one-fourth is consumed by the processor, one-fourth is consumed by the system memory, one-fourth is

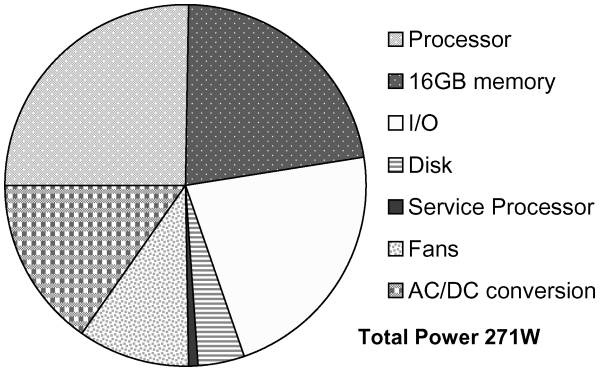


Fig. 3. Power breakdown of T2000 UltraSPARC executing SpecJBB.

consumed by the power supply, and one-fifth is consumed by the I/O interface. Immediately we can see that using a relatively large amount of system memory results in the consumption of a substantial fraction of power. This is expected to increase as the system memory clock frequency increases and system memory capacity increases. We also find that despite using simpler cores that are energy efficient, a processor would still consume a noticeable amount of power. The I/O interface consumes a large amount of power due to the high I/O supply voltage required in off-chip interfaces. The I/O supply voltage is likely to reduce as we scale in the future, but won't scale as much as the core supply voltage. This suggests that system-level integration could further reduce power. Finally, we find that the power supply displays some inefficiency. This is due to the multiple levels of voltage it has to support. In summary, 3D stacking technology has the potential to reduce the power consumed by the processor and the I/O interfaces by pushing system integration to the next level.

2.2 3D Stacking Technology

This section provides an overview of 3D stacking technology. In the past there have been numerous efforts in academia and industry to implement 3D stacking technology [Black et al. 2004; Lee et al. 2000; Koyanagi 2005; Lu 2005; Xue et al. 2003]. They have met with mixed success. This is due to the many challenges that need to be addressed, including (1) achieving high yield in bonding die stacks, (2) delivering power to each stack, and (3) managing thermal hotspots due to stacking multiple dies. However, in the past few years strong market forces in the mobile terminal space have accelerated a demand for small form factors with very low power. In response, several commercial enterprizes have begun offering reliable low-cost die-to-die 3D stacking technologies.

In 3D stacking technology, dies are typically bonded as face to face or face to back. Face-to-face bonds provide higher die-to-die via density and lower area overhead than face-to-back bonds. The lower via density for face-to-back bonds results from the through-silicon vias (TSVs) that have to go through silicon bulk. Figure 4 shows a high-level example of how dies can be bonded using 3D stacking technology. The bond between layers 1 (starting from the bottom)

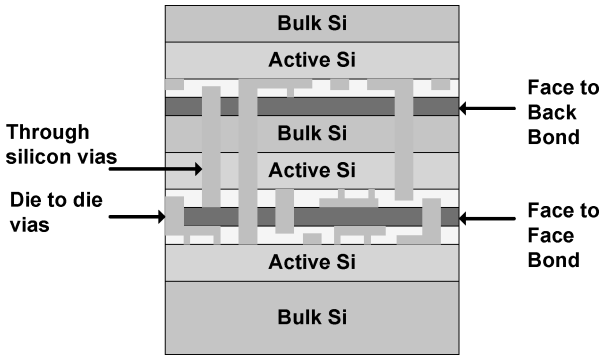


Fig. 4. Example of a 3 layer 3D IC.

and 2 is face to face, while the bond between layers 2 and 3 is face to back. Using the bonding techniques in 3D stacking technology opens up the opportunity of stacking heterogeneous dies together, for example, architectures that stack DRAM and logic manufactured from different process steps. Loi et al. [2006], Ghosh and Lee [2007], and Black et al. [2006] demonstrate the benefits of stacking DRAM on logic. Furthermore, with the added third dimension from the vertical axis, the overall wire-interconnect length can be reduced and wider bus width can be achieved at lower area costs. The parasitic capacitance and resistance for 3D vias are negligible compared to global interconnect. We also note that the size and pitch of 3D vias only adds a modest area overhead. 3D via pitches are equivalent to 22λ for $90nm$ technology, which is about the size of a 6T SRAM cell. They are also expected to shrink as this technology becomes mature.

The ITRS roadmap in Table II predicts deeper stacks being practical in the near future (3D stacking technology parameters are given in Table III). The connections are by vias that run perpendicular to the dies. As noted earlier, the dimensions for a 3D interconnect via vary from $1\sim3\mu m$ with a separation of $1\sim6\mu m$. Current commercial offerings can support 1,000,000 vias per cm^2 [Gupta et al. 2004]. Overall yield using 3D stacking technology is a product of the yield of each individual die layer. Therefore, it is important that the individual die are designed with high yield in mind. Memory stacking is a better choice than logic-to-logic stacking. Memory devices typically show higher yield because fault tolerance fits well with their repetitive structure. For example, re-fusing extra bitlines to compensate for defective cells and applying single-bit error correction logic to memory boosts yield. Several studies, including Ohsawa et al. [2006], show that DRAM yields are extremely high, suggesting that chips built with a single logic layer and several DRAM layers generate yields close to the logic die.

2.3 DRAM Technology

This section discusses the advances in DRAM technology in the server space. DRAM today is offered in numerous forms, usually determined by the application space. In particular, for server platforms, DDR2/DDR3 DRAM has emerged

Table II. ITRS Projection [ITRS 2005] for 3D Stacking Technology, Memory Array Cells and Maximum Power Budget for Power-Aware Platforms

	2007	2009	2011	2013	2015
Low-cost/handheld #die/stack	7	9	11	13	14
SRAM density Mbits/cm^2	138	225	365	589	948
DRAM density Mbits/cm^2 at production	1,940	3,660	5,820	9,230	14,650
Max. Power Budget for cost-performance systems(W)	104	116	119	137	137
Max. Power Budget for low-cost/handheld systems with battery(W)	3.0	3.0	3.0	3.0	3.0

ITRS projections suggest DRAM density exceeds SRAM density by 15∼18×, entailing large capacity of DRAM can be integrated on chip using 3D stacking technology as compared to SRAM.

Table III. 3D Stacking Technology Parameters [Gupta et al. 2004; Banerjee et al. 2001; Lu 2005]

	Face-to-Back	Face-to-Face	RPI	MIT 3D FPGA
Size	$1.2\mu \times 1.2\mu$	$1.7\mu \times 1.7\mu$	$2\mu \times 2\mu$	$1\mu \times 1\mu$
Minimum Pitch	$<4\mu$	2.4μ	N/A	N/A
Feed Through Capacitance	2∼3fF	≈0	N/A	2.7fF
Series Resistance	$<0.35\Omega$	≈0	≈0	≈0

as the primary solution for system memory. FBDIMM DRAM that delivers higher throughput than DDR2/DDR3 is emerging as an alternative, but higher power, solution. RLDRAM and NetRAM [NetRAM 2005; RLDRAM 2008] are also popular DRAM choices for network workloads in the server space. The common properties for these memories are high throughput and low latency. In the server space, DRAM must meet the high-throughput and low-latency demands to deliver high performance. These demands can only be achieved at the price of increasing power consumption in the DRAM I/O interface and the DRAM arrays. As a result, power has increased to a point where the I/O power and DRAM power contribute to a significant amount of overall system power (as we showed in Section 2.1.3). Industry has addressed this concern by reducing the I/O supply voltage and introducing low-power versions of the DDR2 interface at the price of sacrificing both throughput and latency. We will show that DRAM stacked using 3D stacking technology can be implemented to deliver high-throughput and low-latency DRAM interfaces while consuming much less power.

3. METHODOLOGY

To explore the design space for 3D stacking technology, we modeled the benefits gained using 3D technology on a full-system simulator. The architectural aspects of our studies were obtained from a microarchitectural simulator called M5 [Binkert et al. 2006] that is able to run Linux and evaluate full system-level performance. We model multiple servers connected to multiple clients in M5. The client requests are generated from user-level network application programs. We measure server throughput (network bandwidth or transactions per second) to estimate performance. We also estimated die area size based on previous publications and developed models for delay and power. They were derived

from data published by industry and academia [ITRS 2005; Gupta et al. 2004; Rahman and Reif 2000; 3DRISC 2004; Banerjee et al. 2001; Black et al. 2004]. DRAM timing- and power values were obtained from IBM and Micron technology datasheets [MicronDRAM 2008]. A detailed description of our methodology is described in the following subsections.

3.1 Simulation Studies

3.1.1 *Full-System Architectural Simulator.* To evaluate the performance of our server we used the M5 full-system simulator. M5 boots an unmodified Linux kernel on a configurable architecture. Multiple systems are created in the simulator to model the clients and servers, and connected via an ethernet link model. The server side executes Apache (a Web server), Fenice (a video streaming server), MySQL (a database server), and NFS (a file server). The client side executes benchmarks that generate representative requests for dynamic and static Web page content, video stream requests, database queries, and network file commands, respectively. For comparison purposes we defined a Pentium 4-like system [Schutz and Webb 2004], and a chip multiprocessor-like system similar to Kongetira et al. [2005]. We also looked at several configurations using 3D stacking technology on these platforms. We assume that with 3D stacking technology, wider bus widths can be implemented with lower power overhead. Table IV shows the configurations used in our simulations.

3.1.2 *Server Benchmarks.* We use several benchmarks that directly interact with client requests. We used two Web-content-handling benchmarks, SURGE [Barford and Crovella 1998] and SPECweb99 [SPECWeb 1999] to measure Web server performance. Both benchmarks request filesets of more than 1GB. A Web-script-handling benchmark, SPECweb2005 [SPECWeb 2005], using PHP is selected to represent script workloads. A video streaming benchmark, Fenice [LS3 2007], which uses the RTSP protocol along with the UDP protocol, is chosen to measure behavior for on-demand workloads. For a file-sharing benchmark we use an NFS server and stress it with dbench. Finally, we execute two database benchmarks to measure database performance for tier-2 and -3 workloads.

SURGE. The SURGE benchmark represents client requests for static Web content. We modified the SURGE fileset and used a zipf distribution to generate reasonable client requests. Based on the zipf distribution, a static Web page which is approximately 12KB in file size is requested 50% of the time in our client requests. We configured the SURGE client to have 20 outstanding client requests.

SPECweb99. To evaluate a mixture of static Web content and simple dynamic Web content, we used a modified version of SURGE to request SPECweb99 filesets (behavior illustrated in Table I). We used the default configuration for SPECweb99 to generate client requests. Specifically, 70% of client requests are for static Web content and 30% are for dynamic Web content.

