Chapter 9
PicoServer: Using 3D Stacking Technology to Build Energy Efficient Servers

Taeho Kgil, David Roberts, and Trevor Mudge

Abstract With power and cooling increasingly contributing to the operating costs of a datacenter, energy efficiency is the key driver in server design. One way to improve energy efficiency is to implement innovative interconnect technologies such as 3D stacking. Three-dimensional stacking technology introduces new opportunities for future servers to become low power, compact, and possibly mobile. This chapter introduces an architecture called Picoserver that employs 3D technology to bond one die containing several simple slow processing cores with 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. PicoServer is intentionally simple, requiring only the simplest form of 3D technology where die are 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.

9.1 Introduction

Datacenters are an integral part of today's computing platforms. The success of the internet and the continued scalability in Moore's law have enabled internet service providers such as Google and Yahoo to build large-scale datacenters with millions of servers. For large-scale datacenters, improving energy efficiency becomes a critical

T. Kgil (✉)
Intel, 2111 NE 25th Ave, Hillsboro, OR, 97124, USA
e-mail: taeho.kgil@intel.com

task. Datacenters based on off-the-shelf general purpose processors are unnecessarily power hungry, require expensive cooling systems, and occupy a large space. In fact, the cost of power and cooling these datacenters will likely contribute to a significant portion of the operating cost. Our claim can be confirmed in Fig. 9.1 which breaks down the annual operating cost for datacenters. As Fig. 9.1 clearly shows, the cost in power and cooling servers is increasingy contributing to the overall operating costs of a datacenter.

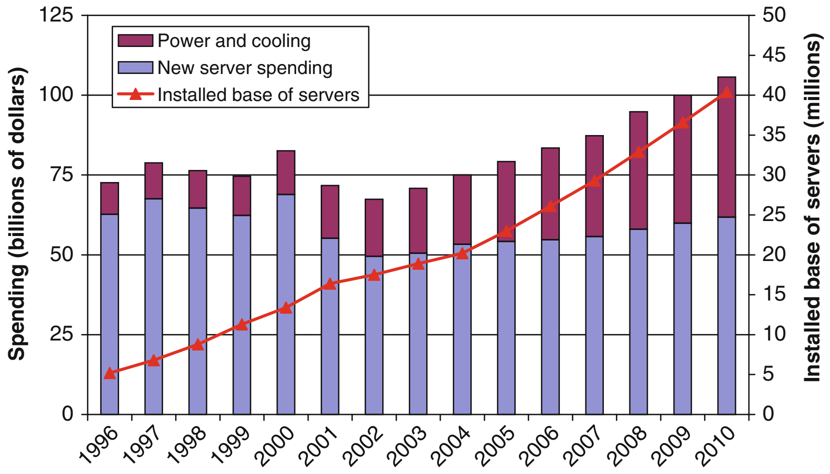


Fig. 9.1 IDC estimates for annual cost spent (1) power and cooling servers and (2) purchasing additional servers [52]

One avenue to designing energy efficient servers is to introduce innovative interconnect technology. Three-dimensional stacking technology is an interconnect technology that enables new chip multiprocessor (CMP) architectures that significantly improve energy efficiency. Our proposed architecture, PicoServer,[1] employs 3D technology to bond one die containing several simple slow processor cores with multiple DRAM dies that form the primary memory. In addition, 3D stacking enables a memory processor interconnect that is of 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. The additional cores mean that they can be run slower without affecting throughput. Slower cores also allow us to reduce power dissipation and with its 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 is stacked on top of one another. Our intent is to minimize risk of realizing a

[1]This chapter is based on the work in [32] and [29]

class of low-cost, low-power, compact server architectures. 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. Three-dimensional stacking enables the following key improvements:

- **High-bandwidth buses between DRAM and L1 caches that support multiple cores – thousands of low-latency connections with minimal 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 inter-chip 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[2] is also reduced because address multiplexing and off-chip I/O pad drivers [47] are not required. Further, it also introduces opportunities to build nonuniform memory architectures with a fast on-chip DRAM and relatively slower off-chip secondary system memory.
- **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 the integration of more cores clocked at a modest frequency – in our work 500-1000 MHz – 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 is thermal containment (see Chapter 3). 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 9.4.5. In fact the ITRS projections of Table 9.2 predict that systems consuming just a few watts do not even reequire a heat sink.

The general architecture of a PicoServer is shown in Fig. 9.2. For the purposes of this work we assume a stack of five to nine dies. The connections are by vias that run perpendicular to the dies. The dimensions for a 3D interconnect via vary from 1 to 3 μm with a separation of 1 to 6 μm. Current commercial offerings

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

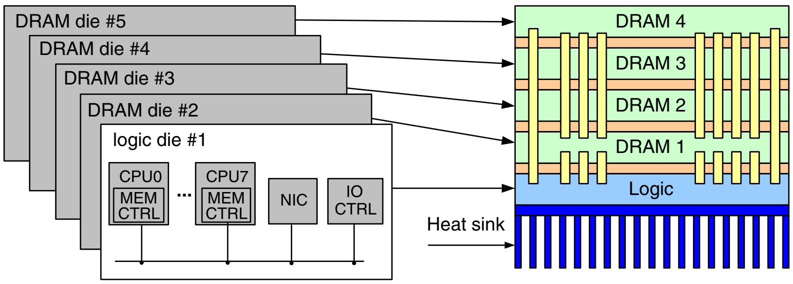


Fig. 9.2 A diagram depicting the PicoServer: a CMP architecture connected to DRAM using 3D stacking technology with an on-chip network interface controller (NIC) to provide low-latency high-bandwidth networking

can support 1,000,000 vias per cm^2 [26]. 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 – the microprocessor cores, the network interface controllers (NICs), and peripherals – are on the bottom layer and conventional capacity-oriented DRAMs occupy the remaining layers. To understand the design space and potential benefits of this new technology, we explored the tradeoffs of different bus widths, number of cores, frequencies, and the memory hierarchy in our simulations. We found bus widths of 1024 bits with a latency of two clock cycles at 250 MHz to be reasonable in our architecture. In addition, we aim for a reasonable area budget constraining the die size area to be below 80 mm^2 at 90 nm process technology. Our 12-core PicoServer configuration which occupies the largest die area is conservatively estimated to be approximately 80 mm^2. The die areas for our 4- and 8-core PicoServer configurations are, respectively 40 mm^2 and 60 mm^2.

We also extend our analysis of PicoServer and show the impact of integrating Flash onto a PicoServer architecture. We provide a qualitative analysis of two configurations that integrate (1) Flash as a discrete component and (2) directly stacks Flash on top of our DRAM + logic die stack. Both configurations leverage the benefits of 3D stacking technology. The first configuration is driven by bigger system memory capacity requirements, while the second configuration is driven by small form factor.

This chapter 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 9.3, we outline our methodology for the design space exploration. In Section 9.4, we provide more details for the PicoServer architecture and evaluate various PicoServer configurations. In Section 9.5, 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 9.6.

9.2 Background

This section provides an overview of 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 work. Finally, we show the advances in DRAM technology. We explain the current and future trends in DRAM used in the server space.

9.2.1 Server Platforms

9.2.1.1 Three-Tier Server Architecture

Today's datacenters are commonly built around a 3-tier server architecture. Figure 9.3 shows a 3-tier server farm and how it might handle a request for service. The first tier handles a large bulk of the requests from the client (end user). Tier 1 servers handle web requests. Because Tier 1 servers handle events on a per-client basis, they are independent and display high levels of thread-level parallelism. For requests that require heavier computation or database accesses, they are forwarded to Tier 2 servers. 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.

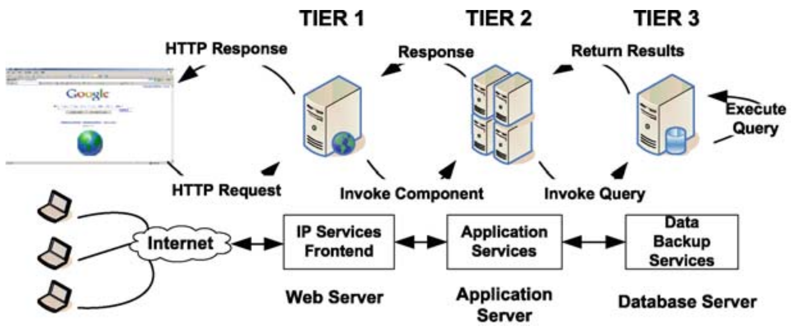


Fig. 9.3 A typical 3-tier server architecture. Tier 1 – web server, Tier 2 – application server, Tier 3 – database server

For example, when a client requests a Java Servlet Page (JSP web 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 (RMI). 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 they 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.

9.2.1.2 Server Workload Characteristics

Server workloads display a high degree of thread-level parallelism (TLP) because connection-level parallelism through client connections can be easily mapped to thread-level parallelism (TLP). Table 9.1 shows the behavior of commercial server workloads. Most of the commercial workloads display high TLP and 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 from 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 such as PHP application servers, video-streaming servers, and decision support systems, out-of-order processors have an IPC between 0.21 and 0.54 for typical server workloads, i.e., at best modest computation loads with an L2 cache of 2 MB. 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 that leads 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 appreciable amount of time spent in kernel code, unlike SPECCPU benchmarks. This kernel code is largely involved in

Table 9.1 Behavior of commercial workloads adapted from [38]

Attribute	Web99	JBOB(JBB)	TPC-C	SAP 2T	SAP 3T 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

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 request access to memory and I/O. Due to the modest computation requirements, 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 naturally 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.

9.2.1.3 Conventional Server Power Breakdown

Figure 9.4 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 1/4 is consumed by the processor, 1/4 is consumed by the system memory, 1/4 is consumed by the power supply, and 1/5 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 size 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 would not scale as fast as the core supply voltage. Therefore, there are many opportunities to further reduce power by integrating system components on-chip. And finally, we find that the power supply displays some inefficiency. This is due to the multiple levels of voltage it has to support. A reduced number of power rails will dramatically improve the power supply efficiency. Three-dimensional stacking technology has the potential to

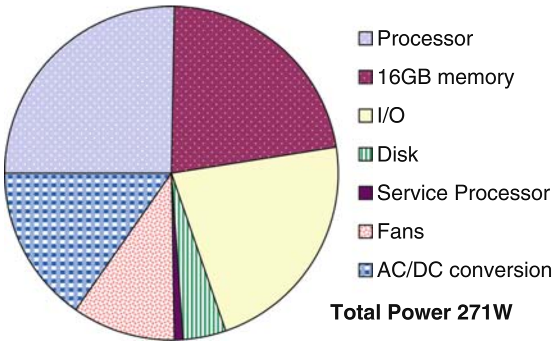


Fig. 9.4 Power breakdown of T2000 UltraSPARC executing SpecJBB

(1) reduce the power consumed by the processor and the I/O interfaces by integrating additional system components on-chip and also (2) improve power supply efficiency by reducing the number of power rails implemented in a server.

