Snatch: Opportunistically Reassigning Power Allocation between Processor and Memory in 3D Stacks*

Dimitrios Skarlatos, Renji Thomas†, Aditya Agrawal‡, Shibin Qin, Robert Pilawa-Podgurski, Ulya R. Karpuzcu♈, Radu Teodorescu†, Nam Sung Kim, and Josep Torrellas

University of Illinois Urbana-Champaign †Ohio State University ‡NVIDIA Corp. ♈University of Minnesota Twin Cities

{skarlat2,sqin3,pilawa,nskim,torrella}@illinois.edu {thomasr,teodores}@cse.ohio-state.edu adityaa@nvidia.com ukarpuzc@umn.edu

***Abstract*—The pin count largely determines the cost of a chip package, which is often comparable to the cost of a die. In 3D processor-memory designs, power and ground (P/G) pins can account for the majority of the pins. This is because packages include separate pins for the disjoint processor and memory power delivery networks (PDNs). Supporting separate PDNs and P/G pins for processor and memory is inefficient, as each set has to be provisioned for the worst-case power delivery requirements.**

In this paper, we propose to reduce the number of P/G pins of both processor and memory in a 3D design, and dynamically and opportunistically divert some power between the two PDNs on demand. To perform the power transfer, we use a small bidirectional on-chip voltage regulator that connects the two PDNs. Our concept, called *Snatch*, is effective. It allows the computer to execute code sections with high processor or memory power requirements without having to throttle performance. We evaluate Snatch with simulations of an 8-core multicore stacked with two memory dies. In a set of compute-intensive codes, the processor *snatches* memory power for 30% of the time on average, speeding-up the codes by up to 23% over advanced turbo-boosting; in memory-intensive codes, the memory snatches processor power. Alternatively, Snatch can reduce the package cost by about 30%.

I. Introduction

3D-stacking is an attractive technology to increase the transistor count of chips [1], [2], [3]. When combining processor and memory dies in a stack, the resulting integration delivers a computer architecture with drastically improved energy, latency, and bandwidth characteristics.

In these architectures, the package cost can often be comparable to the cost of a die, and the number of pins dominates the cost of packages [4]. This is because the higher the pin count is, the bigger the package needs to be.

Many of the package pins are, in fact, power and ground (P/G) pins. Indeed, already in 2D designs of commercial processors, P/G pins can account for about 50% of all the pins [5]. In 3D designs, they can be responsible for an even larger fraction of the total pins. There are two reasons for this. First, the absolute P/G pin count is higher, since the package has to provide P/G pins for both processor and memory. Second, in some 3D platforms at least, the processor uses relatively fewer pins for off-chip memory, since it already has high-bandwidth paths to stacked memory. The result is that P/G pins are major factors in the packaging costs of 3D designs.

It is known that, since applications have phases, the power consumed by a processor die often varies widely over time. The same is true for the power consumed by the memory dies. As an example, Figure 1 shows the variation in the power consumed by the processor and memory dies of a 3D architecture that we will detail later, as it runs the *MG* NAS benchmark on 8 cores.

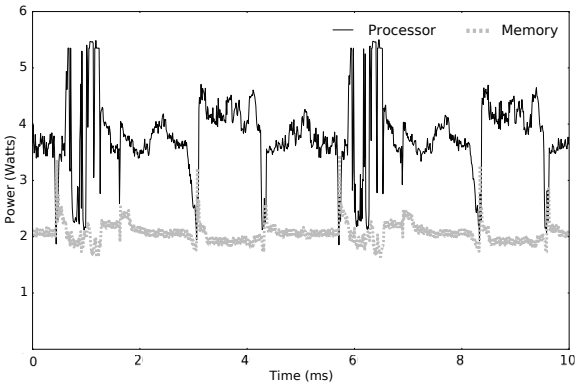


Fig. 1: Time variation of the power consumed by the processor and memory dies of a 3D stack for the *MG* benchmark of NAS with 8 cores.

In current 3D stacks, both the processor and the memory have their own power delivery networks (PDNs) and their own P/G pins. The two systems are separate, and each is provisioned for the worst-case power delivery requirements. However, we observe that it is very unlikely that both processor and memories reach their maximum power demands at the same time. Typically, the program executes a compute-intensive section or a memory-intensive section, but not both at the same time. Hence, when the processor power is high, the memory power tends not to be high, and vice-versa. As a result, providing a large power allocation to each of processor

* This work was supported in part by NSF under grants CCF-1012759, CNS-1116237, CCF-1536795, CCF-1421988, and CCF-1253933; DARPA under PERFECT Contract Number HR0011-12-2-0019; and Samsung Electronics under contract 2016-03835. Kim has a financial interest in AMD and Samsung Electronics.

978-1-5090-3508-3/16/$31.00

and memory — and, hence, a large P/G pin count to each of them — is suboptimal and increases the cost unnecessarily.

In this paper, we propose to reduce the number of P/G pins of both processor and memory, and *dynamically and opportunistically* divert some power between the two PDNs on demand. The result is a sizable reduction of the pin count and, therefore, of the packaging cost. To perform the power transfer, we propose to use a small *bidirectional* on-chip voltage regulator (VR) that connects the two PDNs. The VR can redirect some power from the memory to the processor or vice-versa, depending on which unit needs the power. In addition, the on-chip VR works together with the two off-chip VRs to reduce on-chip voltage transients.

Our concept, called *Snatch*, is effective. Compared to having two decoupled PDNs, each with the P/G pin count to independently satisfy the worst case, Snatch allows the use of notably cheaper packages. Compared to a package with two decoupled PDNs but with only as many P/G pins as Snatch, Snatch can handle code sections with high power requirements without having to throttle performance or accept dangerously-high pin currents. Such high currents can cause electromigration in the pins, and high IR drops and unsafe voltage margins in the PDNs.

We evaluate Snatch with simulations of a low-power 8-core multicore stacked with two DRAM dies. In a set of compute-intensive parallel applications, Snatch enables the processor to *snatch* power allocated to memory for 30% of the time on average, speeding-up the applications by up to 23% relative to an advanced turbo-boosted environment; in a set of memory-intensive applications, the memory snatches power from the processor. Alternatively, Snatch can reduce the package cost by about 30%.

II. Background and Related Work

A. Flexible Power Delivery

Power delivery is a critical part of chip design [6]. Power is delivered through a set of package pins that connect power lines in a board to the on-chip PDN — a network of wires that deliver the power across the die. The resistance in the pins and wires causes a voltage (V_{dd}) drop, generally referred to as IR drop. Because of this drop, the V_{dd} provided by the VRs has to be higher, increasing power consumption. The resistance in the PDN can generally be decreased by adding more pins or increasing the width of the wires — all of which increase manufacturing costs.

Given the cost of power delivery, there have been proposals to improve its flexibility. Specifically, Chen et al. [7] propose configurable pins, which can switch between I/O pins and P/G pins. Such a design enables higher performance, by allowing P/G pins to contribute to I/O transfer when higher off-chip bandwidth is needed. The cost is significant circuit redesign, both at the processor and at the off-chip memory interface — including motherboard redesign. In our work, we want to avoid redesigning chip interfaces.