SPECweb2005. Scripting languages are a popular way to describe Web pages. SPECweb2005 offers three types of benchmarks: a banking benchmark that

Table IV. Commonly Used Simulation Configurations

	OO4-small baseline / w. 3D stacking	OO4-large baseline / w. 3D stacking	Conventional CMP MP4/8 w.o. 3D stacking	PicoMP4/8/12-500MHz/ 1000MHz*
Operating Frequency	4GHz	4GHz	1GHz	500MHz/1GHz
Number of Processors	1	1	4/8	4/8/12
Processor Type	out-of-order	out-of-order	in-order	in-order
Issue Width	4	4	1	1
L1 cache	2 way 16KB	2 way 128KB	4 way 16KB per core	4 way 16KB per core
L2 cache	8 way 256KB 7.5ns unloaded latency	8 way 2MB 7.5ns unloaded latency	8 way 2MB 16ns unloaded latency	N/A
Memory bus width	64bit@400MHz/ 1024bit@250MHz	64bit@400MHz/ 1024bit@250MHz	64bit@250MHz	1024bit@250MHz
System Memory	512MB DDR2 DRAM	512MB DDR2 DRAM	512MB DDR2 DRAM	128MB~512MB DDR2 DRAM

*PicoServer platform using 3D stacking technology. The core clock frequency of PicoServer is typically 500MHz. PicoServer configurations with 1GHz core clock frequency are later used to show the impact of 3D stacking technology.

System memory latencies are generated from DDR2 DRAM models. We assume cores clocked at lower clock frequencies (below 1GHz) have higher L1 cache associativity. L2 cache unloaded latency for single-core and multicore configurations differ due to longer global interconnect lengths in multicore platforms [Laudon 2005].

emulates the online banking activity of a user; an e-commerce benchmark that emulates the online purchase activity; and a support benchmark that emulates the online stream activity. All benchmarks require a dynamic Web page to be generated from a script interpreter. We use a PHP interpreter to measure the behavior of tier-2 servers. The client requests are generated from methods described for SPECweb99 and SURGE clients.

Fenice. On-demand video serving is also an important workload for tier-1 servers. For copyright protection and live broadcasts, the RTSP protocol is commonly used for real-time video playback. Fenice is an open-source streaming project [LS3 2007] that provides workloads supporting the RTSP protocol. We modified it to support multithreading. Client requests were generated with a modified version of nemesi, an RTSP-supporting MPEG player. Nemesi is also from the open-source streaming project. We generated multiple client requests that fully utilized the server CPUs for a high-quality 16Mbps datastream of 720×480-resolution MPEG2 frames.

dbench. This benchmark is commonly used to stress NFS daemons. In our tests we used the in-kernel NFS daemon which is multithreaded and available in standard Linux kernels. We generated NFS traffic using dbench on the client side that stressed the file server. Dbench generates workloads that both read and write to the file server while locking these files so that a different client cannot access it simultaneously.

OLTP. Online transaction processing is a typical workload executed on tier-2 and -3 servers (behavior illustrated in Table I). The TPC council has described in detail benchmarks for OLTP. We used a modified version of TPC-C made available by the Open Source Development Lab (OSDL), called DBT2 [OSDL 2006]. DBT2 generates transaction orders. Our database server is MySQL 5.0. We use the InnoDB storage engine that supports transactions and provides a reasonable amount of scalability for multicores. We generated a 1GB warehouse which is typically used for small-scale computation-intensive databases. We chose a small working-set size due to limitation in simulation time. We selected a buffer pool size accordingly.

DSS. Decision Support System is another typical workload used to evaluate tier-2 and -3 servers. We used TPC-H, the current version of a DSS workload. Again, a modified version of TPC-H available by OSDL (DBT3) [OSDL 2006] is used in this study. We loaded the TPC-H database onto MySQL and used the defined TPC-H queries to measure performance. The query cache is disabled to prevent speedup in query time due to caching. To reasonably reduce our simulation time, we only performed and measured the time for a Q22 query out of the many TPC-H queries. A Q22 query takes a modest amount of time to execute and displays the behaviors illustrated in Table I.

3.2 Estimating Power and Area

Power and area estimation at the architectural level is difficult to do with great accuracy. To make a reasonable estimation and show general trends, we

Table V. Published Power Consumption Values for Various Microprocessors [Clark et al. 2001; ARM11MPcore 2004; Ricci et al. 2005; Schutz and Webb 2004]

	Pentium 4 90nm	ARM11 130nm	Xscale 90nm*	PicoServer MP 90nm†
L1 cache	16KB	16KB	32KB	16KB
L2 cache	1MB	N/A	N/A	N/A
Total Power(W)	89–103W	250mW @ 550MHz	850mW @ 1.5GHz	190mW @ 500MHz
Total Die Area(mm^2)	112	5–6	6–7	4–5

*Die area for a 90nm Xscale excludes L2 cache [Ricci et al. 2005].
†For the PicoServer core, we estimated our power to be in the range of an ARM11, Xscale.

resorted to industry white papers, datasheets, and academia publications on die area, and we compared our initial analytical power models with real implementations and widely used cycle-level simulation techniques. We discuss this further in the next subsections.

3.2.1 *Processors.* We relied to a large extent on the figures reported in Clark et al. [2001], ARM11MPcore [2004], and Ricci et al. [2005] for an ARM processor to estimate processor power and die area. The ARM is representative of a simple in-order 32-bit processor that would be suitable for the PicoServer. Due to the architectural similarities with our PicoServer cores, we extrapolated the die area and power consumption for our PicoServer cores at 500MHz from published data in Clark et al. [2001], ARM11MPcore [2004], and Ricci et al. [2005]. Table V lists these estimates, along with values listed in ARM11MPcore [2004], and Ricci et al. [2005] and a Pentium 4 core for comparison. A die area analysis on the expected die area per core was also conducted. We collected several die area numbers available from ARM, MIPS, PowerPC, and other comparable scalar in-order processors. We also synthesized several 32-bit open-source cores that are computationally comparable to a single PicoServer core. We synthesized them using the Synopsys Physical compiler toolset.

The power values listed in Table V include static power. Our estimates for a 500MHz PicoServer core are conservative compared to the ARM core values, especially with respect to Ricci et al. [2005]. Given that the Xscale core consumes 850mW at 1.5GHz and 1.3V, a power consumption of 190mW at 500MHz for the 90nm PicoServer core is conservative when applying the 3× scaling in clock frequency and the additional opportunities to scale voltage. For power consumption at other core clock frequencies, for example, 1GHz, we generated a power-versus-frequency plot. It follows a cubic law [Flynn and Hung 2004]. We assumed a logic depth of 24 FO4 (fan out of 4) logic gates and used the 90nm PTM process technology [Ricci et al. 2005].

Support for 64 bits in a PicoServer core seems inevitable in the future. We expect the additional area and power overhead for 64-bit support in a PicoServer core to be modest when we look at the additional area and power overhead for 64-bit support in commercially available cores like MIPS and Xeon. As for the L2 cache, we referred to Wendell et al. [2004] and scaled the area and power numbers generated from actual measurements. We assumed the power numbers in Wendell et al. [2004] were generated when the cache-access rate was 100%. Therefore, we scaled the L2 cache power by size and access

Table VI. Parasitic Interconnect Capacitance for On-Chip 2D, 3D and Off-Chip 2D for a 1024-Bit Bus

	130nm	90nm
on-chip 2D 12mm	5.6nF	5.4nF
on-chip 3D 8mm	3.7nF	3.6nF
off-chip 2D	16.6nF	16.6nF

rate while assuming leakage power would consume 30% of the total L2 cache power.

3.2.2 *Interconnect Considering* 3D *Stacking Technology*. For the purposes of this study, we have adopted the data published in ITRS [2005], Gupta et al. [2004], and Rahman and Reif [2000] as typical of 3D stacking interconnects. In general, we found die-to-die interconnect capacitance to be below 3fF. We also verified this with extracted parasitic capacitance values from 3D Magic, a tool recently developed at MIT. The extracted capacitance was found to be 2.7fF, which agrees with the results presented in Gupta et al. [2004]. By comparison with 2D on-chip interconnect, a global interconnect wire was estimated to have capacitance of 400fF per millimeter, based on Ho and Horowitz [2001]. Therefore, we can assume that the additional interconnect capacitance in 3D stacking vias is negligible. As for the number of I/O connections that are possible between dies, a figure of 10,000 connects per square millimeter is reported [Gupta et al. 2004]. Our needs are much less. From our studies, we need roughly 1100 I/O connections: 32 bits for our address bus, 1024 bits for the data bus, and some additional control signals. For estimating the interconnect capacitance on our processor and peripheral layer, we again referred to Ho and Horowitz [2001] to generate analytical and projected values. We selected a wire length of 12mm to account for 1.3 times the width/height of an $80mm^2$ die and scaled the wire length accordingly for smaller die sizes. We assumed we would gain a 33% reduction in wire capacitance compared to a 2D on-chip implementation from projections on interconnect wire length reduction shown in Davis et al. [2005]. Based on these initial values, we calculated the number of repeaters required to drive the interconnect range at 250∼400MHz from hspice simulations. We found we needed only a maximum of 2∼3 repeaters to drive this bus, since the frequency of this wide on-chip bus was relatively slow.

We measured the toggle rate and access rate of these wires and calculated power using the well-known dynamic power equation to calculate interconnect power. Table VI shows the expected interconnect capacitance for 1024 bits in the case of 2D on-chip, 3D stacking, and 2D off-chip implementations. Roughly speaking, on-chip implementations have at most 33% of the capacitance of an off-chip implementation. Furthermore, because the supply voltages in I/O pads (typically 1.8∼2.5V) are generally higher than the core supply voltage, we find the overall interconnect power for an off-chip implementation consumes an order of magnitude more power than an on-chip one. With modest toggle rates, small to modest access rates for typical configurations found in our benchmarks, and a modest bus frequency of 250MHz, we conclude that interdie interconnect power contributes very little to overall power consumption.

Table VII. Bandwidth and Latency

	SDRAM	DDR2 DRAM	XDR DRAM	L2 Cache @1.2GHz	On-chip DRAM 3D IC
Bandwidth (GB/sec)	1.0	5.2	31.3	21.9	31.3
Average access latency(ns)	30ns	25ns	28ns	16ns	25ns*

*Average access latency with no 3D stacking aware optimizations. On-chip DRAM latency expected to reduce by more than 50% [Matick and Schuster 2005] when 3D stacking optimizations are applied.

These results suggest on-chip DRAM can easily provide enough memory bandwidth compared to an L2 cache, as noted in Laudon [2005] and Wendell et al. [2004]. Average access latency for SDRAM and DDR2 DRAM is estimated to be t_{RCD}+t_{CAS}, where t_{RCD} denotes RAS to CAS delay and t_{CAS} denotes CAS delay. For, XDRAM t_{RAC-R} is used, where t_{RAC-R} denotes the read-access time.

3.2.3 *DRAM.* We made DRAM area estimates for the PicoServer using the data in MacGillivray [2005]. Currently, it is reasonable to say that 80mm^2 of chip area is required for 64MB of DRAM in 90nm technology.

Conventional DRAM is packaged separately from the processor and is accessed through I/O pad pins and wires on a PCB. However, for our architecture, DRAM exists on chip and connects to the processor and peripherals through a 3D stacking via. Therefore, the pad power consumed by the packages, necessary for driving signals off-chip across the PCB, is avoided in our design. Using the Micron DRAM spreadsheet calculator [MicronDRAM 2008] modified to omit pad power, and profile data from M5 including the number of cycles spent on DRAM reads, writes, and page-hit rates, we generated an average power for DRAM. We compared the estimated power from references on DRAM and especially with the DRAM power values generated from the SunFire T2000 Server Power Calculator [Sun Fire T2000 2008]. The Micron spreadsheet uses actual current measurements for each DRAM operation: read, write, refresh, bank precharge, etc. We assumed a design with a 1.8V voltage supply.

