Exploring the Design Space for 3D Clustered Architectures

Manu Awasthi, Rajeev Balasubramonian
School of Computing, University of Utah
{manua, rajeev}@cs.utah.edu [*]

Abstract

3D die-stacked chips are emerging as intriguing prospects for the future because of their ability to reduce on-chip wire delays and power consumption. However, they will likely cause an increase in chip operating temperature, which is already a major bottleneck in modern microprocessor design. We believe that 3D will provide the highest performance benefit for high-ILP cores, where wire delays for 2D designs can be substantial. A clustered microarchitecture is an example of a complexity-effective implementation of a high-ILP core. In this paper, we consider 3D organizations of a single-threaded clustered microarchitecture to understand how floorplanning impacts performance and temperature. We first show that delays between the data cache and ALUs are most critical to performance. We then present a novel 3D layout that provides the best balance between temperature and performance. The best-performing 3D layout has 12% higher performance than the best-performing 2D layout.

Keywords: *3D die-stacked chips, clustered architectures, microarchitectural floorplanning, wire delays, cache hierarchies.*

1. Introduction

Interconnect performance [25] and power [24] have emerged as major bottlenecks. The vertical stacking of dies enables low intra-chip distances for signal transmission and helps alleviate the interconnect bottleneck. However, 3D chips will likely experience high power densities and operating temperatures; inter-die insulators within a 3D chip also make it harder for heat to escape to the heat sink on the chip's surface. Every time the operating temperature exceeds a threshold, processors are forced to switch to low-power and low-performance modes. A high thermal emergency threshold leads to high packaging and cooling costs, while a low thermal emergency threshold leads to frequent emergencies and lower performance. Assuming that 3D chips represent the way of the future, it is important that architects pursue innovations that allow these 3D chips to either incur tolerable cooling costs or low performance overheads when dealing with thermal emergencies.

As of now, it is too early to tell if 3D chips do represent the way of the future. Any one of many hurdles can threaten to be a show-stopper: manufacturing limitations, poor yield, cooling limitations, not enough performance benefit, lack of maturity in EDA tools, alternative computing technologies and market forces. In spite of these hurdles, early-stage architecture results are necessary to determine the potential of pursuing the 3D approach. Recent papers have indicated that the potential for performance and power improvement is non-trivial [4, 23, 29, 30, 31, 32, 33, 35, 39]. The most compelling arguments in favor of 3D chips are as follows:

- Future chip multiprocessors (CMPs) that accommodate tens or hundreds of cores will most likely be limited by memory bandwidth. Consider the following design: DRAM dies stacked on top of processing dies, with inter-die vias scattered all over each die surface. By leveraging the entire chip surface area for memory bandwidth (as opposed to part of the chip perimeter in conventional 2D chips), this design can help meet the memory needs of large-scale CMPs. There are indications that Intel designers may consider this possibility [20] and Samsung has already developed 3D chips that stack DRAM on top of a processing die [36].
- It is well-known that as process technologies shrink, wire delays do not scale down as well as logic delays. Future processors will likely be communication-bound, with on-chip wire delays of the order of tens of cycles [1, 18] and on-chip interconnects accounting for half the dynamic power dissipated on a chip [24]. A 3D implementation dramatically reduces the distances that signals must travel, thereby reducing wire delay and wire power consumption.
- Since each die can be independently manufactured, heterogeneous technologies can be integrated on a single chip.

One approach to take advantage of 3D is to implement

[*] This work was supported in part by NSF grant CCF-0430063 and by an NSF CAREER award.

every block of a microprocessor as a 3D circuit, also referred to as *folding*. Puttaswamy and Loh characterize the delay and power advantage for structures such as the data cache [29], register file [31], ALU [32], and issue queue [30]. The delays of these structures can typically be reduced by less than 10%, implying a potential increase in clock speed, or a potential increase in instruction-level parallelism (ILP) by accommodating more register/cache/issue queue entries for a fixed cycle time target. The disadvantage with the folding approach is that potential hotspots (*e.g.*, the register file) are stacked vertically, further exacerbating the temperature problem. Much design effort will also be invested in translating well-established 2D circuits into 3D.

An alternative approach is to leave each circuit block untouched (as a 2D planar implementation) and leverage 3D to stack different microarchitectural structures vertically. The primary advantage of this approach is the ability to reduce operating temperature by surrounding hotspots with relatively cool structures. A second advantage is a reduction in inter-unit wire delay/power and a potentially shorter pipeline. The goal of this paper is to carry out a preliminary evaluation of this alternative approach. Wire delays between microarchitectural blocks may not represent a major bottleneck for small cores. Hence, it is unlikely that 3D will yield much benefit for such cores. Clearly, larger cores with longer inter-unit distances stand to gain more from a 3D implementation. Since we are interested in quantifying the maximum performance potential of 3D for a single core, as an evaluation platform, we employ a large clustered architecture capable of supporting a large window of in-flight instructions. Many prior papers [1, 22, 28] have shown that a clustered microarchitecture represents a complexity-effective implementation of a large core. In a clustered design, processor resources (registers, issue queue entries, ALUs) are partitioned into small clusters, with an interconnect fabric enabling register communication between clusters. Such microarchitectures have even been adopted by industrial designs [22]. A multi-threaded clustered architecture is also capable of simultaneously meeting industrial demands for high ILP, high TLP (thread-level parallelism), and clock speeds [9, 14].

We first present data on the impact of inter-unit wire delays on performance (Section 2). Wire delays between integer ALUs and data caches have the most significant impact on performance. Also, data cache banks are relatively cool structures, while clusters have higher power densities. This implies that the relative placement of clusters and cache banks can have a significant impact on performance and temperature. The baseline 2D clustered architecture is described in Section 3 and 3D design options are explored in Section 4. Performance and temperature results are presented in Section 5. We discuss related work in Section 6 and draw conclusions in Section 7.