It is well known that applications go through phases. Hence, it is attractive to adapt the general allocation of power based on the requirements of the phase being executed. For example, Paul et al. [8] consider a configurable system with GPU cores and an off-chip memory. The GPU cores can be configured by changing the number of compute units and the frequency (f); the memory can change the f as well. The authors reconfigure the system to provide a more powerful GPU in compute-intensive program phases, and a faster memory in memory-intensive phases. In our paper, our goal is to provide power adaptation at much finer granularity within a 3D chip using a bidirectional on-chip VR.

Within a chip, Godycki et al. [9] propose a reconfigurable power distribution network (RPDN). The idea is that multiple on-chip VRs are connected to multiple cores through an RPDN, which can adjust the connections to supply more power to some cores and less power to other cores. The design is used to enable fine-grain V_{dd} scaling, reducing the V_{dd} supplied to idle cores, while providing a higher V_{dd} to cores that are doing useful work.

In our paper, we aim to redirect power between the on-chip processor and on-chip memory PDNs in a 3D stack, by using a small bidirectional on-chip VR working synergistically with off-chip VRs. Our goal is to keep the pin count low, while providing more power to the processor or to the memory when they need it, at the expense of one another.

B. On-Chip Voltage Regulation

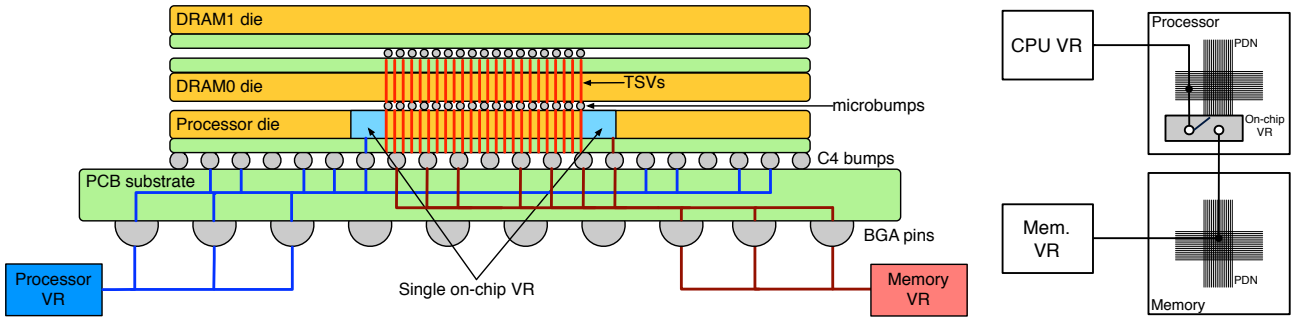
There is significant interest in building multi-phase on-chip VRs (e.g., [10], [11], [12], [13]). These VRs can provide multiple on-chip V_{dd} domains. When operating at high switching frequencies, they can enable fast V_{dd} transitions at nanosecond timescales. Their effectiveness for aggressive power management has been explored in prior work [14].

Most recently, Intel's Haswell-based Xeon processors have deployed on-chip VRs based on in-package inductors. Such design is called Fully-Integrated VR (FIVR) [10]. Because of its large power-delivery capacity, the FIVR has a substantial area and power cost, and is only used in high-end Haswell-based processors.

If we use on-chip VRs in a 3D processor-memory chip, a conventional design needs to employ at least two VRs: one for the processor PDN and one for the memory PDN. This is because processor and memory generally use different V_{dd} levels. In such a design, we need to provision each VR to support the peak power consumption expected in the corresponding PDN. As a result, these VRs have a sizable area and power cost. In our work, we want to employ smaller VRs, so that they have little power and area cost.

C. 3D-Stacked Chip Cost

The total cost of a 3D-stacked chip is the sum of the cost of manufacturing the dies, of performing the 3D bonding of the dies, and of the actual package. The last category is affected by the package type (e.g., flip-chip land grid array (fcLGA)), the package area, and the package pin count. A recent study by Dong et al. [4] carefully analyzes each of the costs, and presents models to estimate them. The paper



(a) Cross-section chip view.

(b) Interconnected PDNs.

Fig. 2: Overview of the layout of the Snatch power delivery network.

shows that the overall package cost for a given package type can be dominated by the package pin count. Moreover, in 3D designs for manycore multiprocessors, the package cost is often comparable to the cost of a die. Hence, if we reduce the number of pins, we are able to reduce the cost of 3D-stack chips significantly.

III. Snatch Design

This section presents the design of the Snatch architecture. It describes Snatch's PDNs (Section III-A), on-chip VR (Section III-B), pin reliability considerations (Section III-C), and a hardware algorithm that dynamically anticipates when reassignments of power are necessary (Section III-D).

A. Power Delivery Networks

Consider a 3D architecture like the one in Figure 2(a), with a processor die at the bottom of the stack, and two memory dies on top. In this environment, conventional designs use two separate power domains: one for the processor die, and one for the memory dies. Each power domain is supplied by an off-chip VR. Each of these VRs delivers power through a set of package pins as shown in the figure, through the PCB substrate in the package, and to a set of C4 bumps. The C4 bumps that deliver power from the processor VR are connected to the metal layer of the processor die; the ones that deliver power from the memory VR are connected to the power TSVs and, from there, to the memory metal layers. The two PDNs are normally isolated.

In conventional designs, the processor and the memory PDNs and their subsystems (off-chip VR, package pins, and C4 bumps) are sized to support the highest power and current that the processor and memory, respectively, are expected to consume. In this paper, we propose to size each of them for only a fraction of their maximum consumption; then, when a PDN needs extra power, it effectively seizes it from the other PDN — a process we term *Snatching*. The result is a substantial packaging cost reduction.

To allow cores to snatch spare power capacity from the underutilized memory PDN, and vice-versa, we connect the two PDNs with a single, small on-chip VR located on the processor die. Figure 2(a) shows where the on-chip VR is, and Figure 2(b) shows how it is connected at a high level.

The on-chip VR is needed because the processor and memory power domains generally use a different V_{dd}. The VR bridges the two PDNs by stepping the voltage down/up as needed. When the processor snatches power, the on-chip VR steps the memory V_{dd} down, and supplies additional current to the processor die; when memory snatches power, the on-chip VR steps the V_{dd} up and supplies the memory dies with additional current. The on-chip VR also allows the two power domains to perform DVFS independently, even when the PDNs are bridged. This includes, for instance, putting the memory in a low-power state while allowing the processor to consume additional power in a compute-intensive code section.

Bridging the two PDNs for power snatching ensures that the current density per C4 remains as in nominal conditions. When the processor is snatching power from the memory, the extra power is delivered through package pins and C4s assigned to the memory. When the memory is snatching power, the opposite occurs. This avoids any increase in IR drop or any degradation in lifetime reliability resulting from insufficient package pins or C4s.

B. On-Chip Voltage Regulation

To enable dynamic reallocation of power between the two PDNs, Snatch uses a small multi-phase on-chip VR, as shown in Figure 3. The VR is on the processor die, and can be implemented to use little area and have a high power efficiency. This is because it only needs to supply a fraction of the power provided by each of the off-chip VRs. Moreover, while the off-chip VRs receive the 12V supply from the power supply unit in the platform, and down-convert it to the on-chip voltages used by processor and memory, the on-chip VR only needs to up- or down-convert a few hundreds of mV.