9.2.2 Three-Dimensional 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 [17, 40, 37, 44, 57]. They have met with mixed success. This is due to the many challenges that need to be addressed. They include (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 enterprises 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 result from the through silicon vias (TSVs) that have to go through silicon bulk. Figure 9.5 shows a high-level example of how dies can be bonded using 3D stacking technology. The bond between layer 1 (starting from the bottom) and 2 is face-to-face, while the bond between layer 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 are manufactured from different process steps. References [43, 24, 16] 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. Three-dimensional via pitches are equivalent to 22λ for 90-nm technology, which is about the size of a 6T SRAM cell. They are also expected to shrink as this technology becomes mature.

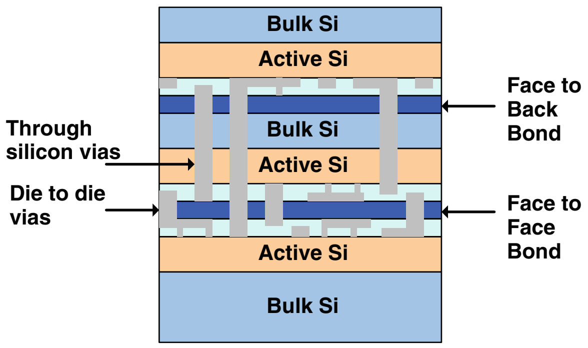


Fig. 9.5 Example of a three-layer 3D IC

The ITRS roadmap in Table 9.2 predicts deeper stacks being practical in the near future. The connections are by vias that run perpendicular to the dies. As noted earlier, the dimensions for a 3D interconnect via vary from 1 to 3 μm with a separation of 1 to 6 μm. Current commercial offerings can support 1,000,000 vias per cm^2 [26].

Table 9.2 ITRS projection [12] for 3D stacking technology, memory array cells, and maximum power budget for power aware platforms. ITRS projections suggest DRAM density exceeds SRAM density by 15–18$\times$ entailing large capacity of DRAM can be integrated on-chip using 3D stacking technology as compared to SRAM

	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
Maximum power budget for cost-performance systems (W)	104	116	119	137	137
Maximum power budget for low-cost/handheld systems with battery (W)	3.0	3.0	3.0	3.0	3.0

Table 9.3 Three-dimensional stacking technology parameters [26, 13, 44]

	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	2–3 fF	≈ 0	N/A	2.7 fF
Series resistance	$<0.35\Omega$	≈ 0	≈ 0	≈ 0

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 is 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 boost yield. Several studies including [48] show that DRAM yields are extremely high suggesting chips built with a single logic layer and several DRAM layers generate yields close to the logic die.

9.2.3 DRAM Technology

This section overviews 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 as the primary solution for system memory. FBDIMM DRAM that delivers higher throughput and higher capacity than DDR2/DDR3 is emerging as an alternative, but higher power, solution. RLDRAM and NetRAM [55, 7] 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 9.2.1.3). Industry has addressed this concern by reducing the I/O supply voltage and introducing low-power versions of the DDR2 interface both at the price of sacrificing 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.

9.3 Methodology

This section describes our methodology in evaluating the benefits of 3D stacking technology. The architectural aspects of our studies were obtained from a microarchitectural simulator called M5 [15] that is able to run Linux and evaluate full system-level performance. We model the benefits gained using 3D stacking technology in this full system simulator. We also 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. Our die area estimations are derived from previous publications and developed models for delay and power [12, 26, 50, 2, 13, 17]. DRAM timing and power values were obtained from IBM and Micron technology datasheets [3]. A detailed description of our methodology is described in the following sections.

9.3.1 Simulation Studies

9.3.1.1 Full System Architectural Simulator

To evaluate the performance of PicoServer we used the M5 full system simulator. M5 boots an unmodified Linux kernel on a configurable architecture. Multiple systems are defined 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 [53] and a chip multiprocessor-like system similar to [36]. 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 9.4 shows the configurations used in our simulations.

9.3.1.2 Server Benchmarks

We use several benchmarks that directly interact with client requests. We used two web content handling benchmarks, SURGE [14] and SPECweb99 [10], to measure web server performance. Both benchmarks request filesets of more than a 1 GB. A web script handling benchmark SPECweb2005 [9] using PHP is selected to represent script workloads. A video-streaming benchmark, Fenice [6], that 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 stressed it with dbench. Finally, we executed 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 12 KB 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 9.1). We used the default configuration for SPECweb99 to generate client requests. Seventy percent of client requests are for static web content and 30% are for dynamic web contents.

SPECweb2005 Scripting languages are a popular way to describe web pages. SPECweb2005 offers three types of benchmarks: a banking benchmark that 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 [6] that provides workloads supporting the RTSP protocol. We modified it to support multithreading. Client requests were generated with a modified version of nemesi, a 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 16 Mbps 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

Table 9.4 Commonly used simulation configurations. System memory latencies are generated from DDR2 DRAM models. We assume cores clocked at lower clock frequencies (below 1 GHz) have higher L1 cache associativity. L2 cache unloaded latency for single-core and multicore configurations differs due to longer global interconnect lengths in multicore platforms [39]

	OO4-small baseline with and w/out 3D stacking	OO4-large baseline/with and w/out 3D stacking	Conventional CMP MP4/8 w/out 3D stacking	PicoMP4/8/12-500 MHz/1000 MHz*
Operating frequency	4 GHz	4 GHz	1 GHz	500 MHz/1 GHz
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 16 KB	2 way 128 KB	4 way 16 KB per core	4 way 16 KB per core
L2 cache	8 way 256 KB 7.5 ns unloaded latency	8 way 2 MB 7.5 ns unloaded latency	8 way 2 MB 16 ns unloaded latency	N/A
Memory bus width	64 bit@400 MHz/1024 bit@250 MHz	64 bit@400 MHz/1024 bit@250 MHz	64 bit@250 MHz	1024 bit@250 MHz
System memory	512 MB DDR2 DRAM	512 MB DDR2 DRAM	512 MB DDR2 DRAM	128 MB – 512 MB DDR2 DRAM

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

the file server. dbench generates workloads that both read and write to the file server while locking these files so that a different client could not access it simultaneously.

OLTP On-line transaction processing is a typical workload executed on Tier 2 and 3 servers (behavior illustrated in Table 9.1). 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 [5]. 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 1 GB warehouse which is typically used for small-scale computation intensive databases. We chose a small working-set size due to simulation time limitations. 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) [5] 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 reduce our simulation time to a reasonable amount, we only performed and measured the time for a Q22 query out of the many TPC-H queries. Q22 query takes a modest amount of time to execute and displays the behaviors illustrated in Table 9.1.

9.3.2 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 to show general trends, we 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.

9.3.2.1 Processors

We relied to a large extent on figures reported in [20, 1, 51] 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 500 MHz from published data in [20, 1, 51]. Table 9.5 lists these estimates along with values listed in [1, 51] and a Pentium 4 core for comparison. An analysis of 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.

Table 9.5 Published power consumption values for various microprocessors [20, 1, 51, 53]

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

*Die area for a 90 nm Xscale excludes L2 cache [51]
†For the PicoServer core, we estimated our power to be in the range of an ARM11, Xscale

The power values listed in Table 9.5 include static power. Our estimates for a 500 MHz PicoServer core are conservative compared to the ARM core values, especially with respect to [51]. Given that the Xscale core consumes 850 mW at 1.5 GHz and 1.3 V, a power consumption of 190 mW at 500 MHz for the 90 nm PicoServer core is conservative when applying the 3$\times$ scaling in clock frequency and the additional opportunities to scale voltage. For power consumption at other core clock frequencies, for example 1 GHz, we generated a power vs. frequency plot. It followed a cubic law [23]. We assumed a logic depth of 24 FO4 (fan out of 4) logic gates and used the 90-nm PTM process technology [51].

Support for 64 bit 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 [56] and scaled the area and power numbers generated from actual measurements. We assumed the power numbers in [56] were generated when the cache access rate was 100%. Therefore, we scaled the L2 cache power by size and access rate while assuming leakage power would consume 30% of the total L2 cache power.

9.3.2.2 Interconnect Considering 3D Stacking Technology

For the purposes of this study, we adopted the data published in [12, 26, 50] 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 [26]. By comparison with 2D on-chip interconnect, a global interconnect wire was estimated to have capacitance of 400fF per millimeter, based on [27]. 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 [26]. 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 [27] to generate analytical and projected values. We selected a wire length of 12 mm to account for 1.3 times the width/height of a 80 mm^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 [22]. Based on these initial values, we calculated the number of repeaters required to drive the interconnect range at 250–400 MHz from hspice simulations. We found we needed only a maximum of two to three 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 interconnect power using the well-known dynamic power equation. Table 9.6 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% capacitance of an off-chip implementation. Furthermore, because the supply voltages in I/O pads – typically 1.8–2.5 V 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 modest bus frequency – 250 MHz, we conclude that inter-die interconnect power contributes very little to overall power consumption.

Table 9.6 Parasitic interconnect capacitance for on-chip 2D, 3D, and off-chip 2D for a 1024 bit bus.

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

9.3.2.3 DRAM

We made DRAM area estimates for the PicoServer using the data in [45]. Currently, it is reasonable to say that 80 mm^2 of chip area is required for 64 MB of DRAM in 90 nm 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 [3], 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 [11]. The Micron spreadsheet uses actual current measurements for each DRAM operation – read, write, refresh, bank precharge, etc. We assumed a design with a 1.8-V voltage supply.

9.3.2.4 Network Interface Controller – NIC

Network interface controller power was difficult to model analytically due to lack of information on 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). We analyzed the datasheet and found the maximum power consumed by this chip to be 743 mW [4]. This power number is for 180–nm technology. We assumed maximum power is consumed when all the input and output pins were active. We then derated this figure based on our measured usage. In addition, we assumed static power at 30% of the maximum chip power. We believe our power model is conservative considering the significant improvements made on NICs since [4].