3.2.4 *Network Interface Controller* (*NIC*). Network interface controller power was difficult to model analytically, due to lack of information of the detailed architecture of commercial NICs. For our simulations, we looked at the National Semiconductor 82830 gigabit ethernet controller. This chip implements the MAC layer of the ethernet card and interfaces, with the physical layer (PHY) using the gigabit media independent interface (GMII) interface. We analyzed the datasheet and found the maximum power consumed by this chip to be 743mW [NSNIC 2001]. This power number is for 180nm technology. We assumed maximum power is consumed when all the input and output pins are active.We then derated this figure based on our measured usage. In addition, we assumed static power at 30% of the maximum chip power.

4. PICOSERVER ARCHITECTURE

Table VII shows the latency and bandwidth achieved for conventional DRAM, XDR DRAM, L2 cache, and on-chip DRAM using 3D stacking technology. With a 1024-bit wide bus, the memory latency and bandwidth achieved in a 3D stacking on-chip DRAM is comparable to an L2 cache and XDR DRAM. This suggests an L2 cache is not needed if stacking is used. Furthermore, the removal of off-chip drivers in conventional DRAM reduces access latency by more than 50% [Matick

and Schuster 2005]. This strengthens our argument that on-chip DRAM can be as effective as an L2 cache. Another example that strengthens our case is that DRAM vendors are producing and promoting DRAM implementations with reduced random-access latency [NetRAM 2005; RLDRAM 2008]. Therefore, our PicoServer architecture does not have an L2 cache and the on-chip DRAM is connected through a shared-bus architecture to the L1 caches of each core. The role of this on-chip DRAM is as a primary system memory.

The PicoServer architecture is comprised of single issue in-order processors that together create a chip multiprocessor which is a natural match to applications with a high level of TLP [Kongetira et al. 2005]. Each PicoServer CPU core is clocked at a nominal value of 500MHz and has an instruction and data cache, with the data caches using a MESI cache-coherence protocol. Our studies showed the majority of bus traffic is generated from cache miss traffic, not cache coherence. This is due to the properties of the target application space and the small L1 caches: 16KB instruction and 16KB data per core. With current densities, the capacity of the on-chip DRAM stack in PicoServer is hundreds of megabytes. In the near future this will rise to several gigabytes, as noted in the Table II. Other components such as the network interface controller (NIC), DMA controller, and additional peripherals that are required in implementing a full system are integrated on the CPU die.

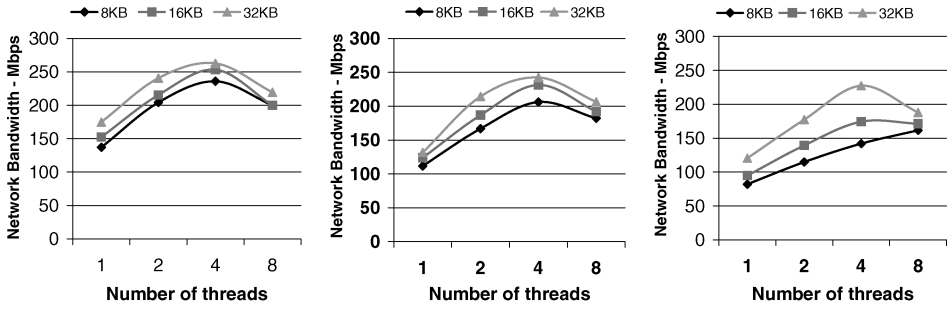
4.1 Core Architecture and the Impact of Multithreading

PicoServer is made up of simple, single issue in-order cores with a five-stage pipeline. A 32-bit architecture is assumed for each core. Branch prediction is still useful in a server workload. Each core has a hybrid branch predictor with a 1KB history table. Our studies showed the accuracy of the branch predictor for server workloads to be about 95%.

Each core also includes architectural support for a shared memory protocol and a memory controller that is directly connected to DRAM. The memory controller responds to shared bus snoops and cache misses. On a request to DRAM, the memory controller delivers the address, as well as data for memory writes or the CPU ID for memory reads. The CPU ID is needed for return routing of read data. Our estimated die area for a single core is $4\sim5\text{mm}^2$ (shown in Table V).

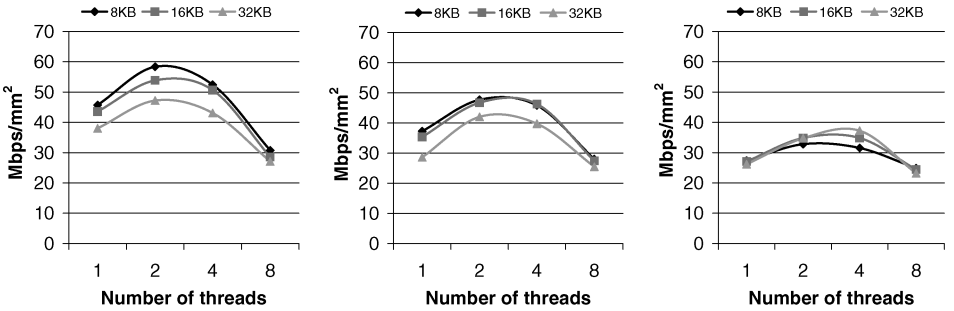
Despite some benefits that can be obtained from multithreading (described in later paragraphs), we assume no support for multithreading due to the limitation in our simulation environment. Without significant modification to a commodity Linux kernel, it is difficult to scale server applications to more than 16 cores or threads. For this reason our study of multithreading examined a single core with multiple threads. We extrapolated this to the multicore core case to show how many threads would be optimal when we leverage 3D stacking technology. Multithreading has the potential to improve overall throughput by switching thread contexts during lengthy stalls to memory.

To study the impact of multithreading in PicoServer, we assume multithreading support that includes an entire thread context: register file, store buffer, and interrupt trap unit. An additional pipeline stage is required to schedule



(a) memory latency = 1 (b) memory latency = 10 (c) memory latency = 100

Fig. 5. Impact of multithreading for varying memory latency on SURGE for varying 4-way set-associative cache sizes (8KB, 16KB, 32KB) and a varying number of threads. We assume the core is clocked at 500MHz.



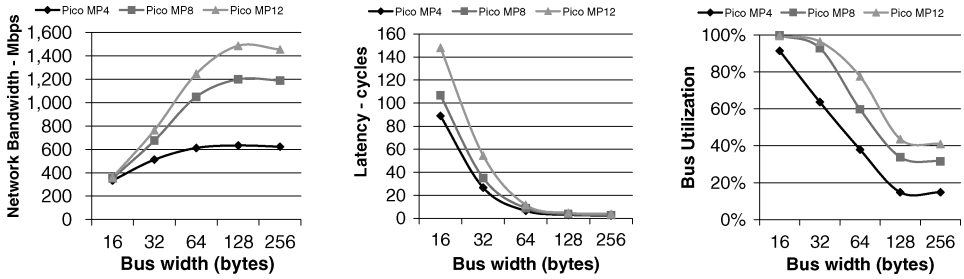
(a) memory latency = 1 (b) memory latency = 10 (c) memory latency = 100

Fig. 6. Impact of multithreading for Mbps/mm^2 when varying memory latency on SURGE. The same setup and assumptions as in Figure 5 are applied.

threads. We assumed a die area overhead of supporting 4 threads to be about 50%. Although Laudon [2005] predicted a 20% die area overhead to support 4 threads in the Niagara core, our cores are much smaller: 5mm^2 versus 16mm^2. Register and architectural state die area estimates from Clark et al. [2001] and Ricci et al. [2005] take up a larger percentage of the total die area. Therefore, we assessed a greater area overhead for PicoServer cores.

In the multithreading study, we varied the number of threads that can be supported and access latency to memory from a single core and measured the network bandwidth (a metric for throughput) delivered by this core. We did our analysis running SURGE because it displayed the highest L1 cache miss rate, which implies it would benefit the most from multithreading. Our metrics used in this study are total network bandwidth and network bandwidth/mm^2. We varied the cache size to see the impact of threading.

Figures 5 and 6 show our simulated results. From these we are able to conclude that threading indeed helps improve overall throughput, however, only to a limited extent when considering the area overhead and the impact of 3D stacking. 3D stacking reduces the access latency to memory by simplifying the



(a) network bandwidth vs. bus width

(b) overall bus-loaded latency vs. bus width

(c) bus utilization vs.bus width

Fig. 7. Network performance on SURGE for various shared bus architectures with an L1 cache of 16KB (each for I and D). We assumed a CPU clock frequency of 500MHz for these experiments. Our bus architecture must be able to handle high bandwidths as the number of processors increases.

core-to-memory interface and reducing the transfer latency. 3D stacked memory can be accessed in tens of cycles, which corresponds to the plots shown in Figures 5(b) and 6(b). The latter plot suggests that if area efficiency and throughput are taken together, a limitation of only 2 threads appears optimal. We also find that the memory and I/O traffic increases as we add additional threads to the core. Therefore, a system must be able to deliver sufficient I/O bandwidth, and memory bandwidth to accommodate the additional threads. Otherwise, threading will be detrimental to overall system throughput.

4.2 Wide Shared-Bus Architecture

PicoServer adopts a wide shared bus architecture that provides high memory bandwidth and fully utilizes the benefits of 3D stacking technology. Our bus architecture was determined from SURGE runs on M5; we limited it to SURGE because it generates a representative cache miss rate per core on our benchmarks. To explore the design space of our bus architecture, we first ran simulations for varying the bus width on a single shared-bus ranging from 128 bits–2048 bits. We varied the cacheline size as well to match the bus width (varied it from 16 bytes–256 bytes). Network bandwidth (a metric for throughput) was measured to determine the impact of bus width on the PicoServer. As shown in Figure (7a), a relatively wide data bus is necessary to achieve scalable network performance to satisfy the outstanding cache miss requests. This is because of the high bus contention on the shared data bus for high bus traffic that is generated for narrow bus widths, as shown in Figures (7b) and (7c). As we decrease the bus width, the bus traffic increases, resulting in a superlinear increase in latency. Reducing bus utilization implies reduced bus arbitration latency, thus improving network bandwidth. Wide bus widths also help speed-up NIC DMA transfers by allowing a large chunk of data be copied in one transaction. A 1024-bit bus width seems reasonable for our typical PicoServer configurations of 4, 8, and 12 cores. Having more cores causes network performance to saturate, unless wider buses are employed. We also looked at interleaved bus architectures but found that with our given L1 cache miss rates,

Table VIII. Projected On-Chip DRAM Size for Varying Process Technologies

	130nm	110nm	90nm	80nm
DRAM stack of 4 layers each layer 40mm^2	64MB	96MB	128MB	192MB
DRAM stack of 8 layers each layer 40mm^2	128MB	192MB	256MB	384MB
DRAM stack of 4 layers each layer 60mm^2	96MB	144MB	192MB	288MB
DRAM stack of 8 layers each layer 60mm^2	192MB	288MB	384MB	576MB
DRAM stack of 4 layers each layer 80mm^2	128MB	192MB	256MB	384MB
DRAM stack of 8 layers each layer 80mm^2	256MB	384MB	512MB	768MB