Given the requirements of this system and the state-of-the-art solutions available [10], [11], [12], [13], we choose to implement a multi-phase *bidirectional* switched inductor converter. It operates as a buck converter when the processor snatches power, and as a boost converter when the memory snatches power. The switched inductor topology naturally supports bidirectional power flow with proper control [15]. Thus, the current of the on-chip VR can be controlled dynamically in both directions.

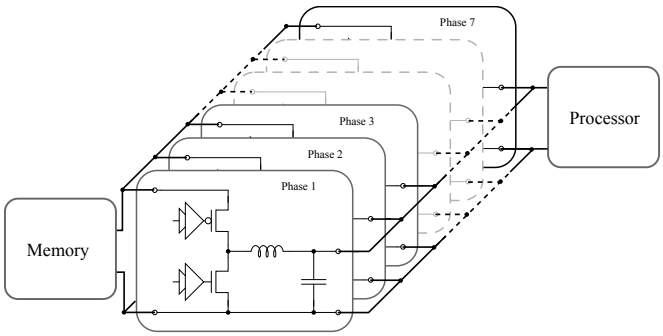


Fig. 3: Schematic of the on-chip VR and its connections to the processor and memory PDNs.

In a switched-inductor based on-chip VR, the inductor choice is critical. We base our design on Intel's FIVR [11], which uses the bottom metal layer of a flip-chip package to implement the air-core inductors. Complete specifications of the FIVR package inductor are not available, but we make a best-effort estimate based on other available information [11], [16], [17]. Moreover, we validate our design with figures of merit from some commercially-available package inductors [18] that have been used in on-chip VR designs [19].

We assume an off-chip VR for the processor that is provisioned for 5.5W, operating from 0.80V (nominal conditions) up to 0.95V. The off-chip VR for the memory is provisioned for 4.5W at 1.1V. Moreover, the on-chip VR can supply up to 2.5W. With these specifications, we use Cadence tools to perform simulations with commercial TSMC 65nm CMOS technology. We assume inductor technology of 2nH and 46mΩ at 100MHz. We derive the design following the procedure outlined in [20], and scale to 22nm technology. The estimated design is shown in Table I.

On-Chip VR Parameters	
Output V_{dd}	0.80 V to 0.95 (processor PDN) 1.1 V (memory PDN)
Power	2.5 W
Number of phases	7
Total PMOS trans.	W=10086.36μm, L=22nm / phase
Total NMOS trans.	W=2241.41μm, L=22nm / phase
Switching frequency	129.8 MHz
Total inductor area	18.9 mm^2 on package substrate
Power efficiency	92% (memory to processor) 90% (processor to memory)

TABLE I: Proposed on-chip VR design at 22nm.

As shown in the table, the design has 7 parallel phases. We implement interleaving to minimize output V_{dd} ripple. The inductor takes significant area, but it is not placed on the die but on the package. We estimate the power efficiency to be 92% when down-converting and 90% when up-converting. Simulations using Cadence validate these estimations.

An important advantage of the on-chip VR is its fast dynamic response, compared to an off-chip VR. As a result, it supports fast V_{dd} transitions. As demonstrated in [11], with proper control design, on-chip VRs can provide a bandwidth of tens of MHz, whereas off-chip VRs offer only tens of KHz. We estimate our design to attain a switching frequency of 129.8 MHz. Therefore, the on-chip VR can absorb load transients within the chip. Furthermore, it can reduce the regulation requirement on the off-chip VR. Section IV-A considers this issue further.

C. Pin Reliability

The lifetime of pins is affected by wear-out induced by electromigration (EM). EM causes gradual mass transport in metal conductors along the direction of an applied electric field, potentially leading to both open- and short-circuit failures. The impact of EM increases with increasing current density (j). Both pins and the rest of the PDN are vulnerable to EM because they experience large uni-directional currents [21]. This sustained stress accelerates the onset of EM-induced failures.

The lifetime of a pin under EM is measured by its Mean Time To Failure (MTTF). Using Black's model [22], we have:

$$MTTF = Aj^{-n}exp(Q/kT) \quad (1)$$

where A is a constant that depends on the pin geometry, Q is the EM activation energy, k is Boltzmann's constant, n is a material-specific constant, and T is the temperature. Following [23], [24], we use an adjusted version of Black's equation to account for current crowding and Joule heating:

$$MTTF = A(cj)^{-n}exp[Q/k(T+\Delta T)] \quad (2)$$

where c is a material-specific constant.

Consider a chip where the pins for the processor PDN and those for the memory PDN are provisioned for the nominal power allocated to the processor and to the memory, respectively. With Snatch, the processor can increase its power consumption above its allocation by snatching power from the memory. This is done without increasing the current in the pins above what they were designed for. Hence, the MTTF of the pins does not decrease. Without Snatch, if the processor increases its power above its nominal allocation, more current flows in its pins than they were provisioned for. Hence, the MTTF of the pins decreases.

D. Snatch Algorithm for Power Reassignment

To understand the Snatch power reassignment algorithm, assume that the off-chip processor and memory VRs are dimensioned to provide power up to P_{PROC} and P_{MEM}, respectively, and the on-chip VR can transfer P_{SNATCH} from one PDN to the other. When the algorithm estimates that one of the units (processor or memory) can use more power, it tries to boost the V_{dd} and f of the unit until it reaches its maximum power (P_{PROC} or P_{MEM}), and then even more until the unit snatches all P_{SNATCH} from the other unit. To be effective, the algorithm only boosts the V_{dd} and f of the processor or memory unit if the code being executed is compute or memory intensive, respectively. If a code section is both compute and memory intensive, the dominating behavior is the one that determines the type of boosting performed.

The algorithm uses three main inputs: the epoch size (E), the Characteristic Table (CT), and the activity factors of processor (P_{act}) and memory (M_{act}). Every E cycles, the algorithm takes power measurements and can potentially change V_{dd} and f. Since these actions involve no software, E can be as short as 10 μs. The CT stores the V_{dd}-f bins available for the processor and for the memory. For each bin i, it stores a conservative estimate of the power P_i that we need to reserve for the average application running at this bin. The bins are shown in Section V-A.

The activity factors (P_{act} and M_{act}) are measurements that estimate if the epoch is compute or memory bound. The algorithm uses them to decide whether to boost the V_{dd} and f of a unit. In our case, P_{act} is the power consumed by the processor in the epoch as a fraction of P_{PROC}. If such value is over a threshold P_{ACT}, the algorithm claims that the epoch is compute bound. Similarly, if M_{act} is over a threshold M_{ACT}, the algorithm claims that the epoch is memory bound.

The algorithm makes cautious decisions. First, to determine what power to assign to the processor and memory units in the next epoch (P_{proc} and P_{mem}), it takes the maximum power consumed by the unit in any of the last N epochs. Section IV-B describes the hardware used to do it. In addition, when the algorithm has increased the V_{dd}-f bin for a unit, it waits for N_{WAIT} epochs before further increasing the unit's bin. This is to avoid costly, repeated changes of settings. However, the algorithm places no restrictions on how frequently V_{dd}-f can be reduced; the power saved can be re-assigned to the other unit.