9.4 PicoServer Architecture

Table 9.7 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 are 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% [47]. 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 [57, 7]. 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 [36]. Each PicoServer CPU core is clocked at a nominal value

Table 9.7 Bandwidth and latency suggest on-chip DRAM can easily provide enough memory bandwidth compared to an L2 cache noted in [39, 56]. 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, $t_{\mathrm{RAC\text{-}R}}$ is used where $t_{\mathrm{RAC\text{-}R}}$ denotes the read access time

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

*Average access latency with no 3D stacking aware optimizations. On-chip DRAM latency expected to reduce by more than 50% [47] when 3D stacking optimizations are applied.

of 500 MHz 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 – 16 KB instruction and 16 KB 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 9.2. 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.

9.4.1 Core Architecture and the Impact of Multithreading

PicoServer is composed 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 1 KB history table. Our studies showed the accuracy of the branch predictor for server workloads is 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, data for memory writes, or 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 4–5 mm^2 (shown in Table 9.5).

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 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 on 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 threads. We assumed a die area overhead of supporting four threads to be about 50%. Although [39] predicted a 20% die area overhead to support four threads in the Niagara core, our cores are much smaller – 5 mm^2 vs. 16 mm^2. Register and architectural state die area estimates from [20, 51] 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 9.6 and 9.7 show our simulated results. From these we are able to conclude threading indeed helps improve overall throughput, however, only to a limited extent when considering the area overhead and the impact of 3D stacking. Three-dimensional stacking reduces the access latency to memory by simplifying the core to memory interface and reducing the transfer latency. Three-dimensional stacked memory can be accessed in tens of cycles which correspond to the plots shown in Figs. 9.6b and 9.7b. The latter plot suggests that if area efficiency and throughput are taken together, limiting to only two 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 and memory bandwidth to accommodate the additional threads. Otherwise, threading will be detrimental to overall system throughput.

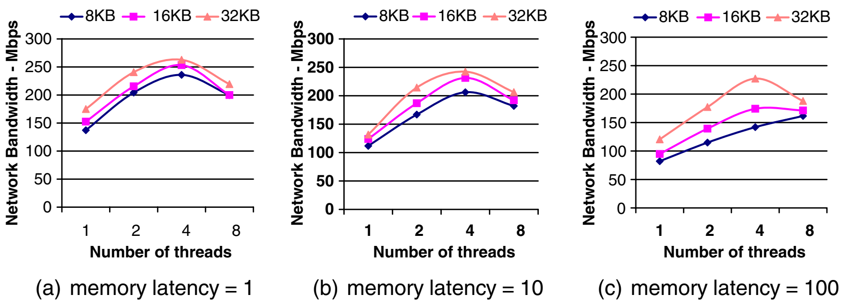


Fig. 9.6 Impact of multithreading for varying memory latency on SURGE for varying four-way set associative cache sizes (8 KB, 16 KB, 32 KB) and varying number of threads. We assume the core is clocked at 500 MHz

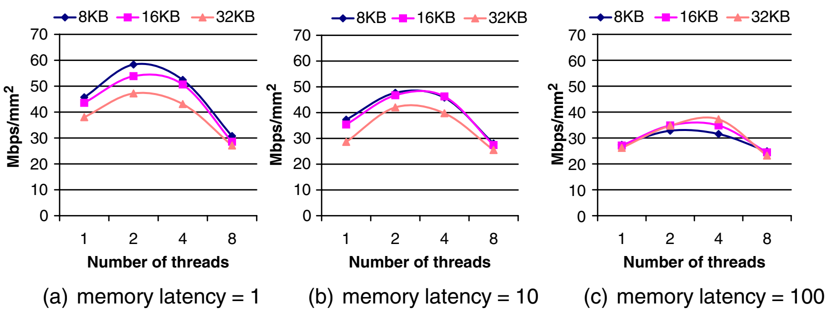


Fig. 9.7 Impact of multithreading for Mbps/mm^2 when varying memory latency on SURGE. The same setup and assumptions in Fig. 9.6 are applied

9.4.2 Wide Shared Bus Architecture

PicoServer adopts a simple 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 our tests 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 bus widths on a single-shared bus ranging from 128 to 2048 bits. We varied the cacheline size as well to match the bus width (varied it from 16 to 256 bytes). Network bandwidth (a metric for throughput) is measured to determine the impact of bus width on the PicoServer. As shown in Fig. 9.8a, 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 Fig. 9.8b, c. 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 to be copied in one transaction. A 1024-bit bus width seems reasonable for our typical PicoServer configurations of 4, 8, and 12 cores. More cores cause 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, 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 the number of bus requests.

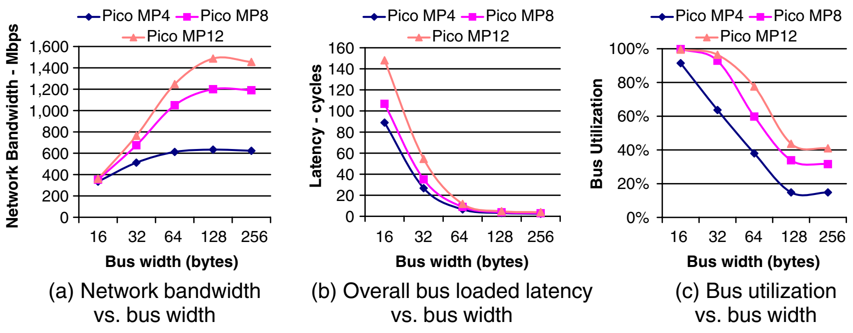


Fig. 9.8 Network performance on SURGE for various shared bus architectures with an L1 cache of 16 KB (each for I and D). We assumed a CPU clock frequency of 500 MHz for these experiments. Our bus architecture must be able to handle high bandwidths as the number of processors increases

9.4.3 On-chip DRAM Architecture

9.4.3.1 Role of On-chip DRAM

Based on the logic die area estimates, we projected the DRAM die size for a 4-core, 8-core, and 12-core PicoServers to be 40, 60, and 80 mm^2, respectively. Table 9.8 shows the on-chip memory alternatives for PicoServers. For example, to obtain a total DRAM size of 256 MB, we assume DRAM is made up of a stack of four layers. For Tier 3 servers we employ eight layers because they rely heavily on system memory size. With current technology – 90 nm, it is feasible to create a four-layer stack containing 256 MB of physical memory for a die area of 80 mm^2. Although a large amount of physical memory is common in server farms (4–16 GB) 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 data center experts [41]. From our measurements on memory usage for server applications shown in Fig. 9.9, we found for many of the server applications (except TPC-C and TPC-H) that a modest amount – around 64 MB – 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 256 MB can be integrated on-chip for four 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 256 MB 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 eight-layer case. From the ITRS roadmap in Table 9.2, 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 layer is quite small – less than 5 W/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

Table 9.8 Projected on-chip DRAM size for varying process technologies. Area estimates are generated based on Semiconductor SourceInsight 2005 [45]. Die size of 80 mm^2 is similar to that of a Pentium M at 90 nm

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

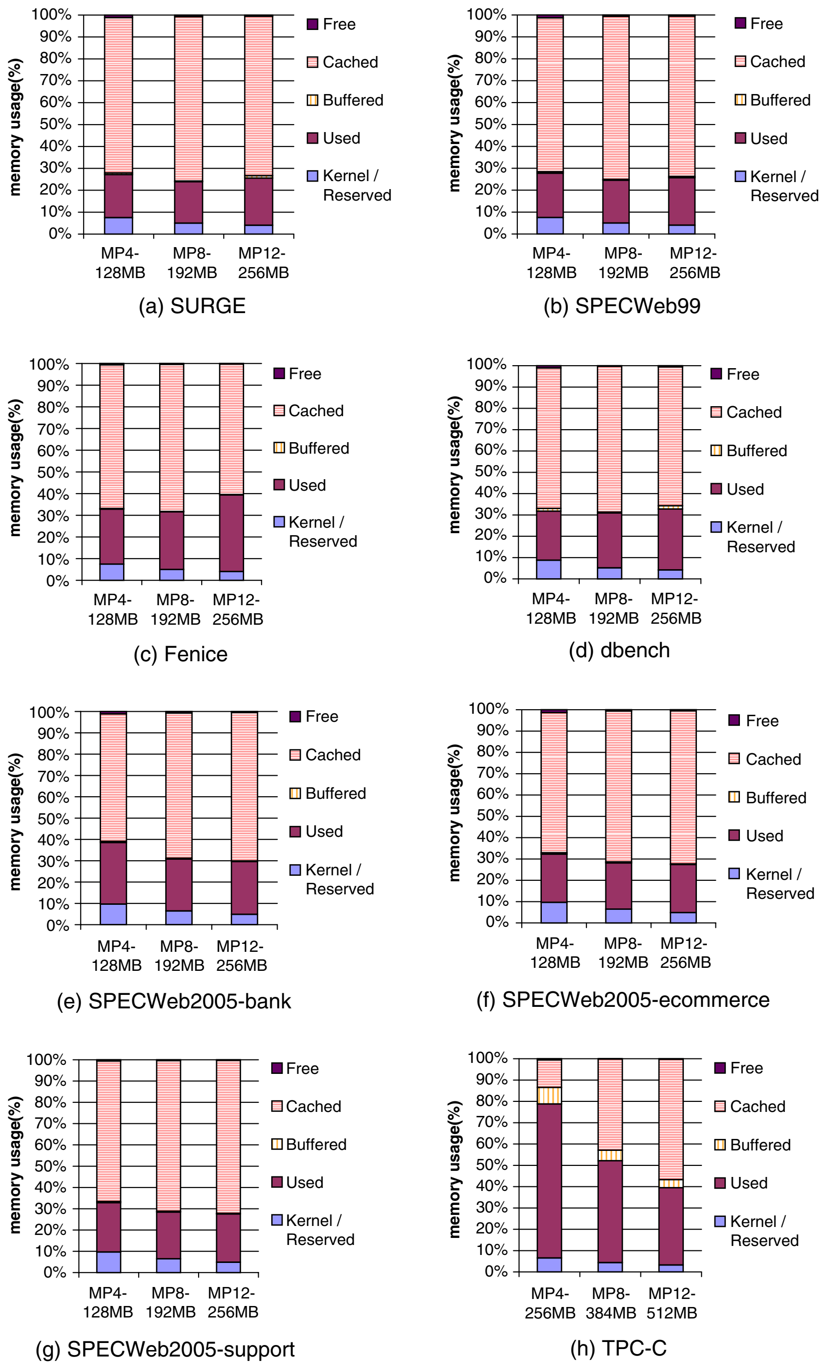


Fig. 9.9 Breakdown in memory for server benchmarks (SURGE, SPECWeb99, Fenice, dbench, SPECWeb2005, TPC-C); TPC-H is excluded because it displayed memory usage similar to TPC-C

slow as hundreds of micro seconds without affecting throughput. An access latency as slow as hundreds of micro seconds implies that Flash memory that consumes less active/standby power can be used as secondary system memory – shown in Section 9.4.6. This idea has been explored in [30, 31]. 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 while the slow system memory would function as a large disk cache device.