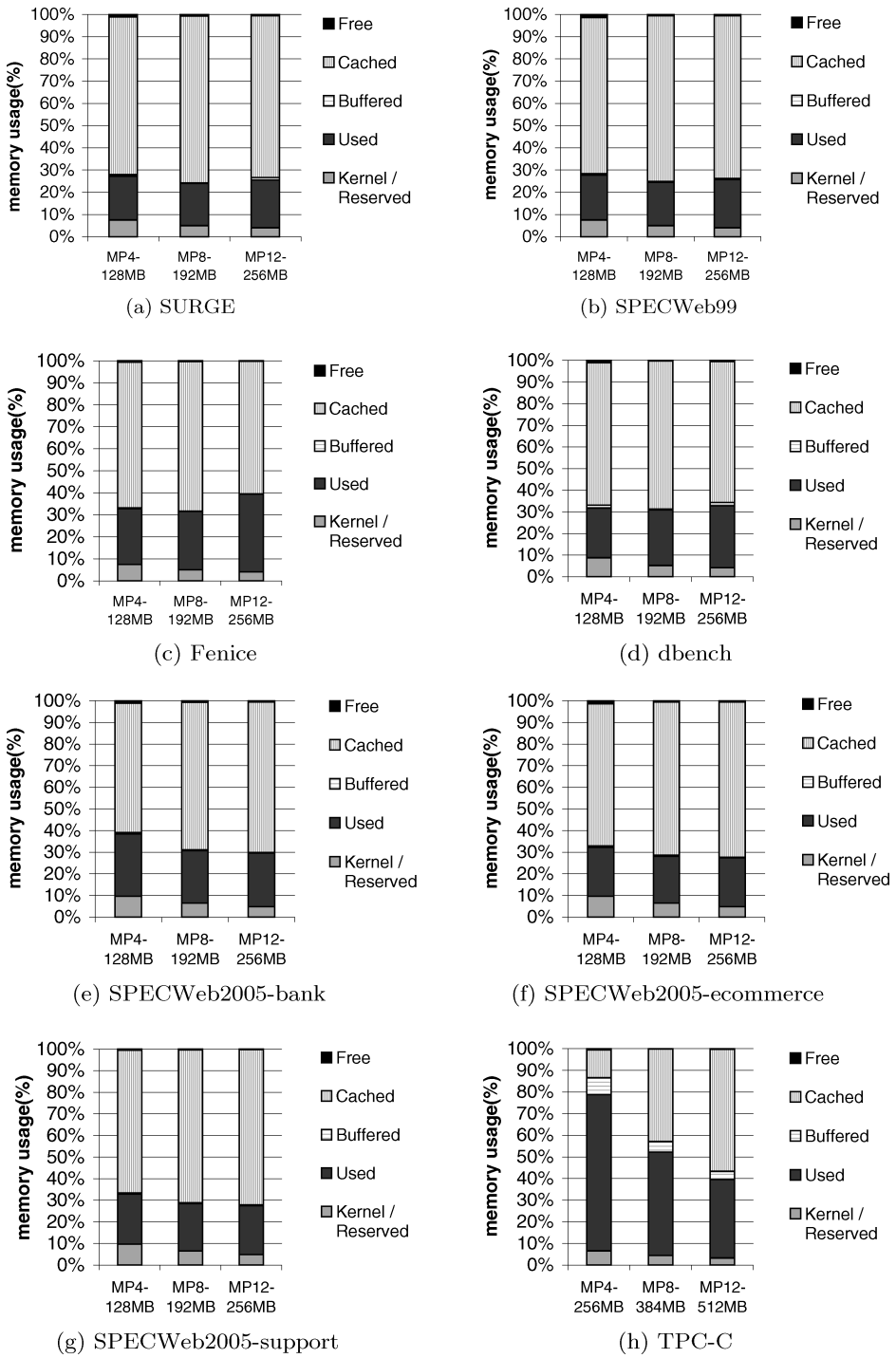
Area estimates are generated based on Semiconductor SourceInsight 2005 [MacGillivray 2005]. 80mm^2 of die size is similar to that of a Pentium M at 90nm.

a 1024-bit bus is wide enough to handle the bus requests. For architectures and workloads that generate higher bus requests as a result of increasing the number of cores to 16 or more, or by having L1 caches with higher miss rates (more than 10%), then interleaving the bus becomes more effective. An interleaved bus architecture increases the number of outstanding bus requests, thus addressing the increase in number of bus requests.

4.3 On-Chip DRAM Architecture

4.3.1 *Role of On-Chip DRAM.* Based on the logic die area estimates, we projected the DRAM die size for a 12-core PicoServer to be 80mm^2, and 40mm^2 and 60mm^2, respectively, for a 4-core and 8-core PicoServer. Table VIII shows the on-chip memory alternatives for PicoServers. For example, to obtain a total DRAM size of 256MB, we assume a DRAM made up of a stack of 4 layers. For tier-3 servers we employ 8 layers because they rely heavily on system memory size. With current technology, namely 90nm, it is feasible to create a 4-layer stack containing 256MB of physical memory for a die area of 80mm^2. Although a large amount of physical memory is common in server farms (4GB–16GB) today, we believe server workloads can be scaled to fit into smaller systems with smaller physical memory, based on our experience with server workloads and discussions with datacenter experts [Lim et al. 2008]. From our measurements on memory usage for server applications shown in Figure 8, we found for many of the server applications (except TPC-C and TPC-H) that a modest amount of (around 64MB) of system memory is occupied by the user application, data, and the kernel OS code. The remainder of the memory is either free or used as a disk cache. When we consider that much of the user memory space in TPC-C and TPC-H is allocated as user-level cache, this is even true for TPC-C and TPC-H. Considering the fact that 256MB can be integrated on-chip for 4 die layers, a large portion of on-chip DRAM can be used as a disk cache. Therefore, for applications that require small/medium filesets, an on-chip DRAM of 256MB is enough to handle client requests.

For large filesets, there are several options to choose from. First, we could add additional on-chip DRAM by stacking additional DRAM dies, as in the 8-layer case. From the ITRS roadmap in Table II, recall that the number of stacked dies we assume is conservative. With aggressive die stacking, we could add more die stacks to improve on-chip DRAM capacity; ITRS projects more than 11 layers in the next 2∼4 years. This is possible because our power density in the logic



(a) SURGE (b) SPECWeb99

(c) Fenice (d) dbench

(e) SPECWeb2005-bank (f) SPECWeb2005-ecommerce

(g) SPECWeb2005-support (h) TPC-C

Fig. 8. Breakdown in memory for server benchmarks (SURGE, SPECweb99, Fenice, dbench, SPECweb2005, TPC-C). TPC-H is excluded because it displayed similar memory usage as TPC-C.

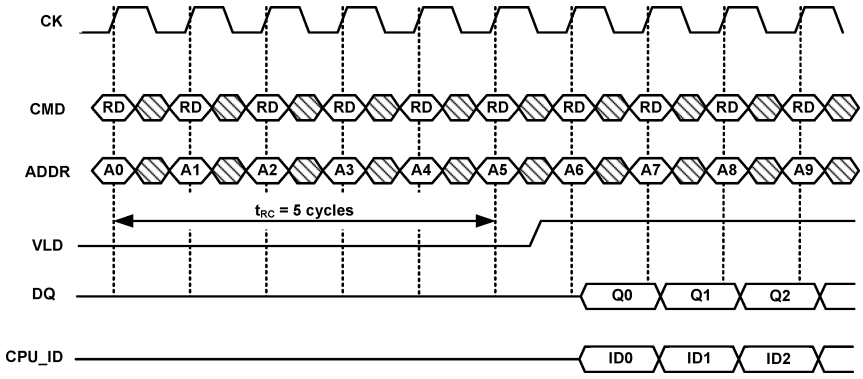


Fig. 9. On-chip DRAM read timing diagram without address multiplexing.

layer is quite small: less than $5W/cm^2$. Another alternative is to add a secondary system memory which functions as a disk cache. For the workloads we considered in this study, we found that the access latency of this secondary system memory could be as slow as hundreds of μs, without affecting throughput. An access latency as slow as hundreds of μs implies that the flash memory that consumes less active/standby power can be used as secondary system memory. This idea has been explored in Kgil and Mudge [2006] and Kgil et al. [2008]. Therefore, for workloads requiring large filesets, we could build a nonuniform memory architecture with fast on-chip DRAM and relatively slower off-chip secondary system memory. The fast on-chip DRAM would primarily hold code, data, and a small disk cache, and the slow system memory would function as a large disk cache device.

4.3.2 *On-Chip DRAM Interface.* To maximize the benefits of 3D stacking technology, the conventional DRAM interface needs to be modified for PicoServer's 3D stacked on-chip DRAM. Conventional DDR2 DRAMs are designed assuming a small pin count and use address multiplexing and burst mode transfer to make up for the limited number of pins. With 3D stacking technology, there is no need to use narrow interfaces and address multiplexing with the familiar two-phase commands, RAS then CAS. Instead, the additional logic required for latching and muxing narrow addresses/data can be removed. The requested addresses can be sent as a single command while data can be driven out in large chunks. Further, conventional off-chip DRAMs are offered as DIMMs made up of multiple DDR2 DRAM chips. The conventional off-chip DIMM interface accesses multiple DDR2 DRAM chips per request. For 3D stacked on-chip DRAM, only one subbank needs to be accessed per request. As a result, 3D stacked on-chip DRAM consumes much less power per request than off-chip DRAM. Figure 9 shows an example of a read operation without multiplexing. It clearly shows that RAS and CAS address requests are combined into a single address request. DRAM vendors already provide interfaces that do not require address multiplexing, such as reduced latency DRAM from Micron [RLDRAM 2008] and NetDRAM [NetRAM 2005] from Samsung. This suggests the interface required for 3D stacked on-chip DRAM can be realized

with only minor changes to existing solutions. Additional die area made available through the simplification of the interface can be used to speed-up the access latency to on-chip DRAM. By investing more die area to subbank the on-chip DRAM, latencies as low as 10*ns* can be achieved.[2]

4.3.3 *Impact of On-Chip DRAM Refresh on Throughput.* DRAM periodically requires each DRAM cell to be refreshed. Retention time of each DRAM cell is typically defined as 64*ms* for industry standard temperature and decreases to 32*ms* in hotter environments. Based on our thermal analysis presented in Section 4.5, our maximum junction temperature was well under the industry standard temperature constraints. As a result, we assumed a 64*ms* refresh cycle per cell. However, refresh circuits are commonly shared among multiple DRAM cell arrays to reduce the die area overhead, reducing the average DRAM refresh interval to approximately 7.8125μs and requiring approximately 200*ns* to complete. Roughly speaking, this implies that a DRAM bank cannot be accessed for a duration of hundreds of CPU clock cycles every ten thousands of CPU clock cycles. To measure the impact of refresh cycles, we modeled the refresh activity of DRAM on M5 and observed the CPI overhead. The access frequency to on-chip DRAM is directly correlated to the amount of L1 cache misses observed. We found that for a 5% L1 cache miss rate and 12 cores clocked at 500MHz (PicoMP12-500MHz running SURGE), this would incur a CPI refresh overhead of 0.03 CPI. This is because many of the L1 cache misses do not occur when a refresh command is executed, resulting in only a marginal performance penalty.