To predict the activity factors of the processor and memory units in the next epoch, the algorithm takes the average P_{act} and M_{act} values in the last N epochs. Also, when the algorithm assigns the power allocation to processor and memory for the next epoch, it always leaves an unallocated power reserve equal to P_{MARGIN}, in case the prediction is inaccurate. Note that if the system attempts to consume more than the maximum power available (i.e., P_{PROC} + P_{MEM}), the corresponding unit is automatically throttled.

The algorithm proceeds in two steps. The first one estimates how much power is available (P_{avail}) and the type of execution regime. Specifically, the algorithm predicts the power that the two units will consume in the next epoch (P_{proc} and P_{mem}) and the activity factor of the two units (P_{act} and M_{act}) — all based on the last N epochs. Then, the value of P_{avail} is computed as P_{PROC}+P_{MEM}-P_{proc}-P_{mem}-P_{MARGIN}. A comparison between P_{act} and P_{ACT}, and between M_{act} and M_{ACT} determines the degree to which the execution is (or is not) compute- or memory-intensive.

The second step decides on the new power assignment. If P_{avail} is positive, the algorithm checks if the execution is compute- or memory-intensive (or which regime dominates, if both are true). If the execution is compute-intensive (and more so than memory intensive), the algorithm tries to assign P_{avail} to the processor. To do so, it checks the reserved power of the current V_{dd}-f bin of the processor (Pi) and the ones for the few notches up ($Pi+1$, $Pi+2$, ...), the current processor power (P_{proc}), and P_{avail}. Based on this, the algorithm may decide to increase the V_{dd}-f bin of the processor by one or more notches. If the execution is memory intensive, a similar algorithm is followed for the memory. If the execution is both compute- and memory-intensive, only one unit is turbo-boosted in this way.

If, instead, P_{avail} is negative, the algorithm tries to reduce the V_{dd}-f bin of one of the units (or both). It starts with the unit with the lowest relative activity. It tries to reduce one or more notches of its V_{dd}-f bin. If the power thus saved is less than P_{avail}, the algorithm tries to reduce the V_{dd}-f bin of the other unit. The goal is to save P_{avail}.

In addition to these actions, if the processor or memory has an activity lower than P_{ACT} or M_{ACT}, respectively, the algorithm attempts to reduce its V_{dd}-f bin a notch to save power. This is done to minimize wasted power.

This Snatch algorithm has several advantages over the seemingly-simple approach of simply reacting to voltage droops as soon as they are detected. First, Snatch makes educated guesses on how much power should be reallocated. Without our algorithm, when a droop occurs in one unit (processor or memory), it is unclear how much power should the unit receive, or how much power is available to be taken from the other unit. The second advantage of Snatch is that, using its predictions, it can provide extra power to a unit and improve performance even when there is no voltage droop. Finally, Snatch provides better power management because it estimates the power needs and availability in advance; if the system simply responded to voltage droops, it would need to issue very fast responses, which are likely less optimal.

IV. Implementation Issues

A. On-Chip VR Reduces Voltage Droops

The fast dynamic response of the on-chip VR allows it to absorb load transients within the chip. To illustrate this benefit, we examine the worst-case scenario where the processor wants to quickly ramp-up its power from 1.67 W (i.e., the power consumed when the processor is idle) to 7.5 W (i.e., the nominal power of the processor plus P_{SNATCH}), snatching the necessary power from the memory. We perform simulations using the model and data provided in [25], which includes the parasitics of the BGA pins and C4 bumps. In the simulation, an off-chip VR supplies the processor power. Details on the design of the off-chip VR can be found in [26].

Figure 4(a) shows the current supplied by the off-chip VR, with and without the on-chip VR. Figure 4(b) shows the resulting voltage transients. In the voltage plot, we show four curves, namely the V_{dd} at the output of the off-chip VR (*External Voltage*), and at the processor die (*Die Voltage*), both with and without the on-chip VR.

Without the on-chip VR, power snatching is not enabled, and the off-chip VR sees the entire magnitude of the current ramp (Figure 4(a)). Moreover, there is a significant V_{dd} droop in the processor die (Figure 4(b)). Note that the processor V_{dd}

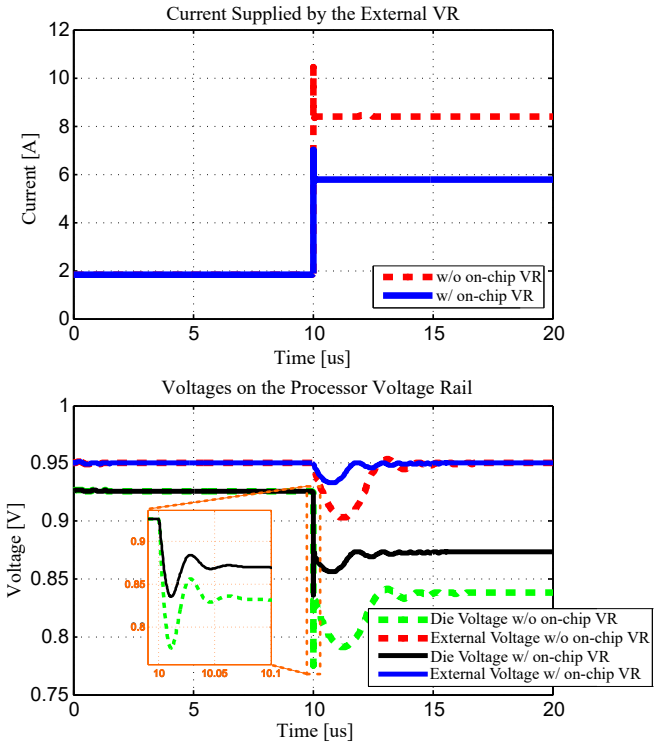


Fig. 4: Current and voltage with and without the on-chip VR.

rail also has an IR voltage drop in steady state. This problem is often fixed by active voltage positioning (AVP) control of the off-chip VR.

On the other hand, with the on-chip VR, power snatching is enabled, and the on-chip VR is able to provide some power to the processor. Since the on-chip VR has a very high current slew rate and high control bandwidth, it can absorb a portion of the processor current ramp, such that a smaller magnitude is seen by the off-chip VR (Figure 4(a)). The result is that the on-chip V_{dd} droop is greatly alleviated (Figure 4(b)).

In other words, the on-chip VR distributes the load current ramp between the off-chip VR for the processor and the off-chip VR for the memory, such that the V_{dd} droop in each individual voltage rail is reduced. The same idea applies when memory snatches power from the processor.

B. Predictor for Power Consumption

As part of the algorithm for power reassignment described in Section III-D, Snatch needs to compute two pairs of values. The first pair is the *maximum* power consumed in any of the last N epochs by the processor and by the memory. This pair is used to predict the future power needs of processor and memory, respectively. The second pair is the *average* of the power consumptions in the last N epochs by the processor and by the memory. This pair is used to estimate the activity of the processor and the memory, respectively. As we will see in Section V, $N = 64$.