9.4.3.2 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 address/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 9.10 shows an example of a read operation without multiplexing. In particular, it 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 [7] and NetDRAM [55] from Samsung. This suggests the interface required for 3D

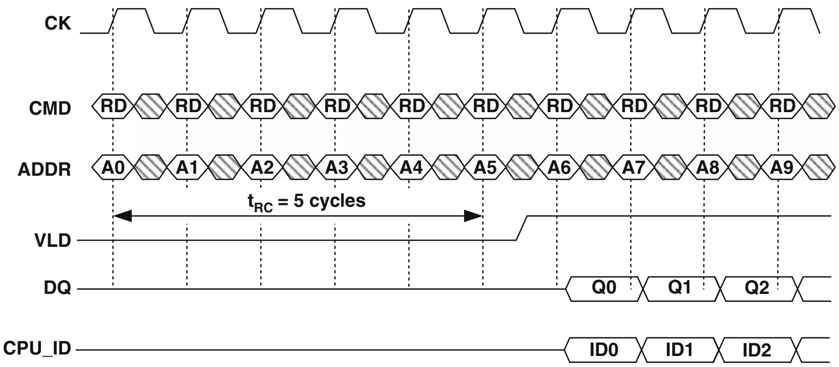


Fig. 9.10 On-chip DRAM read timing diagram without address multiplexing

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.[3]

9.4.3.3 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 9.4.5, 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, thus reducing the average DRAM refresh interval to approximately 7.8125 μs and requiring approximately 200 ns to complete. Roughly speaking, this implies 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 500 MHz (PicoMP12-500 MHz running SURGE), this would incur a CPI refresh overhead of 0.03 CPI. This is because many of the L1 cache misses do not coincide with a refresh command, executed resulting in only a minimal performance penalty.

9.4.4 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 with 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 the 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 eight-chip multiprocessor architecture with one 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 one NIC for every two processors.

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

For example, a four-processor architecture would have two NICs, an eight-processor architecture would have four 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 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 [21]. TCP splicing and handoff are also good examples of intelligent load balancing at higher network layers [46].

9.4.5 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 [28]. 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 our work is aimed at answering whether 3D stacking can provide an advantage in 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 9.9. 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 27°C. Hotspot requires information on properties of the material and power density to generate steady-state temperatures. We extracted 3D stacking properties from [37, 44, 57]

Table 9.9 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 25°C	0.026	1.2×10^3

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 shown in Fig. 9.11.

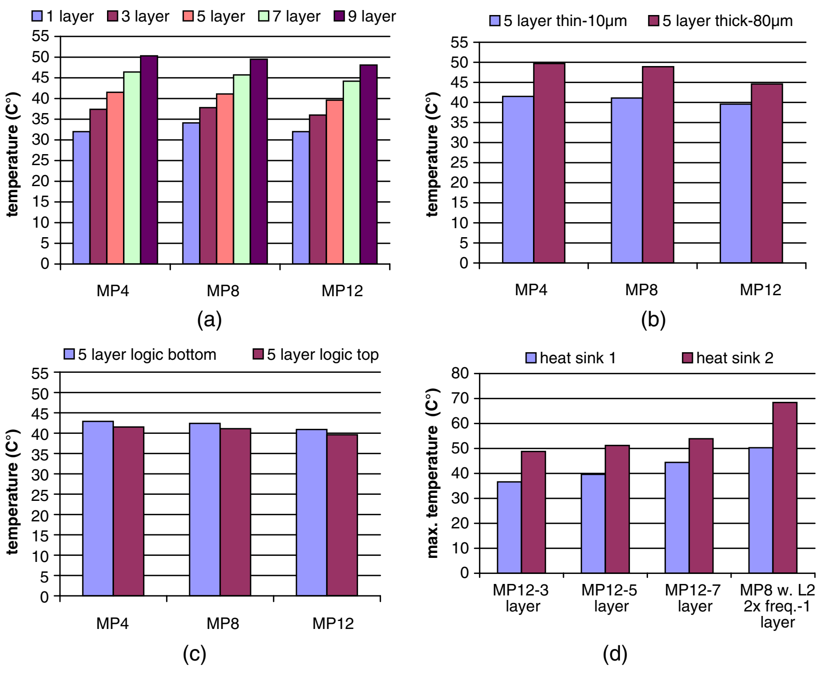


Fig. 9.11 Maximum junction temperature for sensitivity experiments on Hotspot: (**a**) varying the number of layers; (**b**) varying 3D interface thickness; (**c**) varying location of logic die; and (**d**) maximum junction temperature for heat sink quality analysis. A core clock frequency of 500 MHz is assumed in calculating power density. We varied the size of on-chip memory based on the number of layers stacked. One layer assumes no on-chip memory at all

Figure 9.11a shows the sensitivity to the number of stacked layers. We find roughly a 2–3°C increase in maximum junction temperature for each additional 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 9.11b shows the sensitivity to the 3D stacking dielectric interface. We compared the effect of the SiO_2 thickness (the interface material) for 10 and 80 µm. In [17, 37, 44, 57] we find the maximum thickness of the interface material does not exceed 10 µ m for 3D stacking. The 80 µm point is selected to show the impact of heat containment as the thickness is increased substantially. It results in a 6°C 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 9.11c 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 of 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 9.11d 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 above results suggest that heat containment is not a major limitation for the PicoServer architecture. The power density is relatively low. It does not exceed 5 W/cm^2. As a result, the maximum junction temperature does not exceed 50°C. Three-dimensional 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 [19, 25].

9.4.6 Impact of Integrating Flash onto PicoServer

This section examines the architectural impact of directly attaching NAND Flash devices onto a PicoServer architecture, and it is also a case study on 3D-stacked Flash devices integrated onto PicoServer. Flash is emerging as an attractive memory device to integrate onto a server platform primarily due to the rapid rate of density improvement. There are two primary usage models for Flash in the server space: (1) a solid state disk (SSD) and (2) a memory device. It is widely believed that Flash integration improves overall server throughput while consuming lower system memory and disk drive power. For example, when Flash is used as a memory device and assigned the role of a disk cache, the higher density of Flash allows us to implement a larger cache with a higher cache hit rate that consumes lower power than DRAM. Higher cache hit rates reduce accesses to disk which results in improved system performance and reduction in disk power.

However, integrating Flash onto a server platform is not straightforward. There are two key challenges that Flash needs to address to successfully integrate onto a server platform; (1) improving transfer latency to/from Flash and (2) providing sufficient memory (RAM) to efficiently manage Flash. NAND Flash displays higher overall access latency (see Table 9.10) than typical memory devices such as DRAM primarily due to the high transfer latency (low-bandwidth narrow 8- or 16-bit interface) for typical off-the-shelf Flash devices. Although the page read latency into the internal buffer for a SLC NAND Flash page is approximately 25 μs, the transfer latency to read several KBs from a NAND Flash chip is substantially higher. One way to reduce transfer latency is leveraging 3D stacking. It enables the use of a wider bus that accesses a larger amount of data per cycle thus reducing transfer latency.

Table 9.10 ITRS 2007 roadmap for memory technology

	2007	2009	2011	2013	2015
NAND Flash-SLC* (μm^2/bit)	0.0130	0.0081	0.0052	0.0031	0.0021
NAND Flash-MLC* (μm^2/bit)	0.0065	0.0041	0.0013	0.0008	0.0005
DRAM cell density(μm^2/bit)	0.0324	0.0153	0.0096	0.0061	0.0038
Flash write/erase cycles – SLC/MLC†	1E+05/ 1E+04	1E+05/ 1E+04	1E+06/ 1E+04	1E+06/ 1E+04	1E+06/ 1E+04
Flash data retention (years)	10–20	10–20	10–20	20	20

*SLC – single-level cell, MLC – multi-level cell
†write/erase cycles for MLC Flash estimated from prior work [34]

The latency reduction may allow more critical data to be moved from DRAM onto Flash for greater energy savings.

Additionally, the amount of memory (RAM) required to efficiently manage a NAND Flash subsystem scales with capacity. While NAND Flash may still be managed with a small amount of memory, these types of subsystems display a limited read/write bandwidth and accelerate Flash wear out. To meet the memory (RAM) requirements that deliver a high bandwidth, high longevity, and high-capacity NAND Flash subsystem, DRAM is commonly integrated onto the NAND Flash subsystem that stores the Flash manageability code and data. However, the cost of implementing a dedicated DRAM device that is required in a NAND Flash subsystem such as a SSD is appreciable and inefficient. Consolidating the entire code and data available in a server platform into a single DRAM module saves cost and improves efficiency. The system integration benefit of 3D stacking allows Flash manageability code and data to reside in DRAM that is shared with other components in the system (shown in Fig. 9.12).

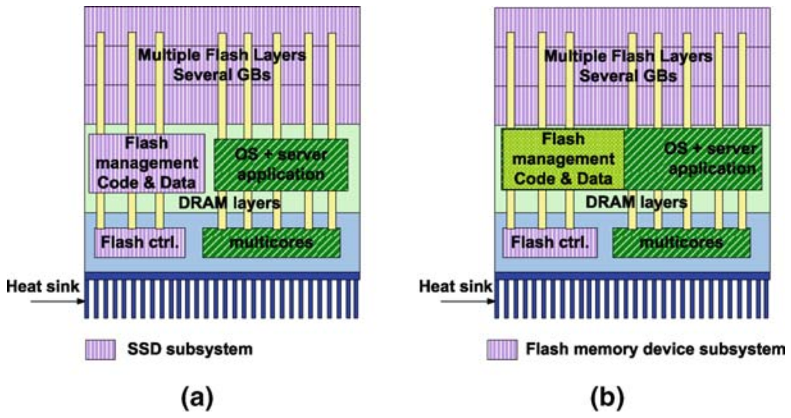


Fig. 9.12 Three-dimensional stacked Flash architecture for (**a**) SSD usage model and (**b**) memory device usage model

For the solid state disk (SSD) usage model, 3D stacking can be used as a way to implement energy efficient on-chip interfaces that can replace the conventional hard disk drive interface. Figure 9.12a shows this approach. Logic for the SSD controller is placed on a die using the same process technology as the CPU cores. The low-level details of Flash management (error checking, wear leveling, and buffer management) are isolated from the processors, providing a simple interface. The memory requirements for Flash manageability are easily satisfied with the integrated on-chip DRAM. A 3D-stacked SSD could provide: (1) lower power, (2) higher random-access throughput, (3) lower latency, and (4) greater physical robustness than disk drives. SSDs consist of a controller and buffer RAM in addition to Flash. In a 3D-stacked PicoServer, exposing an SSD interface in hardware could allow drivers to work without modification. Other benefits over external SSDs include a higher bandwidth interface, the option of data transfer directly into main memory or processor cache, and the ability to allow the PicoServer cores to control the Flash using application-specific algorithms. It has been shown that using a combination of SSD and conventional disk-based storage together can provide higher performance due to the different characteristics of each device [35]. Disk is most effective for high-bandwidth, high-density storage while Flash provides lower latency (especially when reading) and more IOPs (I/O per second). Algorithms dynamically place read-dominated data on SSD while write-dominated data is moved to the HDD for an overall performance improvement.

