Introspective 3D Chips

Shashidhar Mysore Banit Agrawal Navin Srivastava* Sheng-Chih Lin*
Kaustav Banerjee* Timothy Sherwood

Department of Computer Science; *Department of Electrical and Computer Engineering, University of California, Santa Barbara
{shashimc, banit, sherwood}@cs.ucsb.edu, {navins, sclin, kaustav}@ece.ucsb.edu

Abstract

While the number of transistors on a chip increases exponentially over time, the productivity that can be realized from these systems has not kept pace. To deal with the complexity of modern systems, software developers are increasingly dependent on specialized development tools such as security profilers, memory leak identifiers, data flight recorders, and dynamic type analysis. Many of these tools require full-system data which covers multiple interacting threads, processes, and processors. Reducing the performance penalty and complexity of these software tools is critical to those developing next generation applications, and many researchers have proposed adding specialized hardware to assist in profiling and introspection. Unfortunately, while this additional hardware would be incredibly beneficial to developers, the cost of this hardware must be paid on every single die that is manufactured.

In this paper, we argue that a new way to attack this problem is with the addition of specialized analysis hardware built on separate active layers stacked vertically on the processor die using 3D IC technology. This provides a modular "snap-on" functionality that could be included with developer systems, and omitted from consumer systems to keep the cost impact to a minimum. In this paper we describe the advantage of using inter-die vias for introspection and we quantify the impact they can have in terms of the area, power, temperature, and routability of the resulting systems. We show that hardware stubs could be inserted into commodity processors at design time that would allow analysis layers to be bonded to development chips, and that these stubs would increase area and power by no more than $0.021mm^2$ and 0.9% respectively.

Categories and Subject Descriptors C. Computer Systems Organization. [*C.1 Processor Architectures*]
General Terms Design, Performance
Keywords Introspection, Hardware Support for Profiling, 3D Architectures

1. Introduction

Developing high quality software for a modern computer system is no easy task. Performance critical applications are likely to execute for quadrillions of instructions, operate in a complex environment with multiple run-time components, and are increasingly responsible for managing various architectural resources including power and hardware threads. In order to battle this complexity, developers are becoming more dependent on sophisticated software analysis tools. While mixed static-dynamic analysis can be done completely in software through binary instrumentation, the amount of analysis that can be done at test-time is bounded by the performance impact that can be tolerated. In long running or interactive programs, this is especially critical.

To enable run-time analysis with low overhead many researchers have proposed the development of specialized on-chip hardware modules that can assist software developers in building more secure, more bug free, and more efficient applications. For example, a processor may be extended to dynamically insert instructions into the execution stream to profile a program for performance or buffer exploits [15, 16], analysis modules may be added to uncover the performance bottlenecks, hardware performance monitors may track the activities of the cache or branch unit [6, 13, 41, 52, 14, 21], replay boxes may be inserted for tracking down difficult to reproduce bugs [49], and a host of other mechanisms have been proposed in research literature. While the amount of information available at the hardware level makes this a natural place to add new runtime analysis functionality, the inclusion of specialized on-chip hardware is at odds with the cost and marketing constraints of those that build consumer microprocessor systems. Analysis modules can require a significant amount of area and often introduce interconnect congestion because they require signals from many different parts of the chip. In the end, processor developers are reluctant to add anything but the simplest of modules because the added cost cuts directly into the profits. These modules must be replicated on every single processor, regardless of whether they are used by the end user or not.

As an example, consider hardware performance monitors (HPMs). HPMs, such as performance counters, exist on almost every high end commercial processor sold today [17, 18, 24]. These monitors are included in the architecture specification, integrated with the design, fabricated with every single die, and rigorously verified and tested. For all of this work, ninety nine percent of users that buy a machine never use, or even think about, this hardware. It is included almost entirely for the benefit of commercial software developers who use these counters to tune and optimize their production code. While HPMs may be worth while just due to their small size, *any specialized hardware support for developers will be unused in the common case* because most consumers do not *develop* critical code, they execute it. This is not to say that the adding of developer functionality is useless, in fact we argue just the opposite in this paper. It is the case that this additional hardware is only useful to the select, though important, minority of users that write critical code for the rest of us. Given that high performance software analysis tools are needed, the challenge is enabling these techniques with a minimum of impact on typical end-user systems.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ASPLOS'06 October 21–25, 2006, San Jose, California, USA.
Copyright © 2006 ACM 1-59593-451-0/06/0010. . . $5.00

The primary goal of this paper is to explore a new method by which analysis functionality can be added to a processor. Specifically, we propose a new and modular way to add analysis hardware to next generation processors through the use of 3D IC technology. Several 3D technologies, such as those involving inter-die vias, are currently being evaluated in industry as a means of stacking multiple chips together. Some potential applications include the stacking of DRAM or bigger cache directly onto the processor die to alleviate memory pressure [26, 28, 32, 42, 45, 51, 31] and designing stacked chips of multiple processors [3]. While the details of this technology are more fully described in Section 3, the main idea is that two pieces of silicon are bonded together to form a single chip, and the two active layers of the silicon are connected through inter-die vias (called posts) which run vertically between them. This ability to interconnect multiple active layers means that we can consider optionally adding a layer to a processor specifically for analysis which would have access to most of the important signals of the system. A processor with this ability could be sold to developers, while commodity systems would simply not include this extra analysis layer. In this paper we study the potential of 3D IC technology to enable new forms of introspective chips. We more fully elaborate on some of the advantages of 3D introspection over traditional hardware integration in Section 2. To make our analysis concrete, we precisely quantify both the chip bandwidth requirements for full introspection, and the relevant characteristics of 2D and 3D IC technology in Section 3. While there are many advantages of performing analysis on a layer stacked above the main processor, it does not come for free. In Section 4 we consider the architectural impact of a 3D approach in comparison to both a system with no support for introspection and a system with introspection hardware integrated on the same die. We quantify the increase in area, the interconnect overhead, and both the power and thermal impacts of such a design.

2. Introspection in 3D

While software-only schemes are very attractive because of their flexibility and hardware independence, they always require support at the system level and inevitably perturb the software systems being tested no matter how well engineered. While researchers continue to reduce software profiling overheads through clever switching and sampling, developers will always demand heavier forms of dynamic analysis than software alone can provide non-intrusively. Most current machines already make use of some form of performance counters, and most machines now support this idea. While these counters are very useful in quantifying the performance of a machine, it is difficult to use them to assist in more complex analysis methods because of the lack of flexibility to profile application-level events and require significant software management in order to extract useful information [6]. However, if general purpose profiling hardware was added, it could be used to directly instrument and analyze an executing program no matter what software layers were used. Several researchers have already proposed the idea of including these processors on-chip. ProfileMe [21] and Relational Profiling Architecture [23] are flexible and versatile schemes for gathering profile information. Zilles and Sohi [52] in their co-processor approach, design hardware to analyze the stream and compress it to provide concise and distilled profile information to the main processor. Vaswani, et al., in [46] propose a hardware path profiler. While these approaches provide an effective way of processing data that is captured at the processor level, hardware designers in industry have been slow to include these devices for several reasons. In this section we describe the major advantages of building an analysis processor "on top of" rather than "integrated with" the main processor.