Snatch implements these computations by storing the measured values of the power consumed by the processor and the memory in each epoch in two circular shift registers (*CSR*) — one for the processor and another for the memory. There are also two pointers associated with each *CSR*. The first one ($PtrCSR_{max}$) points to the current maximum value in the shift register; the second one ($PtrCSR_{last}$) points to the current last entry in the shift register (i.e., the tail). In addition, for each shift register, two registers hold the sum of all the current values in the shift register (CSR_{sum}) and the average of such values (CSR_{avg}).

The logic to compute the maximum value is as follows. At the end of an epoch, the new power value is stored at the entry after the current tail, and $PtrCSR_{last}$ is updated to point to it. If this power value is greater than or equal than the one pointed to by $PtrCSR_{max}$, then $PtrCSR_{max}$ is updated to point to it. Otherwise, $PtrCSR_{max}$ remains the same. A problem occurs if $PtrCSR_{max}$ is pointing to the value that is currently being shifted out of the shift register, and the new power value that is being shifted in is less than it. In this case, we need to recompute the maximum value in the shift register by traversing the entire shift register. However, we do not need to wait for this computation, and we can use the value that was shifted out for the duration of the new epoch. Note that this approach is conservative, as the new maximum value will always be smaller than the previous one.

The logic to compute the average value (CSR_{avg}) is as follows. At the end of each epoch, the incoming value is added to CSR_{sum}, and the value being shifted out in the *CSR* is subtracted from CSR_{sum}. Finally, to calculate the average (CSR_{avg}), we perform a right shift by six on CSR_{sum}, since N is equal to 64.

Overall, this implementation minimizes the overhead of calculating the maximum and average values, as the critical path for the maximum value contains only a comparison operation. The new average is computed during the current epoch, and is used for the next epoch.

V. Evaluation Methodology

A. Modeled Architecture

We use the SESC [27] cycle-level architecture simulator to model a 3D architecture with an 8-core multiprocessor die and two DRAM memory dies on top of it. Our design closely models contemporary mobile processors with low-power consumption such as [28], [29]. Specifically, as indicated in Section III-B, the off-chip VR for the processor is provisioned for 5.5W, while the off-chip VR for the memory is provisioned for 4.5W. The architecture parameters are shown in Table II. We use 22 nm technology. Each core is 4 issue and out of order. It has private L1 instruction and data caches, and a private L2 cache. A snoopy MESI protocol using a wide bus maintains coherence between the L2s. Each of the two DRAM memory dies has 2 GB of memory.

Table II also shows the parameters of the Snatch algorithm for power reassignment, as described in Section III-D. At the heart of the algorithm lies the Characteristic Table (*CT*), shown in Table III. The Snatch algorithm uses the *CT* to estimate the available power budget, and to make decisions regarding how many upward f–V_{dd} steps are possible, or how

Processor Parameters	
Multicore chip	22nm, eight 4-issue out-of-order cores
Frequency; V_{dd}	1.2–1.5 GHz (Baseline 1.2 GHz); 0.8–0.95 V
Inst. L1 cache	32 KB, 2 way, 2 cycles Round Trip (RT), 64 B line
Data L1 cache	32 KB, 2 way, WT, 2 cycles RT, 64 B line
L2 cache	512 KB, 8 way, WB, private, 10 cycles RT, 64 B line
Network; Coherence	512 b bus; Bus-based snoopy MESI protocol at L2
Stacked DRAM Parameters	
Dies; Channels	2; 4
Ranks/die; Banks/rank	16 (8 per channel); 8
Memory controllers	4 Wide I/O DRAM controllers
Capacity	2 GB/die = 4 GB total in stack
Freq; Data rate; V_{dd}	400–900 MHz (Baseline 400 MHz); DDR; 1.1 V
Snatch Algorithm Parameters (From Section III-D)	
E = 10μs; P_{PROC} = 5.5W; P_{MEM} = 4.5W; P_{SNATCH} = 2W; P_{MARGIN} = 0.5W P_{ACT} = 0.45; M_{ACT} = 0.75; N = 64 epochs; N_{WAIT} = 64 epochs	
Cooling Parameters	
Heatsink type	Passive heatsink
Convection resistance	3.0 °C/W
Dimensions of Stack Layers	
Heat sink	3.0x3.0x0.7 cm^3
DRAM silicon	100 μm
DRAM metal	2 μm
Die-to-die	20 μm
Processor silicon	100 μm
Processor metal	12 μm

TABLE II: Architectural parameters.

many downward f–V_{dd} steps are required so as to stay within the available power budget.

Processor Bins	
Frequency, V_{dd}	Power Upper Bound
1.2 GHz, 0.80 V	5.5 W
1.3 GHz, 0.85 V	6.2 W
1.4 GHz, 0.90 V	6.8 W
1.5 GHz, 0.95 V	7.5 W
Memory Bins	
Frequency, V_{dd}	Power Upper Bound
400 MHz, 1.1 V	4.5 W
900 MHz, 1.1 V	6.5 W

TABLE III: Snatch characteristic table.

The *CT* is a look-up table with (f–V_{dd}, Power) tuples. The power in an f–V_{dd} bin is the estimated upper-bound power that can be consumed in that bin; if the application attempts to consume more power, it is throttled. The difference in power between two consecutive bins is the estimated increase in power needed to move from the lower to the upper bin. Snatch uses it as follows. Assume an application operating at f_i–V_{ddi} whose performance Snatch wants to boost. The Snatch hardware compares P_{avail} to the difference between the power values in the f_{i+1}–V_{ddi+1} entry and in the f_i–V_{ddi} entry. Snatch sets the application to f_{i+1}–V_{ddi+1} only if P_{avail} is larger than or equal to the difference. Multiple f–V_{dd} step changes are possible.

Table II also shows details of the cooling and stack layers modeled. We model a passive heat sink and state-of-the-art stack layers.

B. Power Delivery Network Modeling

We develop a detailed model of the chip's power delivery network, which allows us to determine the IR drop. A chip's power delivery infrastructure consists of both off-chip and on-chip components. The off-chip components include a VR, capacitors used to stabilize V_{dd}, and wires.

Our off-chip power delivery model follows the design layout and component characteristics of the Intel Pentium 4 packaging used in prior work [30], [31], [32], [33]. The values are summarized in Table V. The circuit layout is shown in Figure 5. The impedance of this distributed model is characterized and noted to be very close to those obtained in previous work [30], [31] for similar chip characteristics.

Resistance		Inductance		Capacitance	
R_{pcb}	94$\mu\Omega$	L_{pcb}	21pH	C_{pcb_p}	240μF
R_{pcb_p}	166$\mu\Omega$	L_{pcb_p}	19.536μH	C_{pkg_p}	26μF
R_{pkg}	1$m\Omega$	L_{pkg}	120pH	C_{onDie}	335nF
R_{pkg_p}	541.5$\mu\Omega$	L_{pkg_p}	5.61pH		
R_{grid}	50$m\Omega$	L_{grid}	5.6fH		

TABLE V: RLC component values.