2. Motivation

Floorplanning algorithms typically employ a simulated annealing process to evaluate a wide range of candidate floorplans. The objective functions for these algorithms are usually a combination of peak temperature, silicon area, and metal wiring overhead. In an effort to reduce temperature, two frequently communicating blocks may be placed arbitrarily far apart. As a result, additional pipeline stages are introduced between these blocks just for signal transmission. In modern microprocessors, the delays across global wires can exceed a single cycle. The Intel Pentium4 [17] has a couple of pipeline stages exclusively for signal propagation. As wire delays continue to grow, relative to logic delays and cycle times, we can expect more examples of multi-cycle wire delays within a microprocessor. We extended the Simplescalar-3.0 [5] toolset to model the critical loops in a monolithic superscalar out-of-order processor. In Table 1, we list the salient processor parameters as well as details regarding the critical loops. Figure 1 shows the effect of wire delays between various pipeline stages on average IPC for the SPEC-2k benchmark set.

We see that wire delays between the ALU and data cache degrade IPC the most. Every additional cycle between the ALU and data cache increases the load-to-use latency by two cycles: it takes an extra cycle to communicate the effective address to the cache and an extra cycle to bypass the result to dependent integer ALU operations. Further, it takes longer for the issue queue to determine if a load instruction is a cache hit or miss. Many modern processors employ load-hit speculation, where it is assumed that loads will hit in the cache and dependent instructions are accordingly scheduled. Load hit-speculation imposes an IPC penalty in two ways: (i) In order to facilitate replay on a load-hit mis-speculation, dependent instructions are kept in the issue queue until the load hit/miss outcome is known – this increases the pressure on the issue queue (regardless of whether the speculations are correct or not). (ii) On a load-hit mis-speculation, dependent instructions are issued twice and end up competing twice for resources. The introduction of wire delays between the ALU and data cache increases the time taken to determine if the load is a hit/miss – correspondingly, there is greater pressure on the issue queue and more dependents are issued on a load-hit mis-speculation.

The other noticeable wire delays are between the issue queue and ALUs. These delays also increase the penalties imposed by load-hit speculation. Every other 4-cycle wire delay has less than a 5% impact on IPC. These experiments confirm that the ALUs and data caches must be placed as close to each other as possible during the floor-

Simulation parameters			
Fetch queue size	16	Branch predictor	comb. of bimodal and 2-level
Bimodal predictor size	16K	Level 1 predictor	16K entries, history 12
Level 2 predictor	16K entries	BTB size	16K sets, 2-way
Branch mispredict penalty	at least 10 cycles	Fetch, Dispatch, Commit width	4
Issue queue size	20 Int, 15 FP	Register file size	80 (Int and FP, each)
Integer ALUs/mult-div	4/2	FP ALUs/mult-div	2/1
L1 I-cache	32KB 2-way	Memory latency	300 cycles for the first block
L1 D-cache	32KB 2-way 2-cycle	L2 unified cache	2MB 8-way, 30 cycles
ROB/LSQ size	80/40	I and D TLB	128 entries, 8KB page size

Pipeline stages involved in wire delay	How the wire delay affects performance
Branch predictor and L1I-Cache	Branch mispredict penalty
I-Cache and Decode	Branch mispredict penalty, penalty to detect control instruction
Decode and Rename	Branch mispredict penalty
Rename and Issue queue	Branch mispredict penalty and register occupancy
Issue queue and ALUs	Branch mispredict penalty, register occupancy, L1 miss penalty, load-hit speculation penalty
Integer ALU and L1D-Cache	load-to-use latency, L1 miss penalty, load-hit speculation penalty
FP ALU and L1D-Cache	load-to-use latency for floating-point operations
Integer ALU and FP ALU	dependences between integer and FP operations
L1 caches and L2 cache	L1 miss penalty
Clusters in a clustered microarchitecture	inter-cluster dependences

Table 1. **Simulation parameters for the monolithic superscalar and the effect of wire delays on critical loops.**

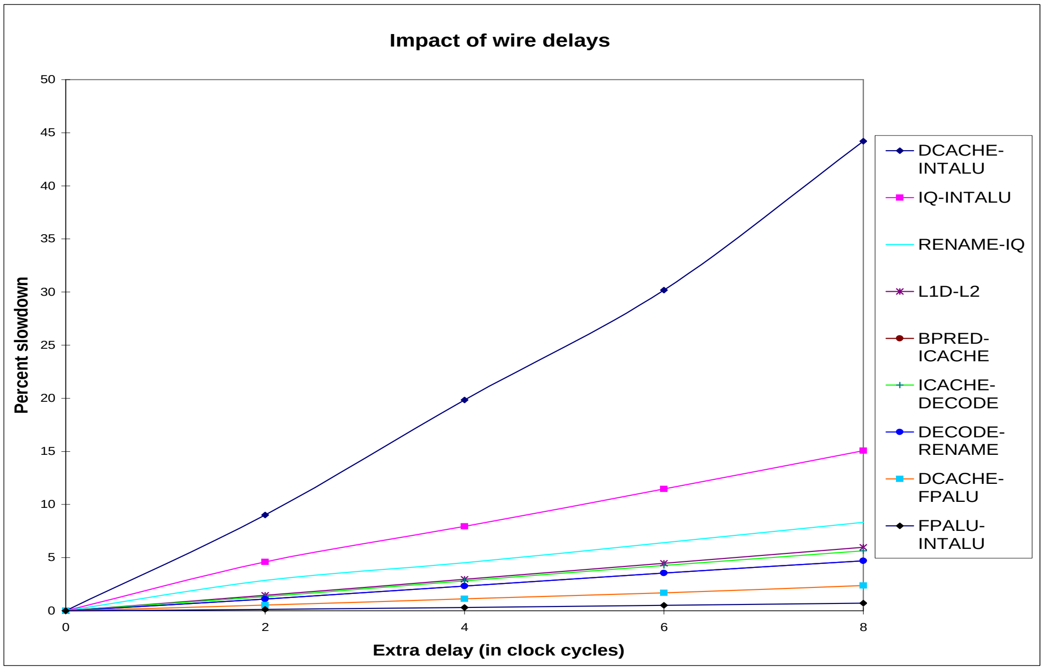


Figure 1. **Impact of inter-unit wire delays on IPC for a monolithic superscalar processor.**

planning process. Hence, for most of this paper, we will assume that the rest of the pipeline is distributed over multiple dies in a 3D chip and we will carefully consider the relative placement of execution units and cache banks.