For the memory device usage model, there have been several proposals that span a wide range of application spaces, including disk buffers [33], disk caches [29, 30, 31], and code/data storage [42, 49]. Figure 9.12b shows how the physical structure is very similar to that of a 3D-stacked SSD except that low-level control of data transfers between Flash is given to the processor cores. This approach increases software complexity, however, achieves higher bandwidth and more efficient management than the SSD usage model. This is because Flash manageability is performed by the processor that has more computation capacity than a Flash controller. When Flash is used as a disk buffer, it is used as a staging buffer between DRAM and disk. Energy savings can be achieved by spinning down the disk for longer periods of time while data drains from the Flash buffer. This scheme also reduces the amount of DRAM in the system and saves 30–40% of the system energy. Similarly, Flash disk caches are simple to implement in an operating system, extending the standard DRAM-based disk cache. Data-intensive server applications such as web servers require large disk caches. By replacing some DRAM with Flash, there are energy savings due to lower idle power and performance gains from having more total cache capacity (due to higher Flash density). Three-dimensional stacking enhances this scheme by satisfying requests to cached data more quickly. Cached pages migrate from secondary Flash storage to the primary page cache in DRAM, so a wide 3D-stacked interconnect completes these data transfers faster and with lower bus energy. There are additional benefits the memory device usage model provides. With small enhancements in the microarchitecture of the Flash controller, code that typically is loaded from NAND Flash and residing in DRAM may directly reside in NAND Flash. The next code block prediction technique described in [42] and "demand-based paging" architecture outlined in [49] have shown the potential benefits. Three dimensional stacking helps this scheme by reducing the energy cost and delay while swapping pages between DRAM and Flash. For the same DRAM and Flash capacities, total energy will be reduced further because program execution completes more quickly, burning less idle energy in DRAM and other system components such as the power supply.

Based on the findings from [29, 30, 31], we present a case study of 3D stacked Flash as a memory device integrated onto a PicoServer architecture.

We first provide an analysis of the OS-managed disk cache behavior running a server workload. Figure 9.13 shows the disk cache access behavior in system memory in a web server. It shows that file access behavior in server workloads displays a short-tailed distribution where a large portion of the disk cache is infrequently accessed and only a small portion of the disk cache is frequently accessed. Further, Fig. 9.14 shows server throughput for varying access latencies to the infrequently accessed files in the disk cache. We are able to observe constant throughput for access latencies of tens to hundreds of microseconds. This is because we can hide

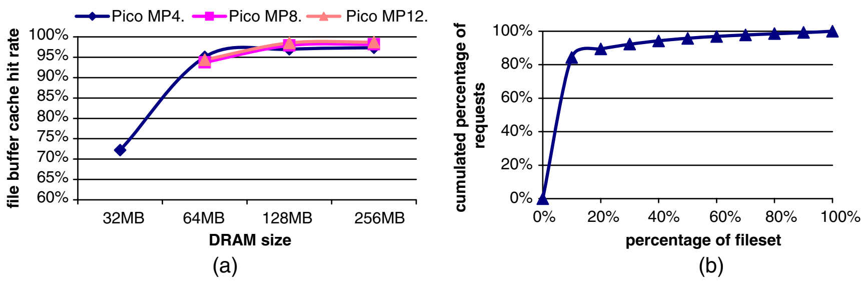


Fig. 9.13 (a) Disk cache access behavior on the server side for client requests. We measured for 4, 8, 12 PicoServer configurations and varied the DRAM size. (b) A typical cumulative distribution function of a client request behavior. Ninety percent of requests are for 20% of the web content files.

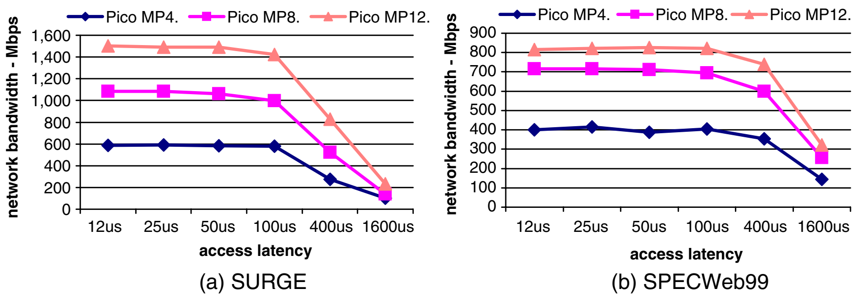


Fig. 9.14 Measured network bandwidth for full system simulation while varying access latency to a secondary disk cache. We assumed a 128 MB DRAM with a slower memory of 1 GB. We measured bandwidth for 4, 8, 12 PicoServer 500 MHz configurations. The secondary disk cache can tolerate access latencies of hundreds of microseconds while providing equal network bandwidth

the lengthy access latency to infrequently accessed files in the disk cache through the multithreaded nature of a server workload. One way to take advantage of this behavior is to integrate Flash as a second-level disk cache that replaces a large portion of DRAM allocated for a disk cache.

Because NAND Flash consumes much less power than DRAM and is more than 4× denser than DRAM (shown in Tables 9.10 and 9.11), a server that integrates Flash in its memory system is expected to be more energy efficient and has larger system memory capacity. Bigger disk cache sizes reduce disk cache miss rate and allow the HDD to be spun down longer. As we show in Table 9.11, HDDs in idle mode consume a significant amount of power.

Table 9.11 Performance, power consumption and cost for DRAM, NAND-based SLC/MLC Flash and HDD.

	Active power	Idle power	Read latency	Write latency	Erase latency
1 Gb DDR2 DRAM	878 mW	80 mW†	55 ns	55 ns	N/A
1 Gb NAND-SLC	27 mW	6 μ W	25 μs	200 μs	1.5 ms
4 Gb NAND-MLC	N/A	N/A	50 μs	680 μs	3.3 ms
HDD‡	13.0 W	9.3 W	8.5 ms	9.5 ms	N/A

† DRAM idle power in active mode. Idle power in powerdown mode is 18 mW
‡ Data for 750 GB hard disk drive [8]

The benefits that are achieved by integrating Flash onto a server can also be applied to PicoServer. Figures 9.15 and 9.16 show two such configurations that integrate Flash onto PicoServer using 3D stacking technology. The first configuration shown in Figs. 9.15a and 9.16a integrates Flash as a discrete component. It stacks eight die layers of Flash using 3D stacking technology to create a separate Flash chip – a discrete component. This Flash chip is connected to PicoServer through a PCB board layout. An off-chip I/O interface (typically PCI express) that is capable of delivering bandwidth of hundreds of megabytes per second is implemented between PicoServer and the discrete Flash chip. Flash is managed (wear-leveling and Flash command interface) by the OS running on the in-order PicoServer cores. The memory capacity of the discrete Flash chip is not constrained by the die area of PicoServer and allows us to stack more dies than PicoServer due to the lower power consumption of Flash. Therefore, discrete Flash chips with tens of gigabytes can be integrated onto a PicoServer architecture. Server workloads with large filesets and modest I/O bandwidth requirements will benefit most from this configuration. One potential drawback to the discrete Flash chip configuration is the idle power consumption of the PCI express interface. Idle power management must be performed on the PCI express I/O pins when they are not active.

The second configuration shown in Figs. 9.15b and 9.16b directly integrates Flash to PicoServer (discussed earlier in the section). It stacks four additional Flash die layers directly on top of the on-chip DRAM using 3D stacking technology. The Flash dies connect to other components in PicoServer through the wide shared bus.

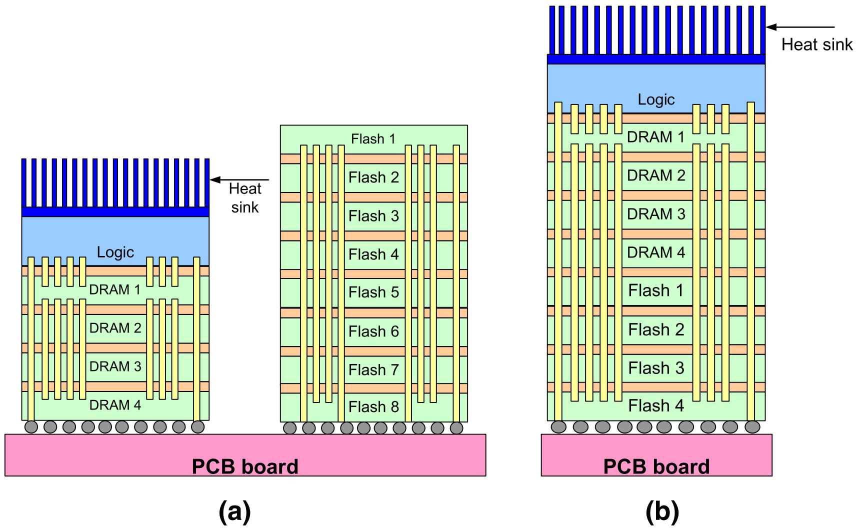


Fig. 9.15 (**a**) Eight-die-stacked Flash integrated as discrete component (PCB layout) in PicoServer (**b**) four-die-stacked Flash directly integrated to PicoServer

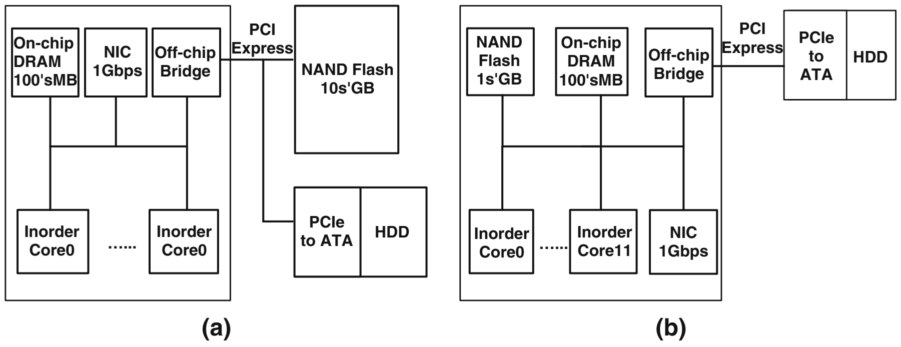


Fig. 9.16 (**a**) High-level block diagram of eight-die-stacked Flash integrated as discrete component (PCB layout) in PicoServer, (**b**) high-level block diagram of four-die-stacked Flash directly integrated to PicoServer

The wide on-chip shared bus interface delivers tens of gigabytes per second bandwidth. Flash is also managed – wear-leveling, Flash command interface – by the OS running on top of the in-order PicoServer cores. Flash capacity is limited by the die area of the on-chip DRAM and logic components in PicoServer. As a result, the Flash capacity is expected to be several gigabytes in size. We expect server workloads that require small filesets and high I/O bandwidth will benefit most from this configuration.