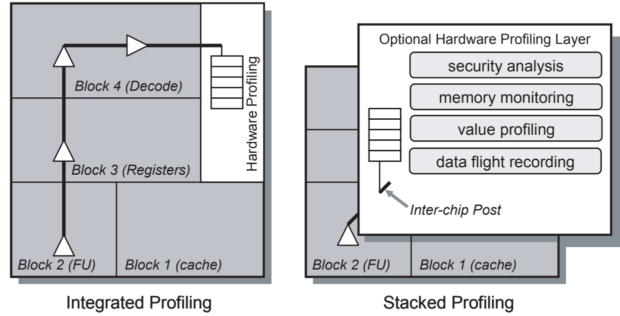

Figure 1. The traditional approach to attacking the hardware profiling problem involves integrating specialized profiling functionality on the same die as the processor. To gather information, long global wires are required which necessarily cross multiple functional blocks. To get high performance, buffers or pipeline latches are required, which in turn require access to silicon which makes for a big mess. Alternatively, in a stacked approach, only a single buffer is required to drive the post up to the analysis layer (which would be an optional feature for software developers)

2.1 Cutting Interconnect Impact

One of the most significant advantages of building monitoring hardware that sits on top of the main processor is that interconnect congestion is drastically reduced. As pointed out in a recent discussion of the challenges facing performance monitoring hardware [2], gathering data from all over the chip for centralized analysis requires a global interconnect that causes some serious headaches. The interconnect will have to cross almost every possible design block and will consume a good deal of the top metal layers. Not only will the interconnect need to join these different regions of the processor, it will also have to run at very high speed. For example, capturing the address of every load instruction would take around 64 Gbps of bandwidth. This data rate, coupled with the long distances required, necessitates wire buffering and even pipeline latches. This in turn requires that silicon is reserved in many different blocks so that the wires can get access to the needed transistors (this problem is more clearly illustrated in Figure 1). In a full custom design, this requires a significant amount of engineering effort spread across every level of the physical and architectural design. Many companies are reluctant to add the complexity of these additional global nets to their designs.

Instead of being forced to route performance data through other blocks, inter-die vias can move data out of plane to a layer specially constructed for gathering and analyzing run-time information (how this is done is described in Section 3.2.2). Of course, this does not come for free, each inter-layer via (or post) could be up to 5μ on a side. Space needs to be reserved for the gates that drive the posts, and switching these large pieces of metal will require some amount of power. In Section 4 we examine how 3D ICs help introspection by quantifying the wires in terms of the number of wire buffers that are needed, the area they will occupy, and power they consume compared to on-die routing. While there is some overhead, the area needed for the posts is localized to the position of the tap (where the profile data is gathered) and no extra coordination is required between the designers of the different blocks. It turns out that because the wires can be much shorter, the power overhead is actually reduced compared to on-die routing.

2.2 Reducing Cost for Commodity Parts

A second advantage that 3D integration provides is a way to reduce the total cost to the end user. The cost for an integrated hard-

ware monitor needs to be paid by every end user, despite the fact that most will never use nor need such functionality. In the United States there are an estimated 225 million PCs [5] in use, which is more than 3 computers for every 4 people, as compared to a total of 700 thousand programmers. Even if every programmer demanded a system with hardware support for debugging, the market of such devices would still be orders of magnitude less than commodity PCs. By fabricating the analysis model with steps that are complementary to (but separate from) the main processor, stacked active layers offers the potential to add monitors on just a small subset of devices without impacting the overall cost of the main processor. Just to be clear, we are advocating the sale of one type of processor which is *always fabricated with connections for hardware monitoring*. The difference between the system we sell to the consumer and the one that is sold to the developer is only whether the hardware monitor devices are actually stacked on top or not. This means we must consider two costs:

(a) The cost of the developer system with hardware monitoring and analysis stacked on top. There is definitely a cost to fabricating systems using 3D technology as it requires mounting the analysis engine, the thermal effects can require the use of more expensive heat sink technology, and the monitoring/analysis layers need to be fabricated and tested. It is difficult to estimate the additional fabrication costs, although many are advocating moving towards 3D IC technology for performance reasons in which case the incremental cost of adding a layer will be small (especially if one analysis layer could be used for multiple different families of chip). In addition to the increased fabrication costs, heat will be generated both by driving the posts and by the active layer of the monitor, which in turn will effect cooling costs. In Section 4 we quantify many of these effects.

(b) The cost of a consumer system with the hardware monitoring/analysis left off. If the average consumer is to buy a system without a mounted analysis engine, we need to measure the incremental cost of making the main processor hardware monitoring compatible. The added cost here is due almost completely to the area consumed by the circuit that drives the post and the vertical column of vias that is required to connect where the post would go (which is now a stub because it is not connected to anything). Again, these area considerations are quantified in Section 4.

2.3 Enabling more Powerful Software Analysis

The final major advantage of stacking a hardware monitor on top of the main processor is the potential it has to open new avenues of research in heavy-weight dynamic program analysis. Current runtime systems are heavily constrained by both the overhead of analysis and the very limited monitoring bandwidth available. A full analysis of the potential of such a system to enable new types of dynamic analysis is beyond the scope of this paper, however there are many examples of such analysis already existing. For example, Mondrian Memory Protection extends the idea of memory protection to include protection on arbitrarily small ranges of memory with permissions for read, write, and execute [47] and has been shown to be effective at identifying many types of software bugs through emulation in software [48]. Unsafe pointer dereference analysis, such as “fat pointers” [27, 38] or unsafe memory region tracking [50], can identify the code that is most likely to be exploited by worms and other network based attacks. Tracking data flow tags through the architecture can point to the suspicious use of data [44, 19] so that worms can be identified in the wild, and data flight recording [49, 36] can allow the playback of architectural state when bugs or attacks are identified. These analysis methods provide a powerful tool, but can cause anywhere from 10 to 10,000 times slowdown. Many have proposed the use of detailed profile information to make informed design and optimization decisions. Procedure and data placement, trace scheduling, value specialization, network load balancing, dynamic compilation, and a whole host of power management techniques can all be precisely guided by a more accurate picture of what a program is doing and how it is interacting with the system.

While all these analysis methods have great potential, a commercially viable way to accelerate them in hardware is needed. Because our monitor is decoupled from the main processor, the amount of area and power which can be allocated to analysis is increased significantly. In Section 4 we explore the limits of our proposed approach in terms of these constraints.

3. Quantifying the Technology

Now that we have discussed the high level ideas behind chip-stacking a hardware monitor, let us consider a concrete example of both a 3D technology and a specific hardware monitor (which we call an analysis engine). A multilayer interconnect could take the form of any number of different competing technologies, including chip-bonding, Multi-chip Modules (MCM) [34], chip-stacking with vias [9, 20], or even wireless superconnect [35]. While chip-bonding and MCM technology are already used in a variety of embedded contexts [4, 8], more aggressive 3D technologies are being heavily researched by several major industrial consortiums. Intel, for example, has been investigating 3D integration to include extra levels of cache. If this technology is included to add extra functionality for consumer machines, it would be an incremental step to add an additional *optional* analysis layer. As this is the most mature next-generation superconnect technology, and given the major investments in its development by industry giants such as Intel, IBM, and Inion, we use this technology as a starting point for our evaluation.

3.1 Manufacturing Posts Between Two Die

One popular method of fabricating 3D integrated chips is to bond together two fully processed wafers on which transistors and wires have been fabricated, such that the wafers completely overlap. The top wafer is first thinned to approximately $\tilde{1}$0-50μm. Optically adjusted bonding is then used to stick this layer to the bottom wafer using an organic adhesive layer ($\tilde{2}\mu m$) of polyimide. After metalization is done on both layers and prior to the bonding process, electrical connections are needed between the two wafers. The connection is made by inter-chip vias, which are etched through the inter-metal-layer dielectric on the top wafer, the thinned top Si wafer itself and through the cured adhesive layer. The inter-chip vias are then formed in these etched holes using chemical vapour deposited (CVD) tungsten which can withstand the high temperatures (400^oC) of the wafer bonding process. In a modern process, these vertical interconnects typically have cross-sections of $5\mu m$ x $5\mu m$ and height of 30-40 μm, whereas a normal metal wire's cross section is of the order of $1\mu m$ x $1\mu m$ [1]. We refer to these inter-chip vias as posts for the rest of our paper. A second approach relies on thermo-compression bonding between metal pads in each wafer. In this case, Cu-Ta pads on both wafers serve as the electrical contacts between the inter-chip vias on the top thinned Si wafer and the uppermost interconnects on the bottom Si wafer. These processes, as well as other processes (for 3D integration of VLSI chips) are described in [8, 9].