4.4 The Need for Multiple NICs on a CMP Architecture

A common problem of servers with large network pipes is handling bursty behavior in the hundreds of thousands of packets that can arrive each second. Interrupt coalescing is one method of dealing this problem. It works by starting a timer when a noncritical event occurs. Any other noncritical events that occur before the timer expires are coalesced into one interrupt, reducing their total number. Even with this technique, however, the number of interrupts received by a relatively low-frequency processor, such as one of the PicoServer cores, can overwhelm it. In our simulations we get around this difficulty by having multiple NICs, one for each of a subset of the processors. For an 8-chip multiprocessor architecture with 1 NIC and on-chip DRAM, we found the average utilization per processor to be below 60%, as one processor could not manage the NIC by itself. To fully utilize each processor in our multiple processor architecture, we inserted 1 NIC for every 2 processors. For example, a 4-processor architecture would have 2 NICs, an 8-processor architecture would have 4 NICs, and so forth.

Although our simulation environment does not support it, a more ideal solution would have a smarter single NIC that could route interrupts to multiple CPUs, each with separate DMA descriptors and TX/RX queues. This could be

[2]In this study, we took a conservative approach and did not apply the latency reduction due to additional subbanking. We only applied latency optimizations due to the removal of off-chip drivers.

Table IX. Thermal Parameters for Commonly Found Materials in Silicon Devices

	Thermal Conductivity (W/m·K)	Heat Capacity (J/m^3·K)
Si	148	1.75×10^6
SiO_2	1.36	1.86×10^6
Cu	385	3.86×10^6
Air at 25C^o	0.026	1.2×10^3

one NIC, either with multiple interface IP addresses or an intelligent method of load-balancing packets to multiple processors. Such a NIC would need to keep track of network protocol states at the session level. There have been previous studies of intelligent load balancing on NICs to achieve optimal throughput on platforms [Congduc 2004]. TCP splicing and handoff are also good examples of intelligent load balancing at higher network layers [Maltz and Bhagwat 1998].

4.5 Thermal Concerns in 3D stacking

A potential concern with 3D stacking technology is heat containment. To address this concern, we investigated the thermal impact of 3D stacking on the PicoServer architecture. Because we could not measure temperature directly on a real 3D stacked platform, we modeled the 3D stack with the grid model in Hotspot version 3.1 [Huang et al. 2004]. Mechanical thermal simulators such as FLOWTHERM and ANSYS were not considered in our studies, due to the limited information we could obtain about the mechanical aspects of the 3D stacking process. However, Hotspot's RC equivalent heat flow model is adequate to show trends and potential concerns in 3D stacking. Because this work describes the usefulness of integrating 3D stacking into the server space, instead of describing the details in heat transfer, we present general trends.

The primary contributors to heat containment in 3D stacking technology are the interface material (SiO_2) and the free air interface between silicon and air, as can be seen in Table IX. Silicon and metal conduct heat much more efficiently. We first configured our PicoServer architecture for various scenarios by: (1) varying the amount of stacked dies; (2) varying the location of the primary heat-generating die, the logic die in the stack; and (3) varying the thickness of the SiO_2 insulator that is typically used in between stacked dies. Our baseline configuration has a logic die directly connected to a heat sink and assumes a room temperature of 27C$^\circ$. Hotspot requires information for properties of the material and power density to generate steady-state temperatures. We extracted 3D stacking properties from Koyanagi [2005], Lu [2005], and Xue et al. [2003] and assigned power density at the component level, based on area and power projections for each component. Components were modeled at the platform-level: processor, peripheral, global bus interconnect, etc. We generated the maximum junction temperature in the PicoServer architecture as shown in Figure 10.

Figure 10(a) shows the sensitivity to the number of stacked layers. We find roughly a 2$\sim$3C$^\circ$ increase in maximum junction temperature for each additional

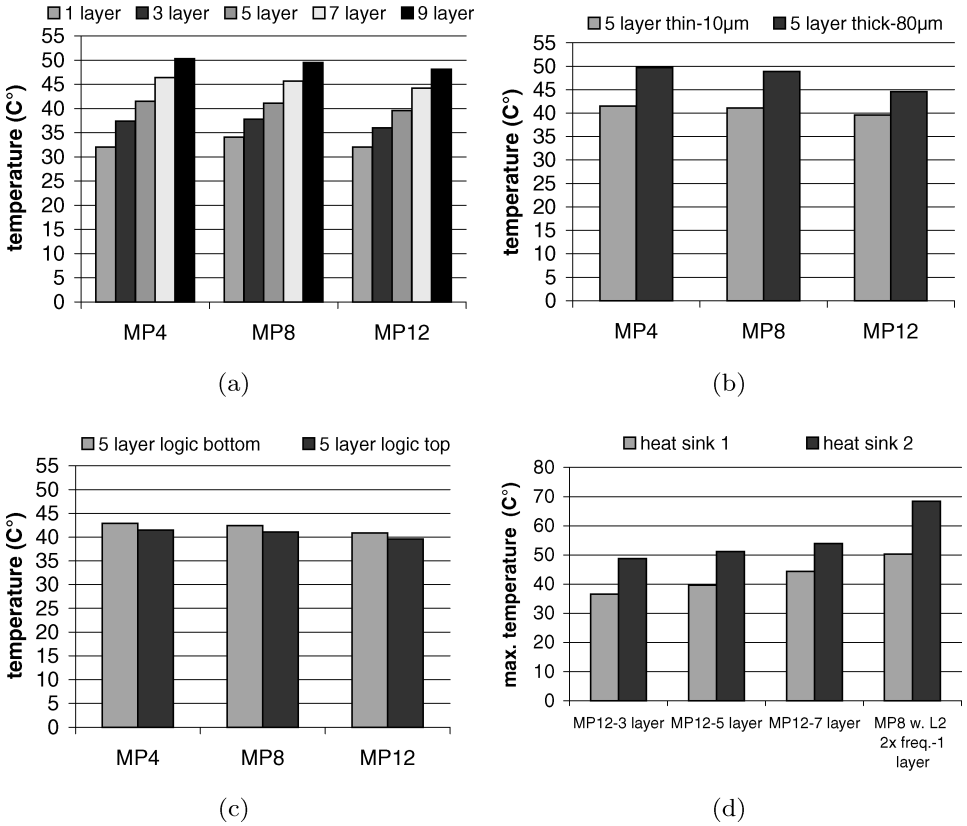


Fig. 10. Maximum junction temperature for sensitivity experiments on Hotspot: (a) varying the number of layers; (b)varying 3D interface thickness; (c)varying location of logic die; (d) maximum junction temperature for heat sink quality analysis. A core clock frequency of 500MHz is assumed in calculating power density. We varied the size of on-chip memory based on the number of layers stacked. 1 layer assumes no on-chip memory at all.

layer stacked. Interestingly, maximum junction temperature reduces as we increase the die area. We believe this is due to our floorplan and package assumptions. Further analysis is needed, and we leave it for future research. Figure 10(b) shows the sensitivity to the 3D stacking dielectric interface. We compared the effect of the SiO_2 thickness (the interface material) for $10\mu m$ and $80\mu m$. In Black et al. [2004], Koyanagi [2005], Lu [2005], and Xue et al. [2003] we find the maximum thickness of the interface material does not exceed $10\mu m$ for 3D stacking. The $80\mu m$ point is selected to show the impact of heat containment as the thickness is increased substantially. It results in a 6C$^\circ$ increase in junction temperature. While notable, this is not a great change, given the dramatic change in material thickness. We assumed the increase in dielectric interface thickness did not increase bus latency because the frequency of our on-chip bus was relatively slow. Figure 10(c) shows the sensitivity to placement in the stack, top or bottom layer. We find the primary heat-generating die is not sensitive to the location of the heat sink.

We also conducted an analysis on the impact of heat-sink quality. We varied the heat sink configuration to model a high-cost heat sink (heat sink 1) and a low-cost heat sink (heat sink 2). Figure 10(d) shows the impact of 3D stacking technology on heat-sink quality. It clearly suggests that a low-cost heat sink can be used on platforms using 3D stacking technology.

The aforementioned results suggest that heat containment is not a major limitation for the PicoServer architecture. The power density is relatively low, not exceeding $5W/cm^2$. As a result, the maximum junction temperature does not exceed $50C^{\circ}$. 3D vias can also act as heat pipes, which we did not take into account in our analysis; however, this can be expected to improve the situation. An intelligent placement would assign the heat-generating layer (the processor layer) adjacent to the heat sink, resulting in a majority of the heat being transferred to the heat sink. There is independent support for our conclusions in Chiang et al. [2001] and Goplen and Sapatnekar [2005].

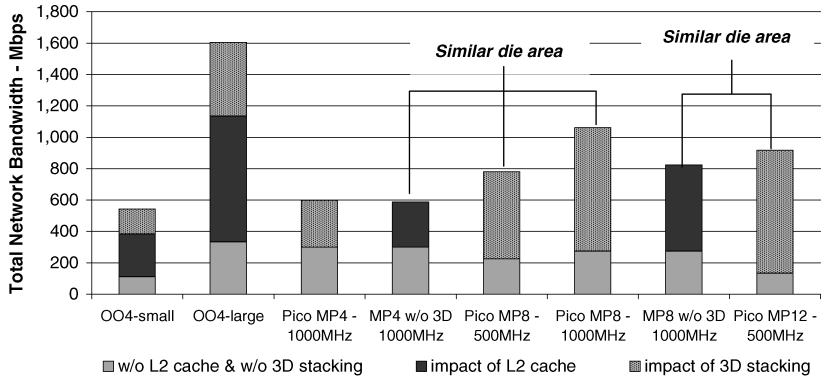
5. RESULTS

To evaluate the PicoServer architecture, two metrics are important: throughput and power. Throughput that can be measured as network bandwidth or transactions per seconds is a good indicator of overall system performance because it is a measure of how many requests were serviced. In this section, we compare various PicoServer configurations to other architectures, first in terms of achievable throughput and then in terms of power. Since the PicoServer has not been implemented, we use a combination of analytical models and published data to make a conservative estimate about the power dissipation of various components. Finally, we present a Pareto chart showing the energy efficiency of the PicoServer architecture.

