Leveraging 3D Technology for Improved Reliability

Niti Madan, Rajeev Balasubramonian
School of Computing, University of Utah
{niti, rajeev}@cs.utah.edu*

Abstract

Aggressive technology scaling over the years has helped improve processor performance but has caused a reduction in processor reliability. Shrinking transistor sizes and lower supply voltages have increased the vulnerability of computer systems towards transient faults. An increase in within-die and die-to-die parameter variations has also led to a greater number of dynamic timing errors. A potential solution to mitigate the impact of such errors is redundancy via an in-order checker processor.

Emerging 3D chip technology promises increased processor performance as well as reduced power consumption because of shorter on-chip wires. In this paper, we leverage the "snap-on" functionality provided by 3D integration and propose implementing the redundant checker processor on a second die. This allows manufacturers to easily create a family of "reliable processors" without significantly impacting the cost or performance for customers that care less about reliability. We comprehensively evaluate design choices for this second die, including the effects of L2 cache organization, deep pipelining, and frequency. An interesting feature made possible by 3D integration is the incorporation of heterogeneous process technologies within a single chip. We evaluate the possibility of providing redundancy with an older process technology, an unexplored and especially compelling application of die heterogeneity. We show that with the most pessimistic assumptions, the overhead of the second die can be as high as either a 7 °C temperature increase or a 8% performance loss. However, with the use of an older process, this overhead can be reduced to a 3 °C temperature increase or a 4% performance loss, while also providing higher error resilience.

Keywords: *reliability, redundant multi-threading, 3D die-stacking, parameter variation, soft errors, dynamic timing errors, power-efficient microarchitecture, on-chip temperature.*

1. Introduction

The probability of erroneous computation increases as we employ smaller device dimensions and lower voltages. Firstly, bit values can be easily corrupted by the charge deposited by high-energy particles [21]. This phenomenon is referred to as a *transient fault* and it results in *soft errors* that corrupt program outputs, but do not render the device permanently faulty. Secondly, parameter variation can cause uncertainties in various device dimensions (such as gate length/width, wire height/width/spacing), that can influence the delay characteristics of circuits [3]. Dynamic conditions such as temperature, supply voltage noise, and cross-coupling effects can also influence the delay of a circuit. Because of these variations, timing constraints are occasionally not met and an incorrect result is latched into the pipeline stage at the end of that clock cycle. We refer to these errors as *dynamic timing errors.* Thirdly, processor components gradually wear out and are rendered permanently faulty, leading to *hard errors.*

Various solutions exist to handle each of the above errors. In the first part of this paper, we outline a design that represents a complexity-effective solution to the problem of comprehensive error detection and recovery. This solution builds upon a number of prior proposals in the area of reliable microarchitecture (such as [1, 12, 19, 22]). In this design, a redundant checker core verifies the computation of a primary core and the checker core employs in-order execution and conservative timing margins. This processor model is then used as an evaluation platform for the second half of the paper. The 3D proposal discussed in the second half represents the primary contribution of this work.

3D technology is emerging as an intriguing prospect for the future (see [44] for a good overview). This technology enables the vertical stacking of dies with low-latency communication links between dies. 3D stacking offers three primary advantages that make it a compelling research direction:

- Stacking of heterogeneous dies: A concrete application of this approach is the 3D stacking of DRAM chips upon large-scale CMPs. Inter-die vias can take advantage of the entire die surface area to implement a high bandwidth link to DRAM, thereby addressing a key bottleneck in CMPs that incorporate nearly a hundred cores [27, 32].
- "Snap-on" analysis engines: Chips employed by application developers can be fitted with additional stacked dies that contain units to monitor hardware activity and aid in debugging [24]. Chips employed by application users will not incorporate such functionality, thereby lowering the cost for these systems.

*We thank the anonymous reviewers for many helpful suggestions. This work was supported in parts by NSF grant CCF-0430063 and NSF CAREER award CCF-0545959.

1072-4451/07 $25.00 © 2007 IEEE
DOI 10.1109/MICRO.2007.31

- Improvement in CPU performance/power: The components of a CPU (cores/cache banks, pipeline stages, individual circuits) can be implemented across multiple dies. By lowering the penalties imposed by long wires, performance and power improvements are possible.

All of the above advantages of 3D can be exploited if we implement the checker core on a die placed above the CPU die: (i) The communication of results between cores is very efficient in terms of delay and power as short inter-die wires can be used. By isolating the checker core to a separate die, the layout and wiring of the main CPU die can be simplified. (ii) CPU dies and checker dies can be independently fabricated and chip manufacturers can offer products with a varying number of checker dies. (iii) The checker cores can be implemented in a process technology that is more resilient to soft errors and dynamic timing errors.

The one salient disadvantage of 3D technology is that it increases on-chip power density and temperature. Temperature increases can have a significant impact on lifetime reliability, soft error rates, leakage power dissipation, dynamic timing error rates, and performance. We consider the above effects and attempt to alleviate them with older process technologies, deeper pipelines, and three-dimensional L2 cache organizations. An evaluation of die yield is beyond the scope of this study. While the fabrication of small-sized dies for a 3D chip will improve yield, this advantage may be offset by complications in the 3D manufacturing process. The effect on yield will become clearer as 3D technology matures.

The major contributions of this paper are as follows:

- We quantify the impact of a 3D checker core on power, temperature, performance, and interconnect overhead.
- We argue for the use of an older process technology for the checker die to improve its error resilience and quantify the impact of this choice on power, temperature, and performance.
- We show that a high-ILP checker can operate at low frequencies, allowing much more timing slack in each pipeline stage and greater error resilience.
- We argue against the use of deep pipelines for the checker core.

The paper thus presents a detailed analysis of the design space to determine if the implementation of checker cores in 3D is an attractive option. In Section 2, we first describe our checker processor model. Section 3 evaluates the impact of leveraging 3D interconnects for inter-core communication. We also quantify the impact of this approach on temperature, interconnect length, and error tolerance. The use of multiple dies enables the use of different parameters for the primary and checker cores. Section 4 analyzes the effect of older technology parameters for the checker on overall power, temperature, and error rates. We discuss related work in Section 5 and summarize our conclusions in Section 6.

2. Baseline Reliable Processor Model

This section describes the implementation of a checker core that redundantly executes a copy of an application thread. This implementation draws on a number of concepts described in recent papers [1, 11, 12, 19, 22] and is capable of detecting and recovering from soft, hard, and dynamic timing errors. We are not claiming that this is an optimal design for comprehensive error coverage, a topic that remains an active area for research. We are simply postulating this design as a potential implementation that can be employed for the rest of our 3D layout analysis.

The architecture consists of two communicating cores that execute copies of the same application for fault-detection. This is often referred to as *redundant multi-threading (RMT)*. One of the cores (the trailing core) executes behind the second core (the leading core) by a certain amount of slack to enable checking for errors. The leading core communicates its committed register results to the trailing core for comparison of values to detect faults (the baseline reliable processor architecture is graphically depicted in Figure 1). Load values are also passed to the trailing core so it can avoid reading values from memory that may have been recently updated by other devices. Thus, the trailing core never accesses the L1 data cache. The leading core is allowed to commit instructions before checking. The leading core commits stores to a store buffer (StB) instead of to memory. The trailing core commits instructions only after checking for errors. This ensures that the trailing core's state can be used for a recovery operation if an error occurs. The trailing core communicates its store values to the leading core's StB and the StB commits stores to memory after checking.