3. Baseline Clustered Architecture

Wire delays play a greater role in large cores with many resources. We will therefore consider cores with in-flight instruction windows as high as 256. Even a medium-scale out-of-order processor such as the Alpha 21264 employs a clustered architecture to support an in-flight instruction window of 80. As an evaluation platform for this study, we adopt a dynamically scheduled clustered microarchitecture.

Centralized Front End

As shown in Figure 2, instruction fetch, decode, and dispatch (register rename) are centralized in our processor model. During register rename, instructions are assigned to one of eight clusters. The instruction steering heuristic is based on Canal *et al.*'s ARMBS algorithm [6] and attempts to minimize load imbalance and inter-cluster communication. For every instruction, we assign weights to each cluster to determine the cluster that is most likely to minimize communication and issue-related stalls. Weights are assigned to a cluster if it produces input operands for the instruction. Additional weights are assigned if that producer has been on the critical path in the past. A cluster also receives weights depending on the number of free issue queue entries within the cluster. Each instruction is assigned to the cluster that has the highest weight according to the above calculations. If that cluster has no free register and issue queue resources, the instruction is assigned to a neighboring cluster with available resources.

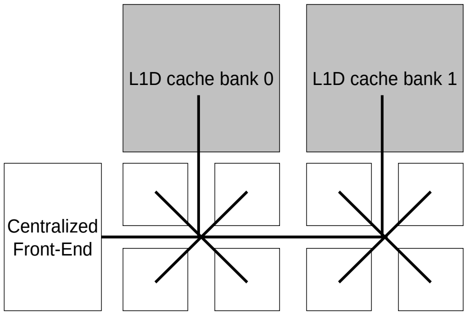


Figure 2. Baseline 2D implementation of the 8-cluster system.

Execution Units

Our clustered architecture employs small computation units (clusters) that can be easily replicated on the die. Each cluster consists of a small issue queue, physical register file, and a limited number of functional units with a single cycle bypass network among them. The clock speed and design complexity benefits stem from the small sizes of structures within each cluster. Dependence chains can execute quickly if they only access values within a cluster. If an instruction sources an operand that resides in a remote register file, the register rename stage inserts a "copy instruction" [6] in the producing cluster so that the value is moved to the consumer's register file as soon as it is produced. These register value communications happen over longer global wires and can take up a few cycles. Aggarwal and Franklin [2] show that a crossbar interconnect performs the best when connecting a small number of clusters (up to four), while a hierarchical interconnect performs better for a large number of clusters.

Cache Organization

In this paper, we consider centralized and distributed versions of the L1 data cache. Our implementations are based on state-of-the-art proposals in recent papers [3, 16, 34, 40]. Load and store instructions are assigned to clusters, where effective address computation happens. The effective addresses are then sent to the corresponding LSQ and L1 data cache bank. For a centralized cache organization, a single LSQ checks for memory dependences before issuing the load and returning the word back to the requesting cluster. When dispatching load instructions, the steering heuristic assigns more weights to clusters that are closest to the centralized data cache.

As examples of decentralized cache organizations, we consider replicated and word-interleaved caches. In a replicated cache, each cache bank maintains a copy of the L1 data cache. This ensures that every cluster is relatively close to all of the data in the L1 cache. However, in addition to the high area overhead, every write and cache refill must now be sent to every cache bank. An LSQ at every cache bank checks for memory dependences before issuing loads. A word-interleaved cache distributes every cache line among the various cache banks (for example, all odd words in one bank and even words in another bank). This ensures that every cluster is relatively close to some of the data in the L1 cache. Word-interleaved caches have larger capacities than replicated caches for a fixed area budget. Once the effective address is computed, it is sent to the corresponding LSQ and cache bank. Load instructions must be steered to clusters that are in close proximity to the appropriate cache bank. Since the effective address is not known at dispatch time, a predictor is employed and the predicted bank is fed as an input to the instruction steering algorithm. A mechanism [40] is required to ensure that memory dependences are maintained even when a store instruction's bank is mispredicted. Initially, each LSQ maintains a dummy entry for every store, preventing

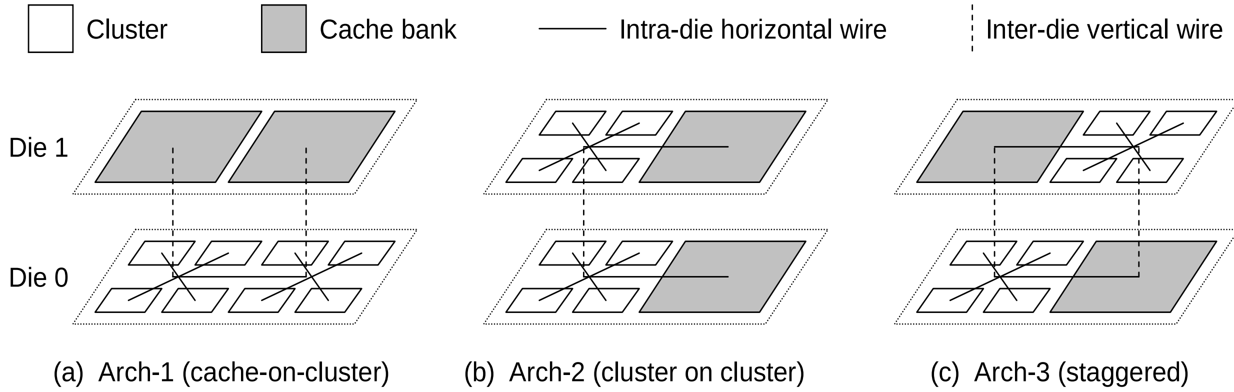

Cluster
Cache bank
Intra-die horizontal wire
Inter-die vertical wire
Die 1
Die 0
(a) Arch-1 (cache-on-cluster)
(b) Arch-2 (cluster on cluster)
(c) Arch-3 (staggered)