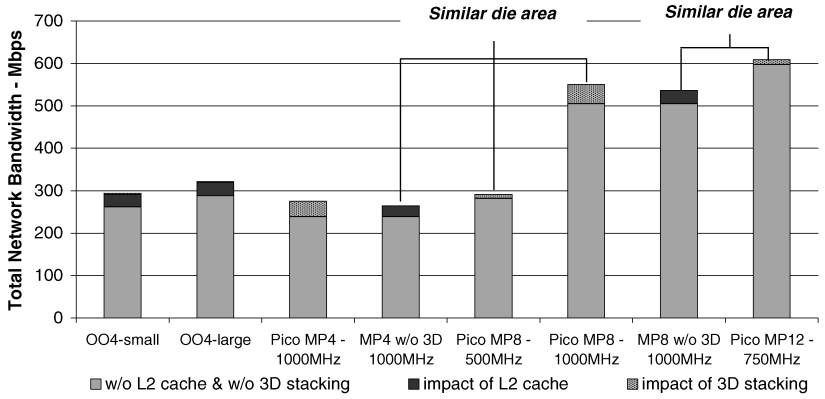
5.1 Overall Performance

Figures 11 and 12 show the throughput for some of our tier-1, -2, and -3 workload runs. Each bar shows the contribution to throughput in three parts: (1) a baseline with no L2 cache and a narrow (64-bit) bus; (2) the baseline, but with an L2 cache; or (3) the baseline with a wide bus, no L2 cache, and 3D stacking for DRAM. Hence, we are able to make comparisons that differentiate the impact of 3D stacking technology with the impact of having an L2 cache. Figure 11 shows that using 3D stacking technology alone improves overall performance as much as or more than having an L2 cache. A fair comparison for a fixed number of cores, for example, would be a Pico MP4-1000MHz versus a conventional CMP MP4 without 3D-1000MHz. In general, workloads that generated modest to high cache miss rates (SURGE, SPECweb99, SPECweb2005, and dbench) showed dramatic improvement from adopting 3D stacking technology. Fenice shows less dramatic improvements, because it involves video stream computations that generate lower cache miss rates. Interestingly, the script language tier-2 benchmark, SPECweb2005, performed well against OO4 configurations that were expressly designed for single-threaded performance.

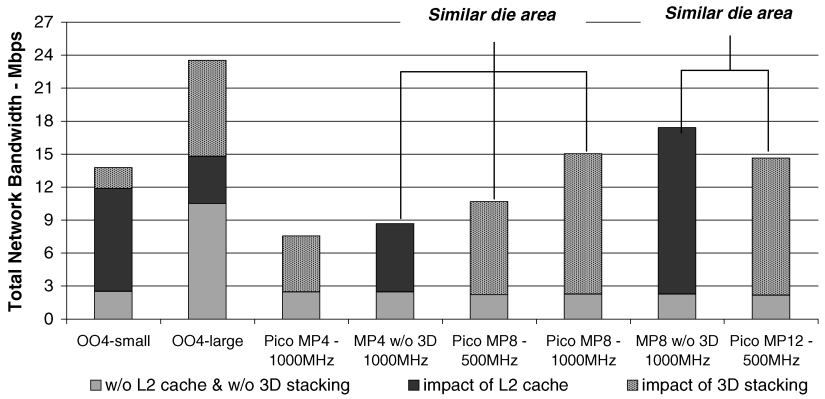
For OO4 configurations, we combine the impact of having an L2 cache and 3D stacking, since the L2 cache latency on a uniprocessor is likely to be smaller



(a) SPECweb99

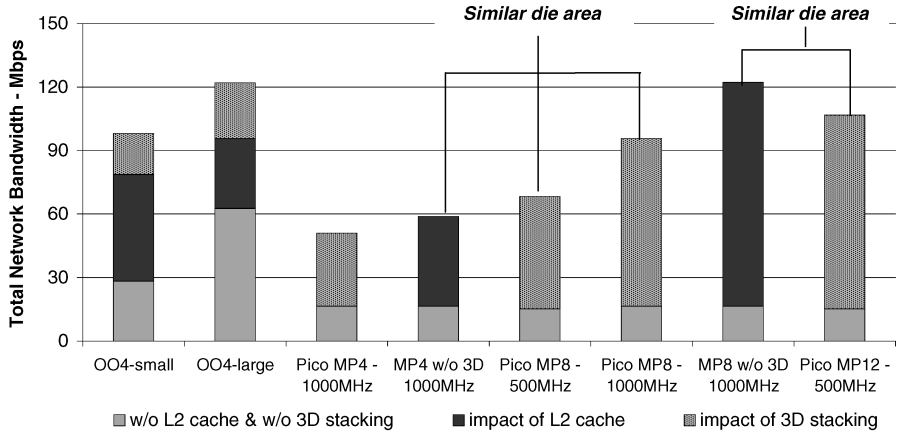


(b) Fenice

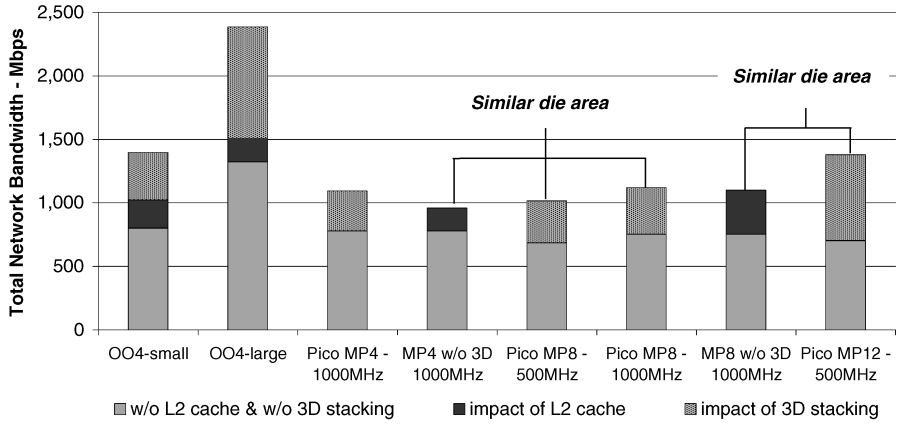


(c) SPECweb2005-bank

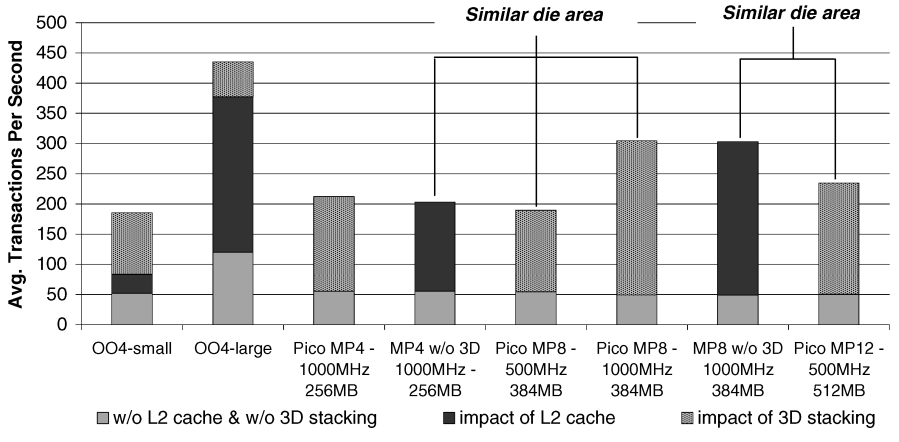
Fig. 11. Throughput measured for varying processor frequency and processor type. For PicoServer CMPs, we fixed the on-chip data bus width to 1024 bits and bus frequency to 250MHz. For a Pentium 4-like configuration, we placed the NIC on the PCI bus and assumed the memory bus frequency to be 400MHz. For an MP4 or MP8 without a 3D stacking configuration, to be fair we assumed no support for multithreading and an L2 cache size of 2MB. The external memory bus frequency was assumed to be 250MHz (SPECweb99, Fenice, SPECweb2005-bank).



(a) SPECweb2005-ecommerce



(b) dbench



(c) TPC-C

Fig. 12. Throughput measured for varying processor frequency and processor type (SPECweb2005-ecommerce, dbench,TPC-C). We applied the same assumptions used in Figure 11.

than the access latency to a large-capacity DRAM, making it less appealing to only have a high-bandwidth on-chip DRAM implemented from 3D stacking. We find that 3D stacking improves performance by 15% on OO4 configurations. When we compare an OO4 architecture without 3D stacking with our PicoServer architecture, a PicoServer MP8 operating at 500MHz performs better than a 4GHz OO4 processor with a small L1 and L2 cache of 16KB and 256KB, respectively. For a similar die area comparison, we believe comparing PicoServer MP8 and a OO4-small architecture is a fair comparison, because the OO4-large architecture requires additional die area for a 128KB L1 cache and an 2MB L2 cache.

If we assume that the area occupied by the L2 cache in our conventional CMP MP4/8 without 3D stacking technology is replaced with additional processing cores (a benefit made possible by using 3D stacking technology), then a comparison in throughput for similar die area can be conducted on the following systems: (1) a Pico MP8-500MHz versus a conventional MP4 without 3D-1000MHz; and (2) a Pico MP12-500MHz versus a conventional MP8 without 3D-1000MHz (for Fenice, compared with a Pico MP12-750MHz). Our results suggest that on average, adding additional processing elements and reducing core clock frequency by half improves throughput and significantly saves on power, as we will show in Section 5.2. For compute bound workloads like Fenice, SPECweb2005-bank, and SPECweb2005-ecommerce, however, Pico MP12-500MHz did not do better than a conventional MP8 without 3D-1000MHz. For SPECweb2005-bank and ecommerce, introducing a 2MB L2 cache dramatically cuts the number of cache misses, reducing the benefit of adding more cores while lowering core clock frequency. Pico MP12-500MHz also did not perform well for TPC-C because of the I/O scheduler. However, we expect Pico MP12-500MHz to perform better for OS kernels with TPC-C-optimized I/O-scheduling algorithms. Our estimated area for adding extra cores is quite conservative, suggesting more cores could be added, to result in even more improvement in throughput.

5.2 Overall Power

Processor power still dominates overall power in PicoServer architectures. Figure 13 shows the average power consumption based on our power estimation techniques for server application runs. We find PicoServer with a core clock frequency of 500MHz is estimated to consume between $2 \sim 3$ Watts for 90nm process technology. Much of the total power is consumed by the simple in-order cores. NIC power also consumes a significant amount, due to the increase in number of NICs when increasing the number of processors. However, as described in Section 4.4, an intelligent NIC designed for this architecture could be made more power efficient, as a more advanced design would only need one. An appreciable amount of DRAM power reduction is also observed due to 3D stacking. The simplified on-chip DRAM interface requires that fewer DRAM subbanks need to be simultaneously accessed per request. Other components such as the interconnect make marginal contributions to overall system power, due to the modest access rates and toggle rates of these components.