To facilitate communication of values between leading and trailing threads, first-in-first-out register value queues (RVQ) and load value queues (LVQ) are used. As a performance optimization, the leading core also communicates its branch outcomes to the trailing core (through a branch outcome queue (BOQ)), allowing it to have perfect branch prediction. If the slack between the two threads is at least as large as the re-order buffer (ROB) size of the trailing thread, it is guaranteed that a load instruction in the trailing thread will always find its load value in the LVQ. When external interrupts or exceptions are raised, the leading thread must wait for the trailing thread to catch up before servicing the interrupt.

As regards transient faults, the assumed fault model is exactly the same as in [12, 22]. The proposed system is designed to detect and recover from transient faults that go on to produce an error in the datapath. The cores are still susceptible to some faults in the control path that may result

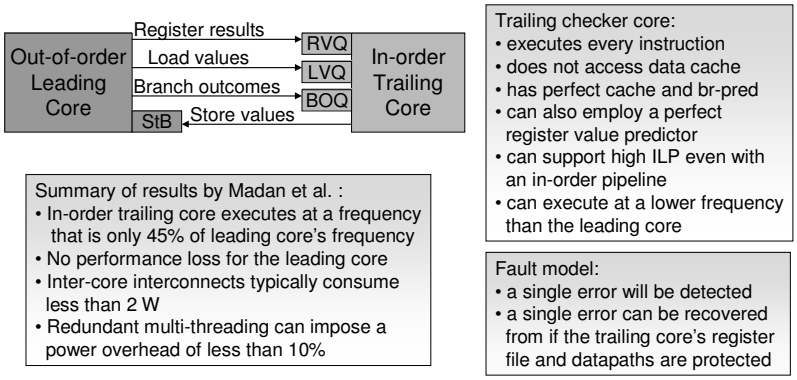


Figure 1. Overview of checker core design proposed in [19].

in errors such as resources not being freed. The following condition is required in order to detect a single transient fault in the datapath:

- The data cache, LVQ, and buses that carry load values must be ECC-protected as the trailing thread directly uses these load values.

Other structures in each core (including the RVQ) need not have ECC or other forms of protection as disagreements will be detected during the checking process. The BOQ need not be protected as long as its values are only treated as branch prediction hints and confirmed by the trailing pipeline.

The following additional condition is required in order to detect and recover from a single transient fault in the datapath:

- When an error is detected, the register file state of the trailing thread is used to initiate recovery. The trailing thread's register file must be ECC-protected to ensure that values do not get corrupted once they have been checked and written into the trailer's register file. Further, the value and the register tag must not be corrupted on their way to the register file after checking.

We assume ECC-protected data cache, LVQ, and trailing register file, enabling detection of and recovery from a single transient fault. Similarly, dynamic timing errors in either core that manifest in corrupted register values will be detected by the checking process. Unfortunately, dynamic timing errors are often correlated and the probability of multiple errors in a single core is non-trivial. Multiple errors in the leading core can still be handled because valid state in the trailing register file will be used for recovery when the first disagreement is flagged. Multiple errors in the trailing core's register file may however not be handled by ECC and may impact our ability to recover from the errors. As will be subsequently described, the microarchitecture of the trailing core is defined in a manner that minimizes this probability. As a result, the trailing core is also more resilient to faults in the control path.

The checker core is a full-fledged core with some logic extensions to enable its operation as a checker. Hence, it is also capable of executing a leading thread by itself, if necessary. Given the redundancy in the architecture, a hard error in the leading core can also be tolerated, although at a performance penalty[1]. The following sub-section describes mechanisms to implement the trailing checker core in a power-efficient manner.

2.1. Power-Efficient Checker Core Design

An RMT system attempts to maintain a roughly constant slack between leading and trailing threads. Since the trailing thread benefits from perfect caching and branch prediction, it tends to catch up with the leading thread. This provides the opportunity to throttle the execution of the trailer in a manner that lowers its power consumption and maintains the roughly constant slack. Building upon the ideas in [19], we consider the following techniques to reduce checker core power consumption.

Dynamic Frequency Scaling (DFS) is a well-established technique that allows dynamic power to scale down linearly with clock frequency [20]. It is a low overhead technique – in Intel's Montecito, a frequency change can be effected in a single cycle [20]. We implement the algorithm in [19] to scale down (up) the frequency of the trailing core when the occupancy of the RVQ falls below (above) a certain threshold. This approach helps reduce dynamic power (and energy) with a negligible impact on performance. To accomodate a slack of 200 instructions, we implement a 200-entry RVQ, an 80-entry LVQ, a 40-entry BoQ and a 40-entry StB.

Since the cores now operate at different frequencies, they must employ GALS-like synchronization [34] for inter-core communication. This usually requires a buffer between cores that can be read/written at different clock edges – the RVQ/LVQ/BoQ buffers can be implemented in this manner. Such an approach typically introduces occasional delays in inter-core communication, but this is not a

[1]We will assume that a mechanism exists to detect the hard error and re-schedule tasks accordingly on the other core if necessary.

bottleneck for the proposed system. There are other overheads and implementation issues when supporting multiple clocks (the need for multiple PLLs or clock dividers, multiple clock trees, etc.). We expect multiple clocks to be the norm in future large heterogeneous multi-cores and do not further consider these overheads in this study.

In-order execution of the trailing core enables an even greater saving in dynamic and leakage power consumption. This approach extends some of the concepts advocated in the DIVA [1] design to general-purpose cores. Unfortunately, for many program phases, an in-order core, even with a perfect D-cache and branch predictor, cannot match the throughput of the leading out-of-order core. Hence, some enhancements need to be made to the in-order core. We employ a form of register value prediction (RVP). Along with the result of an instruction, the leading thread also passes the input operands for that instruction to the trailing thread. Instructions in the trailing core can now read their input operands from the RVQ instead of from the register file. Before the instruction commits, it verifies that the results read from the RVQ match the values in the trailer's register file. Such a register value predictor has a 100% accuracy, unless an earlier instruction's result has been influenced by an error. If an earlier instruction has been affected by an error, the trailing thread will detect it when it verifies the prediction and the subsequent execution will be corrected. With a perfect RVP, instructions in the trailer are never stalled for data dependences and ILP is constrained only by a lack of functional units or instruction fetch bandwidth.

The above two techniques can be combined. An in-order checker core with RVP has high ILP and tends to catch up with the leading core. Depending on the occupancy of the RVQ, the frequency of the checker core is scaled up or down. This yields a complexity-effective reliable processor design (summarized in Figure 1) that employs a low-power and simple in-order core operating at a low frequency to verify the computations of the leading core. In addition to providing tolerance to hard errors in the leading core, the checker core can detect errors caused by high energy particles, noise, and dynamic timing variation. It must be noted that such an efficient checker core is facilitated by investing in a large amount of inter-core traffic (register operands, load values, and branch outcomes are transmitted to the trailing core). This is worthwhile even in a 2D layout because of the significant energy savings it enables in the checker core. The inter-core traffic is an especially worthwhile investment in a 3D layout where communication between dies happens on low-latency and low-power vias.

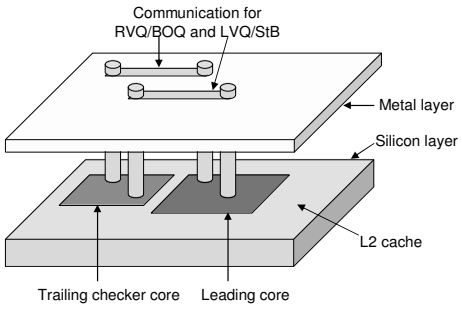
3. Proposed 3D Implementation

In a 2D layout of the reliable processor described in Section 2, the out-of-order leading core and in-order checker core are located next to each other, as shown in Figure 2(a). Inter-core communication of results is implemented on interconnects that are routed over a number of other blocks on higher metal layers. These are typically long pipelined wires with a few repeaters and latches per wire that consume non-trivial amounts of power. These repeaters and latches also require silicon area in the blocks that the interconnects happen to fly over. As a result of these wiring and silicon needs of the checker, the layout of the leading core will be negatively impacted. The integration of such an invasive checker core on a 2D die will likely increase the area, wiring complexity, and cycle time for the leading core.