Figure 3. **Block diagrams for 8-cluster 3D architectures 1, 2, and 3.**

subsequent loads from issuing. Once the store address is known, only the corresponding LSQ tracks that address, while other LSQs remove the dummy entry. Thus, both decentralized caches suffer from the problem that stores have to be broadcast to all LSQs.

4. 3D Clustered Architectures

We consider the three most interesting relative placements for clusters and cache banks in 3D (shown in Figure 3). For most of this discussion, we assume that (i) two dies are bonded face-to-face (F2F [4]), (ii) each cache bank has the same area as a set of four clusters, and (iii) the system has eight clusters and two cache banks. The floorplanning principles can also be extended to greater numbers of clusters, cache banks, and dies. The differentiating design choices for the three architectures in Figure 3 are: (i) How close is a cluster to each cache bank? (ii) Which communication link exploits the low-latency inter-die via? These choices impact both temperature and performance.

Architecture 1 (cache-on-cluster):

In this architecture, all eight clusters are placed on the lower device layer (die 0) while the data cache banks are placed on the upper device layer (die 1). The heat sink and spreader are placed on the upper device layer. The L1 data cache is decentralized and may either be replicated or word-interleaved. The link from each crossbar to the cache banks is implemented with inter-die vias. Inter-die vias are projected to have extremely low latencies and sufficient bandwidth to support communication for 64-bit register values [1]. In such an architecture, communication between two sets of four clusters can be expensive. Such communication is especially encountered for programs with poor register locality or poor bank prediction rates (in the case of a word-interleaved cache). By placing all (relatively hot) clusters on a single die, the rate of lateral heat spreading is negatively impacted. On the other hand, vertical heat spreading is encouraged by placing (relatively) cool cache banks upon clusters.

Architecture 2 (cluster-on-cluster):

This is effectively a rotated variation of Architecture 1. Clusters are stacked vertically, and similarly, cache banks are also stacked vertically. In terms of performance, communication between sets of four clusters is now on faster inter-die vias, while communication between a cluster and its closest cache bank is expensive. In terms of thermal characteristics, the rate of lateral heat spreading on a die is encouraged, while the rate of vertical heat spreading between dies is discouraged.

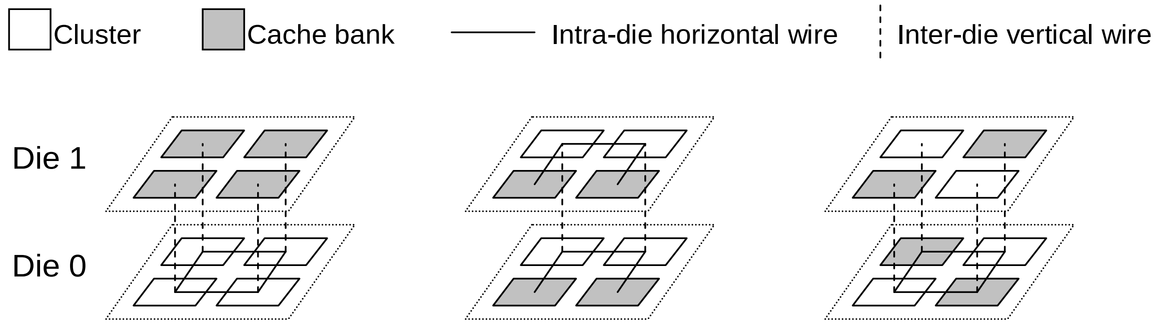
Architecture 3 (staggered):

Architecture 3 attempts to surround hot clusters with cool cache banks in the horizontal and vertical directions with a *staggered* layout. This promotes the rate of vertical and lateral heat spreading. Each set of four clusters has a link to a cache bank on the same die and a low-latency inter-die link to a cache bank on the other die. Thus, access to cache banks is extremely fast. In a word-interleaved cache, bank prediction helps guide a load to a cluster that can access the predicted cache bank with a vertical interconnect. In a replicated cache, a load always employs the corresponding vertical interconnect to access the cache bank. On the other hand, register communication between sets of four clusters may now be more expensive as three routers must be navigated. However, there are two equidistant paths available for register communication, leading to fewer contention cycles. In our experiments, register transfers are alternately sent on the two available paths.

Sensitivity Study:

Most of our evaluation employs a specific 8-cluster 2-bank system to understand how 3D organizations impact performance and temperature characteristics. As future

[1] Inter-die vias have a length of 10μm and a pitch of 5μm [23].



(a) Arch-1 (cache-on-cluster) (b) Arch-2 (cluster on cluster) (c) Arch-3 (staggered)

Figure 4. **Block diagrams for 3D organizations of the 4-cluster system.**

work, we plan to also quantify these effects as a function of number of dies, clusters, cache banks, network characteristics, different resource sizes, etc. For this paper, we repeat our experiments for one other baseline system with four clusters. This helps confirm that our overall conclusions are not unique to a specific processor model.

The second processor model has four clusters and each cluster is associated with a cache bank (either word-interleaved or replicated). The clusters are connected with a ring network. Figure 4 illustrates the two-die organizations studied for the 4-cluster system.