The cross section of the stacked processor layer and analysis layer is shown in Figure 2. In this figure, we show the active layer (where cmos logic is designed), metal layers (for routing), vias (connecting metal layers), buffers (driving vias), and vertical posts (connecting different active layers) because we will use these terms to explain the advantages and overhead of introspective 3D chip in subsequent sections.

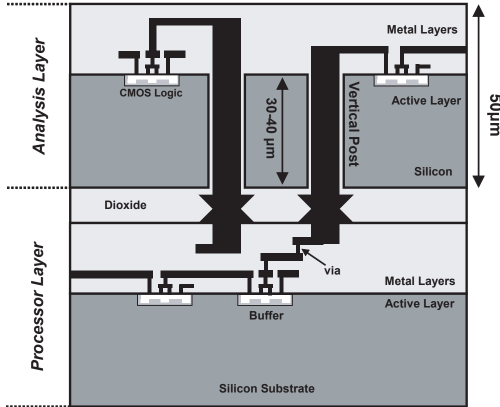

Figure 2. The figure shows the cross section of the introspective 3D chip with processor and analysis layers separated by the dioxide layer. We conservatively estimate that the vertical posts are $50\mu m$ in length with a cross section is $5\mu m$ x $5\mu m$.

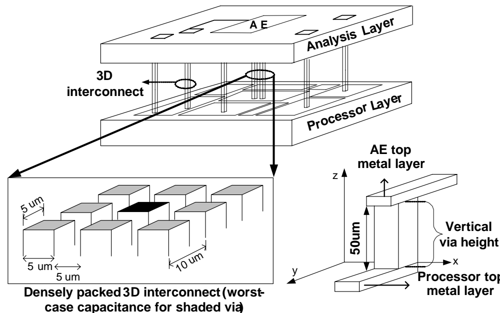

Figure 3. 3D IC with the dimensions. A cross section of posts is shown in enlarged form to show the worst possible coupling capacitance.

3.2 Details of the Interconnect

In this subsection, we present the analysis of interconnects and associated buffers that are used to drive the interconnect. To compare the interconnect and buffers overhead with 2D, we calculate the active layer area overhead (area of active layers), metalization area overhead (area of metal layers), and power consumption for both 2D and 3D ICs and buffers. The analysis is completely different in 3D compared to 2D because of different requirements of number of buffers and complexity of interconnect. In 2D, we need to drive the interconnect across the chip, so we need larger number of buffers, whereas in 3D, the interconnect is going vertically up to the layer above and this requires just one buffer in the processor layer. However, we need to route the interconnects in the analysis layer of 3D chip from the vertical posts which are driven from the processor layer. For 3D interconnect, we use post of height 50 μm and the orientation of these posts are shown in Figure 3. For 2D and the analysis layer in 3D, we consider the optimal buffer insertion to drive the interconnect across the chip, which is explained in detail below.

3.2.1 Buffering Long Wires

Driving a signal across a very long wire (in 2D) requires that the wire is broken up into segments, where each segment is driven with its own buffer (which acts like a repeater in a network [29]). To keep up with the data rate at which profile data is generated every cycle, we would need to insert buffers of the appropriate sizes at regular intervals along this global interconnect. While increasing the size of a buffer increases its drive strength, it also increases the load capacitance seen by the previous driver stage. Similarly, while a larger number of buffers along the wire decreases the wire load on each buffer, it adds to the capacitive load on the overall interconnect. Hence there exists an optimal buffer size (b_{opt}) and inter-buffer separation (l_{opt}) which minimizes the delay along these long global interconnects. For the 130 nm node the optimal buffer size is found to be 74 times the minimum-sized inverter, and the optimal inter-buffer separation is found to be $284\mu m$. The device and interconnect parameters for these calculations are found from [1].

3.2.2 Interconnect overhead analysis

In order to estimate the area overhead for routing these global interconnects and buffers, we separately consider the *active area* occupied by the buffers (the silicon surface area required to implement the inverters) and the *metalization area* occupied by the additional wiring requirements of these global interconnects.

2D Interconnect Overhead - We find that for $0.13\mu m$ technology, the active area required by one optimally sized buffer is 0.21e-4 mm^2. At 2 GHz, driving the capacitance of the wire segment and the input to the next buffer, each buffer will consume 0.290 mW. The static power of a buffer is calculated using the E-cacti tool [33] which is an extended version of Cacti, however the contribution (0.003mW) is negligible compared to the dynamic power (as it is switching a large load with high frequency). Using a processor floorplan (See figure 5, we find the number of buffers that are required to drive the interconnect across the chip and then we can get the interconnect overhead for 2D system with analysis engine.

3D Interconnect Overhead - The overhead that interferes with the processor in the 3D case comes from the buffers needed to drive the posts. As explained before, the global wire routing and additional buffers are all placed on the AE layer and incur no overhead on the microprocessor layer. We also analyze the interconnect overhead in the analysis layer as data comes from posts and is routed over metal wires to the analysis engine. The buffer, post, and global wire analysis is similar to the 2D case except that we need fewer buffers in the processor layer compared to 2D. To find the power consumption of a post, we calculate the capacitance of the post by considering the worst possible case i.e. when a single post is surrounded by 8 posts on all sides as shown in Figure 3. This worst possible case takes care of the maximum possible coupling capacitance loading from surrounding wires. The worst case capacitance of a single post is found (using [37]) to be 0.594e-15 $F/\mu m$. Since we take the length of post to be 50 μm in our case, the capacitance of one post will be 0.297e-13 F. In 0.13 μm technology with a frequency of 2 Ghz, the maximum power consumption of one single post will be 0.071 mW.

Metalization area - The *metalization area* overhead of implementing the analysis engine is evaluated by considering the minimum area on the different metal layers that needs to be reserved for routing the global wires that interconnect the analysis engine to the microprocessor. For the 2D case, it is assumed that these global interconnects are routed on the topmost metal layer of the microprocessor with minimum pitch global wires (670nm for 130nm technology node [1]). In addition, every buffer inserted in the global interconnect is assumed to be interconnected by a stack of vias of minimum width (175nm square [1]) that run through all successive metal layers from the topmost (Metal 8) to the active layer. For the 3D case, it is assumed that a single optimally sized buffer is implemented on the microprocessor layer. This buffer drives the signal through a 3D post ($5\mu m$ square) and the remaining global interconnect routing is implemented in the top metal layer of the stacked

Application Data	Required Posts	Location
Memory Addresses	32×IPC	LSQ
Memory Values	32×IPC	LSQ
Program Counter	32×IPC	Program Counter
Opcodes	3×10×IPC	Integer/FP Queue
Register Names	2×5×IPC	Integer/FP Queue
Register Values	32×IPC	Register File
is_cache_miss	2	MBox
is_branch_miss	3	Branch Predictors
is_tlb_miss	2	Translation Buffer

Table 1. Number of required posts for different profiling application data. For each type of data the number of posts needed is shown in the second column (the vertical interconnection lines that need to be inserted to drive the data to the analysis engine). The third column is the location on the layer-1 processor where the tap drivers need to be placed.

analysis engine (AE) layer. The *metalization area* for the 3D case is only the total area that needs to be reserved on each metal layer of the microprocessor (Metal1 to Metal8) to accommodate the connection from the buffer to the 3D via (which need not form a perfect vertical column). In this case, additional buffers needed for the global interconnect are also implemented in the stacked AE layer, hence they lead to no overhead in the microprocessor layer itself.