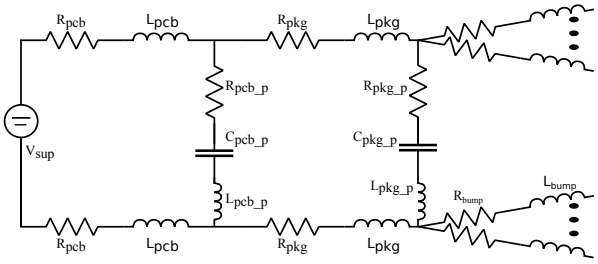


Fig. 5: Off-chip component of the power delivery network.

On the chip, power is delivered through a set of pins and C4 pads. These connect to a network of wires that deliver the required voltage to the various chip components. We model the on-chip power grid using a distributed RLC network similar to those used by prior work [31], [34]. Wiring is modeled as an RL network with two planes — one for the V_{dd} and one for the V_{ss} — connected by capacitors. Current sinks across the capacitors are used to model the current drawn by the various functional units, as in Herrell and Beker [35]. C4 bumps are placed uniformly throughout the entire chip.

The inputs to the model consist of current traces at functional unit granularity. To resolve the power delivery network we use a specialized RLC solver based on a pre-conditioned Krylov-subspace iterative method. We developed such method based on models by Chen and Chen [36]. The model output consists of supply voltage distributions for the CPU and memory dies. This allows the measurement of the IR drop in the different configurations we evaluate.

C. Modeling Infrastructure

We use the SESC [27] architectural simulator, together with the DRAMsim2 [37] memory system simulator modified to model a Wide I/O memory configuration [38], [39], to estimate performance, and McPAT [40] to estimate energy consumption. We use ArchFP [41] to design the processor and memory floorplans, and HotSpot [42] for the thermal analysis of the 3D stack.

To evaluate the architecture, we use 21 applications from three suites, which we logically organize into two groups.

Configurations Evaluated	
Baseline	**Conventional system**: runs at nominal conditions (1.2GHz & 0.8V for the processor, and 400MHz & 1.1V for the memory); no turbo-boosting or power snatching; processor and memory power are limited to P_{PROC} and P_{MEM}, respectively.
FGTurboBoost	**Fine-grained advanced turbo-boosting**: use the Snatch algorithm to enable V_{dd} and f increases in the processor and memory (up to 1.5GHz & 0.95V for the processor, and 900MHz & 1.1V for the memory), getting as much power as needed, but no more than P_{PROC} and P_{MEM}, respectively (i.e., no power snatching).
Snatch	**Snatch system**: use the Snatch algorithm to enable V_{dd} and f increases in the processor and memory (up to 1.5GHz & 0.95V for the processor, and 900MHz & 1.1V for the memory), getting as much power as needed, but no more than P_{PROC}+P_{SNATCH} and P_{MEM}+P_{SNATCH}, respectively (i.e., power snatching is allowed).
Snatch(M→P)	Snatch system except that we disallow the memory from snatching power from the processor (i.e., any power transfer goes from memory to processor).
Snatch(P→M)	Snatch system except that we disallow the processor from snatching power from the memory (i.e., any power transfer goes from processor to memory).

TABLE IV: Configurations evaluated in this paper. In all cases, if the processor or memory attempts to surpass its maximum allocated power, it is automatically throttled.

One group has compute-intensive applications. It contains 10 applications from SPLASH-2 [43] and 7 from NAS [44]. Each experiment runs one of these parallel applications with 8 threads. The other group is more memory-intensive. It contains 4 applications from SPEC2006 [45]. Each experiment runs 8 instances of the same application on the multicore.

The applications and input sets are as follows. From SPLASH-2, we use Barnes (16K particles), Cholesky (tk29.O), FFT (220), FMM (16K), LU (512x512), Radiosity (batch), Radix (2M keys), Raytrace (teapot), Water-Nsquared (512 molecules), and Water-Spatial (512 molecules). From NAS, we use BT and FT (S size), and CG, IS, LU, MG, and SP (W size). From SPEC, we use mcf (train), milc (train), lbm (train), and bzip2 (dryer.jpg).

D. Configurations Evaluated

We evaluate the five configurations in Table IV. Baseline is a conventional system that always runs at nominal conditions: 1.2 GHz and 0.8 V for the processor, and 400 MHz and 1.1 V for the memory. In addition to Snatch, we evaluate three other configurations: FGTurboBoost, Snatch(M→P), and Snatch(P→M). FGTurboBoost is an environment with fine-grained advanced turbo-boosting. It uses the Snatch algorithm of Section III-D to enable V_{dd} and f increases in the processor and memory, getting as much power as needed, but no more than P_{PROC} and P_{MEM}, respectively (i.e., no power snatching). To be conservative, we compare Snatch to FGTurboBoost when we report overall speed-ups, since the only difference between the two is the *snatching* of provisioned power between processor and memory. Snatch(M→P) and Snatch(P→M) are the Snatch system except that we only allow power snatching in one direction — from memory to processor or from processor to memory, respectively. In all cases, if the processor or memory attempts to surpass its maximum allocated power, it is automatically throttled.

VI. Evaluation

A. Hardware Cost

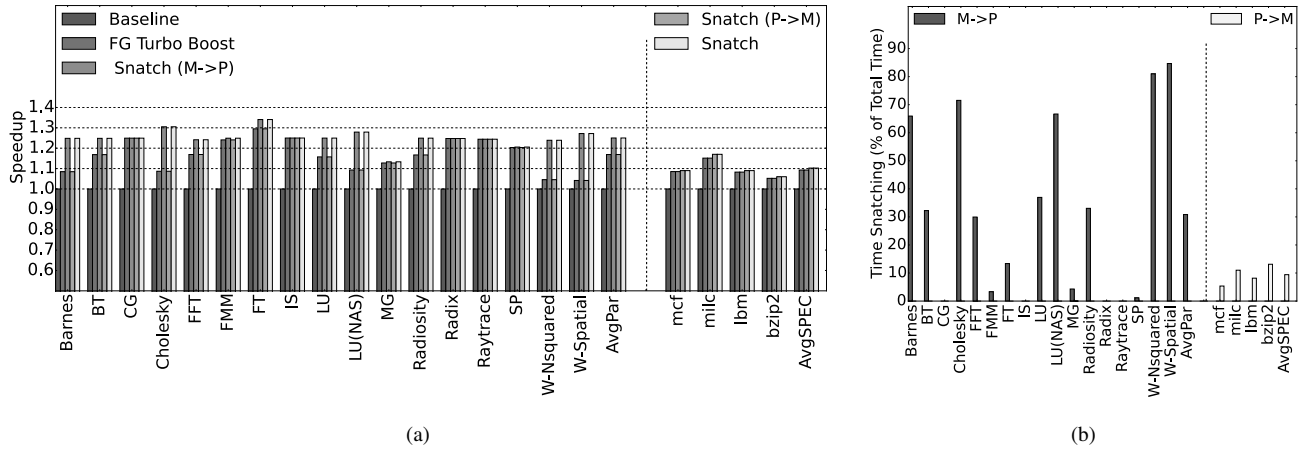
The number of P/G pins in our baseline is provisioned for nominal conditions. This corresponds to a maximum power of P_{PROC} = 5.5W and P_{MEM} = 4.5W, for a total of 10W. Since each pin can handle a maximum current of about 250mA [24], the number of P/G pins in our baseline is approximately 100. With Snatch, we keep the same total power, but can provide up to P_{SNATCH} = 2W to the processor or the memory by snatching power from the other unit. If we used a conventional design and wanted to provide an additional 2W to the processor and 2W to the memory, we would need a total maximum power of 14W. This requires about 140 P/G pins.