5. Results

5.1. Methodology

Our simulator is based on Simplescalar-3.0 [5] for the Alpha AXP ISA. Separate issue queues and physical register files are modeled for each cluster. Major simulation parameters are listed in Table 2. Contention for interconnects and memory hierarchy resources are modeled in detail. Each cluster is assumed to have a register file size of 30 physical registers (integer and floating point, each) and 15 issue queue entries (integer and floating point, each). We present results for 23 of the 26 SPEC2k integer and floating point benchmarks[2]. The programs are simulated for 100 million instruction windows identified by the Simpoint [37] toolkit.

The latencies of interconnects are estimated based on distances between the centers of microarchitectural blocks in the floorplan. Intra-die interconnects are implemented on the 8X metal plane, and a clock speed of 5 GHz at 90nm technology is assumed. Figures 3 and 4 are representative of the relative sizes of clusters and cache banks. Each crossbar router accounts for a single cycle delay. For the topology in Figure 2, for intra-die interconnects, it takes four cycles to send data between two crossbars, one cycle to send data between a crossbar and cluster, and three cycles to send data between the crossbar and 32KB cache bank. All vertical inter-die interconnects are assumed to have a single cycle latency due to their extremely short length (10μm [23]). For intra-die interconnects in the 4-cluster organization, the inter-crossbar latency is 2 cycles and the crossbar-cache latency is 2 cycles (each cache bank is 8KB).

Fetch queue size	64
Branch predictor	comb. of bimodal and 2-level
Bimodal predictor size	2048
Level 1 predictor	1024 entries, history 10
Level 2 predictor	4096 entries
BTB size	2048 sets, 2-way
Branch mispredict penalty	at least 12 cycles
Fetch width	8 (across up to 2 basic blocks)
Issue queue size	15 per cluster (int and fp, each)
Register file size	30 per cluster (int and fp, each)
Integer ALUs/mult-div	1/1 (in each cluster)
FP ALUs/mult-div	1/1 (in each cluster)
L1 I-cache	64KB 2-way
L1 D-cache	64KB 2-way set-assoc (8-clusters), 32KB 2-way set-assoc (4-clusters), 6 cycles, 2-way word-inter/replicated
L2 unified cache	8MB 8-way, 25 cycles
I and D TLB	128 entries, 8KB page size
Memory latency	300 cycles for the first chunk
Address Predictor Table size	64KB

Table 2. **Simplescalar simulator parameters.**

The bank predictor for the word-interleaved cache organization is based on a strided address predictor. The predictor has an average accuracy of 75%. This predictor performs better than a branch predictor-like two-level predictor.

We assume a face-to-face (F2F) wafer-bonding technology for this study. F2F bonding allows a relatively high inter-die via density [23] because of which we assume that the inter-die bandwidth is not a limiting constraint for our experiments.

[2] Sixtrack, Facerec, and Perlbmk are not compatible with our simulation environment.

Parameter	Value
Unit Area (Router+Crossbar)	0.3748 mm^2 [23]
Router+Crossbar Power	119.55mW [23]
Wire Power/Unit Length (Data & Control)	1.422 mW/mm [8]

Table 3. **Interconnect power modeling parameters.**

Parameter	Value
Specific heat capacity (Si)	1.75E+6 J/(m^3/K)
Specific heat capacity (SiO_2)	1.79E+6 J/(m^3/K)
Thermal Resistivity (Si)	1.69 $(W/m/K)^{-1}$
Thermal Resistivity (SiO_2)	40 $(W/m/K)^{-1}$

Table 4. **Hotspot Parameters [11].**

The Wattch power models are employed to compute power consumption of each microarchitectural block. The contribution of leakage to total chip power is roughly 20%. Interconnect power (summarized in Table 3) is based on values for 8X minimum-length wires [8] and a generic Network-on-Chip router [23]. Even though prior studies [13] have shown that inter-die vias consume little power, we consider their marginal power contributions (modeled as wires of length 10μm).

Temperature characteristics are generated by feeding the power values to Hotspot-3.0's [38] grid model with a 500×500 grid resolution. Hotspot does not consider interconnect power for thermal modeling. Hence, consistent with other recent evaluations [19], interconnect power is attributed to the units that they connect in proportion to their respective areas. Hotspot's default heat sink model and a starting ambient temperature of 45 °C is assumed for all experiments. Table 4 provides more details on the HotSpot parameters used. Each die is modeled as two layers - the active silicon and the bulk silicon. A layer of thermal interface material (TIM) is also assumed to be present between the bulk silicon of the top die and the heat spreader [33].

5.2. IPC Analysis

The primary difference between Architectures 1/2/3 (Figure 3) is the set of links that are implemented as inter-die vias. Hence, much of our IPC results can be explained based on the amount of traffic on each set of links. We observed that for a word-interleaved cache, even with a 75% bank prediction accuracy, loads are often not steered to the corresponding cluster (because of load imbalance or other register dependences). Hence, nearly half the cache accesses are to the remote cache bank through the inter-crossbar interconnect. Unless bank predictors with accuracies greater than 95% can be designed, word-interleaved cache organizations will likely continue to suffer from many remote bank accesses. In a replicated cache orga-

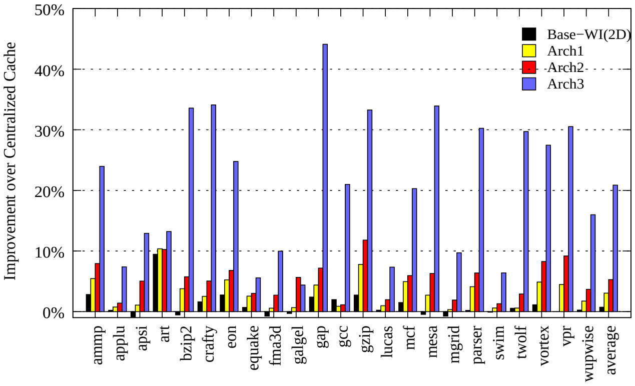


Figure 5. **Relative IPC improvement of word-interleaved architectures over the 2D base case with centralized cache - 8 clusters**