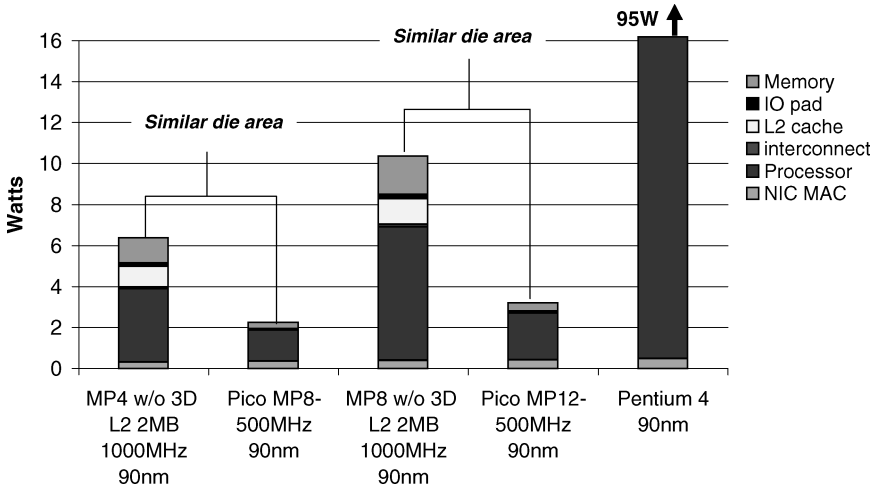


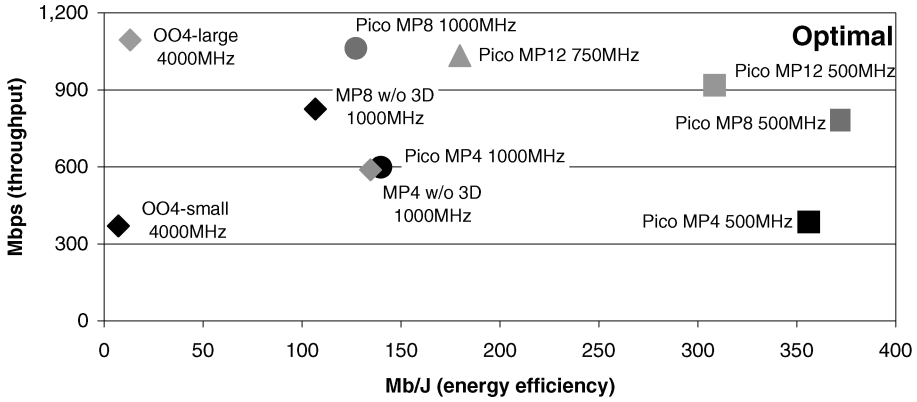
Fig. 13. Breakdown of average power for 4-, 8-, and 12-core PicoServer architectures using 3D stacking technology for 90nm process technology. Estimated power per workload does not vary by a lot because the cores contribute to a significant portion of power. We expect 2∼3 Watts to be consumed at 90nm. An MP8 without 3D stacking operating at 1GHz is estimated to consume 8W at 90nm.

Comparing our PicoServer architecture with other architectures, we see that for a similar die area comparison, we use less than half the power when we compare Pico MP8/12-500MHz with a conventional MP4/8 without 3D stacking but with an L2 cache at 1000MHz. We also recall in Section 5.1 that, in terms of performance, for a similar die area, the PicoServer architectures perform on average 10∼20% better than conventional CMP configurations. Furthermore, we use less than 10% of the power of a Pentium 4 processor while, as we showed in the previous section, performing comparably. At 90nm technology, it can be projected that the power budget for a typical PicoServer platform satisfies the mobile/handheld power constraints noted in ITRS projections. This suggests the potential of implementing server-type applications in ultrasmall form factor platforms.

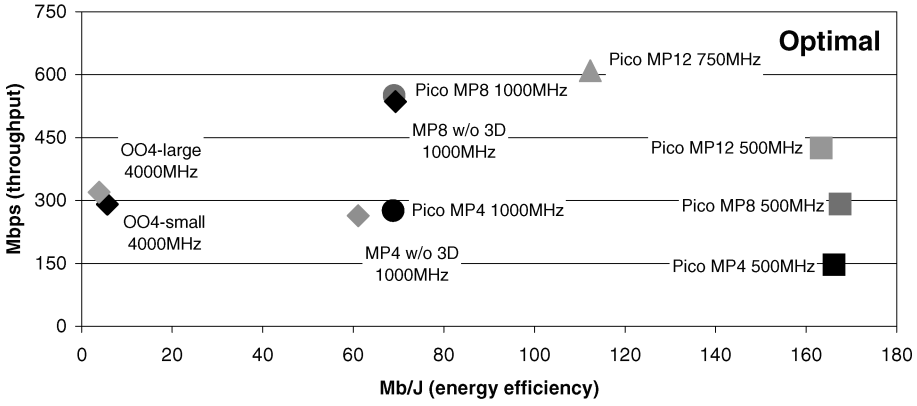
5.3 Energy Efficiency Pareto Chart

In Figures 14 and 15, we present a Pareto chart for PicoServer, depicting the energy efficiency (in Mbs per Joule) and throughput (we only list the major workloads). The points on this plot show the large out-of-order cores, conventional CMP MP4/8 processors without 3D stacking, and the PicoServer with 4, 8, and 12 cores. On the y-axes we present Mbps and transactions per second, and on the x-axes we show Mb/J and transactions per Joule. From Figures 14 and 15, it is possible to find the optimal configuration of processor number and frequency for a given energy-efficiency/throughput constraint.

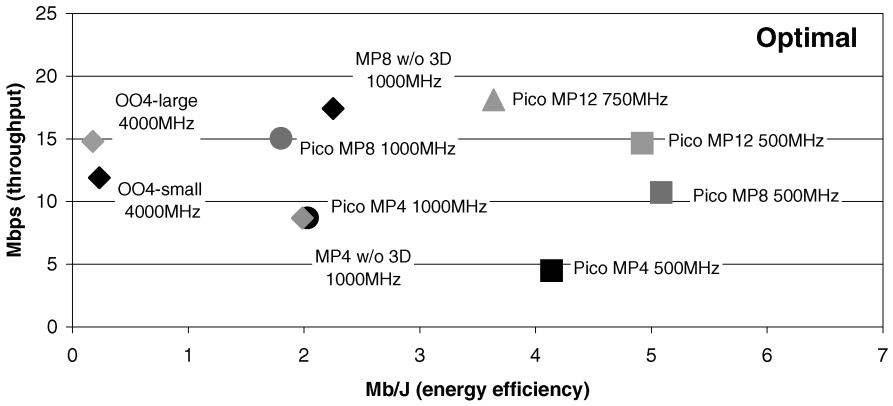
Additionally from Figures 14 and 15, we find the PicoServer architectures clocked at modest core frequency: 500MHz is 2∼4× more energy efficient than conventional chip-multiprocessor architectures without 3D stacking technology. The primary power savings can be attributed to 3D stacking technology



(a) SPECweb99

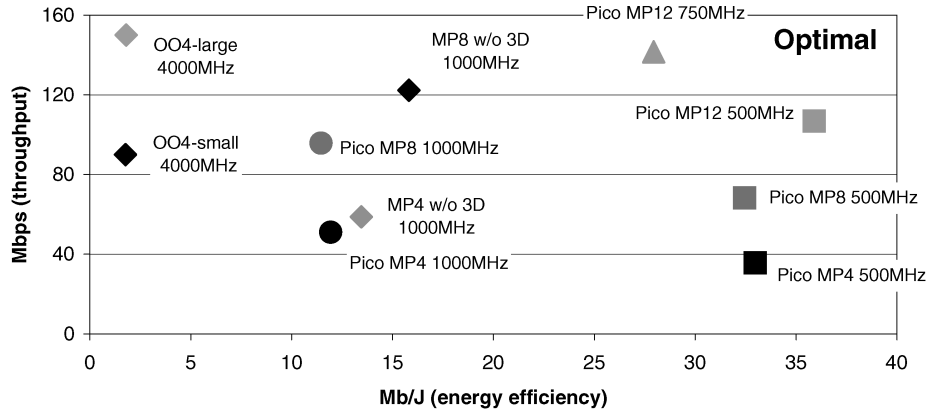


(b) Fenice

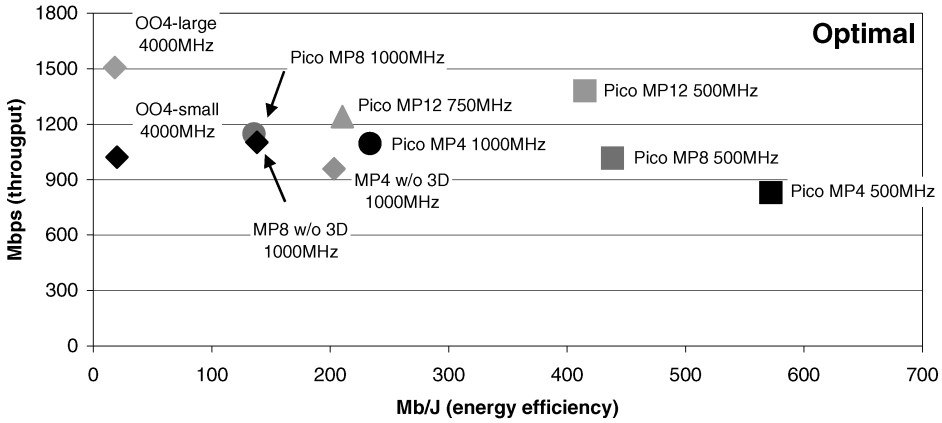


(c) SPECweb2005-bank

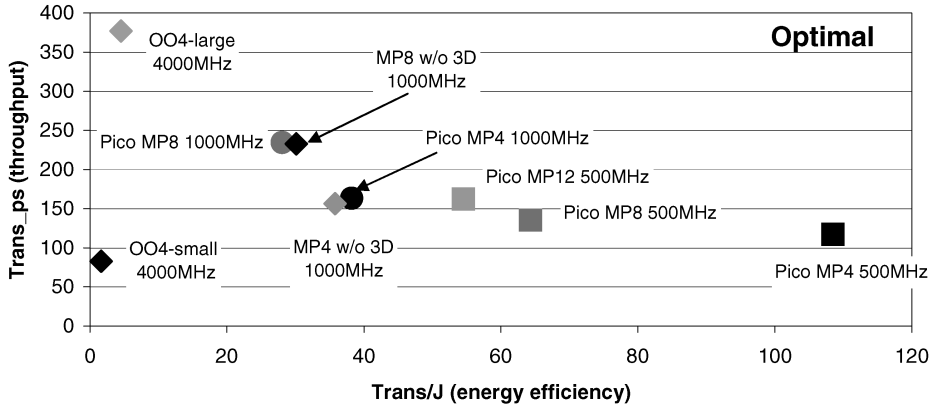
Fig. 14. Energy efficiency, performance Pareto chart generated for 90nm process technology. 3D stacking technology enables new CMP architectures that are significantly energy efficient (SPECweb99, Fenice, SPECweb2005-bank).



(a) SPECweb2005-ecommerce



(b) dbench



(c) TPC-C

Fig. 15. Energy efficiency, performance Pareto chart generated for 90nm process technology. 3D stacking technology enables new CMP architectures that are significantly energy efficient (SPECweb2005-ecommerce,dbench,TPC-C).

that enables a reduction in core clock frequency while providing high throughput. A sweetspot in system-level energy efficiency for our plotted datapoints can also be identified among the PicoServer architectures when comparing Pico MP4-500MHz, MP8-500MHz, and MP12-500MHz. These sweetspots in energy efficiency occur just before diminishing returns in throughput are reached as parallel processing is increased by adding more processors. The increase in parallel processing raises many issues related to inefficient interrupt balancing, kernel process/thread scheduling, and resource allocation that result in diminishing returns. Independent studies have shown the OS can be tuned to scale with many cores [Bryant et al. 2004] and Shah et al. [2007] is an example of such an implementation. However, we feel further investigation is necessary and leave such work for future research.