From this simple calculation, we can see the advantages of Snatch. Compared to the conventional system above with the same total maximum power, it reduces the number of P/G pins from 140 to 100, which is nearly 30%. This reduces the package cost substantially, as the package cost is nearly linearly proportional to the package pin count [4]. On the other hand, adding this small on-chip VR increases the processor die area a little, but it has practically no impact on the package cost, as package sizes change in a discrete manner. Lastly, note that the on-chip VR inductor is not on the processor die. It utilizes a layer in the package and, therefore, does not increase the die area (or the material cost). Hence, overall, Snatch practically reduces the package cost by about 30%.

B. Impact of Snatch on Performance and Power

1) Performance Improvements: Figure 6(a) shows the performance improvements attained by the different configurations running our applications. On the left side, we have bars for the parallel applications, followed by the average (*AvgPar*); on the right side, we have bars for the SPEC workloads, followed by the average (*AvgSPEC*). For each application, we show bars for Baseline, FGTurboBoost, Snatch(M→P), Snatch(P→M), and Snatch, all normalized to Baseline.

Consider the parallel applications first, which are generally compute intensive. Snatch speeds-up these applications by 13–34% relative to Baseline, and by 0–23% relative to FGTurboBoost. The gains over FGTurboBoost are substantial, and are the result of the f–V_{dd} boost enabled by power



(a) (b)

Fig. 6: Execution speedup of various Snatch configurations (a), and fraction of time when Snatch snatches power (b).

snatching between processor and memory. Note that many applications already do well with FGTurboBoost. This is because their power requirements with maximum turbo-boosting do not exceed the Baseline power provisioning. In particular, 7 out of the 17 applications do not require power snatching. On the other hand, applications such as Water-Spatial show the potential of Snatch. For these applications, the processor power consumption at nominal conditions is close to P_{PROC}, and the memory power consumed is well below its allocation.

From the figure, we see that Snatch(P→M) gains practically nothing over FGTurboBoost, and that Snatch(M→P) performs as well as Snatch. This is because these are compute-intensive applications, and the memory never needs to snatch power.

Consider now the SPEC workloads, which are more memory-intensive. FGTurboBoost already speeds-up these workloads over the Baseline significantly. This is because, in these workloads, the P_{PROC} and P_{MEM} power budgets are ample enough for the memory to run at the higher f, and there is not much extra benefit from snatching from the processor. As a result, Snatch only provides a small improvement of 0.5-2% over FGTurboBoost. We also see that Snatch(M→P) gains practically nothing over FGTurboBoost, and that Snatch(P→M) performs as well as Snatch.

2) Characterizing Snatch: To understand the performance improvements, Figure 6(b) shows the fraction of the time when power snatching occurs in Snatch. The figure is organized with the parallel applications and their average (*AvgPar*) on the left, and the SPEC workloads and their average (*AvgSPEC*) on the right. From the figure, we see that, in the parallel applications, only the processor ends up snatching power, while in the SPEC workloads, only the memory ends-up snatching power.

If we focus on the parallel applications first, we see that, on average, processors snatch power from the memories about 30% of the time. The height of individual bars is correlated with the difference between the Snatch and FGTurboBoost bars in Figure 6(a) for the same application. We see that, in some applications like CG, IS, Radix, and Raytrace, processors do not need to snatch power, because they are able to turbo-boost to the maximum f–V_{dd} bin without any snatching. In other applications, like Water-Spatial, processors can use extra power beyond their default P_{PROC} allocation; this extra power is attained by snatching power from memory. In other applications, such as FFT, the processor can use the extra power for the majority of the execution time, but often there is no extra power available from the memory. As a result, the speed-up is modest.

Inspecting the SPEC workloads, we see that, on average, memories snatch power from the processor about 10% of the time. The memories already attain most of the power with FGTurboBoost and, therefore the impact of snatching on the execution time is very small.

Figure 7 shows the accuracy of the Snatch predictor algorithm described in Section III-D. The figure shows multiple bars for each application, and for the average of all 21 applications. Each bar showcases a different prediction scenario. The first bar shows the fraction of epochs when the prediction is *Safe*. We define a safe prediction when the predicted available power (P_{avail}) is no higher than the actual available power plus the safety margin P_{MARGIN}. We see that, on average, 99% of the epochs fall in this category. Hence, Snatch's predictions are very safe. In the remaining 1% of the epochs, the system gets automatically throttled.

The next three bars depict the case when the predictor makes a prediction that is *Not Wasteful*. A wasteful prediction underestimates the available power by more than P_{MARGIN}. This means that we had more power available and missed the opportunity to improve performance. We consider three scenarios: the predictor makes a wasteful prediction for at least three epochs in a row, for at least 10 epochs in a row, and for at least 20 epochs in a row. The duration of the wasteful predictions is significant: short durations are more acceptable that longer ones, because there is a 10μs overhead in changing the frequency. Hence, it is not worthwhile to change frequencies for short wasteful periods.

The three bars labeled *NoWaste* show 1 minus the fraction

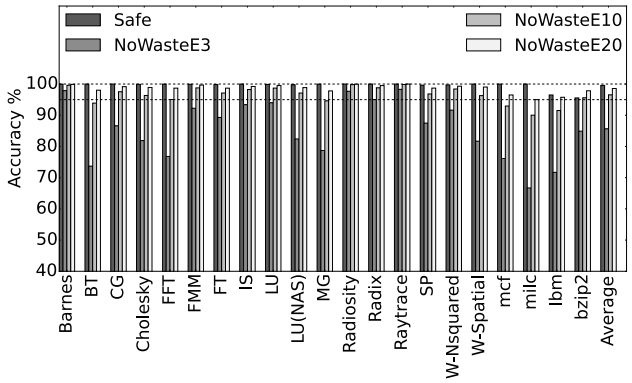


Fig. 7: Accuracy of the Snatch predictor algorithm.

of epochs when the predictor makes a wasteful prediction for at least 3, 10, and 20 epochs in a row. From the figure we see that, on average, such values are 85.6%, 96.5% and 98.5%, respectively. As we can see, the predictor rarely makes long wasteful predictions.

3) Power Increase: Figure 8 compares the *average* power consumed by the Baseline and Snatch configurations, broken down into processor and memory. The figure has bars for each application and for the average. Recall that the maximum power available is P_{PROC}+P_{MEM} = 10W.

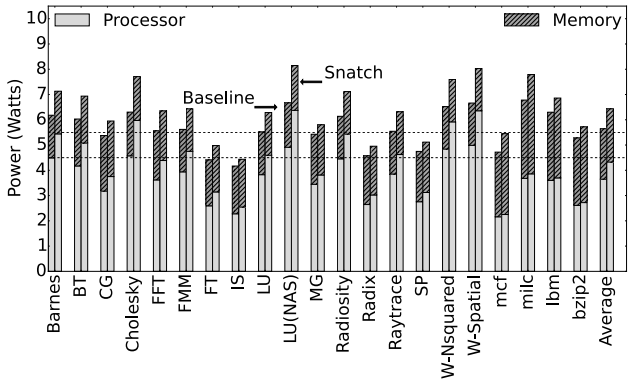


Fig. 8: Power consumed by Baseline and Snatch.