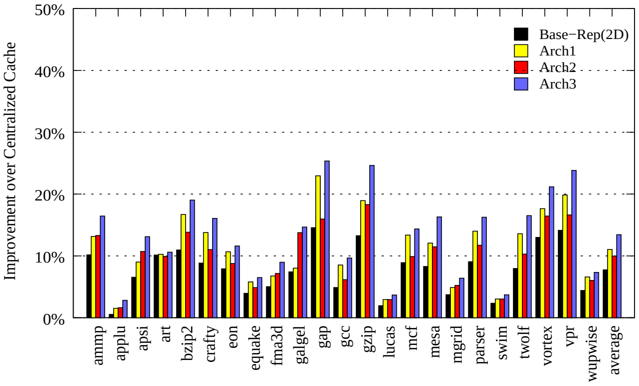


Figure 6. **Relative IPC improvement of replicated data-cache architectures over 2D centralized cache architecture - 8 clusters**

Access type	Word-int Cache 3D – arch1	Word-int Cache 3D – arch2	Word-int Cache 3D – arch3	Replicated Cache 3D – arch1	Replicated Cache 3D – arch2	Replicated Cache 3D – arch3
Local load accesses	4.50	5.89	4.80	4.12	5.24	4.13
Remote load accesses	9.10	8.90	7.30	0	0	0
Inter-crossbar register traffic	8.30	7.54	5.80	8.15	7.00	5.53

Table 5. Average network latencies (in cycles) for different types of interconnect messages.

nization, all load requests are sent to the local cache bank. About half as many register transfers are sent on the inter-crossbar interconnect between clusters. Table 5 shows the average network latencies experienced by loads and register transfers in the most relevant 8-cluster architectures.

For Figures 5 and 6, we fix the 2D 8-cluster system with a centralized cache as the baseline. A 2D system with a word-interleaved cache performs only 2% better than the baseline, mostly because of the poor bank prediction rate. A 2D system with a replicated cache performs about 7.7% better than the baseline. The replicated cache performs better in spite of having half the L1 data cache size – the average increase in the number of L1 misses in moving from a 64KB to a 32KB cache was 0.85%. A replicated cache allows instructions to not only be close to relevant data, but also close to relevant register operands. However, store addresses and data are broadcast to both cache banks and data is written into both banks (in a word-interleaved organization, only store addresses are broadcast to both banks).

Figures 5 and 6 show IPC improvements for word-interleaved and replicated cache organizations over the 2D baseline. The word-interleaved organizations are more communication-bound and stand to gain much more from 3D. The staggered architecture-3 performs especially well (20.8% better than the baseline) as every cluster is relatively close to both cache banks, bank mis-predictions are not very expensive, and multiple network paths lead to fewer contention cycles. Architecture-2 performs better than Architecture-1 because it reduces the latency for register traffic, while slowing down access for correctly bank-predicted loads. The opposite effect is seen for the replicated cache organizations because Architecture-2 slows down access for all loads (since every load accesses the local bank).

With the replicated cache, architecture-3 is similar to architecture-1 as regards cache access, but imposes greater link latency for inter-cluster register communication. Because there are multiple paths for register communication, architecture-3 imposes fewer contention cycles. As can be seen in Table 5, the average total latency encountered by register transfers is lowest for architecture-3, for both word-interleaved and replicated organizations. The net result is that architecture-3 performs best for both cache organizations. The move to 3D causes only a 5% improvement for a replicated cache organization, while it causes an 18% improvement for the word-interleaved organization. For the architecture-3 model, the word-interleaved and replicated organizations have similar latencies for instructions (Table 5), but the word-interleaved organization has twice as much L1 cache capacity. It is interesting to note that an organization such as the word-interleaved cache, which is quite un-attractive in 2D has the best performance in 3D (arch-3).

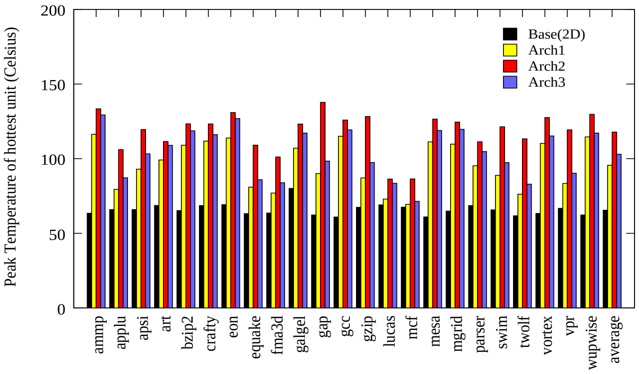


Figure 7. On-chip Peak temperatures for 8-cluster organizations (°C)

The conclusions from our sensitivity analysis with a 4-cluster organization are similar. Compared to a 2D baseline with a centralized cache, the 3D word-interleaved architectures 1, 2, and 3 yield an improvement of 9%, 10%, and 16%, respectively. The 3D replicated architectures 1, 2, and 3 yield improvements of 9%, 13%, and 15%, respectively. The move from 2D to 3D yields an improvement of 9.7% for the word-interleaved and 8% for the replicated cache organizations.