3.3 Profiling Requirements

As mentioned previously, gathering profile information is very crucial to many optimizations and design problems. In this subsection we explain where in the processor we need to put the *taps* in and how many posts are required to draw out the required data to the analysis engine. In order to ensure that the profiling hardware will be flexible enough to perform a wide variety of analysis methods, we need to capture many different signals as described in Table 1. The second and third columns of the table shows the number of posts that need to be inserted per tap and the location on the processor where the tap needs to be placed. This gives us an estimate of the number of posts or wires that needs to be accommodated for all relevant information to be passed on to an analysis engine. Based on the requirements shown in table 1 we estimate that 1024 bits of profile data will be generated each cycle, which will in turn require 1024 wires or posts. Figure 5 shows the different blocks in a Pentium 4 processor floorplan, and our estimate of where the data needs to be gathered.

3.4 An Example Hardware Monitor

Designing an analysis engine capable of performing a variety of online program analysis is no trivial task. On one end, a counter is probably the simplest mechanism to aid program analysis, while on the other end we could have complex analysis processors capable of running tens of profiling algorithms, enabling multiple optimizations, and performing analysis over different profile data of the running program all at the same time. Rather than exploring this massive design space, in this section we start by describing the design and analysis of a simple example analysis engine based on past work. We then use this as the layer-2 analysis engine in the proposed 3D architecture. For all analysis and design process in this paper we assume 2 GHz clock rate and 0.13-μ technology at 1.1V. Our purpose here is to examine a concrete example to argue the feasibility of an introspective 3D chip.

Analysis Engine Architecture - Looking at the operations which a programmable analysis engine might most frequently perform, we find that an associative lookup followed by counter increments, simple manipulations on a set of counters, and a periodic

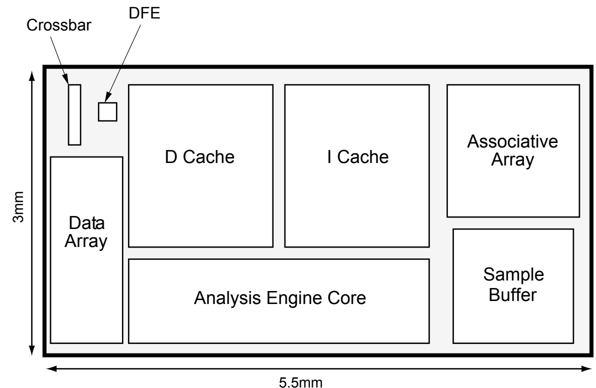

Figure 4. A floorplan of the layer-2 analysis engine. 1024 bits of profile data flow from layer-1 to this layer-2 through the 3D posts. This data is then filtered and appropriate routines are called in to process the data by the crossbar and the DFE. The floorplan of an XScale processor was used to determine the size and shape of the analysis engine core, the data cache, and the instructions cache. An associative array and a data array are used to speed lookup and profiling operations.

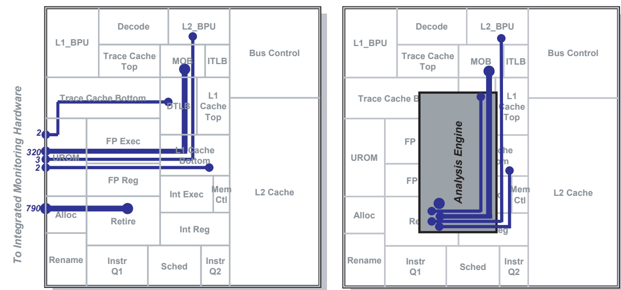

Figure 5. The figure on the left shows the locations where chip profile data will need to be gathered if the profiling hardware is integrated with the chip (a P4 in this case). If the analysis engine is located on the periphery of the chip, significant signals will have to be global, even if they are placed to minimize wire length (as they are in this picture). The numbers by the end of each line show the number of bits. The figure on the right shows an analysis engine stacked on top, connected with posts. There is no new global routing on the processor layer, and the global routing on the analysis layer is minimized because the analysis can be centrally located.

sequence of complex computations are very common functions. We would like to provide at least these features on our layer-2 analysis engine and we consider the architecture of an analysis engine based on the profiling co-processor proposed by Zilles and Sohi [52]. We have made a couple of modifications to the co-processor architecture to suit the 3D IC architecture including additional features to make more complex analysis possible and increasing the size of memory.

Specifically, the architecture of the layer-2 comprises of the following components: a core analysis engine, an associative array to perform fast lookups, a data array, a crossbar which filters the layer-1 data down to only that which is needed for analysis at the layer-2 processor, a decoder and field extractor (DFE) which determines what processing to do with profile data captured at

Component	Area in mm^2	Dyn Power in mW	Stat Power in mW
Associative Array	2.42	371	2.23
Data Array	2.23	464	4.19
DFE	0.03	128	0.08
Sample Buffer	2.07	586	4.76
CrossBar	0.054	103	0
AE Core	3	300	1.71
D-Cache	3	350	3.6
I-Cache	3	350	3.6
Total	15.82	2652	18.46

Table 2. Power and area requirements for the components in the analysis engine

the layer-1 processor, and a sample buffer to store the profile data before the analysis engine picks it up for processing. The floorplan of the architecture is shown in Figure 4. The analysis engine, at the core, is a RISC microprocessor in a six-layer metal 0.13-μm, which implements the ARM^{TM} V.5TE as described in [12]. In our analysis, we select a sample buffer of size 16 KB with with one read and one write port, an associative array of 16 KB with one read-write port, a data array of 32 KB with one read-write port, and instruction & data caches each of 32 KB.

Hardware Overhead - The area and power for this analysis processor are calculated based on the designs described in [12]. In [12], Clark et al. describe an implementation of the Intel XScale Microarchitecture. We scale the static power, dynamic power and area of the XScale core from 0.18 μm to 0.13 μm technology as described in [10]. We find that the area of this core is about 9 mm^2 and that it would consume 1.1 W dynamic power and 9 mW static power for 0.13 μm technology at room temperature ($300K$). We break down the area, dynamic power, and static power of the core into the analysis engine (AE) core, the D-cache and the I-cache using the floorplan provided in the paper [12] and using E-cacti tool [33].

We extend the Ecacti tool to get the dynamic/static power and area of data array, DFE, sample buffer. Ecacti does not support the static power modeling of associative arrays and hence we get a rough estimate of the static power using the model of Butts et al. [11]. We consider the values of k_{design} for RAM cell and associative CAM cell from [11] to scale the results accordingly. Table 2 shows the power and area requirements for each of the components in analysis engine on layer-2. By summing the hardware overhead of each component, we find that the analysis engine requires about 16 mm^2 and consumes about 2.7 W dynamic power and 18.5 mW static power at room temperature. Later in Section 4.4 we will evaluate the thermal impact of stacking this design with a P4 processor.

Programmability - While working out all the implementation details of an example system using 3D introspection is beyond the scope of this paper, it is worth mentioning how this analysis layer might be used and programmed. The core can be programmed by the main processor to include a variety of filters, to perform sampling, and more importantly to actually store run-time analysis information (such as legal memory ranges). The filters set the crossbar at the layer-2 to transfer only the relevant bits among the 1024 bits flowing into the analysis engine every cycle. The Decoder and field extractor (DFE) is programmable to choose among the fields of the incoming instruction to be profiled (which is fetched from the sample buffer). The DFE also decides what type of profiling to do on this instruction. The profiling code itself, is loaded onto the analysis engine by the layer-1 processor.