9.5 Results

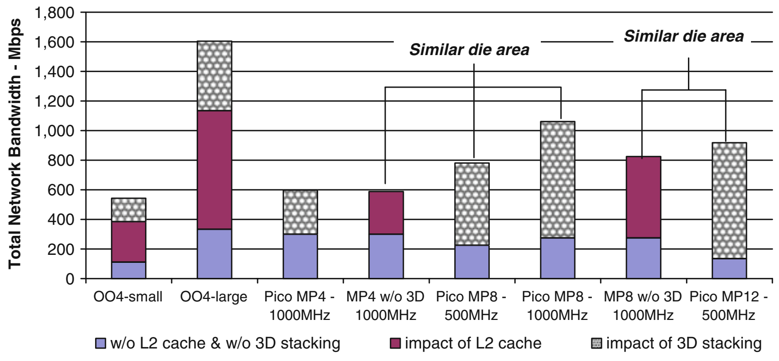
To evaluate the PicoServer architecture two metrics are important – throughput and power. Throughput that can be measured as network bandwidth or transactions per seconds is a good indicator of overall system performance because it is a measure of how many requests were serviced. In this section, we 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.

9.5.1 Overall Performance

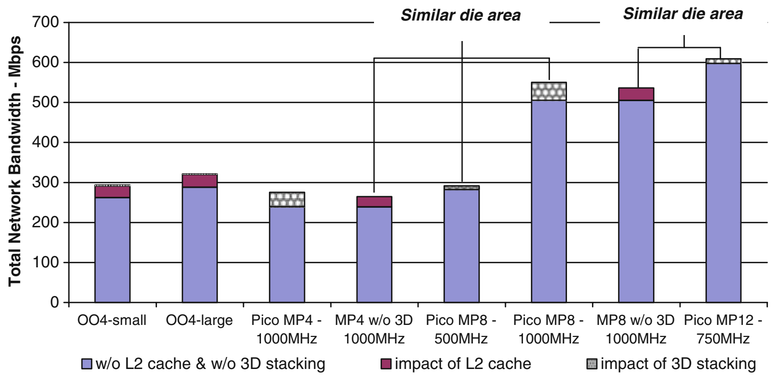
Figures 9.17 and 9.18 show the throughput for some of our Tier 1–3 workload runs. Each bar shows the contribution to throughput in three parts: (1) a baseline with no L2 cache and a narrow (64 bit) bus; (2) the baseline but with an L2 cache; or (3) the baseline with wide bus, no L2 cache, and 3D stacking 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 9.17 shows that 3D stacking technology alone improves overall performance equal to or more than having an L2 cache. A fair comparison for a fixed number of cores, for example, would be a Pico MP4-1000 MHz vs. a conventional CMP MP4 without 3D-1000 MHz. 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 the OO4 configurations that have been 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 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 500 MHz performs better than a 4 GHz OO4 processor with a small L1 and L2 cache of 16 KB and 256 KB, 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 requires additional die area for a 128 KB L1 cache and a 2 MB L2 cache.

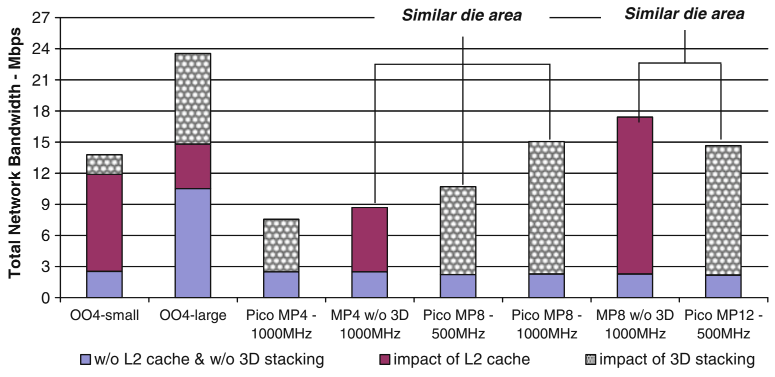
If we assume that the area occupied by the L2 cache in our conventional CMP MP4/8 without 3D stacking technology is replaced with additional processing cores – a benefit made possible by using 3D stacking technology – a comparison in



(a) SPECweb99



(b) Fenice



(c) SPECweb2005-bank

Fig. 9.17 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 250 MHz. For a Pentium 4-like configuration, we placed the NIC on the PCI bus and assumed the memory bus frequency to be 400 MHz. For a MP4, MP8 without 3D stacking configuration, to be fair we assumed no support for multithreading and an L2 cache size of 2 MB. The external memory bus frequency was assumed to be 250 MHz (SPECweb99, Fenice, SPECweb2005-bank)

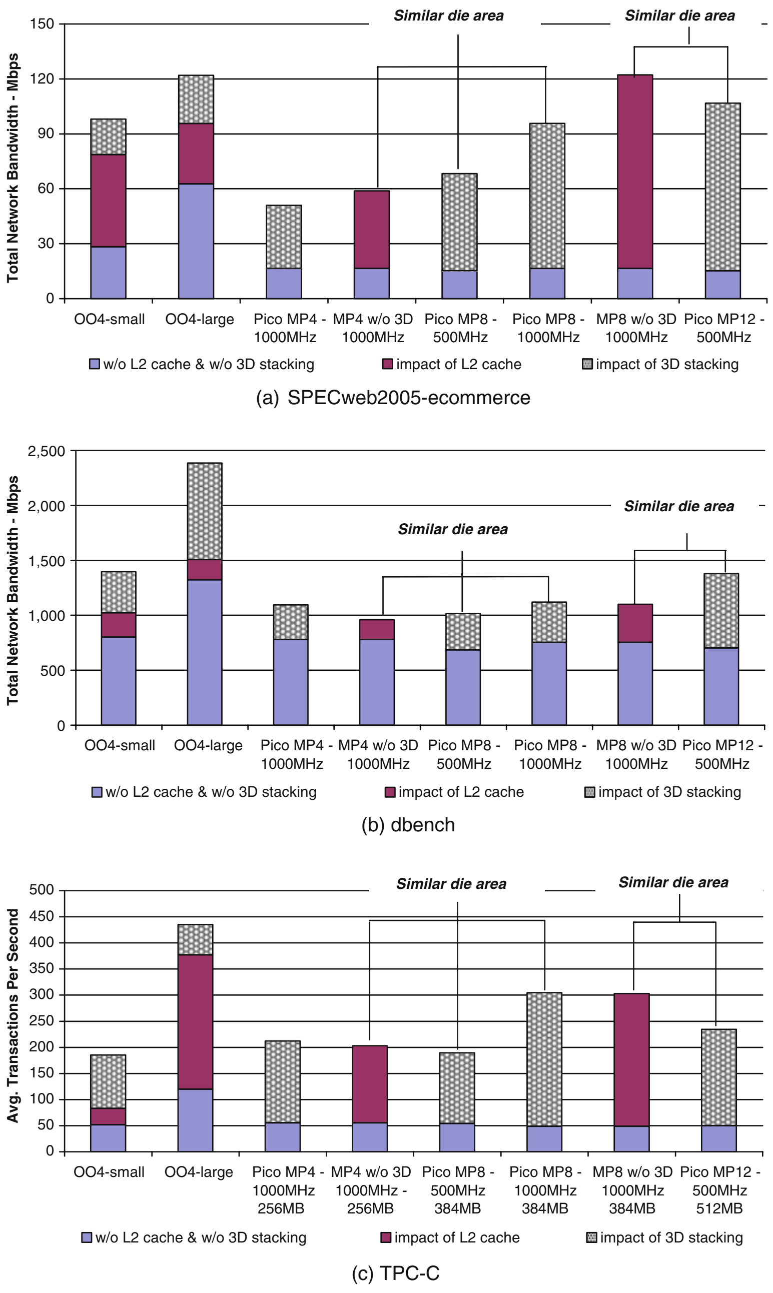


Fig. 9.18 Throughput measured for varying processor frequency and processor type (SPECweb2005-ecommerce, dbench, TPC-C). We applied the same assumptions used in Fig. 9.17

throughput for similar die area can be conducted on the following systems: (1) a Pico MP8-500 MHz vs. a conventional MP4 without 3D-1000 MHz and (2) a Pico MP12-500 MHz vs. a conventional MP8 without 3D-1000 MHz (for Fenice, compare with a Pico MP12-750 MHz). Our results suggest that on average, additional processing elements and reducing core clock frequency by half improve throughput and significantly saves on power – shown in Section 9.5.2. For compute-bound workloads like Fenice, SPECWeb2005-bank and SPECWeb2005-ecommerce, however Pico MP12-500 MHz did not do better than a conventional MP8 without 3D-1000 MHz. For SPECWeb2005-bank and ecommerce, introducing a 2 MB L2 cache dramatically cuts the number of cache misses reducing the benefit of adding more cores while lowering core clock frequency. Pico MP12-500 MHz also did not perform well for TPC-C because of the I/O scheduler. However, we expect Pico MP12-500 MHz 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.

9.5.2 Overall Power

Processor power still dominates overall power in PicoServer architectures. Figure 9.19 shows the average power consumption based on our power estimation techniques for server application runs. We find that PicoServer with a core clock frequency of 500 MHz is estimated to consume between 2 and 3 Watts for 90 nm process technology. Much of the total power is consumed by the simple in-order

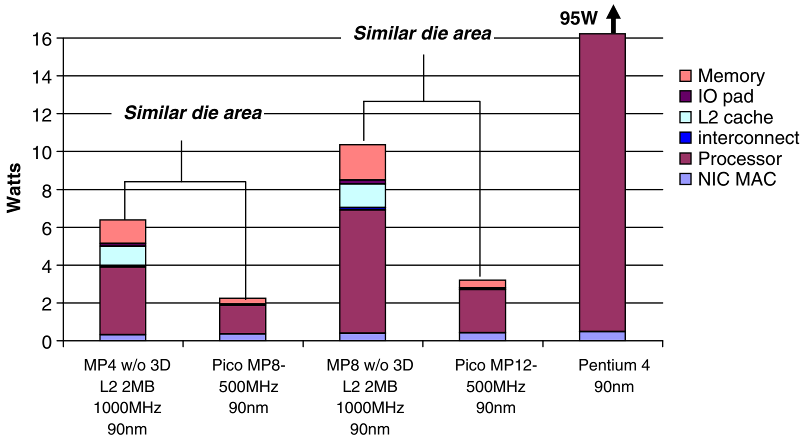


Fig. 9.19 Breakdown of average power for 4, 8, and 12 processor PicoServer architectures using 3D stacking technology for 90 nm process technology. Estimated power per workload does not vary by a lot because the cores contribute to a significant portion of power. We expect 2–3 W to be consumed at 90 nm. An MP8 without 3D stacking operating at 1 GHz is estimated to consume 8 W at 90 nm

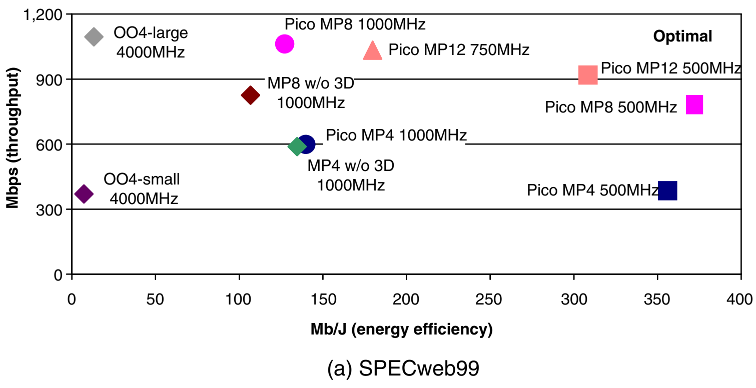
cores. NIC power also consumes a significant amount due to the increase in the number of NICs when increasing the number of processors. However, as described in Section 9.4.4, an intelligent NIC designed for this architecture could be made more power efficient as a more advanced design would only need 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.

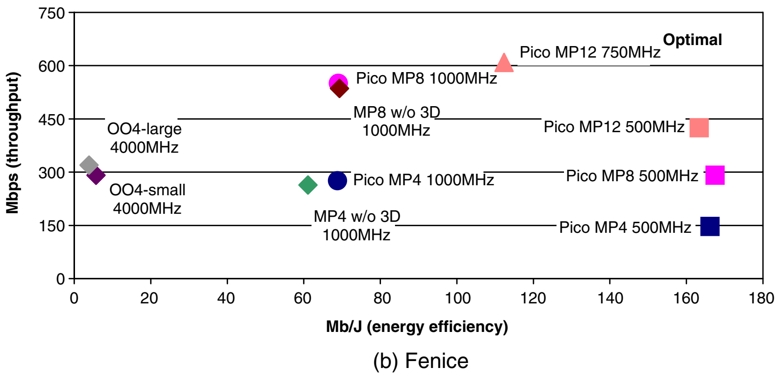
Comparing our PicoServer architecture with other architectures, we see that for a similar die area comparison, we use less than half the power when we compare Pico MP8/12-500 MHz with a conventional MP4/8 without 3D stacking but with an L2 cache at 1000 MHz. We also recall in Section 9.5.1 that performance-wise 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 and, as we showed in the previous section, perform comparably. At 90-nm technology, it can be projected that the power budget for a typical PicoServer platform satisfies mobile/handheld power constraints noted in ITRS projections. This suggests the potential for implementing server-type applications in ultra-small form factor platforms

9.5.3 Energy Efficiency Pareto Chart

In Figs. 9.20 and 9.21, 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, the 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 Figs. 9.20 and 9.21, it is possible to find the optimal configuration of processor number and frequency for a given energy efficiency/throughput constraint.

Additionally from Figs. 9.20 and 9.21, we find the PicoServer architectures clocked at modest core frequency – 500 MHz are 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 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 MP4-500 MHz, MP8-500 MHz, and MP12-500 MHz. 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 return. Independent studies





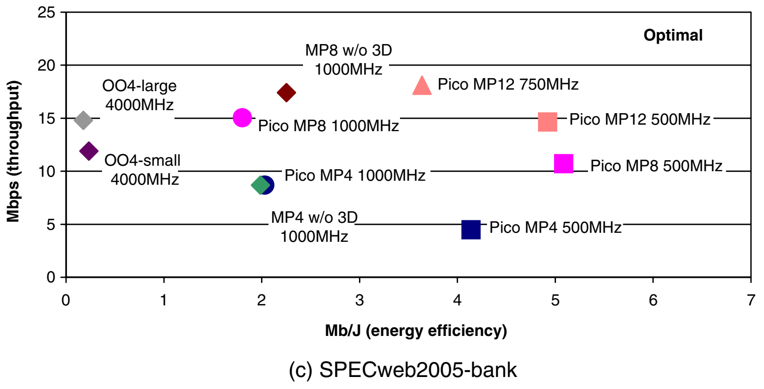


Fig. 9.20 Energy efficiency, performance pareto chart generated for 90-nm process technology. Three-dimensional stacking technology enables new CMP architectures that are significantly energy efficient. (SPECWeb99, Fenice, SPECweb2005-bank)

have shown the OS can be tuned to scale with many cores. The works [18] and [54] are examples of such an implementation. However, we feel further investigation is necessary and leave such work for future research.

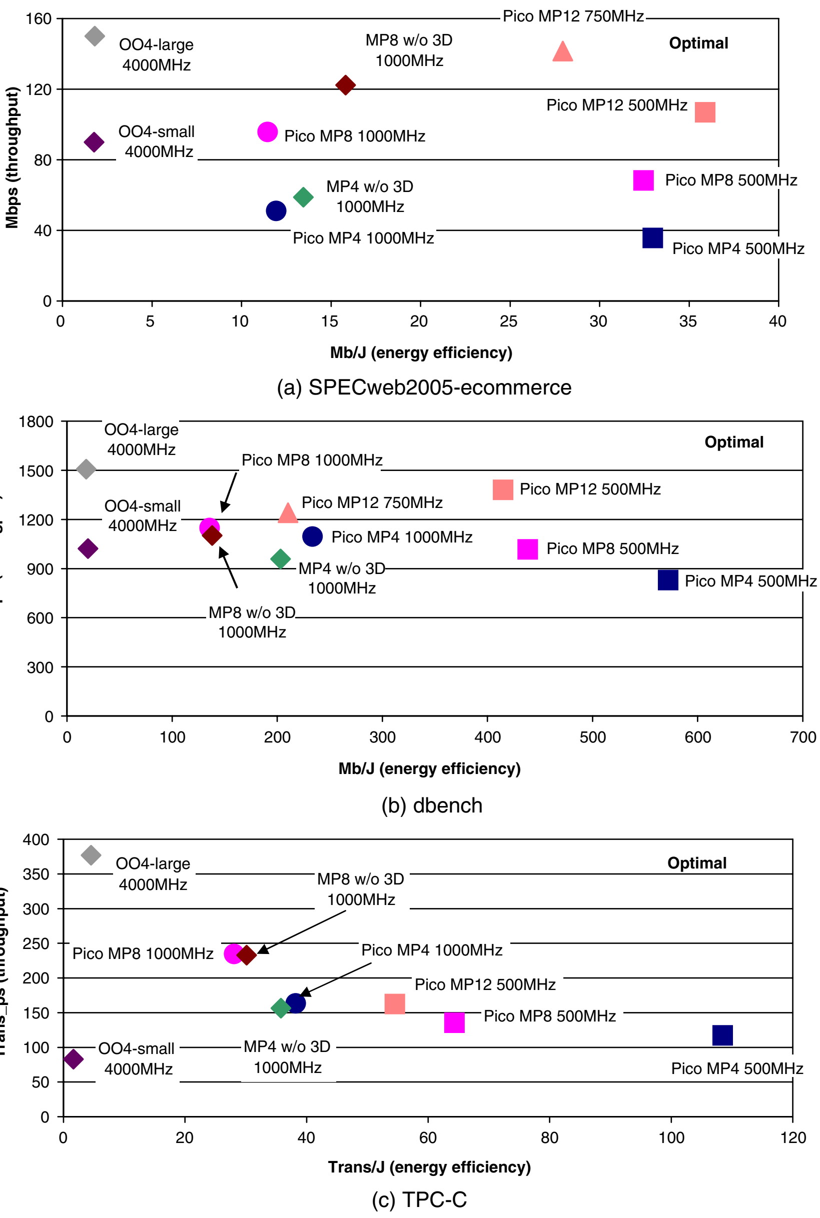


Fig. 9.21 Energy efficiency, performance pareto chart generated for 90-nm process technology. Three-dimensional stacking technology enables new CMP architectures that are significantly energy efficient. (SPECweb2005-ecommerce, dbench, TPC-C)

9.6 Conclusions

Throughout this chapter, we showed that 3D stacking technology can be leveraged to build energy efficient servers. For a wide range of server workloads, the resulting systems have significant energy efficiency in a compact form factor. A 12-way PicoServer running at 500 MHz can deliver 1 Gbps of network bandwidth within a 3 W power budget using a 90-nm process technology. These power results are two to three times 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. Compared to a conventional 8-way 1 GHz chip multiprocessor with a 2 MB L2 cache, an area-equivalent 12-way PicoServer running at 500 MHz yields an improvement in energy efficiency of more than 2×. The absolute power values are also expected to scale with process technology. We expect to see additional cores and even lower power for 65-nm and 45-nm process technology-based PicoServer platforms.

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. Three dimensional 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.

Acknowledgments This work is supported in part by the National Science Foundation, Intel, and ARM Ltd.

References

1. ARM 11 MPcore. http://www.arm.com/products/CPUs/ARM11MPCoreMultiprocessor.html.
2. FaStack 3D RISC super-8051 microcontroller. http://www.tachyonsemi.com/OtherICs/datasheets/TSCR8051Lx_1_5Web.pdf.
3. The Micron system-power calculator. http://www.micron.com/products/dram/syscalc.html.
4. National semiconductor DP83820 10 / 100 / 1000 Mb/s PCI ethernet network interface controller.
5. OSDL DataBase Test Suite. http://www.osdl.net/lab_activities/kernel_testing/osdl_database_test_suite/.
6. (LS)^3-libre streaming, libre software, libre standards an open multimedia streaming project. http://streaming.polito.it/.
7. RLDRAM memory. http://www.micron.com/products/dram/rldram/.
8. Seagate Barracuda. http://www.seagate.com/products/personal/index.html.
9. SPECweb2005 benchmark. http://www.spec.org/web2005/.
10. SPECweb99 benchmark. http://www.spec.org/osg/web99/.
11. Sun Fire T2000 Server Power Calculator. http://www.sun.com/servers/coolthreads/t2000/calc/index.jsp.
12. ITRS roadmap. Technical report, 2005.