5.3. Thermal Analysis

As shown in the previous sub-section, the best 3D organization out-performs the best 2D organization by 12%. The primary benefit of 3D is that cache banks can be placed close to clusters, allowing high performance even for word-interleaved caches with poor bank prediction rates. The architectures that place cache banks close to clusters also have favorable thermal characteristics. For the 8-cluster system, Figure 7 shows the peak temperature attained by each architecture, while Figure 8 shows the average temperature of the hottest on-chip unit (typi-

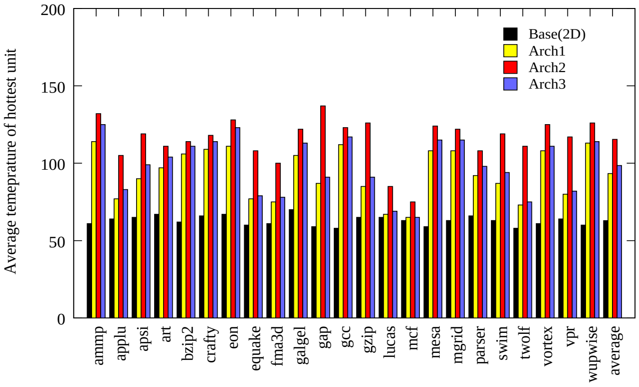


Figure 8. Average temperatures of hottest on-chip unit, for 8-cluster organizations (°C)

cally one of the issue or load/store queues). A similar profile is also observed for the 4-cluster system and the profile is largely insensitive to the choice of word-interleaved or replicated banks. Architectures 1 and 3 that stack cache upon cluster are roughly 12 °C cooler than architecture-2 that stacks cluster upon cluster. Thus, staggered architecture 3 not only provides the highest performance, but also limits the increase in temperature when moving to 3D. The lateral heat spreading effect played a very minor role in bringing down architecture 3's temperature – in fact, it was hotter than architecture 1 because of its higher IPC and power density. All 3D organizations suffer from significantly higher temperatures than the 2D chip. Note that our thermal model assumes Hotspot's default heat sink and does not take into account the ability of the inter-die vias to conduct heat to the heat sink. An advantage of 3D is the reduction in interconnect power. On an average, for the 8-cluster configurations we recorded a decrease of 8, 11, and 10 % respectively for architectures 1, 2, and 3.

6. Related Work

While the VLSI community has actively pursued tools to implement 3D circuits [10, 12], the computer architecture community is only beginning to understand the implications of 3D architectures. A research group at Intel reported [4] a 15% improvement in performance and power by implementing an IA-32 processor in 3D. As described in Section 1, Puttaswamy and Loh examine 3D implementations of specific structures such as the data cache [29], register file [31], ALU [32], and issue queue [30]. They also study the temperature profile of an Alpha 21364-like processor that is implemented across up to four dies and report a temperature increase of up to 33 Kelvin [33]. For that study, most RAM, CAM, and ALU structures are folded across the dies. In this paper, we attempt no folding of individual structures. Two other groups have examined the effect of folding the L1 data cache [35, 39]. A recent paper [23] explores a CMP with a 3D network of processing cores and L2 cache banks. Each core in the CMP is implemented on a single die and the core is surrounded (horizontally and vertically) by L2 cache banks to reduce temperature. The L2 cache is implemented as a Non-Uniform Cache Architecture (NUCA) and the 3D implementation enables about a 50% reduction in average L2 access time.

Clustered (partitioned) processors [15, 21, 28] have received much attention over the past decade. The Alpha 21264 [22] is an example of a commercial design that has adopted such a microarchitecture. While interest in ILP has waned in recent years, clustered multi-threaded architectures [9, 14] may simultaneously provide high ILP, high TLP, and high clock speeds. This paper is the first study of a 3D implementation of a clustered architecture. Temperature studies involving clustered architectures include those by Chaparro et al. [7], Nelson et al. [27], and Muralimanohar et al. [26]. The design of distributed data caches for clustered architectures has been evaluated by Zyuban and Kogge [40], Gibert et al. [16], and Balasubramonian [3].

7. Conclusions

3D technology can benefit large high-ILP cores by reducing the distances that signals must travel. In this paper, we consider various 3D design options for a clustered architecture. Placing caches and clusters in close proximity in 3D enables high performance and relatively low temperature. We show that a word-interleaved cache with a staggered 3D placement performs 12% better than the best 2D design (with a replicated cache). For future work, we plan to study the effect of scaling the system to more dies, clusters, cache banks, etc. We also plan to design more accurate bank predictors that may enable better locality of computation and data, thereby leveraging the benefits of 3D.

References

[1] V. Agarwal, M. Hrishikesh, S. Keckler, and D. Burger. Clock Rate versus IPC: The End of the Road for Conventional Microarchitectures. In *Proceedings of ISCA-27*, pages 248–259, June 2000.

[2] A. Aggarwal and M. Franklin. An Empirical Study of the Scalability Aspects of Instruction Distribution Algorithms for Clustered Processors. In *Proceedings of ISPASS*, 2001.

[3] R. Balasubramonian. Cluster Prefetch: Tolerating On-Chip Wire Delays in Clustered Microarchitectures. In *Proceedings of ICS-18*, June 2004.

[4] B. Black, D. Nelson, C. Webb, and N. Samra. 3D Processing Technology and its Impact on IA32 Microprocessors. In *Proceedings of ICCD*, October 2004.

[5] D. Burger and T. Austin. The Simplescalar Toolset, Version 2.0. Technical Report TR-97-1342, University of Wisconsin-Madison, June 1997.

[6] R. Canal, J. M. Parcerisa, and A. Gonzalez. Dynamic Code Partitioning for Clustered Architectures. *International Journal of Parallel Programming*, 29(1):59–79, 2001.

[7] P. Chaparro, J. Gonzalez, and A. Gonzalez. Thermal-effective Clustered Micro-architectures. In *Proceedings of the 1st Workshop on Temperature Aware Computer Systems, held in conjunction with ISCA-31*, June 2004.

[8] L. Cheng, N. Muralimanohar, K. Ramani, R. Balasubramonian, and J. Carter. Interconnect-Aware Coherence Protocols for Chip Multiprocessors. In *Proceedings of ISCA-33*, June 2006.

[9] J. Collins and D. Tullsen. Clustered Multithreaded Architectures – Pursuing Both IPC and Cycle Time. In *Proceedings of the 18th IPDPS*, April 2004.