4. Architectural Ramifications

In the sections leading up to this we have discussed the individual pieces to the larger puzzle of introspection through 3D IC technology. To actually weigh the advantage of such a scheme we need to consider 4 types of systems –

1. Basic system (S_{base}) - The base case where the cost of the chip to the consumer is minimum but there is no hardware support for analysis. All area, power, and thermal impacts of additional hardware support should be considered in relation to this.
2. System with integrated profiling hardware ($S_{integrated}$) - This involves designing and fabricating the analysis engine along with the processor on the same die. This affects the routing on the chip since the profile data needs to flow from the place where it is generated from the taps on the processor to the place where it is processed in the analysis engine, and may impact design complexity significantly.
3. System with profiling hardware stacked on top ($S_{stacked}$) - In this system the profile data now flows through the posts to the analysis engine which is stacked directly over the processor. As mentioned earlier, the design and fabrication costs of such a processor can be decoupled from that of the analysis engine. However, the cost of manufacturing this 3D chip is definitely higher compared to the base system. We show later in this section that $S_{stacked}$ requires an additional $0.021mm^2$ active area and 1.4% more power than S_{base}. We also show that additional active area required for $S_{integrated}$ is more by a factor of 20 than what is required by $S_{stacked}$ and also that $S_{stacked}$ consumes less than half as much power as $S_{integrated}$ does.
4. System with profiling stubs (no stacking) (S_{stubs})- As we have argued before, not all consumers would require introspection and analysis on their chips. In fact, most consumers (at least 99.7% of them) are end users who are only interested in using computers for applications. In the $S_{integrated}$ version, we have no other option but to design and verify S_{base} (for the 99.7%) and $S_{integrated}$ (for the 0.3% developer community). However, in the case of $S_{stacked}$, since the analysis engine design is decoupled from the base processor, all we need to do on the base processor is to make provisions for stacking an analysis engine. This way, we need to design just one type of processor with stubs in it. Though these stubs have a very low impact on the power and area, we think it is important to evaluate it because this is the one that is sold to 99.7% of the consumers. We later show that this additional provisioning for analysis support adds about $0.021mm^2$ of active area with less than 1% increase in power.

We now evaluate these four systems with respect to their routability, area, power, and thermal effects and show that $S_{stacked}$ for developers and S_{stubs} for the end-users makes some difficult-to-achieve profiling and analysis techniques practicable and cost effective.

4.1 Routability

For an analysis engine to perform any introspection, the data needs to flow from the taps on the host processor to the analysis engine (as shown in Figure 5). In the S_{base} configuration, no such wires are required because no hardware analysis is performed, whereas for $S_{integrated}$ and $S_{stacked}$, wires travel from a functional block on the host processor towards a logical connector in the analysis engine. In $S_{integrated}$ the logical connectors are at the boundary of the processor from where the analysis engine can fetch the profile data. In the case of $S_{stacked}$, wires terminate in layer-2 as shown

in figures 3 and 5. In either case, the long wires will need to be segmented and buffered as discussed in Section 3.2.2.

Figure 5 shows the wires that need to be drawn from the processor core into the analysis engine, and Table 1 shows both the number of vertical posts required to capture all necessary information and the locations on the physical processor from where we need to draw wires. Based on this information we find that the total wire length required for $S_{integrated}$ is 5682.3mm. We earlier discussed that segments of length $285um$ should be used for optimal buffering. Hence we require on the order of $5682.3mm/285um = 20,000$ buffers for $S_{integrated}$. That is a significant number of buffers and distributing them throughout the chip adds greatly to design complexity.

In the case of our proposed $S_{stacked}$ configuration, we do not require any wiring on the processor layer other than the posts. One buffer is required to drive each vertical post, and each post carries the signal up until it reaches the analysis engine layer. Hence there is no additional horizontal wiring required on the processor layer in the $S_{stacked}$ configuration. $S_{stacked}$ still requires 1024 buffers (one buffer per vertical post) to drive the signal from the host processor to the analysis engine. This reduces the number of buffers (almost 20 times less than $S_{integrated}$) and it further eases the design of layer-1 because there is no need to coordinate between different functional blocks. In terms of routability, $S_{stacked}$ and S_{stubs} are essentially the same as both have an identical layer-1.

4.2 Area

With additional wires and buffers on the processor layer to transmit profile information, there is an added space requirement. As described earlier, every wire that needs to be laid occupies some space on the metal layer (for the wire and the via, as shown in figure 2) and the buffers occupy space on the active layer. While there can be a non-linear area impact due to increased interconnect in the face of significant congestion, we estimate the area impact from the number of buffers and wire area needed which is likely to be a conservative estimate.

The area overhead is, therefore, a contribution of the wire area at the top most metal layer (also referred to as the global layer) (A_{wire}), via area (A_{via}) and the area at the active layer by the buffers (A_{active}). $A_{wire} + A_{via}$ is called the metalization area ($A_{metalization}$). In $S_{integrated}$, the horizontal wires total $5683.2mm$ in length and assuming a global layer in $0.13um$ technology where the wire pitch is $670nm$ [1] (as also described in Section 3.2.2) A_{wire} is $670nm \times 5683.2mm = 3.8mm^2$. The area taken by one buffer is $0.000021mm^2$ as calculated in Section 3.2.2. For $S_{integrated}$, A_{active} is $20000 \times 0.000021 = 0.41mm^2$ which then needs to be distributed over the chip.

Figure 6 shows the increase in area for the three configurations over S_{base}. Since $S_{integrated}$ has all wires on layer-1, the first bar only shows increases in layer-1 components, which is about $4.2mm^2$. *This is without even considering the extra 16.5 mm^2 required for the integrated analysis engine*, it is just the area needed on the processor layer for the buffers.

The bar for S_{stub} (which also has no layer-2) just has an increase in area due to the posts (which is not connected to anything) and then drivers for those posts. The total amount of area required is around $0.021mm^2$, or an estimated 0.008% increase. Including stubs should not impact the cost of a system compared to S_{base}.

In the case of $S_{stacked}$, we have the area of the buffers and posts in layer-1 (as for S_{stub}), but we now have to route the data from the posts to the analysis engine. While this does not impact the size of the processor layer, it is worth considering the routing complexity at the second level. However, as can be seen in the Figure 5, the routing is significantly easier in the second layer because we can more centrally locate the analysis engine. If fact we can place it

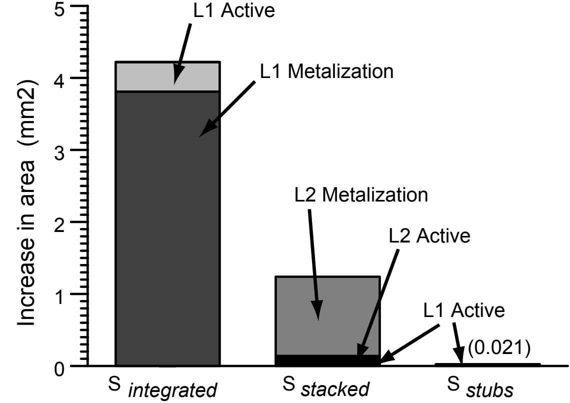

Figure 6. The figure shows the increase in area of different configurations compared to the S_{base} configuration. L1 and L2 indicate the area required on layer-1 and layer-2 respectively. Negligibly small amount of area overhead on layer-1 in $S_{stacked}$ and S_{stub} is one of the major advantages of using 3D IC technology for introspection.

directly over the core as long as the thermal impact is not too large as we study in subsequent sections.