13. K. Banerjee, S. J. Souri, P. Kapur, and K. C. Saraswat. 3-D ICs: A novel chip design for improving deep-submicrometer interconnect performance and systems-on-chip integration. *Proceedings of the IEEE*, 89(5):602–633, May 2001.
14. P. Barford and M. Crovella. Generating representative web workloads for network and server performance evaluation. In *Measurement and Modeling of Computer Systems*, pp. 151–160, 1998.
15. N. L. Binkert, R. G. Dreslinski, L. R. Hsu, K. T. Lim, A. G. Saidi, and S. K. Reinhardt. The M5 simulator: Modeling networked systems. *IEEE Micro*, 26(4):52–60, Jul/Aug 2006.
16. B. Black, M. Annavaram, N. Brekelbaum, J. DeVale, L. Jiang, G. H. Loh, D. McCaule, P. Morrow, D. W. Nelson, D. Pantuso, P. Reed, J. Rupley, S. Shankar, J. P. Shen, and C. Webb. Die stacking (3D) microarchitecture. In *International Symposium on Microarchitecture*, December 2006.
17. B. Black, D. Nelson, C. Webb, and N. Samra. 3D processing technology and its impact on iA32 microprocessors. In *Proceedings of International Conference on Computer Design*, pp. 316–318, 2004.
18. R. Bryant, J. Hawkes, J. Steiner, J. Barnes, and J. Higdon. Scaling Linux to the Extreme From 64 to 512 Processors. In *Linux Symposium*, July 2004.
19. T.-Y. Chiang, S. J. Souri, C. O. Chui, and K. C. Saraswat. Thermal analysis of heterogeneous 3-D ICs with various integration scenario. In *IEDM Technical Digest*, pp. 681–684, December 2001.
20. L. T. Clark, E. J. Hoffman, J. Miller, M. Biyani, Y. Liao, S. Strazdus, M. Morrow, K. E. Verlarde, and M. A. Yarch. An embedded 32-b microprocessor core for low-power and high-performance applications. *IEEE Journal of Solid State Circuits*, 36(11):1599–1608, November 2001.
21. E. L. Congduc. Packet classification in the NIC for improved SMP-based internet servers. In *Proceedings of International Conference on Networking*, February 2004.
22. W. R. Davis, J. Wilson, S. Mick, J. Xu, H. Hua, C. Mineo, A. M. Sule, M. Steer, and P. D. Franzon. Demystifying 3D ICs: The pros and cons of going vertical. *IEEE Design & Test of Computers*, 22(6):498–510, 2005.
23. M. J. Flynn and P. Hung. Computer architecture and technology: Some thoughts on the road ahead. In *Proceedings on International Conference on Engineering of Reconfigurable Systems and Algorithms*, pp. 3–16, 2004.
24. M. Ghosh and H.-H. S. Lee. Smart Refresh: An Enhanced Memory Controller Design for Reducing Energy in Converntional and 3D Die-Stacked DRAMs. In *International Symposuim on Microarchitecture*, December 2007.
25. B. Goplen and S. S. Sapatnekar. Thermal via placement in 3D ICs. In *Proceedings of International Symposium on Physical Design*, pp. 167–174, April 2005.
26. S. Gupta, M. Hilbert, S. Hong, and R. Patti. Techniques for producing 3D ICs with high-density interconnect. www.tezzaron.com/about/papers/ieee_vmic_2004_finalsecure.pdf.
27. R. Ho and M. Horowitz. The future of wires. *Proceedings of the IEEE*, 89(4), April 2001.
28. W. Huang, M. R. Stan, K. Skadron, K. Sankaranarayanan, S. Ghosh, and S. Velusam. Compact thermal modeling for temperature-aware design. In *Proceedings Design Automation Conference*, June 2004.
29. T. Kgil. *Architecting Energy Efficient Servers*. PhD thesis, University of Michigan, 2007.
30. T. Kgil and T. Mudge. FlashCache: a NAND flash memory file cache for low power web servers. In *Proceedings of International Conference on Compilers, Architecture and Synthesis for Embedded Systems*, October 2006.
31. T. Kgil, D. Roberts, and T. Mudge. Improving NAND flash based disk caches. In *Proceedings of International Symposium on Computer Architecture*, June 2008.
32. T. Kgil, A. Saidi, N. Binkert, S. Reinhardt, K. Flautner, and T. Mudge. PicoServer: Using 3D stacking technology to build energy efficient servers. *ACM Journal on Emerging Technologies in Computing Systems*, 2009.

33. M. G. Khatib, B. J. van der Zwaag, P. Hartel, and G. J. M. Smit. Interposing flash between disk and DRAM to save energy for streaming workloads. In *ESTIMedia*, 2007.
34. K. Kim and J. Choi. Future outlook of NAND flash technology for 40 nm node and beyond. In *Workshop on Non-Volatile Semiconductor Memory*, pp. 9–11, February 2006.
35. I. Koltsidas and S. D. Viglas. Flashing up the storage layer. In *VLDB*, August 2008.
36. P. Kongetira, K. Aingaran, and K. Olukotun. Niagara: A 32-way multithreaded Sparc processor. *IEEE Micro*, 25(2):21–29, March 2005.
37. M. Koyanagi. Different approaches to 3D chips. http://asia.stanford.edu/events/Spring05/slides/051205-Koyanagi.pdf.
38. S. R. Kunkel, R. J. Eickemeyer, M. H. Lipasti, T. J. Mullins, B. O'Krafka, H. Rosenberg, S. P. VanderWiel, P. L. Vitale, and L. D. Whitley. A performance methodology for commercial servers. *IBM Journal of Research and Development*, 44(6):851–872, 2000.
39. J. Laudon. Performance/watt: the new server focus. *SIGARCH Computer Architecture News*, 33(4):5–13, 2005.
40. K. Lee, T. Nakamura, T. Ono, Y. Yamada, T. Mizukusa, H. Hashimoto, K. Park, H. Kurino, and M. Koyanagi. Three-dimensional shared memory fabricated using wafer stacking technology. In *IEDM Technical Digest*, pp. 165–168, December 2000.
41. K. Lim, P. Ranganathan, J. Chang, C. Patel, T. Mudge, and S. Reinhardt. Understanding and designing new server architectures for emerging warehouse-computing environments. In *Proceedings of International Symposium on Computer Architecture*, June 2008.
42. J.-H. Lin, Y.-H. Chang, J.-W. Hsieh, T.-W. Kuo, and C.-C. Yang. A NOR emulation strategy over NAND flash memory. In *13th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*, August 2007.
43. G. L. Loi, B. Agrawal, N. Srivastava, S.-C. Lin, T. Sherwood, and K. Banerjee. A thermally-aware performance analysis of vertically integrated (3-D) processor-memory hierarchy. In *Proceedings Design Automation Conference*, June 2006.
44. J. Lu. Wafer-level 3D hyper-integration technology platform. www.rpi.edu/ luj/RPI_3D_Research_0504.pdf.
45. G. MacGillivray. Process vs. density in DRAMs. http://www.eetasia.com/ARTICLES/2005SEP/B/2005SEP01_STOR_TA.pdf.
46. D. A. Maltz and P. Bhagwat. TCP splicing for application layer proxy performance. Research Report RC 21139, IBM, March 1998.
47. R. E. Matick and S. E. Schuster. Logic-based eDRAM: origins and rationale for use. *IBM Journal of Research and Development*, 49(1):145–165, January 2005.
48. T. Ohsawa, K. Fujita, K. Hatsuda, T. Higashi, T. Shino, Y. Minami, H. Nakajima, M. Morikado, K. Inoh, T. Hamamoto, S. Watanabe, S. Fujii, and T. Furuyama. Design of a 128-Mb SOI DRAM using the floating body cell (FBC). *IEEE Journal of Solid State Circuits*, 41(1), January 2006.
49. C. Park, J.-U. Kang, S.-Y. Park, and J.-S. Kim. Energy-aware demand paging on NAND flash-based embedded storages. In *ISLPED*, pp. 338–343, 2004.
50. A. Rahman and R. Reif. System-level performance evaluation of three-dimensional integrated circuits. *IEEE Transactions on VLSI*, 8(6):671–678, December 2000.
51. F. Ricci, L. T. Clark, T. Beatty, W. Yu, A. Bashmakov, S. Demmons, E. Fox, J. Miller, M. Biyani, and J. Haigh. A 1.5 GHz 90 nm embedded microprocessor core. In *Proceedings of the IEEE Symposium on VLSI Circuits*, pp. 12–15, June 2005.
52. J. Scaramella. Enabling technologies for power and cooling. http://h71028.www7.hp.com/enterprise/downloads/Thermal_Logic.pdf.
53. J. Schutz and C. Webb. A scalable X86 CPU design for 90 nm process. In *Proceedings of IEEE International Solid-State Circuits Conference*, February 2004.
54. M. Shah, J. Barreh, J. Brooks, R. Golla, G. Grohoski, N. Gura, R. Hetherington, P. Jordan, M. Luttrell, C. Olson, B. Saha, D. Sheahan, L. Spracklen, and A. Wynn. UltraSPARC T2: A highly-threaded, power-efficient, SPARC SOC. In *Asian Solid-State Circuirts Conference*, November 2007.

55. J. Truong. Evolution of network memory. http://www.jedex.org/images/pdf/jack_troung_samsung.pdf.
56. D. Wendell, J. Lin, P. Kaushik, S. Seshadri, A. Wang, V. Sundararaman, P. Wang, H. McIntyre, S. Kim, W. Hsu, H. Park, G. Levinsky, J. Lu, M. Chirania, R. Heald, and P. Lazar. A 4 MB on-chip l2 cache for a 90 nm 1.6 GHz 64b SPARC microprocessor. In *Proceedings of IEEE International Solid-State Circuits Conference*, February 2004.
57. L. Xue, C. C. Liu, H.-S. Kim, S. Kim, and S. Tiwari. Three-dimensional integration: Technology, use, and issues for mixed-signal applications. *IEEE Transactions on Electron Devices*, 50:601–609, May 2003.