[10] J. Cong and Y. Zhang. Thermal-Driven Multilevel Routing for 3-D ICs. In *Proceedings of ASP-DAC*, January 2005.

[11] CRC Press. CRC Handbook of Chemistry. http://www.hpcpnetbase.com/.

[12] S. Das, A. Chandrakasan, and R. Reif. Three-Dimensional Integrated Circuits: Performance, Design Methodology, and CAD Tools. In *Proceedings of ISVLSI*, 2003.

[13] 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.

[14] A. El-Moursy, R. Garg, D. Albonesi, and S. Dwarkadas. Partitioning Multi-Threaded Processors with a Large Number of Threads. In *Proceedings of ISPASS*, March 2005.

[15] K. Farkas, P. Chow, N. Jouppi, and Z. Vranesic. The Multicluster Architecture: Reducing Cycle Time through Partitioning. In *Proceedings of MICRO-30*, pages 149–159, 1997.

[16] E. Gibert, J. Sanchez, and A. Gonzalez. Effective Instruction Scheduling Techniques for an Interleaved Cache Clustered VLIW Processor. In *Proceedings of MICRO-35*, pages 123–133, November 2002.

[17] G. Hinton, D. Sager, M. Upton, D. Boggs, D. Carmean, A. Kyker, and P. Roussel. The Microarchitecture of the Pentium 4 Processor. *Intel Technology Journal*, Q1, 2001.

[18] R. Ho, K. Mai, and M. Horowitz. The Future of Wires. *Proceedings of the IEEE*, Vol.89, No.4, April 2001.

[19] W.-L. Hung, G. Link, Y. Xie, N. Vijaykrishnan, and M. J. Irwin. Interconnect and thermal-aware floorplanning for 3d microprocessors. *isqed*, 0:98–104, 2006.

[20] J. Rattner. Predicting the Future, 2005. Keynote at Intel Developer Forum (article at http://www.anandtech.com/tradeshows/showdoc.aspx?i=2367&p=3).

[21] S. Keckler and W. Dally. Processor Coupling: Integrating Compile Time and Runtime Scheduling for Parallelism. In *Proceedings of ISCA-19*, pages 202–213, May 1992.

[22] R. Kessler. The Alpha 21264 Microprocessor. *IEEE Micro*, 19(2):24–36, March/April 1999.

[23] F. Li, C. Nicopoulos, T. Richardson, Y. Xie, N. Vijaykrishnan, and M. Kandemir. Design and Management of 3D Chip Multiprocessors Using Network-in-Memory. In *Proceedings of ISCA-33*, June 2006.

[24] N. Magen, A. Kolodny, U. Weiser, and N. Shamir. Interconnect Power Dissipation in a Microprocessor. In *Proceedings of System Level Interconnect Prediction*, February 2004.

[25] D. Matzke. Will Physical Scalability Sabotage Performance Gains? *IEEE Computer*, 30(9):37–39, September 1997.

[26] N. Muralimanohar, K. Ramani, and R. Balasubramonian. Power Efficient Resource Scaling in Partitioned Architectures through Dynamic Heterogeneity. In *Proceedings of ISPASS*, March 2006.

[27] N. Nelson, G. Briggs, M. Haurylau, G. Chen, H. Chen, D. Albonesi, E. Friedman, and P. Fauchet. Alleviating Thermal Constraints while Maintaining Performance Via Silicon-Based On-Chip Optical Interconnects. In *Proceedings of Workshop on Unique Chips and Systems*, March 2005.

[28] S. Palacharla, N. Jouppi, and J. Smith. Complexity-Effective Superscalar Processors. In *Proceedings of ISCA-24*, pages 206–218, June 1997.

[29] K. Puttaswamy and G. Loh. Implementing Caches in a 3D Technology for High Performance Processors. In *Proceedings of ICCD*, October 2005.

[30] K. Puttaswamy and G. Loh. Dynamic Instruction Schedulers in a 3-Dimensional Integration Technology. In *Proceedings of GLSVLSI*, April 2006.

[31] K. Puttaswamy and G. Loh. Implementing Register Files for High-Performance Microprocessors in a Die-Stacked (3D) Technology. In *Proceedings of ISVLSI*, March 2006.

[32] K. Puttaswamy and G. Loh. The Impact of 3-Dimensional Integration on the Design of Arithmetic Units. In *Proceedings of ISCAS*, May 2006.

[33] K. Puttaswamy and G. Loh. Thermal Analysis of a 3D Die-Stacked High-Performance Microprocessor. In *Proceedings of GLSVLSI*, April 2006.

[34] P. Racunas and Y. Patt. Partitioned First-Level Cache Design for Clustered Microarchitectures. In *Proceedings of ICS-17*, June 2003.

[35] P. Reed, G. Yeung, and B. Black. Design Aspects of a Microprocessor Data Cache using 3D Die Interconnect Technology. In *Proceedings of International Conference on Integrated Circuit Design and Technology*, May 2005.

[36] Samsung Electronics Corporation. Samsung Electronics Develops World's First Eight-Die Multi-Chip Package for Multimedia Cell Phones, 2005. (Press release from `http://www.samsung.com`).

[37] T. Sherwood, E. Perelman, G. Hamerly, and B. Calder. Automatically Characterizing Large Scale Program Behavior. In *Proceedings of ASPLOS-X*, October 2002.

[38] K. Skadron, M. R. Stan, W. Huang, S. Velusamy, and K. Sankaranarayanan. Temperature-Aware Microarchitecture. In *Proceedings of ISCA-30*, pages 2–13, 2003.

[39] Y.-F. Tsai, Y. Xie, N. Vijaykrishnan, and M. Irwin. Three-Dimensional Cache Design Using 3DCacti. In *Proceedings of ICCD*, October 2005.

[40] V. Zyuban and P. Kogge. Inherently Lower-Power High-Performance Superscalar Architectures. *IEEE Transactions on Computers*, March 2001.