One issue that comes up with optionally including a second layer is how to deal with I/O. On a typical chip, pads are fabricated on the top-most level of metal. ITRS numbers indicate that a processor may need about 1000 pads for I/O and another 1000 pads just for power and ground. There are two issues here: First, in the case of S_{stub}, communication stubs must not significantly interfere with the pad placement. Given that the stubs are insignificant in size compared to the pads we believe they could be easily squeezed in between the pads with negligible effect. The second problem is for $S_{stacked}$. In that case all of the I/O, power, and ground for the main processor will have to be routed vertically *through* the analysis layer. This should not pose a significant problem as the analysis layer has the same footprint as the processor layer, but is far less dense (so routing around these extra vias in the analysis layer should be easily solvable). It may increase the amount of Analysis Layer (L2) Metalization area estimated by our analysis by a factor of 2 or 3, but it should impact neither the cost of S_{stub} nor the thermal issues of $S_{stacked}$. However a full and careful analysis of 3D IC I/O issues is outside the scope of this paper.

4.3 Power

An analysis engine, when running, will necessarily draw power. That power will come from one of two major sources, the wires required for routing (including stubs, posts, and buffers) and the analysis engine itself. As we demonstrate in this section, if the wires are routed across chip, interconnect power can easily dwarf the power consumed by a second processor. In trying to build an analysis engine that is as non-intrusive as possible it is important to minimize this power consumption.

In the case of $S_{integrated}$, since the wires are drawn from the processor to the analysis engine as shown in figure 5, power is only required to drive the signal horizontally across the chip (P_{horiz}). Each buffer consumes 0.29mW and, as we discussed above, $S_{integrated}$ will require 20000 buffers. That means a total of $5862mW$ is required. The bars in Figure 7 show the increase in power with respect to the S_{base}. We see that in figure 7, just for transferring profile data across the chip (L1 P_{horiz}), $S_{integrated}$ consumes 24% more than S_{base}. When combined with even a simple analysis engine, this is a 34% increase in power.

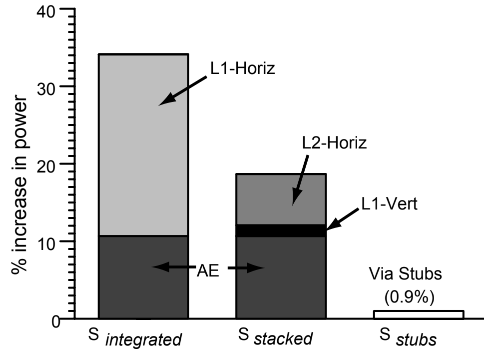

Figure 7. The bar graph shows the percentage increase in power of different configurations compared to S_{base}. AE is the power consumed by the example analysis engine we consider in this paper. L1 and L2 represent the power consumption to transfer profile data in layer-1 and layer-2 respectively. The $Horiz$ factor is the drive signals on the same plane and the $Vert$ component is to drive it across layers. $S_{stacked}$ results in less than 2% increase at layer-1 and S_{stubs} less than 1%.

3D IC technology does not fully solve this problem, as $S_{stacked}$ needs power to drive the profile data up to the second layer (L1 P_{vert}) and then to transfer the data to a suitable location on the layer-2 analysis engine (L2 P_{horiz}). However, because the posts are quite small, and because the total distance traveled for wires can be much less with careful placement, the interconnect power can be reduced from 23% to 8%.

As we have mentioned, to be able to simplify the design and verification process, we wanted to maintain a single fabrication process for layer-1 for $S_{stacked}$ and S_{stubs}. Hence, though there is not a layer-2 in S_{stubs}, we need to make room for a possible mounting of layer-2. This means that we will have some vias which remain dangling on layer-1 metal which will add to the power consumption, though by a very small factor. We evaluate the S_{stub}'s additional power requirement due to the buffers and vias hanging on in layer-1's metal layer (assuming we do not add extra circuitry to stop them from driving if disconnected). Even with these buffers driving, because the load is significantly less (the posts are not attached) there is less than a 1% increase in power compared to S_{base}. It may be possible to even further reduce this with careful design.

4.4 Thermal

Until now we have focused on evaluating the area and power impact for 4 different configurations. While power and area play major roles in the cost of a system, the cooling cost is also extremely important. To address these thermal problems researchers have proposed a variety of techniques including using performance counters for power [25], tracking runtime temperature [30], and even thermal management at the microarchitectural level [43]. The total system power in part determines the system temperature and hence the cooling cost of the system.

One potential problem with a 3D approach is that you might stack a hot analysis engine onto an already critical processor hotspot. In fact, that is exactly what you want to do with an analysis engine, as usually the hottest, most active, parts of the chip yield the most information. To examine this effect, we have performed a detailed thermal analysis of $S_{stacked}$.

For the 3D introspective chip, the total system power required includes all layer-1 and layer-2 horizontal and vertical driver buffer powers, the power consumed by the analysis engine, and the power consumed by the host processor itself. For a base case (S_{base}) we use industry data from the Pentium 4 processor, we then examine the impact of adding the stacked component ($S_{stacked}$). Using layout geometry, power distribution, and physical parameters of all the components in the processor and analysis engine we are able to model the estimated thermal impact. A full-chip realistic packaging model is incorporated and takes into account both vertical and lateral heat transfer paths. Electrothermal couplings [7] are embedded into heat parabolic partial differential equations [39] and the equations are solved in a self-consistent manner using the Alternating-Direction-Implicit (ADI) method [40, 22].

The thermal profiles can be seen in figure 8 (they are better seen in color). The left most figure in the first row is the temperature profile of the base Pentium 4 processor. Each column of figures represents a different configuration. Immediately to the right of the base case is the case for $S_{stacked}$, with the analysis engine (shown on bottom) stacked on to of the P4 (shown on top). To obtain a pessimistic case, we have placed the analysis engine directly on top of the hottest part of the P4 core. In addition to showing the thermal gradients, the temperature of the hottest spot (which is usually used to determine packaging requirements) is shown in bold. We can see that S_{base} maximum temperature is 54.8^oC and for $S_{stacked}$ the hotspot is 55.9^oC. This small 1.1^oC rise in maximum system temperature provides evidence that cooling a $S_{stacked}$ should be feasible, and even if slightly more expensive, would not impact developers too significantly.

While a simple analysis engine such as $S_{stacked}$ would provide basic functionality, the amount of computation it could perform would be limited compared to the size of the stream of data available. With terabytes of program data per second at your disposal, it might make sense to develop a specialized high-throughput analysis engine to make good use of it all. While we do not develop such an architecture here, we do consider two more aggressive designs for the purpose of examining their thermal profiles in the hopes that it may be useful to those that will. Specifically we consider a simple tiling of analysis engines mounted on layer-2. The third column in figure 8 shows the temperature profile for a Pentium 4 (top) with four analysis engines mounted on layer-2 (bottom). Interestingly, while the total average temperature goes up, the temperature of the hottest point on the chip decreases slightly from 55.9^oC to 55.8^oC. Despite the fact that there is now a factor 4 times more heat being generated on layer-2, the analysis engines have been positioned so as to not directly overlap with the hot-spot. Finally, we consider a case where 8 analysis engines fully tile the second layer which raises the temperature to 57.5^oC. While hot-spot mitigation is an important area of research already, we simply point out that in building an 3D analysis layer, if hot-spots are avoided it could enable far more computation on the layer-2. Of course this could lead to longer interconnects and more power, but this is a tradeoff better left to future work.

5. Conclusions

Enabling programmers and software developers to more easily track down bugs, identify performance bottlenecks, and secure their code against attacks needs to be one of the primary concerns of system designers at all levels, including computer architects. One method in which architecture could aid in attacking these problems is through the creation of machines which support intensive dynamic analysis methods with a minimum of interference on the software. We propose that hardware support assisting in these endeavors should be detached from the typical end-user system. One way of detaching this functionality is to have an auxiliary analysis engine capable of performing all of the required dynamic analysis, and to stack this analysis engine on top of the main processor with 3D IC technology.