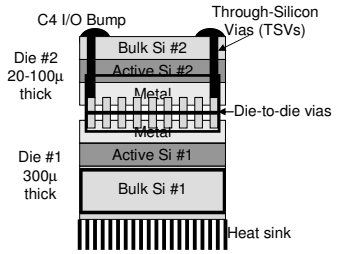
In a 3D die-stacked chip, two separately manufactured dies can be bonded together in various ways. In this paper, we focus on Face-to-Face (F2F) bonding of two dies, as shown in Figure 2(b). Intra-die communication happens on conventional interconnects implemented on that die's metal layers. Inter-die communication happens on inter-die (or die-to-die) vias that are simply an extension of the vias used to access each die's metal stack. A collection of these inter-die vias is referred to as an "inter-die via pillar" [16]. Thus, a register value in the lower die can now be driven vertically on an inter-die via pillar to a metal layer on the die above and then routed horizontally to the unit on the upper die that needs the value. With an appropriate layout, the horizontal traversal on the upper die can be made short. With such an organization, the primary requirement of the inter-core interconnects is metal space to implement the inter-die via pillars. The isolation of the checker core and its associated logic and wiring to a separate die minimizes the impact on the leading core's floorplan, wiring, and cycle time. The lower die can therefore also be sold as a stand-alone entity that offers reduced reliability (and nearly the same performance and power as a 2D processor without the pillars). For customers that require high reliability, the lower die is combined with a corresponding upper die that incorporates a checker core to monitor the register values produced on the inter-die pillars (shown in Figure 2(c)). This "snap-on" functionality of 3D dies was recently proposed by Mysore et al. [24] to implement software profiling and debugging tools. This paper proposes a snap-on 3D checker core and various aspects of such an organization are discussed next. The next section evaluates the use of an older process technology to implement the checker core. It must be noted that a primary benefit of the proposed 3D organization is the minimal impact on the layout, wiring, and cycle time of the leading core; quantifying this "non-invasive" effect of the 3D checker is beyond the scope of this paper.

3.1. Methodology

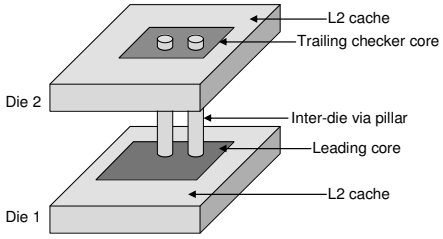
Our thermal model is based on the Hotspot-3.1 [37] grid model. The modeling parameters and thermal constants for each layer are as described in [2, 26] and reproduced in Table 3. The heat sink is placed close to the bottom



(a) 2D-2a baseline processor with checker core



(b) Example 3D stack with face-to-face (F2F) bonding (based on figure in [2])



(c) 3D-2a processor with checker core on stacked die (metal layers not shown)

Figure 2. Baseline (a) and proposed (c) processor models and example F2F 3D stack (b). Figure not drawn to scale.

Branch Predictor	Comb. bimodal/2-level (per core)	Bimodal Predictor Size	16384
Level 1 Predictor	16384 entries, history 12	Level 2 Predictor	16384 entries
BTB	16384 sets, 2-way	Branch Mpred Latency	12 cycles
Instruction Fetch Queue	32	Fetch width/speed	4/1
Dispatch/Commit Width	4	IssueQ size	20 (Int) 15 (FP)
Reorder Buffer Size	80	LSQ size	40
Integer ALUs/mult	4/2	FP ALUs/mult	1/1
(Single thread) L1 I-cache	32KB 2-way	L1 D-cache	32KB 2-way, 2-cyc
L2 NUCA cache	6MB 6-way	Frequency	2 GHz
I and D TLB	256 entries, 8KB page size	Memory Latency	300 cycles for the first chunk

Table 1. Simplescalar Simulation Parameters

Block	Area	Average Power
Leading Core	19.6 mm^2	35 W
In-order Core	5 mm^2	7 W / 15 W
1MB L2 Cache Bank	5 mm^2	0.732 W dynamic per access and 0.376 W static power
Network Router	0.22 mm^2	0.296 W

Table 2. Area and Power Values for various Blocks

Bulk Si Thickness die1(next to heatsink)	750μm
Bulk Si Thickness die2 (stacked die)	20μm
Active Layer Thickness	1μm
Cu Metal Layer Thickness	12μm
D2D via Thickness	10μm
Si Resistivity	0.01 $(mK)/W$
Cu Resistivity	0.0833$(mK)/W$
D2D via Resistivity (accounts for air cavities and die to die interconnect density)	0.0166 $(mK)/W$
HotSpot Grid Resolution	50x50
Ambient temperature	47 °C

Table 3. Thermal Model Parameters

die that includes the high-power out-of-order leading core. The leading core's floorplan is based on the Alpha EV7. The Wattch [4] power model has been extended to model power consumption for the processor at 65 nm technology, at 2 GHz, and 1 V supply voltage. We have added gate capacitance and metal capacitance scaling factors to scale Wattch's 90 nm model to 65 nm as done in Orion [42]. The aggressive clock gating model (cc3) has been assumed throughout and a turn-off factor of 0.2 has been used to account for higher leakage power in a 65 nm process. The power and area values for the L2 cache's network routers have been derived from Orion [42]. These power values are fed to the Hotspot thermal model to estimate temperatures. For our performance simulations, we employ a multi-threaded version of Simplescalar-3.0 [5] for the Alpha AXP ISA.

For our cache power, delay, and area models, we employ CACTI-4.0 [39]. For large L2 cache organizations with non-uniform cache access (NUCA), we employ the methodology in [23] to compute delay and power per access. For example, when modeling a large 15 MB L2 cache, the cache is partitioned into 15 1 MB banks. The banks are accessed via a grid network where each hop consumes four cycles (one cycle for the link latency and three cycles for the router latency). The link latency is computed based on wire delay speeds for top level metal [23] and the router latency is typical of conventional 4-stage routers where the switch and virtual channel allocator stages execute in parallel [8]. The area of the router is taken into account in estimating the area of each cache bank. The size of

the large cache is in keeping with modern trends (the Intel Montecito has a 12 MB L3 cache for each core [20]). The cache partitioning not only allows improved performance from non-uniform access, it also allows us to leverage additional cache banks on the second die in a seamless manner. Our baseline single-die processor has 6 1 MB banks surrounding the core and operates as a 6-way 6 MB L2 cache. This model is referred to as "2d-a" throughout the paper, where "a" refers to the total relative area. When the second die is added, the spare area on the second die allows us to implement nine additional 1 MB banks, modeled as a total 15-way 15 MB L2 cache. This model is referred to as "3d-2a" because it has twice the total area of the "2d-a" baseline. For comparison purposes, we also model a 2D die that incorporates a checker core and a 15-way 15 MB L2 cache, thus providing as many transistors as the 3D chip and referred to as "2d-2a". It must be noted that the 2d-2a model has a larger surface area and hence a larger heat sink than the other two models. All relevant simulation parameters are summarized in Tables 1, 2 and 3.

As an evaluation workload, we use the 7 integer and 12 floating point benchmark programs from the SPEC2k suite that are compatible with our simulator. The executables were generated with peak optimization flags. The programs are simulated over 100 million instruction windows identified by the Simpoint framework [36].

We model two different policies for L2 NUCA cache access. In the first approach, we distribute the sets across cache banks; given an address, the request is routed to a unique cache bank. While this approach is simpler, it causes all banks to be uniformly accessed, potentially resulting in long average cache access times. In a second approach, we distribute the ways across cache banks. Since a block can now reside anywhere, we need a mechanism to search for the data. Instead of sending the request to multiple banks, we maintain a centralized tag array near the L2 cache controller. The tags are first looked up before forwarding the request to the corresponding L2 cache bank. With this policy, one of the L2 cache banks in the floorplan is replaced by this centralized tag structure. Depending on the approach, an increase in cache size is modeled as an increase in the number of sets or ways. We adopt the distributed-sets policy for most of the evaluations in the paper.

3.2. Thermal Overheads of 3D checkers

We first evaluate the thermal impact of implementing a checker core on a separate die. Wattch does not provide accurate relative power estimates for in-order and out-of-order cores as it does not model control paths. There is also a wide spectrum in power and area characteristics of commercial implementations of in-order cores. For example, each core in the Sun Niagara consumes 8 W [14] and the Alpha in-order cores EV4 and EV5 consume 1.83 W and 4.43 W respectively at 100 nm technology [15]. Hence, we present most of our results for a range of in-order core power numbers and highlight the conclusions for two in-order core implementations: one that consumes an optimistic 7 W and another that consumes a pessimistic 15 W.

Figure 3(a) shows the floorplan of a baseline 2D implementation of a single-die single-core processor model (model 2d-a) and Figure 3(b) shows the floorplan of the upper die in a 3D chip that includes the in-order core (upper die of model 3d-2a). The layout in Figure 3(c) shows a second baseline 2D layout (2d-2a) that has as much total area and the same structures as the 3D processor implementation, but implemented in 2D. The floorplan of the out-of-order core is modeled loosely after the EV7 core's floorplan. Since the original EV7 floorplan is based on a 130 nm process, we scale the area down using non-ideal scaling factors for SRAM and logic, as described in [10] for a 65 nm process. To reduce temperature and the length of the inter-core interconnects in 3D, the inter-core buffers of the checker core on the upper die are placed as close as possible to the cache structures of the leading core. We have also attempted to reduce temperature by placing L2 cache banks above the hottest units in the leading core, when possible.

Figure 4 shows the peak temperatures averaged across all benchmark programs for the two baseline 2D organizations and for the 3D implementation (for various checker core power values). Per-benchmark results are shown in Figure 5. The horizontal bar (labeled "2d-a") shows the temperature of the baseline 2D chip that has a 6 MB L2 cache and no checker core (the chip that will be purchased by customers with low reliability requirements). The light bars (labeled "3d-2a") represent the reliable chip with an additional checker core and 9 MB L2 cache snapped on the "2d-a" baseline. The dark bars (labeled "2d-2a") represent a 2D implementation of the 3D chip (with 15 MB L2 cache and checker core). If the checker power is low enough (less than 10 W), the 2d-2a baseline has a lower temperature than the 2d-a baseline because of the lateral heat spreading promoted by the many (relatively) cool L2 cache banks and its larger heat sink. Each L2 cache bank is also cooler in the 2d-2a case as the same number of L2 accesses are distributed across more banks. We also observe that the thermal overheads due to 3D stacking are not high for most checker core values. For a checker core dissipating 7 W, the 3D model is only 4.5 degrees hotter than the 2d-2a model and 4 degrees hotter than the 2d-a model. If the checker core consumes roughly half the power of the leading core (15 W), it is 7 degrees hotter than the 2d-a model. Thus, 3D integration does have a cost. Even though 3D integration enables a 3.4 W power reduction, relative to the 2d-2a model, because of shorter interconnects (clarified in the next sub-section), it still leads to higher power densities. Not surprisingly, the temperature increase is a strong function of the power density of the hottest block in

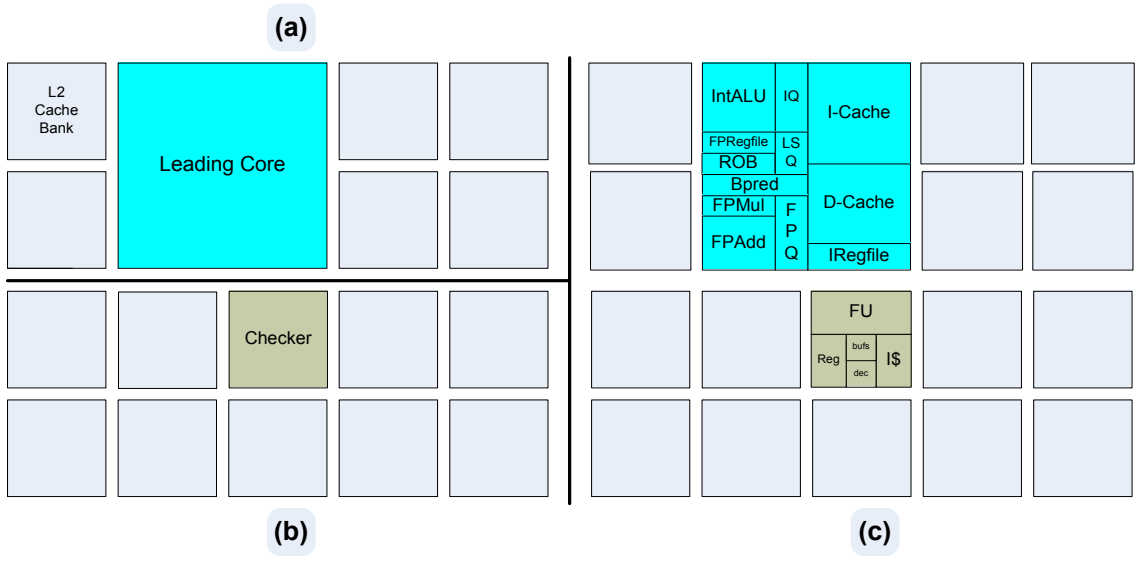


Figure 3. **Floorplans for each processor model: (a) represents the baseline 2d-a model and the bottom die of the 3d-2a model, (b) represents the upper die of the 3d-2a model, (c) represents the 2d-2a model.**

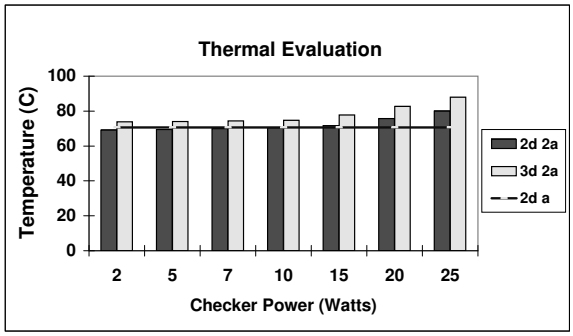


Figure 4. **Thermal overhead analysis of 3D checker**

the in-order core. If the power density of the 15 W checker is further doubled, the temperature rise can be as high as 19 degrees, although, this is an admittedly pessimistic scenario. Higher temperatures will either require better cooling capacities or dynamic thermal management (DTM) that can lead to performance loss.

We also modeled the effect of temperature on leakage power in L2 cache banks. Very few banks in 3d have higher temperature if stacked above a hot unit in the main core. We found the overall impact of temperature on leakage power of caches to be negligible.

It is possible to lower temperatures by not employing the space on the top die for L2 cache. For the 3d-2a case with the 7 W (15 W) checker core, if the top die's checker core is surrounded by inactive silicon, the temperature reduces only by 2 degrees (1 degree). As we show in the next sub-section, the impact of this choice on performance is marginal given the working sets of the SPEC2k applications. However, it is unlikely that manufacturers will choose to waste silicon in this manner, especially since the cache needs of multi-core chips will likely be higher. Hsu et al. [13] show that for heavily multi-threaded workloads, increasing the cache capacity by many mega-bytes yields significantly lower cache miss rates. A second approach to lowering temperature is to move the checker core to the corner of the top die (while incurring a higher cost for inter-core communication). Our experiments show that this approach helps reduce temperature by about 1.5 degrees.

Results Summary: A low-power (7 W) checker core causes a temperature increase of only 4.5 degrees (compared to an unreliable baseline), while a high-power (15 W) checker core can increase temperature by 7 degrees.

3.3. Performance

Figure 6 shows performance of the leading core for all three models without any thermal constraint and using the NUCA policy where sets are distributed across banks. We also show the performance of a model (3d-Checker) where the top die consists of only the checker core and no additional L2 cache banks. As described in Section 2, the checker core is operated at a frequency such that it rarely stalls the leading thread and imposes a negligible performance overhead. The 3d-checker model confirms this fact as its performance is the same as the 2d-a baseline that does not have a checker core and has an equal cache capacity. Hence, most of the performance differences in the 2d-a, 2d-2a, and 3d-2a models can be attributed to the L2 cache organization. For the SPEC benchmarks, a 15 MB L2 cache does not offer much better cache hit rates than a 6 MB cache (the number of L2 misses per 10K instructions reduces from 1.43 to 1.25). The 2d-2a model performs a little worse than the 2d-a model because cache values are more spread out and the average latency for an L2 hit increases from 18 cycles (2d-a) to 22 cycles (2d-2a). The

Checker core doesn't do the slowdown. it is in fact the L2 cache

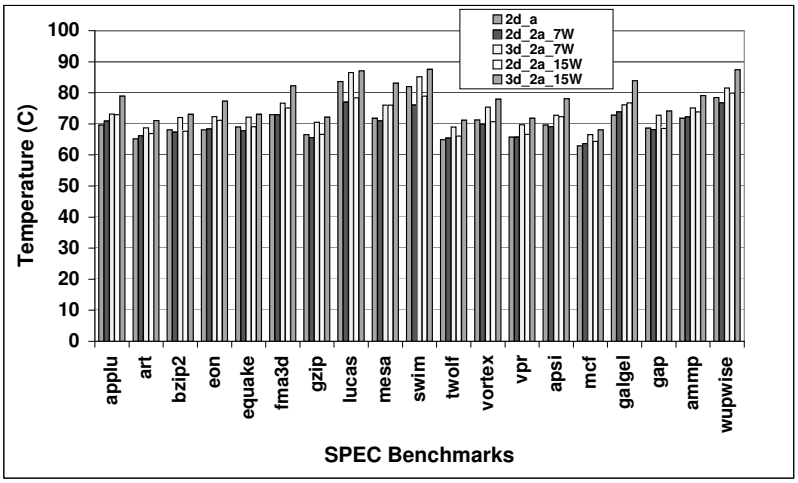


Figure 5. **Thermal overhead analysis of 3D checker for each benchmark**

move to 3D does not help reduce the average L2 hit time compared to 2d-a as the average horizontal distance to the banks remains the same. Relative to the 2d-2a baseline, the 3D chip yields a 5.5% performance improvement.

Even with a NUCA policy that distributes ways, we observe the same relative results between 2d-a, 2d-2a, and 3d-2a. However, the performance with the the distributed-way policy is slightly ($< 2\%$) better than with the distributed-set policy because data blocks tend to migrate closer to the L2 cache controller (especially since the program working sets for SPEC2k are typically smaller than the L2 capacity).

The 3D reliable processor causes a temperature increase that can impact packaging/cooling cost, leakage power, and even reliability. Hence, it is also important to evaluate the various processor models while maintaining a constant thermal constraint. We execute the 3D processor at a lower voltage and frequency than the 2D baseline processor until its average peak temperature matches that of the baseline. Similar to the methodology in [2], we assume that voltage and frequency scale linearly for the voltage ranges considered here. We observed that a 3D processor with a 7 W checker (15 W checker) operating at a frequency of 1.9 GHz (1.8 GHz) has the same thermal characteristics as a 2D baseline processor operating at a frequency of 2 GHz. Thus, assuming constant thermals, the design of a 3D reliable processor entails a performance loss of 4.1% for the 7 W in-order checker core and 8.2% for the 15 W checker core (the performance loss is a little less than the frequency reduction because memory latency is un-changed).

Results Summary: Without a thermal constraint, the second die has a negligible impact on performance because the additional cache space neither improves hit rates nor degrades average access time. The extra cache space may be more valuable if it is shared by multiple threads in a large multi-core chip [13]. With a thermal constraint, the performance loss (4%/8%) is a function of the checker core power (7 W/15 W).

Data	Width	Placement of via
Loads	$load\_issue\_width \times 64$	LSQ
Branch outcome	$branch\_pred\_ports \times 1$	Bpred
Stores	$store\_issue\_width \times 64$	LSQ
Register values	$issue\_width \times 192$	Register File
Die to die L2 cache transfer	384	L2 Cache Controller

Table 4. **D2D interconnect bandwidth requirements**

3.4. Interconnect Evaluation

We next consider the changes to interconnect power and area as a result of the 3D organization. Our methodology is based on the modeling described in [24].

To quantify the requirement for the number of die-to-die (d2d) vias, we compute the data that gets communicated across both dies per cycle. As described in the previous section, the leading core sends load values, register values, and branch outcomes to the checker core and the checker core sends store values back. Table 4 quantifies the maximum per-cycle data bandwidth of these inter-die transfers and where the d2d vias need to be placed. The maximum bandwidth is a function of the core's issue/execution width. For a 4-wide core, 1025 vias are required between the leading and checker cores. In addition, we introduce a 384-bit via pillar to transmit addresses and data between the L2 cache controller on the lower die and the L2 banks on the upper die (64-bit address, 256-bit data, and 64-bit control signals).

The d2d vias impose a minor power and performance overhead. State-of-the-art 3D integration employs thin dies that results in d2d via lengths from 5 μm to 20 μm [9]. Assuming a d2d via length of 10 μm, the worst-case capacitance of a d2d via surrounded by 8 other vias is computed as 0.594e-15 $F/\mu m$. The capacitance for a via of length 10 μm is therefore 0.59e-14 F. The dynamic power at 65 nm (2 GHz frequency and 1 V) is calculated to be 0.011 mW for each d2d via. The total power consumed by all 1409 vias adds up to only 15.49 mW. State-of-the-art width of each via is 5 μm [9]. The area overhead for all

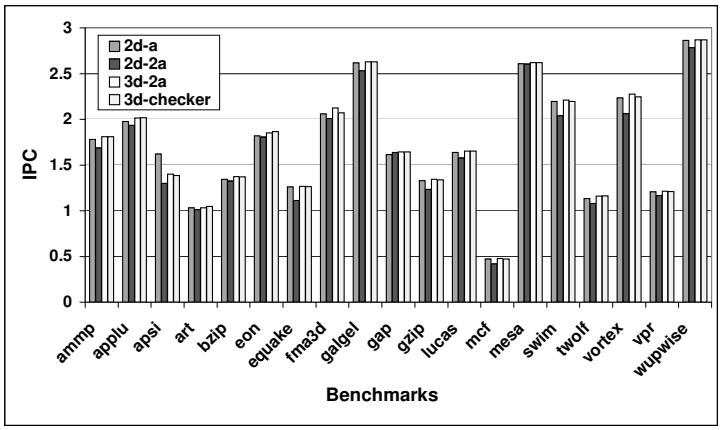


Figure 6. **Performance Evaluation (with a NUCA policy that distributes sets across banks).**

the vias is 0.07 mm^2 assuming that the width and spacing between each via is 5 μm.

We next look at metalization area savings due to reduction in horizontal interconnect length. The total length of horizontal wires in 2D for inter-core communication is 7490 mm and in 3D is 4279 mm. If we assume that these are global wires then the total metalization area (given by $pitch \times length$, pitch of 210 nm at 65 nm technology) in 2d is $1.57mm^2$ and in 3d is $0.898mm^2$ resulting in net metal area savings of 42%. Similarly, we compute net metal savings in L2 interconnect for a NUCA cache. The 2d-a baseline has the least metal area requirement of $2.36mm^2$, and the 2d-2a baseline has a requirement of $5.49mm^2$. The 3D NUCA layout requires $4.61mm^2$, about 16% lower than 2d-2a.

Finally, we look at power consumption of these metal wires. We assume that the interconnect is power-optimized and compute the total power based on methodology in [6]. We observe that the L2 cache and inter-core interconnect power consumed by the 2d-a baseline is 5.1 W, 2d-2a consumes 15.5 W, and 3d-2a consumes 12.1 W. The net power savings due to 3D integration compared to 2d-2a is 3.4 W. The additional 10 W interconnect power in moving from 2d-a to 3d-2a is primarily because of the additional L2 banks – the transfer of register and load values incurs an overhead of only 1.8 W.

Results Summary: The power and area overheads of the inter-die vias are marginal. The horizontal interconnects that feed the checker core consume only 1.8 W. The interconnects for the larger cache on the top die are responsible for an additional 9 W of power.

3.5. Conservative Timing Margins

In the previous section, we argued that the checker core is capable of detecting a single dynamic timing error that manifests in an incorrect register value. Recovery is also almost always possible with an ECC-protected register file, unless the timing error (or soft error) happens when writing the result into the checker core's register file after verification. Unfortunately, dynamic timing errors are often correlated and multiple such errors in the same or successive cycles can happen. The first such error will be detected by the checker core, but recovery may not be possible if the checker's register file state is corrupted beyond repair (ECC may not be able to repair multiple bit flips, nor a result being written to the wrong register). Further, the checker core is not resilient to faults in the control path. Hence, the processor's ability to recover from an error can be greatly improved if the checker core is made more resilient to dynamic timing errors and soft errors. In this sub-section, we explore the option of introducing conservative timing margins in every pipeline stage of the checker core. First, this reduces the chances of a soft error as there is enough slack within the cycle time to allow the circuit to re-stabilize after the pulse and mask the error. Second, even with timing variations, it improves the likelihood that the result is ready when the pipeline latch receives the clock edge.

We first assume that the peak frequency of the checker core is maintained the same as the leading core's peak frequency. To introduce a conservative timing margin in each stage, we must perform less work in each stage. Hence, the work required of the checker core is now distributed across many more pipeline stages, *i.e.*, the checker is implemented as a deep pipeline. Note that the deep pipeline is not being exploited to improve the checker's clock frequency, but to provide more slack in each pipeline stage.

Unfortunately, the power dissipation of a processor increases as its pipeline depth increases due to the higher number of required latches and increased bypassing complexity. To model the power overheads of deep pipelining, we employ the analytical models proposed by Srinivasan et al. [38] to compute power-optimal pipeline depths. More details of this model and assumptions can be found in the original work [38].

Table 5 summarizes the impact of pipeline depth on total power, relative to a baseline pipeline that has a cycle time of 18 FO4. While performing less work in each pipeline stage helps reduce error rates, the power cost is enormous.

Pipeline depth	Dynamic Power Relative	Leakage Power Relative	Total Power Relative
18 FO4	1	0.3	1.3
14 FO4	1.65	0.32	1.97
10 FO4	1.76	0.36	2.12
6 FO4	3.45	0.53	3.98

Table 5. Impact of pipeline scaling on power overheads in terms of baseline dynamic power consumption

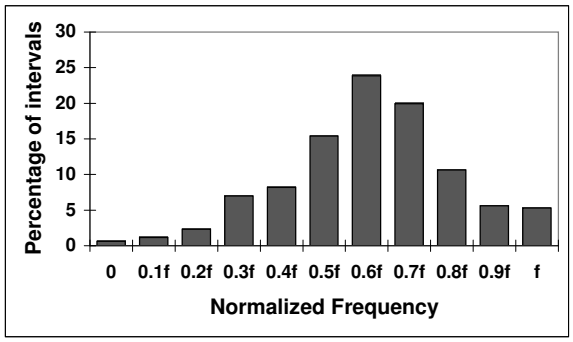


Figure 7. Histogram showing percentage of intervals at each normalized frequency.

Even if the circuits in each stage take up 14 FO4 gate delays, the power consumption of the checker core increases by 50%. Besides, it may be difficult to pipeline the structures of the processor into the required number of stages. Hence, we don't consider this option any more in this paper.

Fortunately, our checker core is designed such that it often operates at a frequency much lower than the peak frequency. As described in Section 2, the checker core lowers its frequency if the slack between the leading and checker core decreases. In Figure 7, we show a histogram of the percentage of time spent at each frequency level. For most of the time, the checker operates at 0.6 times the peak frequency of 2 GHz. This means that the circuits in each pipeline stage have usually finished their operation within half the cycle time and minor variations in circuit delay will likely not impact the value latched at the end of the pipeline stage. Thus, a natural fall-out of our checker core design is that it is much more resilient to dynamic timing errors and soft errors.

Results Summary: Deep pipelining has an inordinate power overhead. Because of its high ILP, the checker core can typically operate at a fraction of its peak frequency. This allows each pipeline stage to already have plenty of slack and tolerate noise.

4. The Impact of Heterogeneity on the Checker

3D integration enables the snap-on functionality of a checker core for customers that require high reliability. It also enables each die to be manufactured in a different process. This has especially favorable implications for the design of a checker core. Unlike the leading core (that must be optimized for performance), the checker core can afford to give up some performance for a manufacturing process that provides high reliability. This allows the checker core to be more resilient to faults in the control path, something that cannot be handled by the register checking process. Further, if the checker core has a much lower probability for an error, its result can be directly employed in case of a disagreement between the leading and checker cores. If the checker core is equally likely to yield an error as the leading core, recovery requires an ECC-protected checker register file and possibly even a third core to implement triple modular redundancy (TMR). In this section, we consider the use of an older process technology to implement the checker core's die. We have seen that the checker core is already more reliable because it has conservative timing margins – the older process further augments checker core reliability.

The use of an older process technology has the following impact:

- Parameter variations and dynamic timing errors are reduced.
- The critical charge required to switch a transistor's state increases, reducing the probability of a soft error.
- Leakage power is reduced (note that a large fraction of the die is L2 cache that dissipates large amounts of leakage).
- Dynamic power increases because of higher switching capacitance.
- Delay of a pipeline stage increases, possibly necessitating a lower clock speed and poorer performance.

We next analyze each of these factors.

Tech gen nm	Vth Variability	Circuit Performance Variability	Circuit Power Variability
80	26%	41%	55%
65	33%	45%	56%
45	42%	50%	58%
32	58%	57%	59%

Table 6. Impact of Technology Scaling on Variability

Figure 8 shows scaling of per-bit SRAM soft-error rate across process nodes due to neutron and alpha particles. Figure 9 shows the probability of per-bit multiple-bit upset (MBU) rate as a function of decreasing critical charge. Even though single-bit error rates per transistor are reducing at future technologies, the overall error rate is increasing because of higher transistor density. The multi-bit error rate per transistor is increasing at future technologies and can be problematic even with a checker core. Further, Table 6 shows ITRS data [35] exhibiting projected variability (as a +/- percentage change from nominal) in threshold voltage, performance, and power, as a function of process technology. All of this data argues for the use of an older technology to implement an error-tolerant checker core die.

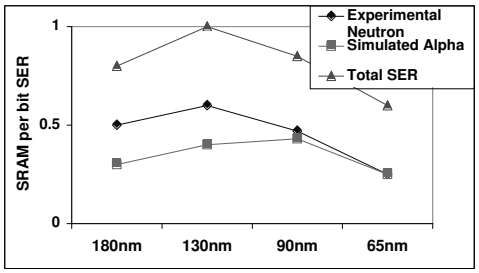


Figure 8. **SRAM Single-bit Soft Error Scaling Rate**

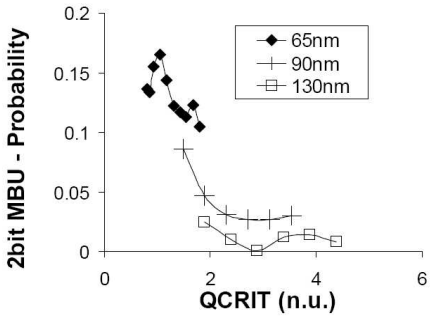


Figure 9. **SRAM Multi-bit Soft Error Probability [33]**

To model power consumption of the same core implemented on different process technologies, we employ device model parameters from ITRS reports [35] (summarized in Table 7). These were substituted in equations for leakage and dynamic power to generate the relative power estimates for various technologies (listed in Table 8).

The data in Table 8 shows that if the older process is 90 nm, the dynamic power consumption is more than 2x and 3x the dynamic power of the core implemented in a more recent technology (65 nm and 45 nm, respectively). However, leakage power is reduced quite significantly by moving to the older process. Thus, the use of an older technology may increase/decrease the overall power dissipation depending on the relative contribution of leakage and dynamic power. If the die containing the checker core is primarily composed of L2 cache banks, leakage likely accounts for a large fraction of die power. It must be noted that these arguments are less clear for newer process technologies as new materials are expected to alleviate leakage concerns (as is observed for the comparison of 65 nm and 45 nm technology nodes in Table 8).

For our power-hungry checker core and L2 cache models, we observed a total power dissipation of 18 W for the checker die at 65 nm technology (14.5 W for the checker core and 3.5 W for the 9 MB of L2 cache banks). If we maintain a constant die area and migrate to 90 nm technology, the checker core now consumes 23.7 W and the 5 MB of L2 cache consumes 1.2 W, leading to a net power increase of 6.9 W. In spite of the power increase, we observe a drop in temperature (of 4 °C, compared to the 3D chip with homogeneous 65 nm technology dies). This is a result of an increase in checker core area and an overall reduction

Tech gen nm	Voltage V	Gate length nm	Capacitance per um	Sub-threshold leakage current per um
90	1.2	37	8.79E-16	0.05
65	1.1	25	6.99E-16	0.2
45	1	18	8.28E-16	0.28

Table 7. **Impact of technology scaling on various device characteristics**

Tech gen nm/nm	Dynamic Power Relative	Leakage Power Relative
90/65	2.21	0.4
90/45	3.14	0.44
65/45	1.41	0.99

Table 8. **Impact of technology scaling on power consumption**

in its power density.

Finally, the older process technology causes an increase in the delay of each circuit and pipeline stage. We have assumed a peak clock frequency of 2 GHz in this study, meaning each pipeline stage can take as long as 500 ps (including overheads for latch, skew, and jitter). If the checker core is now implemented in a 90 nm process instead of a 65 nm process, a pipeline stage that took 500 ps in 65 nm process, now takes 714 ps. This limits the peak frequency of the checker core to 1.4 GHz. Our simulations have shown that a checker core needs to operate at an average frequency of only 1.26 GHz to keep up with a 2 GHz leading core. If the checker core begins to fall behind the trailer, it must increase its frequency, but is limited to a peak frequency of only 1.4 GHz. This modification causes only a minor slowdown to the leading core (3%). The longer delays for the circuits imply less slack until the next clock edge. On average, given the frequency profile of the checker core, there still remains non-trivial slack in each pipeline stage even with the slower circuits.

The L2 cache banks in the older process operate at the same frequency (2 Ghz) as the other banks on the lower die. The access time for each L2 cache bank in the older process increases by a single cycle. Overall cache access times are not greatly impacted as the fewer cache banks lead to fewer hops on average.

Results Summary: An older process increases overall power consumption, but reduces the power density for the hottest units on the chip, allowing overall temperature to decrease by up to 4 degrees. Thus, with a constant thermal constraint, overall performance actually improves over the design with homogeneous dies. A 15 W checker core implemented in an older process will either increase temperature by 3 degrees or suffer a performance loss of 4% under a constant thermal constraint (relative to the 2D baseline). The older process is more resilient to faults and a large portion of the execution continues to enjoy conservative timing margins for the pipeline stages.

Discussion: We observed that the DFS heuristic for matching throughput can play a significant role in power and ther-

mal properties of the checker. An aggressive heuristic that can slow down the checker core further results in lower temperature values. However, such an aggressive mechanism can stall the main core more frequently and result in performance loss compared to an unreliable 2D baseline. In our models, we have employed a less aggressive heuristic that doesn't degrade the main core's performance by itself. This heuristic does lead to higher temperatures, which in turn lead to thermal emergencies and lower performance.

5. Related Work

Many fault-tolerant architectures [1, 12, 19, 22, 28, 30, 31, 41, 43] have been proposed over the last few years. Not all of these designs are amenable to a convenient 3D implementation. Most reliable implementations that execute the redundant thread on a separate core will likely be a good fit for a 3D implementation. While our evaluation focuses on the specific implementation outlined in [19], the results will likely apply to other RMT implementations that provide low-overhead redundancy on a separate core.

There has been much recent work on architectural evaluations involving 3D chips. Many have focused on improving single-core performance and power [2, 7, 26, 44]. Some of this attention has focused on implementing a cache in 3D [25, 29, 40], even in the context of a multi-core NUCA layout [16]. Few studies have considered using additional dies entirely for SRAM or DRAM [2, 17, 18].

The work in this paper is most closely related to the work by Mysore et al. [24]. That work focuses on implementing software profiling and analysis engines on the second die. Further, it does not explore microarchitectural design choices for the second die and does not consider heterogeneous dies. Many of the findings of this study are specific to the implementation of a checker core on the second die: most notably, the effect of technology, pipelining, and frequency scaling on the ability of the checker core to resist transient and dynamic timing errors. A software profiling and analysis application, on the other hand, is different in its needs: it may be more tolerant of errors and less tolerant of performance slowdowns.

6. Conclusions

In this paper, we propose a 3D reliable processor organization and evaluate many of its design considerations. A checker core on the upper die helps reduce interconnect lengths and part of the upper die can also be employed to increase L2 cache size. The isolation of the checker core to a separate die reduces the impact on the leading core's layout, wiring, and cycle time. The snap-on functionality allows customers the option to detect and recover from failures, while not impacting the manufacturing cost for customers that are less concerned about reliability. The most significant impact of this design is a temperature increase of up to 7 °C or alternatively, a performance loss of up to 8% if a constant thermal constraint is imposed. The impact on temperature and performance is much less if we assume a simple low-power microarchitecture for the in-order core. We also show that the checker core is more error-resilient because it typically operates at a much lower frequency. Error tolerance can be further increased by implementing the upper die on an older process that suffers less from dynamic timing errors and soft errors. The move to an older technology increases power consumption, but reduces temperature because the power density of the hottest block is lowered. This helps limit the overhead of the checker core to a 3 °C temperature increase or a 4% performance loss. Our results largely argue in favor of heterogeneous dies for this specific application domain of low-overhead redundancy.

References

[1] T. Austin. DIVA: A Reliable Substrate for Deep Submicron Microarchitecture Design. In *Proceedings of MICRO-32*, November 1999.

[2] B. Black, M. Annavaram, E. Brekelbaum, J. DeVale, L. Jiang, G. Loh, D. McCauley, P. Morrow, D. Nelson, D. Pantuso, P. Reed, J. Rupley, S. Shankar, J. Shen, and C. Webb. Die Stacking (3D) Microarchitecture. In *Proceedings of MICRO-39*, December 2006.

[3] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi, and V. De. Parameter Variations and Impact on Circuits and Microarchitecture. In *Proceedings of DAC*, June 2003.

[4] D. Brooks, V. Tiwari, and M. Martonosi. Wattch: A Framework for Architectural-Level Power Analysis and Optimizations. In *Proceedings of ISCA-27*, pages 83–94, June 2000.

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

[6] 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.

[7] J. Cong, A. Jagannathan, Y. Ma, G. Reinman, J. Wei, and Y. Zhang. An Automated Design Flow for 3D Microarchitecture Evaluation. In *Proceedings of ASP-DAC*, January 2006.

[8] W. Dally and B. Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann, 1st edition, 2003.

[9] S. Das, A. Fan, K.-N. Chen, and C. Tan. Technology, Performance, and Computer-Aided Design of Three-Dimensional Integrated Circuits. In *Proceedings of International Symposium on Physical Design*, April 2004.

[10] H. de Vries. Die area scaling factor. (`http://www.chip-architect.com /news/2007_02_19_Various_Images.html`).

[11] D. Ernst, N. Kim, S. Das, S. Pant, T. Pham, R. Rao, C. Ziesler, D. Blaauw, T. Austin, T. Mudge, and K. Flautner. Razor: A Low-Power Pipeline Based on Circuit-Level Timing Speculation. In *Proceedings of MICRO-36*, December 2003.

[12] M. Gomaa, C. Scarbrough, and T. Vijaykumar. Transient-Fault Recovery for Chip Multiprocessors. In *Proceedings of ISCA-30*, June 2003.

[13] L. Hsu, R. Iyer, S. Makineni, S. Reinhardt, and D. N. l. Exploring the Cache Design Space for Large Scale CMPs. In *Proceedings of dasCMP*, November 2005.

[14] K. Krewell. Sun's Niagara Pours on the Cores. *Microprocessor Report*, 18(9):11–13, September 2004.

[15] R. Kumar, K. Farkas, N. Jouppi, P. Ranganathan, and D. Tullsen. Single ISA Heterogeneous Multi-Core Architectures: The Potential for Processor Power Reduction. In *Proceedings of the 36th International Symposium on Micro-Architecture*, December 2003.

[16] 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.

[17] C. C. Liu, I. Ganusov, M. Burtscher, and S. Tiwari. Bridging the Processor-Memory Performance Gap with 3D IC Technology. *IEEE Design and Test of Computers*, 22:556–564, November 2005.

[18] G. Loi, B. Agrawal, N. Srivastava, S. Lin, T. Sherwood, and K. Banerjee. A Thermally-Aware Performance Analysis of Vertically Integrated (3-D) Processor-Memory Hierarchy. In *Proceedings of DAC-43*, June 2006.

[19] N. Madan and R. Balasubramonian. Power Efficient Approaches to Redundant Multithreading. *IEEE Transactions on Parallel and Distributed Systems (Special issue on CMP architectures)*, Vol.18, No.8, August 2007.

[20] C. McNairy and R. Bhatia. Montecito: A Dual-Core, Dual-Thread Itanium Processor. *IEEE Micro*, 25(2), March/April 2005.

[21] S. Mukherjee, J. Emer, and S. Reinhardt. The Soft Error Problem: An Architectural Perspective. In *Proceedings of HPCA-11 (Industrial Session)*, February 2005.

[22] S. Mukherjee, M. Kontz, and S. Reinhardt. Detailed Design and Implementation of Redundant Multithreading Alternatives. In *Proceedings of ISCA-29*, May 2002.

[23] N. Muralimanohar and R. Balasubramonian. Interconnect Design Considerations for Large NUCA Caches. In *Proceedings of ISCA*, June 2007.

[24] S. Mysore, B. Agrawal, N. Srivastava, S. Lin, K. Banerjee, and T. Sherwood. Introspective 3D Chips. In *Proceedings of ASPLOS-XII*, October 2006.

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

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

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

[28] J. Ray, J. Hoe, and B. Falsafi. Dual Use of Superscalar Datapath for Transient-Fault Detection and Recovery. In *Proceedings of MICRO-34*, December 2001.

[29] 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.

[30] S. Reinhardt and S. Mukherjee. Transient Fault Detection via Simultaneous Multithreading. In *Proceedings of ISCA-27*, pages 25–36, June 2000.

[31] E. Rotenberg. AR-SMT: A Microarchitectural Approach to Fault Tolerance in Microprocessors. In *Proceedings of 29th International Symposium on Fault-Tolerant Computing*, June 1999.

[32] 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`).

[33] N. Seifert, P. Slankard, M. Kirsch, B. Narasimham, V. Zia, C. Brookreson, A. Vo, S. Mitra, and J. Maiz. Radiation Induced Soft Error Rates of Advanced CMOS Bulk Devices. In *Proceedings of IEEE International Reliability Physics Symposium*, 2006.

[34] G. Semeraro, G. Magklis, R. Balasubramonian, D. Albonesi, S. Dwarkadas, and M. Scott. Energy Efficient Processor Design Using Multiple Clock Domains with Dynamic Voltage and Frequency Scaling. In *Proceedings of HPCA-8*, pages 29–40, February 2002.

[35] Semiconductor Industry Association. International Technology Roadmap for Semiconductors 2005. `http://www.itrs.net/Links/2005ITRS/Home2005.htm`.

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

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

[38] V. Srinivasan, D. Brooks, M. Gschwind, P. Bose, V. Zyuban, P. Strenski, and P. Emma. Optimizing Pipelines for Power and Performance. In *Proceedings of MICRO-35*, November 2002.

[39] D. Tarjan, S. Thoziyoor, and N. Jouppi. CACTI 4.0. Technical Report HPL-2006-86, HP Laboratories, 2006.

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

[41] T. Vijaykumar, I. Pomeranz, and K. Cheng. Transient-Fault Recovery via Simultaneous Multithreading. In *Proceedings of ISCA-29*, May 2002.

[42] H.-S. Wang, X. Zhu, L.-S. Peh, and S. Malik. Orion: A Power-Performance Simulator for Interconnection Networks. In *Proceedings of MICRO-35*, November 2002.

[43] N. Wang, J. Quek, T. Rafacz, and S. Patel. Characterizing the Effects of Transient Faults on a High-Performance Processor Pipeline. In *Proceedings of DSN*, June 2004.

[44] Y. Xie, G. Loh, B. Black, and K. Bernstein. Design Space Exploration for 3D Architectures. *ACM Journal of Emerging Technologies in Computing Systems*, 2(2):65–103, April 2006.