6. CONCLUSIONS

In this article, we show the potential of 3D stacking technology to build energy efficient servers. For a wide set of server workloads, the resulting systems have significant energy efficiency in a compact form factor. A 12-way PicoServer running at 500MHz can deliver 1Gbps of network bandwidth within a 3W power budget using a 90nm process technology. These power results are 2∼3× better than a multicore architecture without 3D stacking technology and an order of magnitude better than what can be achieved using a general purpose processor. The ability to tightly couple large amounts of memory to the cores through wide and low-latency interconnect pays dividends by reducing system complexity and creates opportunities to implement system memory with nonuniform access latency. 3D technology enables core-to-DRAM interfaces that are high throughput while consuming low power. With the access latency of on-chip DRAM being comparable to the L2 cache, the L2 cache die area can be replaced with additional cores, resulting in core clock frequency reduction while achieving higher throughput. Compared to a conventional 8-way 1GHz chip multiprocessor with a 2MB L2 cache, an area-equivalent 12-way PicoServer running at 500MHz yields an improvement in energy efficiency of more than 2×.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for providing feedback.

REFERENCES

3DRISC. 2004. FaStack 3D RISC super-8051 microcontroller. http://www.tachyonsemi.com/OtherICs/datasheets/TSCR8051Lx_1_5Web.pdf.

ARM11MPcore. 2004. ARM 11 MPcore. http://www.arm.com/products/CPUs/ARM11MPCoreMultiprocessor.html.

Banerjee, K., Souri, S. J., Kapur, P., and Saraswat, K. C. 2001. 3-D ICs: A novel chip design for improving deep-submicrometer interconnect performance and systems-on-chip integration. *Proc. IEEE 89,* 5 (May), 602–533.

Barford, P. and Crovella, M. 1998. Generating representative Web workloads for network and server performance evaluation. In *Measurement and Modeling of Computer Systems*. 151–160.

Binkert, N. L., Dreslinski, R. G., Hsu, L. R., Lim, K. T., Saidi, A. G., and Reinhardt, S. K. 2006. The M5 simulator: Modeling networked systems. *IEEE Micro 26,* 4 (Jul.-Aug.), 52–60.

BLACK, B., ANNAVARAM, M., BREKELBAUM, N., DEVALE, J., JIANG, L., LOH, G. H., MCCAULE, D., MORROW, P., NELSON, D. W., PANTUSO, D., REED, P., RUPLEY, J., SHANKAR, S., SHEN, J. P., AND WEBB, C. 2006. Die stacking (3D) microarchitecture. In *the International Symposium on Microarchitecture*.

BLACK, B., NELSON, D., WEBB, C., AND SAMRA, N. 2004. 3D processing technology and its impact on iA32 microprocessors. In *Proceedings of the International Conference on Computer Design*, 316–318.

BRYANT, R., HAWKES, J., STEINER, J., BARNES, J., AND HIGDON, J. 2004. Scaling Linux to the extreme from 64 to 512 processors. In *the Linux Symposium*.

CHIANG, T.-Y., SOURI, S. J., CHUI, C. O., AND SARASWAT, K. C. 2001. Thermal analysis of heterogeneous 3-D ICs with various integration scenario. In *IEDM Tech. Digest*, 681–684.

CLARK, L. T., HOFFMAN, E. J., MILLER, J., BIYANI, M., LIAO, Y., STRAZDUS, S., MORROW, M., VERLARDE, K. E., AND YARCH, M. A. 2001. An embedded 32-b microprocessor core for low-power and high-performance applications. *IEEE J. Solid State Circ. 36,* 11 (Nov.), 1599–1608.

CONGDUC, E. L. 2004. Packet classification in the NIC for improved SMP-based Internet servers. In *Proceedings of the International Conference on Networking*.

DAVIS, W. R., WILSON, J., MICK, S., XU, J., HUA, H., MINEO, C., SULE, A. M., STEER, M., AND FRANZON, P. D. 2005. Demystifying 3D ICs: The pros and cons of going vertical. *IEEE Des. Test Comput. 22,* 6, 498–510.

FLYNN, M. J. AND HUNG, P. 2004. Computer architecture and technology: Some thoughts on the road ahead. In *Proceedings of the International Conference on Engineering of Reconfigurable Systems and Algorithms*, 3–16.

GHOSH, M. AND LEE, H.-H. S. 2007. Smart refresh: An enhanced memory controller design for reducing energy in conventional and 3D die-stacked DRAMs. In *Proceedings of the International Symposium on Microarchitecture*.

GOPLEN, B. AND SAPATNEKAR, S. S. 2005. Thermal via placement in 3D ICs. In *Proceedings of the International Symposium on Physical Design*, 167–174.

GUPTA, S., HILBERT, M., HONG, S., AND PATTI, R. 2004. Techniques for producing 3D ICs with high-density interconnect. `www.tezzaron.com/about/papers/ieee_vmic_2004_finalsecure.pdf`.

HO, R. AND HOROWITZ, M. 2001. The future of wires. *Proc. IEEE 89,* 4 (Apr.).

HUANG, W., STAN, M. R., SKADRON, K., SANKARANARAYANAN, K., GHOSH, S., AND VELUSAM, S. 2004. Compact thermal modeling for temperature-aware design. In *Proceedings of the Design Automation Conference*.

ITRS 2005. ITRS roadmap. Tech. Rep.

KGIL, T. 2007. Architecting energy efficient servers. Ph.D. thesis, University of Michigan.

KGIL, T. AND MUDGE, T. 2006. FlashCache: A NAND flash memory file cache for low power Web servers. In *Proceedings of the International Conference on Compilers, Architecture and Synthesis for Embedded Systems*.

KGIL, T., ROBERTS, D., AND MUDGE, T. 2008. Improving NAND flash based disk caches. In *Proceedings of the International Symposium on Computer Architecture*.

KONGETIRA, P., AINGARAN, K., AND OLUKOTUN, K. 2005. Niagara: A 32-way multithreaded Sparc processor. *IEEE Micro 25,* 2 (Mar.), 21–29.

KOYANAGI, M. 2005. Different approaches to 3D chips. `http://asia.stanford.edu/events/Spring05/slides/051205-Koyanagi.pdf`.

KUNKEL, S. R., EICKEMEYER, R. J., LIPASTI, M. H., MULLINS, T. J., O'KRAFKA, B., ROSENBERG, H., VANDERWIEL, S. P., VITALE, P. L., AND WHITLEY, L. D. 2000. A performance methodology for commercial servers. *IBM J. Res. Develop. 44,* 6.

LAUDON, J. 2005. Performance/Watt: The new server focus. *SIGARCH Comput. Archit. News 33,* 4, 5–13.

LEE, K., NAKAMURA, T., ONO, T., YAMADA, Y., MIZUKUSA, T., HASHIMOTO, H., PARK, K., KURINO, H., AND KOYANAGI, M. 2000. Three-Dimensional shared memory fabricated using wafer stacking technology. In *IEDM Tech. Digest*, 165–168.

LIM, K., RANGANATHAN, P., CHANG, J., PATEL, C., MUDGE, T., AND REINHARDT, S. 2008. Understanding and designing new server architectures for emerging warehouse-computing environments. In *Proceedings of the International Symposium on Computer Architecture*.

Loi, G. L., Agrawal, B., Srivastava, N., Lin, S.-C., Sherwood, T., and Banerjee, K. 2006. A thermally-aware performance analysis of vertically integrated (3-D) processor-memory hierarchy. In *Proceedings of the Design Automation Conference*.

LS3 2007. $(LS)^3$-Libre streaming, Libre software, Libre standards an open multimedia streaming project. http://streaming.polito.it/.

Lu, J. 2005. Wafer-Level 3D hyper-integration technology platform. www.rpi.edu/~luj/RPI_3D_Research_0504.pdf.

MacGillivray, G. 2005. Process vs. density in DRAMs. http://www.eetasia.com/ARTICLES/2005SEP/B/2005SEP01_STOR_TA.pdf.

Maltz, D. A. and Bhagwat, P. 1998. TCP splicing for application layer proxy performance. Res. Rep. RC 21139, IBM. March.

Matick, R. E. and Schuster, S. E. 2005. Logic-Based eDRAM: Origins and rationale for use. *IBM J. Res. Develop. 49,* 1 (Jan.).

MicronDRAM 2008. The Micron system-power calculator. http://www.micron.com/support/part_info/powercalc.

Mudge, T. 2001. Power: A first-class architectural design constraint. *IEEE Comput. 34,* 4 (Apr.).

NetRAM. 2005. Evolution of network memory. http://www.jedex.org/images/pdf/jack_troung_samsung.pdf.

NSNIC 2001. National semiconductor DP83820 10 / 100 / 1000 Mb/s PCI ethernet network interface controller.

Ohsawa, T., Fujita, K., Hatsuda, K., Higashi, T., Shino, T., Minami, Y., Nakajima, H., Morikado, M., Inoh, K., Hamamoto, T., Watanabe, S., Fujii, S., and Furuyama, T. 2006. Design of a 128-Mb SOI DRAM Using the Floating Body Cell (FBC). *IEEE J. Solid State Circ. 41,* 1 (Jan).

OSDL. 2006. OSDL dataBase test suite. http://www.osdl.net/lab_activities/kernel_testing/osdl_database_test_suite/.

Rahman, A. and Reif, R. 2000. System-Level performance evaluation of three-dimensional integrated circuits. *IEEE Trans. VLSI 8*.

Ricci, F., Clark, L. T., Beatty, T., Yu, W., Bashmakov, A., Demmons, S., Fox, E., Miller, J., Biyani, M., and Haigh, J. 2005. A 1.5GHz 90nm embedded microprocessor core. In *Proceedings of the Symposium on VLSI Circuits*.

RLDRAM. 2008. RLDRAMA memory. http://www.micron.com/products/dram/rldram/.

Schutz, J. and Webb, C. 2004. A scalable X86 CPU design for 90 nm process. In *Proceedings of the International Solid-State Circuits Conference*.

Shah, M., Barreh, J., Brooks, J., Golla, R., Grohoski, G., Gura, N., Hetherington, R., Jordan, P., Luttrell, M., Olson, C., Saha, B., Sheahan, D., Spracklen, L., and Wynn, A. 2007. UltraSPARC T2: A highly-threaded, power-efficient, SPARC SOC. In *Asian Solid-State Circuirts Conference*.

SPECWeb. 1999. SPECweb99 benchmark. http://www.spec.org/osg/web99/.

SPECWeb. 2005. SPECweb2005 benchmark. http://www.spec.org/web2005/.

Sun Fire T2000. 2008. Sun Fire T2000 server power calculator. http://www.sun.com/servers/coolthreads/t2000/calc/index.jsp.

Wendell, D., Lin, J., Kaushik, P., Seshadri, S., Wang, A., Sundararaman, V., Wang, P., McIntyre, H., Kim, S., Hsu, W., Park, H., Levinsky, G., Lu, J., Chirania, M., Heald, R., and Lazar, P. 2004. A 4MB on-chip l2 cache for a 90nm 1.6GHz 64b SPARC microprocessor. In *Proceedings of the International Solid-State Circuits Conference*.

Xue, L., Liu, C. C., Kim, H.-S., Kim, S., and Tiwari, S. 2003. Three-Dimensional integration: Technology, use, and issues for mixed-signal applications. *IEEE Trans. Electron Devices 50*, 601–609.

Received November 2007; revised May 2008; accepted June 2008