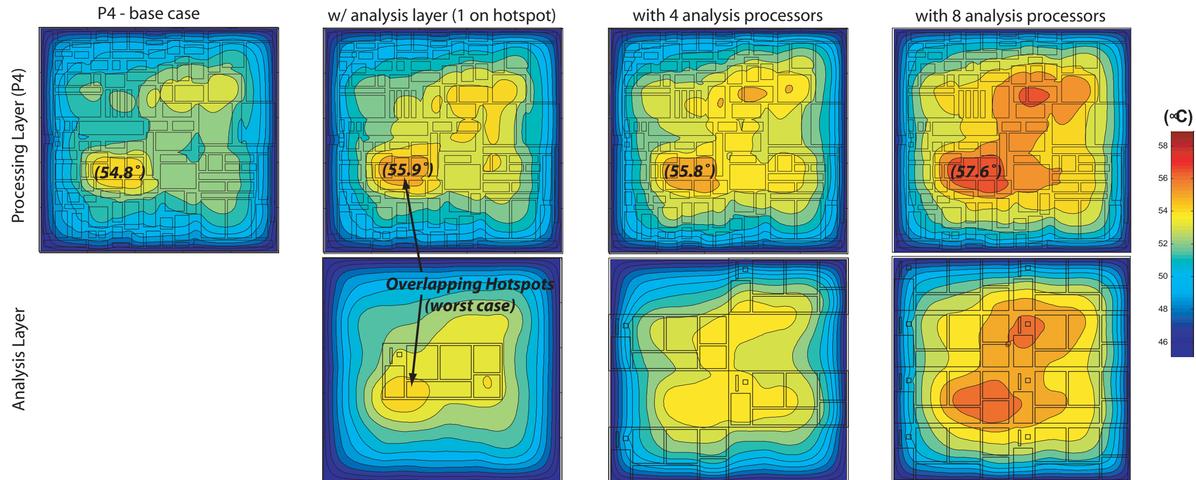

Figure 8. The first row shows the thermal profiles of Pentium 4 and their variation with addition of analysis engines (from left to right). The second row shows the thermal profiles for the analysis layer. The first column has just the S_{base} profile, whereas the second, third and fourth show thermal profiles of the two layers when one, four, and eight analysis engines respectively are mounted on Pentium 4 base.

One of the biggest advantages of this approach is that the cost of specialized analysis hardware is *decoupled* from the highly cost sensitive consumer market. In doing so, users can still buy their cheap high performance machines because the only extra hardware they are paying for are stubs. The additional cost of the hardware to perform online analysis, the cost of the interconnect to route the performance data, and the cost of the complexity of handling that global interconnect, are all eliminated. The hardware stubs that are left increase area and power by no more than $0.021mm^2$ and 0.9% respectively, numbers which might be further reduced with careful design. At the same time, developers and users both benefit from the increased analysis power of dynamic monitoring tools. Even though our argument, like most arguments in systems, is economic in nature, we have used the metrics of area, power, routability, and temperature to quantify one possible design. While the thermal impact of stacking two hot cores together is always a concern in 3D design, we show that the effect is manageable for both our sample system and for a system 8 times more powerful. Given that developers would need to pay more for this additional hardware anyways, the incremental cost of additional cooling should be a minor.

While we have done a detailed analysis of one possible system, there are many open research questions remaining, including finding the best design for an analysis layer. Given that the profile data rates available from a 3D introspective chip are very large, a more throughput oriented analysis architecture will be needed to exploit the full potential of the data rates. With the 3D techniques we present in this paper, we hope to open the door to a rich design space with the dimensions of analysis functionality, generality, area, and thermal impact.

Acknowledgments

The authors would like to thank the anonymous reviewers for their helpful feedback. This work was supported in part by NSF Career Grant CCF-0448654, NSF Grant CNS-0524771, a seed grant from the Boeing Corporation, and a UC-MICRO/Intel grant.

References

[1] International Technology Roadmap for Semiconductors, 2001.

[2] Workshop on Hardware Performance Monitor Design and Functionality in conjunction with HPCA-11, 2005.

[3] N. Goldsman A. Akturk and G.Metze. Self-Consistent Modeling of Heating and MOSFET Performance in 3-D Integrated Circuits. *IEEE Transactions on Electron Devices*, 52(11):2395–2403, 2005.

[4] Cristinel Ababei, Yan Feng, Brent Goplen, Hushrav Mogal, Tianpei Zhang, Kia Bazargan, and Sachin Sapatnekar. Placement and Routing in 3D Integrated Circuits. *IEEE Design and Test of Computers*, 22(6):520–531, Nov/Dec 2005.

[5] Computer Industry Almanac. http://www.c-i-a.com.

[6] J. Anderson, W. Weihl, L. Berc, J. Dean, S. Ghemawat, M. Henziger, S. Leung, R. Sites, M. Vandevoorde, and C. Waldspurger. Continuous Profiling: Where Have All the Cycles Gone? *ACM Transactions on Computer Systems (TOCS)*, 15(4):357–390, November 1997.

[7] K. Banerjee, S-C. Lin, A. Keshavarzi, S. Narendra, and V. De. A Self-Consistent Junction Temperature Estimation Methodology for Nanometer scale ICs with Implications for Performance and Thermal Management. In *IEEE International Electron Devices Meeting (IEDM)*, pages 887–890, 2003.

[8] Kaustav Banerjee, Shukri J. Souri, Pawan Kapur, and Krishna C. Saraswat. 3-d ics: A Novel Chip Design for Improving Deep Submicron Interconnect Performance and Systems-on-Chip Integration. *Proceedings of the IEEE*, 89(5):602–633, May 2001.

[9] Benkart et al. 3D Chip Stack Technology using Through-chip Interconnects. *IEEE Design and Test of Computers*, 22(6):512–518, Nov/Dec 2005.

[10] Shekhar Borkar. Design challenges of Technology Scaling. *IEEE Micro*, 19(4):23–29, 1999.

[11] J. Adam Butts and Gurindar S. Sohi. A Static Power Model for Architects. In *MICRO 33: Proceedings of the 33rd annual ACM/IEEE international symposium on Microarchitecture*, pages 191–201, New York, NY, USA, 2000. ACM Press.

[12] Lawrence T. Clark, E. J. Hoffman, J. Miller, M. Biyani, Y. Liao, S. Strazdus, M. Morrow, K. E. Velarde, and M. A. Yarch. An embedded 32-b microprocessor core for low-power and high-performance applications. volume 36, pages 1599–1608, November 2001.

[13] T. M. Conte, B. A. Patel, and J. S. Cox. Using Branch Handling Hardware to Support Profile-driven Optimization. In *Proceedings of the International symposium on Microarchitecture*, pages 12–21, November 1994.

[14] T. M. Conte, M. Kishore N., and M. Ann Hirsch. Accurate and Practical Profile-driven Compilation using the Profile Buffer. In *Proceedings of the 29th Annual International Symposium on Microarchitecture*, December 1996.

[15] Marc L. Corliss, E Christopher Lewis, and Amir Roth. Dise: A Programmable Macro Engine for Customizing Applications. In *Proceedings of the Thirtieth International Symposium on Computer Architecture (ISCA-30)*, June 2003.

[16] Marc L. Corliss, E Christopher Lewis, and Amir Roth. Low-overhead Debugging via Flexible Dynamic Instrumentation via Dise. In *Proceedings of the Eleventh International Symposium on High-Performance Computer Architecture (HPCA-11)*, pages 303–314, February 2005.

[17] Digital Equipment Corporation. Alpha 21164 Microprocessor Hardware Reference Manual. 1995.

[18] Intel Corporation. Pentium(r) Pro Processor Developer's Manual. In *McGraw-Hill*, June 1997.