Going from Baseline to Snatch, we see that the average processor power and memory power consumption increase in all the applications. On average, the increase is 0.68W and 0.11W for processor and memory, respectively. Note that an increase in the power consumption is expected, since we are increasing the V_{dd} and f of the processor and memory. However, the increase in power is modest. We also note that the average power consumption is not a good indicator of the potential of power snatching in the application. The reason is that an application has a variety of phases, with different behaviors.

4) Temperature Increase: Table VI shows the temperature of different components in our architecture (shown in Figure 2(a)) for the Baseline and Snatch configurations. We run our thermally-worst application, namely Water-Spatial. We report the maximum temperature at the processor die, the lower memory die, the upper memory die, and at the package level.

	Baseline	Snatch	Difference
Processor Die	47 °C	52 °C	+5 °C
Memory Die 1	44.5 °C	48.5 °C	+4 °C
Memory Die 2	43 °C	46.6 °C	+3.6 °C
Package	41 °C	44.2 °C	+3.2 °C

TABLE VI: Maximum temperature at the different dies and at the package for Baseline and Snatch, running the Water-Spatial application.

The table shows that, at worst, we increase the temperature at the processor die by 5 °C. This is the die at the bottom of the stack, and hence furthest away from the heatsink. The tiny on-chip VR does not affect the peak temperature, since it is placed surrounding the TSV bus of the memory system at the center of the die, and close to the cooler L2 caches. Overall, the proposed on-chip VR is negligible in terms of thermals because of its small size, its low power consumption, and its location. Note also that the memory dies are well below the nominal refresh limit of 90 °C. At the package level, the temperature does not surpass 44.2 °C.

C. IR Drop

We evaluate the IR drop for the processor and memory dies under worst-case conditions. On the memory side, we show results only for the top die because it experiences the largest drop. Table VII summarizes the results. The largest IR drop is measured on the processor die due to the higher power consumption. The IR drop on the Baseline system is 63mV under worst-case power consumption. When the Baseline system is turbo-boosted without Snatch assistance (FGTurboBoost), the IR drop increases by 19% to 75mV due to the additional power consumed by the processor.

Processor Die			
	VR V_{dd} (V)	Lowest V_{dd} (V)	IR drop (mV)
Baseline	0.80	0.737	63
FGTB	0.95	0.875	75
Snatch	0.95	0.896	54
Top Memory Die			
	VR V_{dd} (V)	Lowest V_{dd} (V)	IR drop (mV)
Baseline	1.1	1.077	23
FGTB	1.1	1.063	37
Snatch	1.1	1.079	21

TABLE VII: IR drop measurements on the processor and memory dies for the Baseline, FGTurboBoost (FGTB), and Snatch systems. VR stands for off-chip voltage regulator.

The Snatch system provides additional power delivery capacity to support the extra power consumed. The IR drop in Snatch is only 54mV, even lower than in the baseline system despite the higher power consumption.

A similar behavior is observed in the memory die. The baseline IR drop is low at only about 23mV. Turbo-boosting

memory increases the IR drop substantially to 37mV. Snatch reduces the IR drop slightly below the baseline at only 21mV.

As we can see from the FGTurboBoost results, turbo-boosting adds stress on the power delivery network, increasing the IR drop. Snatch alleviates these effects enabling higher power consumption without increasing the stress on the PDNs.

D. Reliability

In this section, we compare the MTTF of the pins in two designs. One is Snatch, where the processor P/G pins are dimensioned for P_{PROC} but the processor can steal P_{SNATCH} from the memory, and the memory P/G pins are dimensioned for P_{MEM} but the memory can steal P_{SNATCH} from the processor. The second design is a conventional design, with separate PDNs for processor and memory. In this design, the processor and memory P/G pins are still dimensioned for P_{PROC} and P_{MEM}, respectively, but the processor ends-up consuming P_{PROC}+P_{SNATCH} when needed, and the memory ends-up consuming P_{MEM}+P_{SNATCH} when needed. As a result, in this second design that we call *NoSnatch*, the pins carry more current, and electromigration reduces their lifetime.

To derive the pin MTTF in the two designs, we use Equation 2. We use n = 1.8, Q = 0.8eV, c = 10, and ΔT=40°C [23], [46]. Our calculations show that, for the *NoSnatch* design, the MTTF per pin at maximum load and 87°C operation is ≈3.65 years. On the other hand, for the Snatch design, the MTTF increases to ≈6.84 years, as the current density is lower. To attain this same MTTF in a conventional design, we would need to increase the number of P/G pins by about 30% (Section VI-A); only then would the current density be similar to the Snatch design.

Figure 9 shows the cumulative distribution function for the TTF per pin in years for the two designs. The TTF per pin follows a log-normal distribution with μ =MTTF, and $\sigma = 0.5$ [46]. For each x value in years, the y coordinate in Figure 9 corresponds to the probability of the pin to fail at or before x. We observe that the Snatch design improves pin reliability notably, by keeping the current load in the P/G pins low.

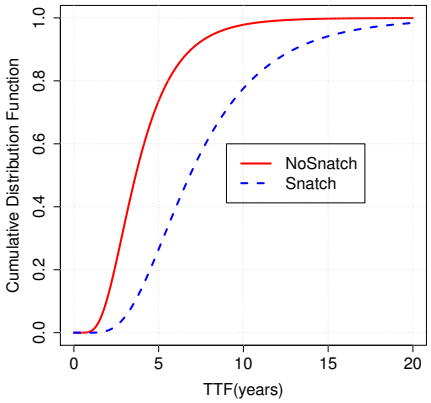


Fig. 9: Comparing the TTF per pin in the *NoSnatch* and Snatch designs.

VII. Conclusion

The pin count largely determines the cost of die packaging, which is often comparable to the cost of a die. In 3D processor-memory designs, P/G pins can account for the majority of the pins, hence significantly determining the package cost. To address this problem, this paper presented *Snatch* — a novel technique to reduce the number of P/G pins of both processor and memory PDNs, and dynamically and opportunistically divert some power between the two PDNs on demand. To perform the power transfer, Snatch uses a small bidirectional on-chip VR that connects the two PDNs, and works together with the two off-chip VRs to limit on-chip voltage transients. Snatch allows the computer to execute code sections with high power requirements without having to throttle performance.

We evaluated Snatch with simulations of a low-power 8-core multicore stacked with two memory dies. For fairness, we compared Snatch to an advanced turbo-boosting environment that uses as much power as the PDN allows but cannot snatch power from the other PDN. In a set of compute-intensive parallel codes, Snatch enabled the processor to take memory power for 30% of the time on average, speeding-up the codes by up to 23% over advanced turbo-boosting. In a set of more memory-intensive serial codes, the memory snatched processor power for 10% of the time on average. However, since turbo-boosting already sped-up the applications significantly, Snatch ended-up speeding-up the codes by only up to 2% over advanced turbo-boosting. Finally, