Processor Design in 3D Die-Stacking Technologies

THREE-DIMENSIONAL DIE-STACKING INTEGRATION STACKS MULTIPLE LAYERS OF PROCESSED SILICON WITH A VERY HIGH-DENSITY, LOW-LATENCY LAYER-TO-LAYER INTERCONNECT. AFTER PRESENTING A BRIEF BACKGROUND ON 3D DIE-STACKING TECHNOLOGY, THIS ARTICLE GIVES MULTIPLE CASE STUDIES ON DIFFERENT APPROACHES FOR IMPLEMENTING SINGLE-CORE AND MULTICORE 3D PROCESSORS AND DISCUSSES HOW TO DESIGN FUTURE MICROPROCESSORS GIVEN THIS EMERGING TECHNOLOGY.

Gabriel H. Loh
Georgia Institute of Technology

Yuan Xie
Pennsylvania State University

Bryan Black
Intel

Three-dimensional integration is an emerging fabrication technology that vertically stacks multiple integrated chips. The benefits include an increase in device density; much greater flexibility in routing signals, power, and clock; the ability to integrate disparate technologies; and the potential for new 3D circuit and microarchitecture organizations. This article provides a technical introduction to the technology and its impact on processor design. Although our discussions here primarily focus on high-performance processor design, most of the observations and conclusions apply to other microprocessor market segments.

3D integration technology overview

Although there are several candidate variants on 3D integration technology, at the heart of all of them is the vertical stacking of two or more individual integrated chips. (This article doesn't cover processes that "grow" multiple layers of devices such as multiple-layer buried substrate [MLBS] technology.) This stacking provides multiple levels of devices and multiple layers of traditional on-chip metal interconnect. One approach for 3D integration is wafer-to-wafer bonding. (See the "Constructing a 3D stack" sidebar for an explanation of how multiple whole silicon wafers are stacked into 3D integrated chips.)

When considering the 3D arrangement of two silicon dies, there are two natural topologies: face to face and face to back, where a die's "face" is the side with the metallization and its "back" is the side with the silicon substrate. A copper-copper bonding process builds an interdie connection, also called a *die-to-die* (d2d) or *3D via*, by depositing the copper of half of the via on each die, and then bonding the two dies together with a thermocompression process. A chemical-mechanical polishing process thins one die to reduce the distance of communication between stacked layers, and for external I/O and power.

3D interconnects

From a processor designer's perspective, the most important attribute of a given 3D fabrication technology is the size and pitch of the d2d vias. A very fine (dense) d2d via pitch enables a correspondingly fine partitioning of processor structures across multiple layers, whereas a larger d2d via pitch

0272-1732/05/$20.00 © 2007 IEEE Published by the IEEE Computer Society.

Constructing a 3D stack

Die-stacking 3D integration involves constructing multiple component dies through conventional fabrication processes. The 3D construction is a back-end-of-the-line (BEOL) process. Depending on the exact technology, we can bond (stack) entire wafers to other wafers and then dice them into separate 3D integrated circuits (ICs). Individual dies can also be directly stacked, and there are many combinations involving individual dies, partial wafers, and complete wafers, such as die-on partial-wafer stacking. The exact unit of stacking is unimportant for the following discussion, although it impacts the achievable die-to-die (d2d) via pitch (full-wafer bonding is harder to align but provides better manufacturing throughput).

Our discussion here is for a copper-copper bonding, face-to-face, two-die stack:

1. We start with two processed wafers (although for clarity we only show individual dies in Figure A).
2. Similar to building conventional vias between different metal layers, we deposit copper via stubs that connect to the top-level metal.
3. We arrange the two wafers face to face and subject them to thermocompression. The bonding time, amount of pressure, and temperature affect the quality of the bond between the two halves of the d2d via stubs. The pressure and temperature effectively cause the two ends of the copper stubs to fuse together. The entire area between two dies will likely be completely populated by d2d vias because, in addition to providing a signal path, the d2d vias serve as the primary heat conduction path between the multiple layers as well as the mechanical means of holding the die together.
4. We use chemical-mechanical polishing (CMP) to thin one layer of the 3D stack to only 10 to 20 μm.
5. The thinning allows the backside or through-silicon vias (TSVs) that implement the external I/O and power/ground connections to be relatively short, thereby minimizing both IR and L di/dt losses.

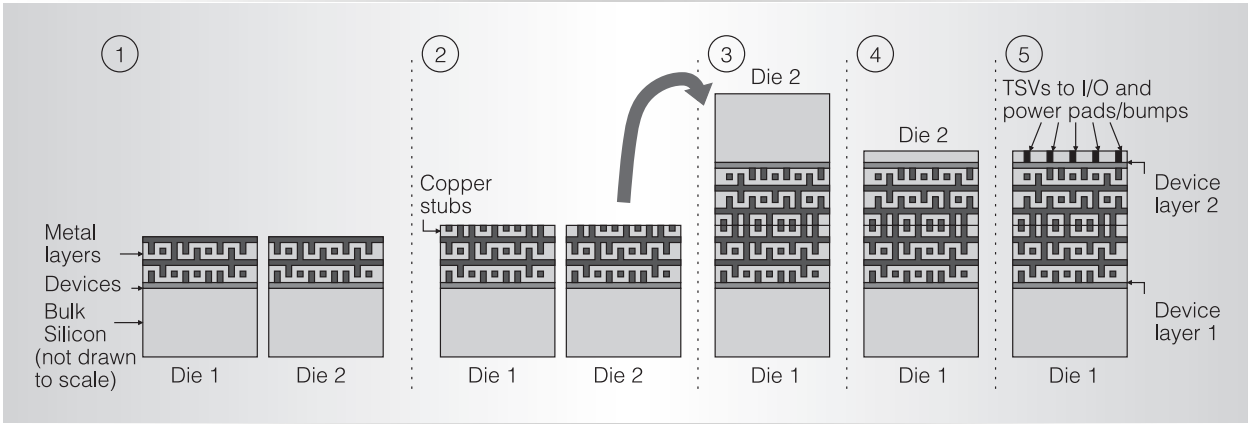

Figure A. Fabrication steps for face-to-face bonding.

reduces the available interdie bandwidth per unit area. Current fabrication technologies can provide d2d via pitches in the range of 10 μm × 10 μm down to 1 μm×1 μm.[1] As a point of reference, a six-transistor static RAM (6T SRAM) cell in a 65-nm process takes approximately 0.7 μm^2.[2] The d2d via pitch is still significantly larger than an individual transistor; however, the micron-level pitch is still fine enough to enable interesting 3D organizations. In a face-to-face bonding scheme, the d2d via size (cross-sectional area when viewed from above) is the same throughout. For a face-to-back organization, the d2d via's size at the bonding site will be similar to that of the face-to-face case, but in the device layer, the d2d via can be smaller to minimize the layout impact of the neighboring circuits and devices, as Figure 1 shows. In either bonding style, the via pitch at the bonding interface limits the number of d2d communication paths. The d2d via size is important in the face-to-back approach to minimize the impact of the d2d vias on the layout of transistors.

The d2d via's size determines the possible 3D partitionings of processor blocks and functional units. The latency of communication across a d2d via greatly affects the overall usefulness of such 3D partitionings. In a face-to-face topology, the d2d vias are deposited on the top layer of the respective dies and therefore the interdie distance is

For a larger stack, repeat the steps. Other bonding techniques besides copper-copper are similar in spirit but might involve special dielectric glues or slightly different processing steps. In a stack of more than two layers, thinning also reduces the effective thermal resistance observed by each subsequent layer that is farther removed from the primary heat removal paths—that is, the heat spreader and heat sink.

After dicing, the individual stacks are packaged (not shown in Figure A).

Constructing a face-to-back stack is similar (see Figure B), but it requires some extra handling of the thinned layer.

1. We again start with the two processed wafers. The wafer to be thinned must first be attached to a ''handle wafer.''
2. We use CMP to thin the wafer down. At this point, the 10- to 20-μm thick wafer is structurally unsound and would physically break if left on its own; the handle provides the necessary mechanical support.
3. We deposit the two halves of the d2d vias. On the ''face'' wafer, constructing the via stubs is similar to the face-to-face case. On the ''back'' wafer, we etch the vias in a fashion similar to the face-to-face's I/O and power TSVs.
4. We bond the two dies through thermocompression.
5. We release the thinned die from the handle. At this point, the unthinned die provides the necessary structural support for the thinned die. Because the thinned die's face is exposed, we can interface I/O and power pads in the same way as a conventional unstacked, planar chip.

Finally, we dice the wafer and package the individual 3D ICs.

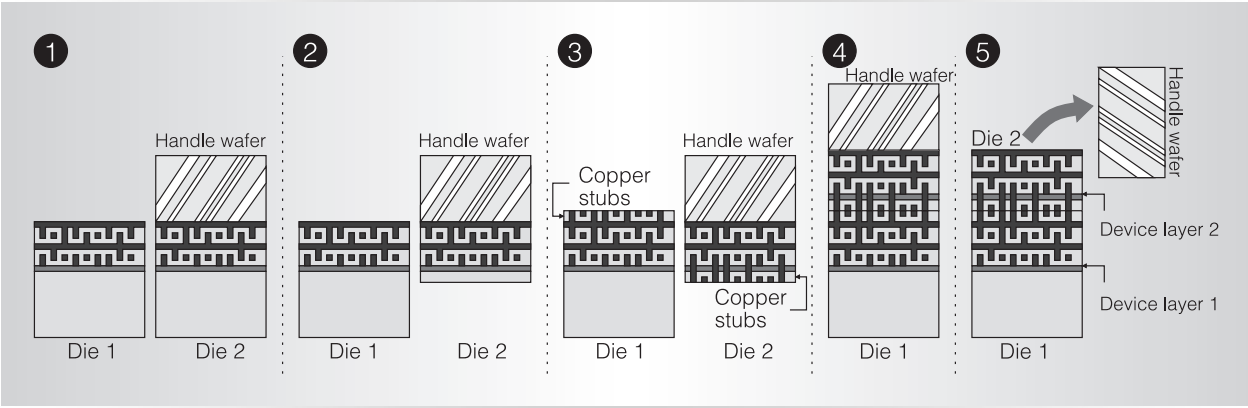

Figure B. Fabrication steps for face-to-back bonding.

small. For a face-to-back topology, the die-thinning process reduces the thickness of the die to 10 to 20 μm, which implies that the length of the d2d via is similar. In a face-to-face topology, the additional metal due to the d2d via has a small impact on overall latency. As Figure 2a illustrates, the d2d via increases the path's resistance—effectively the resistive-capacitive (RC) delay—by only 35 percent compared to a full stack of vias connecting metal 1 to metal 9. This implies that a d2d via can replace even short on-chip interconnects of only a few tens of microns without impacting latency. For longer wires, the wire-length reduction will bring an improvement in both latency and power.

Our Spice simulations indicate that an inverter driving 1 mm of metal takes 225 ps based on a 70-nm Predictive Technology Model (PTM).[3] The same inverter driving through a full via stack, a 10-μm-long d2d via, and then another via stack only requires 8 ps, as Figure 2b illustrates. For reference, the fan-out-of-four (FO4) delay (that is, the time required for a minimum-sized inverter to drive four equivalently sized inverters, frequently used as an approximation for one ''gate delay'') in the same technology is 22 ps.

The d2d vias' exact latency depends on the exact bonding technology, the driving circuits, the loading capacitance, and other factors. However, from the perspective of

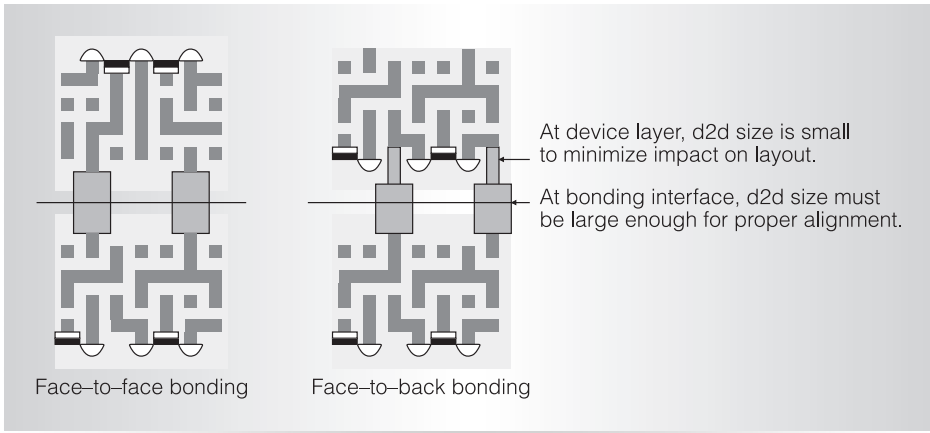

Figure 1. Cross-sectional view of the die-to-die interface for face-to-face and face-to-back bonding arrangements.

routing between processor functional unit blocks (FUBs) and designing 3D FUBs and circuits, it's sufficient simply to consider the d2d via as equivalent to a short run of conventional metal in terms of latency and power. The 3D vias in a copper-copper bonding process don't require special I/O drivers, buffers, or pads; they are simply a small RC component of signal routing.

Power delivery

Another important characteristic of the d2d vias is that they serve for both signal routing and power delivery. Some fraction of all the d2d vias must be reserved for connecting the power and ground planes of each of the individual dies. Our calculations indicate that as much as 30 percent of the available d2d vias must be used for power and ground, although this number can vary depending on the power requirements for each layer. Our simulation results also show that power delivery and supply droop issues for a two-layer 3D stack are no different than those for a processor implemented on the next generation of a conventional process (for example, twice the current transistor density). This is primarily because the electrical characteristics of d2d vias are similar to the on-die vias used for power delivery in a planar (2D) process.

In addition to on-chip power distribution, power must be supplied to the chip from off the chip as well. For a face-to-back topology, we can use conventional pads for power delivery. However, the face-to-face topology requires backside or through-silicon vias (TSVs) for I/O and power. Inductive droops for off-chip power delivery to a face-to-face 3D stack isn't a major hurdle because the inductance of a single 10-µm-wide TSV is less than 2.5 pH for a single return path. Many return paths exist in a full chip, which further reduces the effective inductance. This additional inductance has little effect compared to the switching noise observed in the on-die power distribution networks of existing processors.

If a processor implemented in a two-layer 3D stack requires only one half of the original 2D footprint, this also implies that there are only half as many pins available for power delivery. If the fraction of pins used for power and ground remain the same, then we'd expect the current density per pin to increase by a factor of two. Our simulation results indicate that the physical characteristics of the TSVs can easily support these increased current demands. Furthermore, this analysis is somewhat pessimistic in that it assumes that the 3D processor consumes the same amount of power as the original 2D baseline design. In practice, a 3D-organized processor should be able to eliminate a significant amount of on-chip metal, leading to a corresponding reduction in total power demand. Reducing power can in turn reduce the number of required pins for power delivery as well as the fraction of d2d vias reserved for on-chip power distribution.

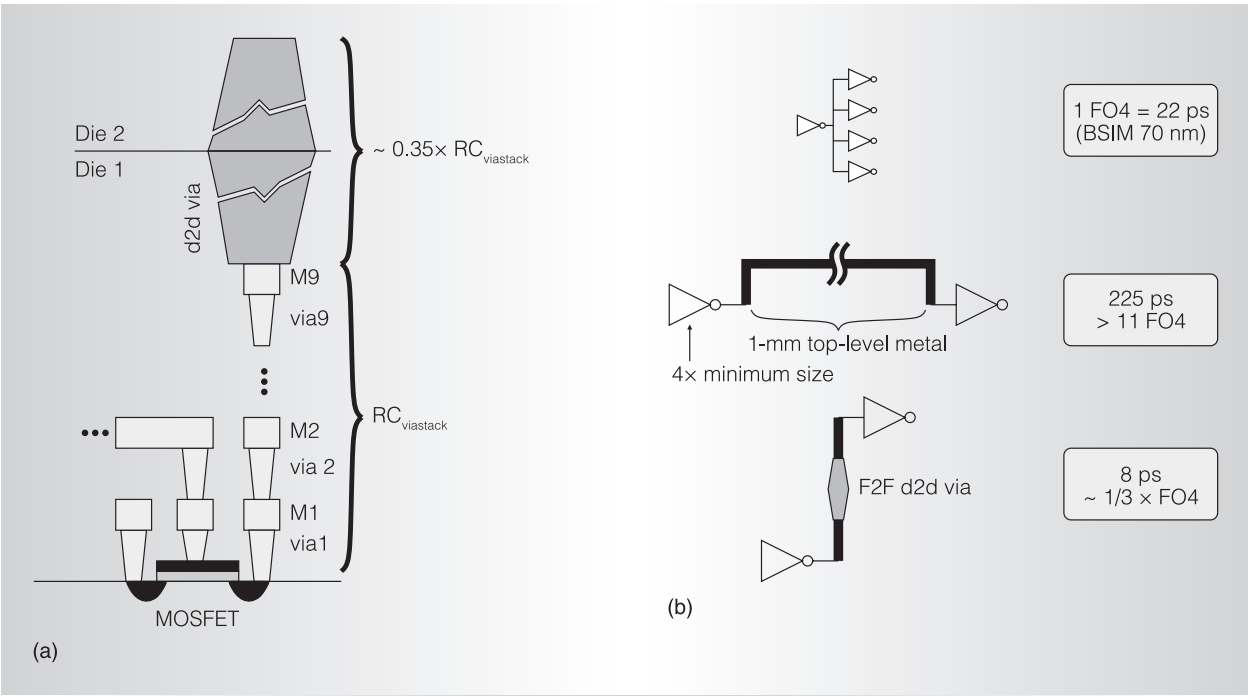

Figure 2. Resistance-capacitance (RC) of a die-to-die (d2d) via relative to a full via stack (a) and signal propagation delays for a d2d via, 1 mm of wire, and a fan-out-of-four (FO4) delay (b).

Thermals

One of the key challenges in using 3D integration technology is managing thermals. Naively stacking *n* layers of transistors would increase the processor power density by a factor of *n* as well, which would most likely lead to significant thermal issues. Two potential problems can lead to increases in chip temperature. The first is that in a stack of multiple layers of silicon, each successive layer is farther removed from the heat sink, thereby increasing the effective thermal resistance observed by that layer. The second is the aforementioned increase in power density.

We conducted thermal simulations of a commercial 65-nm, dual-core, high-performance processor using a production-quality tool at Intel. The results indicated a worst-case chip temperature of 81°C. We also considered a hypothetical two-die 3D-stacked version of the same processor where each block has been partitioned across the two layers such that the overall chip now fits in exactly half the original footprint. We assumed that this 3D processor still consumes the same power as the 2D baseline, but because its footprint has been halved, its power density doubles. As a result, our thermal simulations indicate a worst-case temperature of 98°C. For comparison, we also simulated the original 2D processor shrunk to the next process generation. That is, we have the same number of transistors, but again in half the area, and therefore twice the power density.

Surprisingly, our thermal simulation data showed that this next-generation 2D processor exhibits nearly the same worst-case temperature as our 3D configuration. From these results, we conclude that the 3D structure in and of itself isn't a major thermal problem for 3D processors; successfully controlling thermals in a 3D design will require carefully managing the power density.

One might argue that our previous thermal comparison is unfair because a processor implemented in the next-generation technology would have the benefit of lower total power consumption due to the reduction in the device feature size. Although this is true, we also haven't taken account of the fact that a processor's 3D implementation can achieve lower total power as well by using 3D routing to reduce global

Table 1. Spectrum of 3D design approaches, from coarser- to finer-level granularity.

Stacking unit	Potential benefits	Redesign effort	Examples
Entire core	Low: Power and performance of individual components unchanged. Some benefit in reducing footprint of clock and power networks.	Low: Reuse existing 2D design	Core-on-core, Cache-on-core
Functional unit blocks (FUBs)	Medium: Reduced latency and power of global routes provides simultaneous performance improvement with power reduction.	Medium: Must re-floorplan and retime paths. Need 3D block-level place and route tools. Existing 2D FUBs can be reused.	ALU-on-ALU (faster bypass), DL1-on-ALU (faster loads)
Logic gates (FUB splitting)	High: Reduced latency/power of global, semiglobal, and local routes. Further area reduction due to compact layout and resizing opportunities.	High: Need new 3D circuit designs, methodologies, and layout tools. Reuse existing 2D standard cell libraries.	Cache splitting, ALU bit-splitting
Transistors	High: Possible further reductions in area, latency, and power. Transistor size relative to d2d pitch makes gains unlikely except for large, complex gates.	Extreme: Almost no reuse	NMOS/PMOS* partitioning, Domino

* n-channel metal oxide semiconductor (NMOS) and p-channel MOS (PMOS)

interconnect. If processor architects, circuit designers, and layout engineers can exploit the additional degrees of freedom afforded by 3D organization to provide a power reduction comparable to that of a technology shrink, then the 3D thermal problem will be no worse than what they already must deal with in conventional planar technologies. Although this is still a significant challenge, it's by no means a showstopper or an insurmountable roadblock for the widespread adoption of 3D technology.

3D microarchitectures

3D integration provides two key advantages for the processor architect: the potential for significant wire reduction through 3D placement and routing, and the ability to integrate disparate fabrication technologies without disrupting existing process flows. This article primarily focuses on the wire-reduction benefits.

At a high level, 3D clearly can eliminate wiring by placing two distant objects on top of each other and replacing the wire with a d2d via. However, choosing the stacking granularity leads to different design options, trade-offs, benefits, and complexities. Table 1 and Figure 3 highlight a range of possible stacking granularities. We will first qualitatively discuss the different options in this design spectrum and then provide a more detailed example by examining a range of 3D implementations of on-chip caches.

Partitioning granularity

At one extreme, the designer can choose to stack macroscopic blocks (see Figure 3a). Examples include stacking the L2 cache on top of the CPU core or stacking separate cores in a multicore design. One advantage of this approach is the potential for significantly reusing existing 2D design efforts and intellectual property. A disadvantage is that this approach removes or reduces few critical wires, which limits the performance and power benefits. Each processor core is identical to its original 2D version and therefore has the same performance and power characteristics. This approach can reduce a multicore processor's core-to-core communication paths, but this grossly under-utilizes the available band-

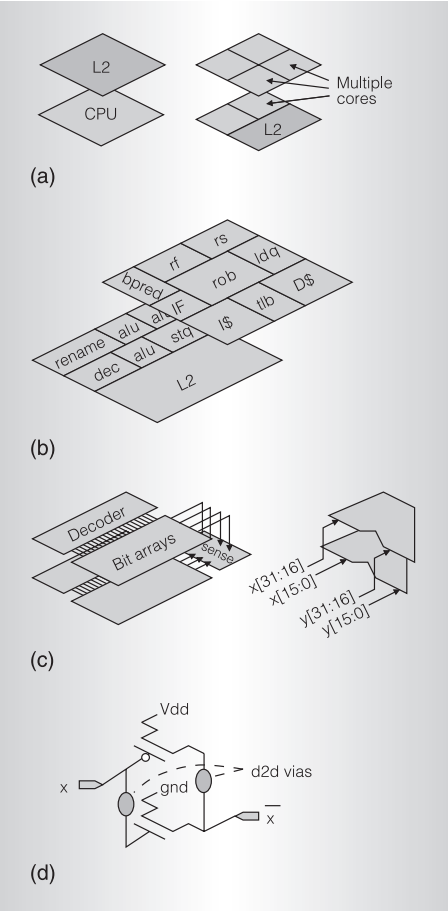

Figure 3. 3D design approaches range from coarse to fine grained, depending on the granularity of the unit to be stacked: entire core (a), functional unit blocks (FUBs) (b), logic gates (c), and transistors (d). See Table 1 for details on each approach.

width of the d2d interface (the interface provides on the order of millions of d2d vias per cm^2, while the core-to-core interface needs no more than a few thousand connections).

To extract more benefit from 3D integration, the architect or designer will likely need to repartition processor functionality at a finer level. The stacking option in Figure 3b partitions an individual processor core across two or more layers by stacking different functional unit blocks (FUBs). For example, we might stack arithmetic logic units directly above the register file to reduce the length of the wires for delivering operands to the computation units as well as the wires used for writing results back to the register file. These data paths are typically wide (multiple operands, each at 32, 64, or even 128 bits for instruction sets that support wide multimedia or SIMD instructions) with high activity factors. This organization can reduce critical paths throughout the processor, yielding simultaneous performance and power benefits. With this approach, each individual block is still fundamentally a planar component, which still provides the potential for some design reuse.

The third level of 3D partitioning splits individual FUBs across multiple layers (see Figure 3c). Examples include splitting a cache's word lines or bitlines, stacking the entries of a processor's reservation stations, or bit-slicing functional units across multiple dies. We can place an individual circuit's different gates on different layers. The benefit of such an approach is that it helps eliminate significant intrablock wiring, which can provide substantial power and performance benefits for wire-dominated blocks such as the instruction scheduler, result bypass network, and large multiported SRAMs—for example, a physical register file or register alias table. In addition to the intrablock wire reduction, this type of partitioning would also reduce each block's footprint, which leads to a more compact overall floorplan and reductions in global wiring. This approach has the potential for far greater performance and power benefits than the coarser-grained approaches, but it also requires a substantial design effort to implement 3D versions of all the FUBs.

The finest level of 3D partitioning is at the transistor level (see Figure 3d). At an extreme, a 3D circuit would have all n-channel metal oxide semiconductor (NMOS) transistors placed on one die and all p-channel MOS (PMOS) devices placed on another. Although current and projected d2d via interfaces won't likely have the density to support such a fine degree of stacking for all CMOS gates, there might be special cases where this approach can be useful. Large circuits that involve a large number of highly clustered transistors can benefit from such fine partitioning. The

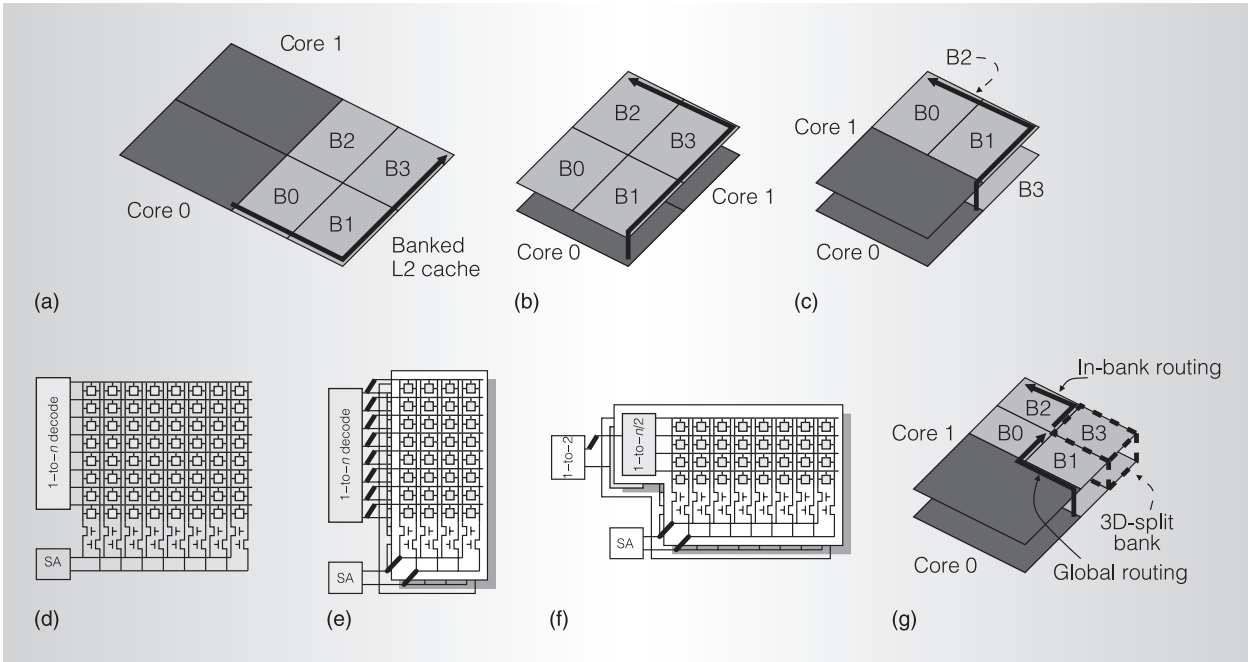

Figure 4. Implementing a cache in 3D. A baseline 2D processor with L2 cache (a), a L2 cache 3D stacked above the cores (b), L2 cache banks 3D stacked on each other (c), a schematic view of a cache's static RAM array (d), a 3D SRAM array with stacked bitlines (e), a 3D SRAM array with stacked word lines (f), and a 3D L2 cache in relation to the cores (g).

individual bit cells of a highly ported register file could be port-partitioned to greatly reduce the area per cell, which in turn would provide a substantial reduction in access latency and power consumption.[4]

Large domino circuits are also candidates for fine-grained 3D partitioning. One option is to place latches, clocking, and all precharge logic on one layer and all data and evaluation transistors on the other. Another option is to place a dual-rail dynamic logic gate's $f(\cdot)$ and $f'(\cdot)$ circuits on separate layers. Transistor-level 3D partitioning gives the designer an extraordinary level of layout flexibility, but it requires a complete redesign of the circuits at a low level.

The partitioning granularity will be largely determined by the achievable d2d via pitch. If the d2d via pitch's size is too large relative to that of a transistor, the designer will be forced to use a coarser partitioning strategy. The placement of backside TSVs (whether for I/O and power or d2d vias in a face-to-back stacking) must pass through the device layer. Therefore, they introduce additional obstructions to transistor layouts that we must also properly take into account.

3D cache example

An important aspect of 3D integration is that the d2d vias are sufficiently dense to enable interesting partitionings of the processor blocks. Depending on the exact density, the partitioning strategies may change. In this section, we show a range of possible two-die 3D cache designs, each targeted for a different level of partitioning. Figure 4a illustrates a baseline 2D processor with a large L2 cache.

We first consider a coarse-grained approach to 3D stacking that stacks an entire cache on top of one or more processor cores (see Figure 4b). This approach minimizes the use of d2d vias because only the bus between the cache and the cores crosses between the two die. A 128-byte cache line requires 1,024 d2d vias (one d2d via per bit of data) plus no more than a few hundred more bits for address and coherency buses. For a die size of 1 to 2 cm^2, the required d2d via density to support this cache interface is only a few thousand vias per

cm^2—4,096 vias per cm^2 assuming the original 2D die is 1 cm^2, the last-level cache takes half that area, and 2,048 bits are needed to communicate data, addresses, and any other control information. This is orders of magnitude less than the density that copper-copper bonding 3D technology can provide, even by the most conservative estimates.

Pessimistically, this organization does not improve the worst-case wiring distance (which we show with arrows in Figure 4) compared to the 2D baseline case. Depending on the exact physical location of the cache access ports, there might be some benefit from removing a few cycles of global routing delay to access the cache. Initial commercial 3D processors are more likely to take this approach because it's a relatively low-risk design requiring minimal redesign effort, a starting point on the evolutionary path to more finely partitioned 3D structures.

The next 3D cache design is a bank-stacked organization. The idea is that each individual bank of a cache can remain unchanged from the original 2D design, but that we place the collection of banks one on top of the other. Figure 4c illustrates how this bank-stacked topology reduces the global interconnect distance required to route from the edge of the cache to the farthest bit cell. This approach provides some degree of latency and power benefit in the long-haul wires but doesn't take advantage of 3D within the individual banks.

For a 1-Mbyte data cache (8-way set associative, 64 banks), we observed a 9.7 percent improvement in access latency with a simultaneous 31.5 percent reduction in energy per read access. If the primary bank decoder needs to route the address and data to each of B individual banks, the cache requires $B \times 4{,}096$ d2d vias per cm^2 to fulfill the interdie communication requirements (under the same assumptions as before). It would require $B = 256$ to saturate a one-million d2d via per cm^2 interface. However, even for highly banked caches, the d2d interface must only support the bandwidth required for the maximum number of simultaneous accesses per cycle, which is typically on the order of ones rather than hundreds. Once routed to the correct layer, additional routing can steer the access to the correct bank.

We next explore the opportunities for 3D within each cache bank at the SRAM array level. The SRAM array primarily consists of a large 2D grid of individual 6T SRAM cells, accompanied by some peripheral logic such as the row decoders and sense amplifiers. In a two-die 3D stack, we can fold the SRAM cells along the *x*- or *y*-axis, leading to splitting either the word lines or the bitlines across the two layers.[5–7]

A third possible organization bit-slices the SRAM array such that, for example, all even bits are placed on one die, and all odd bits are placed on the other. Figures 4d, 4e, and 4f schematically illustrate a 2D SRAM array in addition to two stacked-array topologies. The lengths of either the word lines or the bitlines will be approximately halved in the 3D organization depending on the split orientation, resulting in a latency and power reduction as well as a reduction in the entire array's footprint.

Figure 4g also illustrates that because we've reduced each bank's footprint, the overall cache's footprint is approximately the same as the bank-stacked topology. Overall, this provides wire reduction within each bank as well as to and from the banks. The split-bitline organization results in 21.6 and 30.4 percent reductions in latency and energy per access, respectively, and the split-word-line provides 13.6 and 32.8 percent reductions.

Compared to the bank-stacked approach, this finer level of partitioning provides greater performance and power benefits but also requires a more involved redesign of the SRAM arrays. One interesting conclusion from these results is that the best folding strategy might change depending on the design objectives. In our simulations, the word lines accounted for more delay than the bitlines, so using 3D to reduce the length of the word lines provides a lower overall latency. However, if we were designing to minimize energy consumption, then reducing the length of the bitlines is of greater benefit. This is because the large number of bitlines that must be toggled and sensed in parallel account for a much larger fraction of energy consumption than the switching of a few word lines. The d2d via

requirements are greater than that of the bank-stacked organization because each data bit's output and its complement need to be routed to a sense amp; Figures 4e and 4f show the two d2d vias per sense amp. The d2d via interface provides sufficient density to support two d2d vias per sense amp. There are some additional d2d vias required for splitting the word lines or for fanning out the address bits to a stacked decoder, but we can use one row decoder to drive the word lines for multiple bits of the SRAM, so this component of the d2d via requirement isn't as great. The bit-sliced organization doesn't even require the d2d vias for the sense amps.

For SRAM arrays in particular, partitioning below the level of word line or bitline splitting is unlikely to provide any additional benefit. One possibility for a finer partitioning would be to place each inverter of the SRAM's storage unit on a different layer. Other possibilities are having each bitline on a different layer or even splitting the NMOS and PMOS transistors onto different layers (as we mentioned earlier). However at this point, each of these options requires one or more d2d vias per 6T cell, and even in current technologies, the size of the d2d via already exceeds that of an SRAM cell. The resulting layout would cause the SRAM array's size to grow, thereby reducing the potential benefit from a 3D organization (if not making it worse than the 2D baseline). For a highly ported SRAM, such as a superscalar processor's register alias table or physical register file (but certainly not the last-level cache), each individual SRAM is much larger in size because of the extra access gates and therefore might actually be amenable to a cell-splitting organization.[8]

From this example of a 3D cache design, we can draw a few conclusions about using 3D integration in processor design. First, eliminating critical wires can result in simultaneous latency and power reductions; contrast this with the typical case where a power reduction usually results in a performance drop as well. Second, we can use different partitioning strategies to match or target the communication density of a given d2d via interface. This is particularly important for the continued usefulness of 3D if the d2d via pitch can't scale at the same rate as device feature-size reductions. Third, different partitioning strategies or 3D layouts might yield different trade-offs with respect to power, performance, and area; they should be carefully considered in the context of the overall design goals and constraints.

Mixed-process integration

3D's ability to provide mixed-process integration can have significant consequences on the design of microprocessors. A natural mixed-process application is to stack one or more dies of high-density dynamic RAM (DRAM) on top of a conventional high-performance CMOS process. In a system with relatively low memory requirements, all of main memory may be stacked on top the processor cores (see the "State of the art in 3D processor design" sidebar). Otherwise, the DRAM can be used to implement a large on-chip cache to maintain larger working sets closer to the processor cores. If either of these techniques results in a significant reduction in average memory access time, then this can affect the optimal design of the underlying processor cores as well—that is, fewer misses might require less instruction-level parallelism (ILP) to be exposed to cover average load latencies.

3D integration can let us include analog components in the stack as well. For example, on-stack DC-DC converters can simultaneously provide a multitude of power supply voltage levels that can enable fine-grained dynamic voltage and frequency scaling (DVFS) on a core-by-core or even a block-by-block basis.[9] We can stack a special transistorless layer of high-κ capacitors to provide widespread and highly effective decoupling capacitors (decaps) to the power distribution network to counteract di/dt problems in high-frequency designs without impacting the underlying processor floorplan and circuit layouts. Other applications include the on-stack integration of RF/wireless and networking components. In the long term, emerging technologies such as carbon nanotube transistors or other nano and quantum devices may be integrated on the stack,

State of the art in 3D processor design

Here, we briefly highlight some recent literature related to 3D processor design. We focus on microprocessor and microarchitecture in particular and, therefore, don't include many important works related to 3D manufacturing and technology characterization.

The following works propose and study 3D organizations where a single microprocessor core hasn't been split across multiple layers.

- Main memory (DRAM) stacked on a processor. G. Loi et al., "A Thermally-Aware Performance Analysis of Vertically Integrated (3-D) Processor-Memory Hierarchy," *Proc. Design Automation Conf.* (DAC 06), ACM Press, 2006, pp. 991-996; and C. Liu et al., "Bridging the Processor-Memory Performance Gap with 3D IC Technology," *IEEE Design & Test of Computers*, vol. 22, no. 6, 2005, pp. 556-564.
- Pico-Server: Multi-layer DRAM 3D stack on top of a multi-core. T. Kgil et al., "PicoServer: Using 3D Stacking Technology to Enable a Compact Energy Efficient Chip Multiprocessor," *Proc. Conf. Architectural Support for Programming Languages and Operating Systems* (ASPLOS 06), ACM Press, 2006, pp. 117-128.
- Introspective 3D processor: 3D stacked support for monitoring. S. Mysore et al., "Introspective 3D Chips," *Proc. Conf. Architectural Support for Programming Languages and Operating Systems* (ASPLOS 06), ACM Press, 2006, pp. 264-273.
- Cache stacked on a RISC processor. S. Kühn et al., "Performance Modeling of the Interconnect Structure of a Three-Dimensional Integrated RISC Processor/Cache System," *IEEE Trans. Components, Packaging, and Manufacturing Technology*, vol. 19, no. 4, Nov. 1996, pp. 719-727.
- 3D Multi-core with network-in-memory. F. Li et al., "Design and Management of 3D Chip Multiprocessors Using Network-in-Memory," *Proc. 33rd Ann. Int'l Symp. Computer Architecture* (ISCA 06), IEEE CS Press, 2006, pp. 130-141.
- Intel Core 2 with 3D-stacked L2 cache. B. Black et al., "Die Stacking (3D) Microarchitecture," *Proc. 39th Ann. IEEE/ACM Int'l Symp. Microarchitecture* (MICRO 06), IEEE CS Press, 2006, pp. 469-479.

The following studies analyze 3D processors where functional unit blocks (FUBs) are stacked on top of each other, but each individual FUB is still an inherently 2D structure.

- Automated 3D floorplanning algorithms for FUB-stacked processors. J. Cong et al., "An Automated Design Flow for 3D Microarchitecture Evaluation," *Proc. Asia and South Pacific Design Automation Conf. 2006* (ASP-DAC 06), ACM Press, 2006, pp. 384-389; and M. Healy et al., "Multiobjective Microarchitectural Floorplanning for 2D and 3D ICs," *IEEE Trans. CAD*, vol. 26, no. 1, 2007, pp. 38-52.
- Intel Pentium 4 based 3D processor. B. Black et al., "3D Processing Technology and Its Impact on IA32 Microprocessors," *Proc. Int'l Conf. Computer Design* (ICCD 04), IEEE CS Press, 2004, pp. 316-318.

The remaining papers show research on circuit- and gate-level 3D implementations of several common microprocessor blocks. These four cover 3D FUB-split on-chip caches:

- J. Mayega et al., "3D Direct Vertical Interconnect Microprocessors Test Vehicle," *Proc. Great Lakes Symp. VLSI* (GLSVLSI 03), ACM Press, 2003, pp. 141-146.
- P. Reed et al., "Design Aspects of a Microprocessor Data Cache using 3D Die Interconnect Technology," *Proc. Int'l Conf. Integrated Circuit Design and Technology* (ICICDT 05), IEEE Press, 2005, pp. 15-18.
- K. Puttaswamy and G. Loh, "Implementing Caches in a 3D Technology for High Performance Processors," *Proc. Int'l Conf. Computer Design* (ICCD 05), IEEE CS Press, 2005, pp. 525-532.
- Y. Tsai et al., "Three-Dimensional Cache Design Using 3DCacti," *Proc. Int'l Conf. Computer Design* (ICCD 05), IEEE CS Press, 2005, pp. 519-524.

The rest cover other types of 3D FUB splits:

- 3D FUB-split adders. J. Mayega et al., "3D Direct Vertical Interconnect Microprocessors Test Vehicle," *Proc. Great Lakes Symp. VLSI* (GLSVLSI 03), ACM Press, 2003, pp. 141-146; and K. Puttaswamy and G. Loh, "The Impact of 3-Dimensional Integration on the Design of Arithmetic Units," *Proc. Int'l Symp. Circuits and Systems* (ISCAS 06), CD-ROM, IEEE Press, 2006.
- 3D FUB-split dynamic instruction schedulers. K. Puttaswamy and G. Loh, "Dynamic Instruction Schedulers in a 3-Dimensional Integration Technology," *Proc. Great Lakes Symp. VLSI* (GLSVLSI 06), ACM Press, 2006, pp. 153-158.

This list isn't meant to be exhaustive, but it serves as a starting point for exploring some of the existing 3D research already published.

thereby providing an evolutionary path to the revolutionary technologies of tomorrow.

Converting wire reduction to performance

We now discuss 3D in the context of an entire processor as opposed to a single block such as an on-chip cache. At the level of a complete microarchitecture, there are a few general design "styles" for translating 3D's wire reducing abilities into actual delivered performance. Largely orthogonal to the following techniques, the footprint reduction provided by 3D can result in a more compact clock distribution network, which significantly reduces the power consumed by the clock network. The shorter distance

Figure 5. An Intel Pentium 4-based microprocessor. Original 2D floorplan with two critical paths shown with arrows (a), and a refloorplanned 3D version of the same processor with critical paths removed by stacking (b).

covered by the network can also reduce timing margins due to clock skew and jitter.

Eliminating pipelined wires

Deeply pipelined, high-frequency microarchitectures contain a significant amount of wiring. The clock-cycle time might be so small that two communicating blocks might be unable to communicate with each other within a single clock cycle. As a result, many of the medium to long wires must be pipelined. For example, the Intel Pentium 4 20-stage branch misprediction pipeline contains two stages that are just pipelined wires (for example, to drive the branch misprediction signal from the execution units back to the front end).[10] In the subsequent Intel Pentium 4 in 90 nm (Prescott), the branch misprediction pipeline is 31 stages, which is likely to contain even more stages of pipelined wire.[11]

Based on the design of an Intel Pentium 4-family processor, we created an equivalent 3D floorplan (see Figure 5). To minimize redesign and floorplanning efforts, we use a partitioning scheme similar to the block (FUB) stacking we depicted in Figure 3. However, for a few special blocks, we also used a finer 3D FUB splitting. In particular, we stacked the L2 cache on itself using an odd-even bit-sliced organization, and we stacked the reservation stations (issue queue) with half of the entries on each die. Much of our effort focused on known performance-sensitive pipelines such as the load-to-use pipeline and the register file to floating-point unit data path (see Figure 5a).

By targeting the heavily pipelined wires, we were able to eliminate 25 percent of the pipe stages in the processor. (A pipe stage in this context refers to stages in the branch misprediction pipeline and to all other stages such as pipelined cache and memory accesses and post-commit resource deallocation.) Overall, the pipeline modifications resulted in approximately 15 percent improved performance, with the largest contributors coming from reducing floating-point instruction execution latency and a reduction in store instruction lifetime. For this processor design, the reduction in the clock grid resulted in a 15 percent power reduction. Our power estimates are conservative because we didn't take into account the additional power reductions we can achieve by splitting the L2 cache and the scheduler. Even so, the resulting increase in performance with simultaneous power reductions yields an attractive improvement in the processor's overall performance-per-watt ratio.

A naive two-die 3D stack could result in the accidental doubling of power density along with the corresponding thermal consequences. Our baseline 2D processor already has several thermal hotspots and many very hot locations (just slightly cooler than the worst-case hotspots), so it's critical to minimize increases to the peak power density when stacking for 3D.

Part of our floorplanning involved an iterative process of placing blocks, observing the resultant power densities and repairing the outliers. This resulted in a 1.3-times increase in the peak-power density (significantly better than the 2-times worst-case scenario, which caused a 14°C increase in the worst hotspot located over the instruction scheduler). Although the temperature increase isn't negligible, existing (albeit perhaps more expensive) cooling solutions can handle it. Furthermore, some of the performance improvement can be traded through voltage-frequency scaling to reduce the temperature. For our 3D processor, we found that an 8 percent reduction in voltage and frequency results in an overall 34 percent power reduction compared to the 2D baseline, which is enough for thermal break even—that is, the worst-case temperature is the same as that in the baseline 2D configuration. At the same time, we still retain an 8 percent performance improvement (approximately half of the original 15 percent) over the baseline.

Increasing clock frequency

The previous technique improved performance by using 3D to reduce the number of clock cycles required to perform many different operations. A second approach is to convert the latency benefits into a faster clock frequency. Generally, this approach would be difficult to apply to an existing processor microarchitecture because a well-balanced and well-tuned design would have hundreds or thousands of worst-case timing paths that would all have to be reduced to provide an overall clock-frequency improvement. However, designing a new microarchitecture specifically targeting a 3D fabrication process, we could more easily translate the wire-reduction benefits into a shorter clock cycle. Remember that while conventional techniques for increasing clock speed also increase the number of pipeline stages (which results in a penalty in instructions per cycle), 3D doesn't necessarily increase the pipe depth because the clock-speed reduction comes from reducing the total time to complete a task rather than dividing the task into smaller parts.

For this study, we started with a processor modeled on the Alpha 21364, which includes a 21264-based core (see Figure 6a) and 1.5 Mbytes of on-chip L2 cache. We 3D-split the majority of the FUBs on top of themselves, resulting in the floorplan that Figure 6b shows. For the main integer execution core (see Figure 6c), we selectively used FUB stacking for some blocks (such as the arithmetic units) and FUB splitting for others (register files and issue queue). The different stackings target different critical timing paths. For example, the arithmetic units are primarily logic-dominated, so reducing intrablock wiring wouldn't likely result in significant timing benefits.

In these cases, we chose a global arrangement that would reduce the length of the result bypass paths to attack a different critical latency. On the other hand, the latency of the issue logic is frequently one of the primary cycle-time limitations and the circuitry also contains a significant wire component. Therefore, gate-level 3D stacking ends up being an effective approach for reducing this block's critical-timing paths.

We chose the issue logic and result bypass delay to represent a processor's critical-timing paths.[12] Based on HSpice simulations, we estimate that a 21364-style processor implemented in a 70-nm process would observe a 10.3 percent improvement in clock frequency. (Our issue queue was 11.0 percent faster, and the bypass network was 10.3 percent faster.) After accounting for IPC decreases due to an increase in the number of cycles required to access main memory, our simulations reported an overall performance benefit of 8.7 percent.

Our thermal simulations indicate that the increased power density of the 3D organization results in a 17°C increase in worst-case temperature, which is on the same

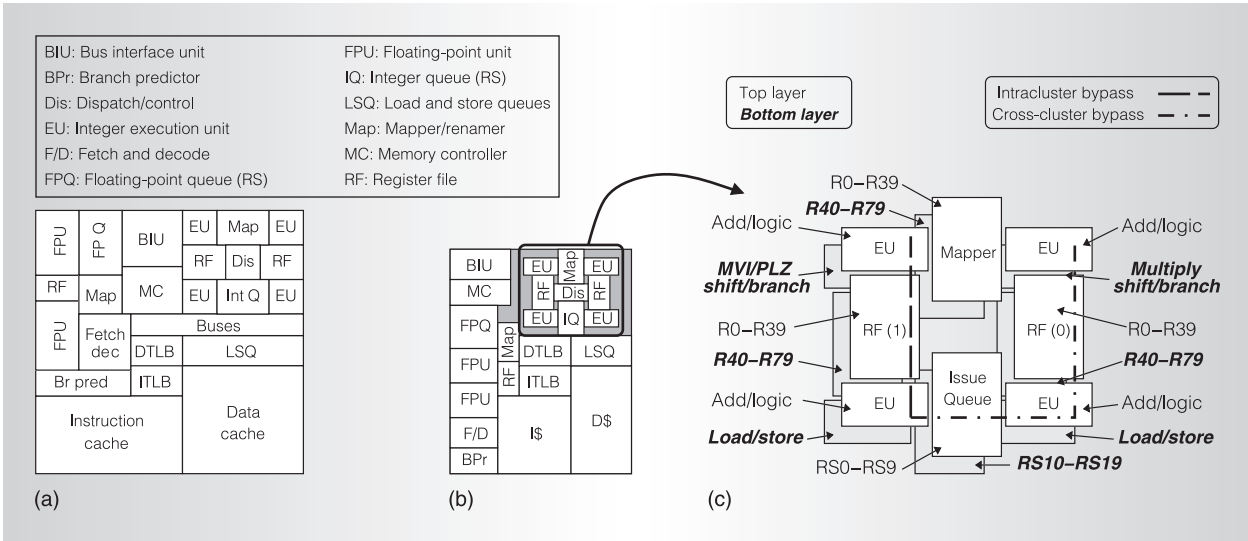

Figure 6. The Alpha 21364's 21264-based execution core. Original 2D floorplan (a), 3D floorplan with most units self-stacked on top of themselves (b), and a detailed 3D organization of the main execution logic (EBox) (c).

order as for the design in the previous section. (We generated the results for the Pentium 4 and 21364 using different sets of timing, power, and thermal estimation tools.)

Exposing more instruction-level parallelism

The last approach that we discuss doesn't attempt to reduce latency or improve clock frequency. The previous two design studies employed a 3D organization that effectively used the same total silicon area. When provided with twice the transistor density, a designer might choose to use twice as many devices rather than shrinking a constant number of devices into a smaller footprint. Our third approach to 3D processor design is to take advantage of both the timing slack from metal reduction and higher device density from 3D stacking to implement larger microarchitecture structures. The goal is to expose more ILP.

Modern processors use many structures to temporarily buffer operations and their data in an attempt to aggressively execute instructions in data-flow order. The sizes of many of these structures are limited by timing constraints, and as a result, they might induce performance-degrading stalls when the resources provided by the structures (such as physical registers) have been completely allocated. Increasing the sizes of these blocks in a 2D implementation might not be feasible due to the impact on cycle time and silicon area budget.

In a 3D implementation, we found that we could substantially increase many of these structures without impacting clock speed. For a 21264-style processor core, a two-die 3D-stacked physical register file could support 50 percent more registers and 50 percent more branch predictor state. Also, the issue queue size can be twice as large, the L1 data cache can be doubled, and the load and store queues can be 37 percent larger. This uses more total silicon area, but the overall per-die footprint is still less than the original 2D processor, so there's no risk of running up against the fabrication reticle limit. The overall performance impact is a 7.3 percent improvement, or a 9.7 percent improvement if we use better branch prediction algorithms to better use the larger instruction buffers.[8]

Hybrid optimization

In each of the brief design studies we outlined here, we took an existing processor design and effectively performed a 3D retrofit using a single approach. It's likely that a well-balanced 3D design will incorporate some of each technique, depending on the design constraints and targets of

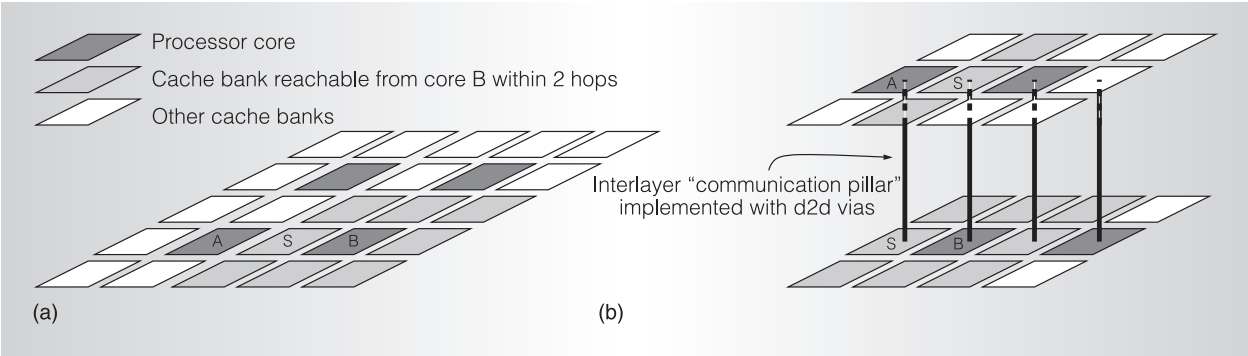

Figure 7. Processor core and nearby cache banks for 2D and 3D organizations. 2D schematic floorplan of a multicore dynamic nonuniform cache architectures (DNUCA) architecture (a), and a 3D version with improved data locality (b).

the different parts of the processor. Clock frequency improvements are probably the most effective for overall performance, but this requires fixing many timing paths. A design team might use 3D to quickly take care of the worst performers and pick off other reasonably low-effort timing paths. At same time, they can attempt to reduce the number of stages in pipelined wire made possible by a more compact 3D floorplan. Finally, depending on the left-over timing and area budget, they can attempt to increase critical structure sizes to expose more ILP.

In the context of power- and temperature-sensitive designs, design teams might need to use similar 3D techniques to target FUBs that consume too much power or result in thermal hotspots. A complete, holistic approach to processor design is necessary to fully exploit future 3D technologies.

3D for multicores

We can also explore the benefits of 3D for a multicore processor. Designers could apply the 3D techniques we describe in this section to a single-core system as well, but the benefits are more significant in a multicore environment.

Network on chip for CMP

One of the trends in high-performance microprocessor design is increasing the size of on-chip memory. This is particularly important for multicore systems where the on-chip memory must simultaneously cope with the combination of working sets from multiple threads. On the other hand, the impact of wire delay is exacerbated as technology scales, causing either a long constant memory access time or a nonuniform memory access time for large on-chip caches. This trend has stimulated the concept of nonuniform cache architectures (NUCA). In a multicore design, the inherent problem of a NUCA cache architecture is managing data the cores share. We can use data migration, such as that used in dynamic NUCA (DNUCA),[13] to address this problem.

Because of d2d vias' low latency, 3D integration can help increase the memory locality (the number of memory banks with low access latency) dramatically, such that it reduces the average L2 cache access time. Figure 7 shows a processor core and several nearby cache banks for both 2D and 3D organizations. The shaded cache banks indicate those reachable by core B in two hops or less. In the 3D case, more banks are reachable for a fixed number of hops than in the 2D case (11 versus eight). Figure 7 also illustrates an increase in the number of cache banks accessible by more than one core for a fixed latency (we marked banks reachable by both A and B in a single hop with an S), which can significantly reduce the number of cache line migrations. Our experimental results show that, compared to a 2D-DNUCA architecture, the 3D version reduces the average L2 access time by approximately 50 percent and the number of cache line migrations by 90 percent.[14] This improves the average latency of

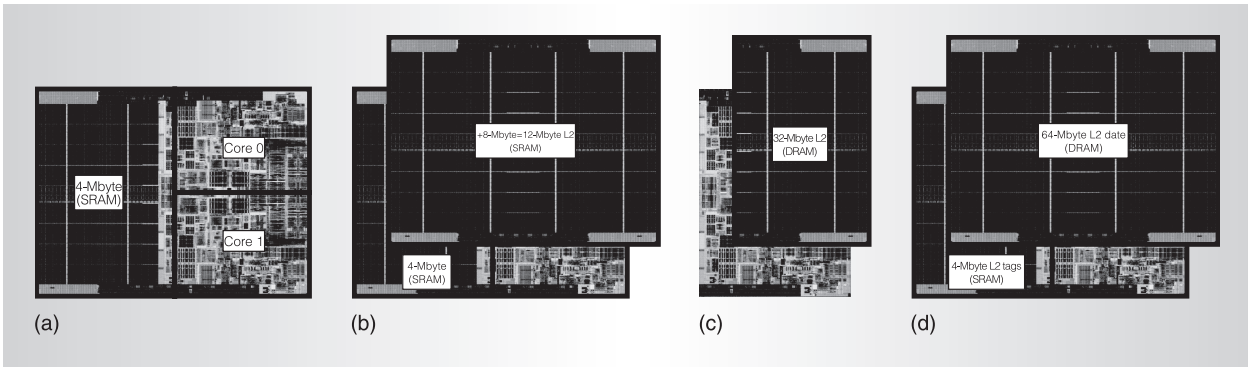

Figure 8. Using 3D integration to increase the storage capacity of on-chip memory. The original 2D floorplan (a), 3D stacking a second die of SRAM (b), replacing the original cache with a stacked die of DRAM (c), and stacking a layer of DRAM on top of the original processor (d).

accesses to shared memory, and the corresponding reduction in cache block migration results in lower power and reduced network contention compared to a 2D implementation.

Stacking a shared cache

Another approach to designing a 3D multicore system is to use the increased device density to increase the L2 cache's size. As opposed to directly decreasing critical paths' wire lengths, this stacking approach increases the chip's storage capacity to reduce the frequency of accessing one of the slowest paths—the off-chip main memory access. Figure 8 shows a baseline dual-core floorplan with 4 Mbytes of L2 cache and three variants using additional 3D-stacked SRAM or DRAM to increase the L2 cache's size. The first variant uses the additional transistors to triple the size of a conventional SRAM cache to 12 Mbytes. The second replaces the SRAM cache with a 32-Mbyte DRAM cache that's much larger but slower compared to an SRAM implementation. The third uses a large 64-Mbyte DRAM L2 cache on the top die and implements the L2 tag array with the faster SRAM on the bottom die.

To evaluate the impact of the increased capacity of the 3D stacked cache, we modeled the dual-core configurations shown in Figure 8 with the Recognition, Mining, and Synthesis benchmarks.[15] The RMS workloads are multithreaded and memory-intensive, and they represent emerging application areas in various areas including financial modeling, data mining, physics modeling, ray tracing, and security-oriented image recognition.

For all three 3D configurations, the larger caches are effective at converting off-die bus accesses into on-die cache hits. For the RMS workloads, the 32-Mbyte stacked DRAM cache provides a 13 percent reduction in the average memory access latency while simultaneously reducing off-chip bandwidth requirements by a factor of three. (Increasing it to 64 Mbytes didn't provide much additional performance.) Similar to the other 3D case studies we present in this article, the combination of higher performance and lower power yields a system with a significantly improved performance-per-watt ratio.

Before 3D processors become mainstream, many open research challenges must be addressed. Several research groups in academia and industry are already attacking many of these problems (see the "State of the art in 3D processor design" sidebar), but many issues remain. In particular, 3D will require new CAD and electronic design automation tools to assist designers and engineers in building 3D processors. Needs include new 3D place-and-route algorithms, floorplanning tools, 3D visualization and layout for 3D circuit implementation, modeling tools for 3D parasitics and timing estimation, and others. Another critical problem is in testing 3D chips. A fault on a single layer of a multidie

stack can render the entire stack inoperable. 3D processors will have to be created with a design-for-test methodology to possibly enable preassembly testing of individual dies or wafers to maximize the stacked product's yield. This might be challenging in the presence of finely partitioned 3D structures where a single die might only contain half of a complete circuit.

In terms of overall microarchitecture design, we have so far only considered "porting" traditional computing paradigms to 3D. Research is needed to innovate new microarchitectures designed from the ground up specifically to exploit 3D's strengths while mitigating its potential weaknesses (particularly thermals). Other research questions include designs for a large number of stacked layers, designs for later-generation 3D processes in which the device size is significantly smaller than the minimum d2d via pitch, and the microarchitectural impact of system-level integration. That is, what should a processor or even the entire system architecture look like when DRAM, networking, wireless, and graphics are all integrated into the same stack?

There's an abundance of interesting and important 3D processor design and research problems out there, and 3D will continue to advance rapidly in the coming years. MICRO

References

1. S. Das et al., "Technology, Performance, and Computer-Aided Design of Three-Dimensional Integrated Circuits," *Proc. Int'l Symp. Physical Design* (ISPD 04), ACM Press, 2004, pp. 108-115.
2. F. Arnaud et al., "A Functional 0.69 μm2 Embedded 6T-SRAM Bit Cell for 65nm CMOS Platform," *Proc. 19th Symp. VLSI Technology*, IEEE Press, 2003, pp. 342-351.
3. Y. Cao et al., "New Paradigm of Predictive MOSFET and Interconnect Modeling for Early Circuit Design," *Proc. 2000 Custom Integrated Circuits Conf.* (CICC 00), IEEE Press, 2000, pp. 201-204.
4. K. Puttaswamy and G.H. Loh, "Implementing Register Files for High-Performance Microprocessors in a Die-Stacked (3D) Technology," *Proc. Int'l Symp. VLSI* (ISVLSI 06), IEEE CS Press, 2006, pp. 384-389.
5. K. Puttaswamy and G.H. Loh, "Implementing Caches in a 3D Technology for High Performance Processors," *Proc. Int'l Conf. Computer Design* (ICCD 05), IEEE CS Press, 2005, pp. 525-532.
6. P. Reed, G. Yeung, and B. Black, "Design Aspects of a Microprocessor Data Cache using 3D Die Interconnect Technology," *Proc. Int'l Conf. Integrated Circuit Design and Technology* (ICICDT 05), IEEE Press, 2005, pp. 15-18.
7. Y.-F. Tsai et al., "Three-Dimensional Cache Design Using 3DCacti," *Proc. Int'l Conf. Computer Design* (ICCD 05), IEEE CS Press, 2005, pp. 519-524.
8. Y. Xie et al., "Design Space Exploration for 3D Architecture," *ACM J. Emerging Technologies in Computer Systems,* vol. 2, no. 2, Apr. 2006, pp. 65-103.
9. G. Schrom et al., "Feasibility of Monolithic and 3D-Stacked DC-DC Converters for Microprocessors in 90nm Technology Generation," *Proc. Int'l Symp. Low Power Electronics and Design* (ISLPED 04), IEEE Press, 2004, pp. 263-268.
10. G. Hinton et al., "The Microarchitecture of the Pentium 4 Processor," *Intel Technology J.*, Q1 2001; ftp://download.intel.com/technology/itj/q12001/pdf/art_2.pdf.
11. D. Boggs et al., "The Microarchitecture of the Pentium 4 Processor on 90nm Technology," *Intel Technology J.*, vol. 8, no. 1, 2004.
12. S. Palacharla, *Complexity-Effective Superscalar Processors*, doctoral thesis, Dept. of Computer Science, Univ. of Wisconsin, 1998.
13. C. Kim et al., "An Adaptive, Non-Uniform Cache Structure for Wire-Delay Dominated On-Chip Caches," *Proc. 10th Symp. Architectural Support for Programming Languages and Operating Systems* (ASPLOS 02), ACM Press, 2002, pp. 211-222.
14. F. Li et al., "Design and Management of 3D Chip Multiprocessors Using Network-in-Memory," *Proc. 33rd Int'l Symp. Computer Architecture* (ISCA 06), IEEE CS Press, 2006, pp. 130-141.
15. P. Dubey, "Recognition, Mining and Synthesis Moves Computers to the Era of Tera," *Technology@Intel Magazine* 2005; http://www.intel.com/technology/magazine/computing/recognition-mining-synthesis-0205.pdf.

Gabriel H. Loh is an assistant professor in the College of Computing at the Georgia Institute of Technology. His research interests include computer architecture, processor microarchitecture, circuit design, and 3D integration technology. He received his PhD in computer science from Yale University. He is a recipient of the National Science Foundation CAREER award.

Yuan Xie is an assistant professor in the Computer Science and Engineering Department at Pennsylvania State University. His research interests include VLSI design, computer architecture, and embedded systems design. He received his PhD in computer engineering from Princeton University. He is a recipient of Semiconductor Research Corporation's Inventor Recognition Award and National Science Foundation CAREER award.

Bryan Black is the research manager at Intel Labs in Austin, Texas. His research interests are in computing systems, computer architecture, and emerging process technologies. He received his PhD in electrical and computer engineering from Carnegie Mellon University.

Direct questions and comments about this article to Gabriel H. Loh, 265 Ferst Drive., KACB 2404, Georgia Inst. of Technology, College of Computing, Atlanta, GA 30332-0765; loh@cc.gatech.edu.