[19] Jedidiah R. Crandall and Frederic T. Chong. Minos: Control Data Attack Prevention Orthogonal to Memory Model. In *MICRO 37: Proceedings of the 37th annual IEEE/ACM International Symposium on Microarchitecture*, pages 221–232, Washington, DC, USA, 2004. IEEE Computer Society.

[20] Davis et al. Demystifying 3D ICs: The pros and cons of going Vertical. *IEEE Design and Test of Computers*, 22(6):498–510, Nov/Dec 2005.

[21] Jeffrey Dean, James E. Hicks, Carl A. Waldspurger, William E. Weihl, and George Z. Chrysos. ProfileMe : Hardware support for instruction-level profiling on out-of-order processors. In *International Symposium on Microarchitecture*, pages 292–302, 1997.

[22] J. Douglas and H. H. Rachford. On the numerical solution of heat conduction problems in two or three space variables. *Transactions on American Mathematical Society*, pages 421–439, 1956.

[23] Timothy Heil and James E. Smith. Relational Profiling: Enabling Thread-level Parallelism in Virtual Machines. In *MICRO 33: Proceedings of the 33rd annual ACM/IEEE international symposium on Microarchitecture*, pages 281–290, New York, NY, USA, 2000. ACM Press.

[24] MIPS Technologies Inc. MIPS R10000 Microprocessor User's Manual. 1995.

[25] Canturk Isci and Margaret Martonosi. Runtime power monitoring in high-end processors: Methodology and empirical data. In *MICRO 36: Proceedings of the 36th annual IEEE/ACM International Symposium on Microarchitecture*, page 93, Washington, DC, USA, 2003. IEEE Computer Society.

[26] Philip Jacob, Okan Erdogan, Aamir Zia, Paul M. Belemjian, Russell P. Kraft, and John F. McDonald. "Predicting the performance of a 3D processor-memory chip stack". *IEEE Design and Test of Computers*, 22(6):540–547, Nov/Dec 2005.

[27] Trevor Jim, Greg Morrisett, Dan Grossman, Michael Hicks, James Cheney, and Yanling Wang. Cyclone: A safe dialect of C. In *USENIX Annual Technical Conference*, June 2002.

[28] Michael B. Kleiner, Stefan A. Kühn, and Werner Weber. Performance improvement of the memory hierarchy of RISC systems by applications of 3-D technology. In *ISCAS*, pages 2305–2308, 1995.

[29] Rajesh Kumar. Interconnect and noise immunity design for the Pentium 4 processor. In *DAC '03: Proceedings of the 40th conference on Design automation*, pages 938–943, New York, NY, USA, 2003. ACM Press.

[30] Kyeong Jae Lee and Kevin Skadron. Using performance counters for runtime temperature sensing in high-performance processors. In *19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05)*, April 2005.

[31] Gian Luca Loi, Banit Agrawal, Navin Srivastava, Sheng-Chih Lin, Timothy Sherwood, and Kaustav Banerjee. A Thermally-Aware Performance Analysis of Vertically Integrated (3-D) Processor-Memory Hierarchy In *Proceedings of the 43nd Design Automation Conference (DAC)*, June 2006.

[32] Christianto C. Liu, Ilya Ganusov, Martin Burtscher, and Sandip Tiwari. Bridging the processor-memory performance gap with 3D IC technology. *IEEE Design Test*, 22(6):556–564, 2005.

[33] M. Mamidipaka and Nikil Dutt. eCACTI: An Enhanced Power Model for On-chip Caches. Technical Report CECS TR-04-28, September 2004.

[34] Claude Massit and Nicolas Gerard. Three-dimensional multichip module United States Patents, US 5373189, December 1994.

[35] Miura et al. A 195gb/s 1.2w 3D-stacked inductive inter-chip wireless superconnect with transmit power control scheme. In *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, pages 264–265, Feb 2005.

[36] Satish Narayanasamy, Gilles Pokam, and Brad Calder. Bugnet: Continuously recording program execution for deterministic replay debugging. In *32nd Annual International Symposium on Computer Architecture (ISCA'05)*, pages 284–295, 2005.

[37] K. Narbos and J. White. Fastcap: A multipole accelerated 3D capacitance extraction program. *IEEE Trans. on CAD*, 10(11):1447–1459, 1991.

[38] George C. Necula, Scott McPeak, and Westley Weimer. Ccured: Type-safe retrofitting of legacy code. In *POPL '02: Proceedings of the 29th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 128–139, New York, NY, USA, 2002. ACM Press.

[39] M. N. Ozisik. Boundary value problems of heat conduction, 2002.

[40] D. W. Peaceman and H. H. Rachford. The numerical solution of parabolic and elliptic differential equations. *Journal of the Society for Industrial and Applied Mathematics (SIAM)*, pages 28–41, 1995.

[41] R. V. Peri, S. Jinturkar, and L. Fajardo. A Novel Technique for Profiling Programs in Embedded Systems. In *ACM Workshop on Feedback-Directed and Dynamic Optimization*, 1999.

[42] Kiran Puttaswamy and Gabriel H. Loh. Implementing caches in a 3D technology for high performance processors. newblock In *IEEE International Conference on Computer Design (ICCD) 2006*, pages 525–532, October 2005.

[43] Kevin Skadron, Mircea R. Stan, Wei Huang, Sivakumar Velusamy, Karthik Sankaranarayanan, and David Tarjan. Temperature-aware microarchitecture. In *ISCA*, pages 2–13. IEEE Computer Society, 2003.

[44] G. Edward Suh, Jae W. Lee, David Zhang, and Srinivas Devadas. Secure Program Execution via Dynamic Information Flow Tracking. In *ASPLOS-XI: Proceedings of the 11th international conference on Architectural support for programming languages and operating systems*, pages 85–96, New York, NY, USA, 2004. ACM Press.

[45] Yuh-Fang Tsai, Yuan Xie, N. Vijaykrishnan, and Mary Jane Irwin. Three-dimensional cache design exploration using 3DCacti. In *IEEE International Conference on Computer Design*. IEEE, October 2005.

[46] Kapil Vaswani, Matthew J. Thazhuthaveetil, and Y. N. Srikant. A Programmable Hardware Path Profiler. In *CGO '05: Proceedings of the international symposium on Code generation and optimization*, pages 217–228, Washington, DC, USA, 2005. IEEE Computer Society.

[47] Emmett Witchel, Josh Cates, and Krste Asanovic. Mondrian memory protection. In *ASPLOS-X: Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 304–316, New York, NY, USA, 2002. ACM Press.

[48] Emmett Witchel, Junghwan Rhee, and Krste Asanovic. Mondrix: memory isolation for linux using mondriaan memory protection. In *SOSP '05: Proceedings of the twentieth ACM symposium on Operating systems principles*, pages 31–44, New York, NY, USA, 2005. ACM Press.

[49] Min Xu, Rastislav Bodik, and Mark D. Hill. A "Flight Data Recorder" for enabling full-system multiprocessor deterministic replay. In *ISCA '03: Proceedings of the 30th Annual International Symposium on Computer Architecture*, pages 122–135, New York, NY, USA, 2003. ACM Press.

[50] Suan Hsi Yong and Susan Horwitz. Protecting C programs from attacks via invalid pointer dereferences. In *ESEC/FSE-11: Proceedings of the 9th European software engineering conference held jointly with 11th ACM SIGSOFT international symposium on Foundations of software engineering*, pages 307–316, New York, NY, USA, 2003. ACM Press.

[51] Annie Zeng, James Lu, Kenneth Rose, and Ronald J. Gutmann. "First-order performance prediction of cache memory with wafer-level3d integration. *IEEE Design and Test of Computers*, 22(6):548–555, Nov/Dec 2005.

[52] Craig B. Zilles and Gurindar S. Sohi. A Programmable Co-processor for Profiling. In *Proceedings of the 7th International Symposium on High Performance Computer Architecture*, 2001.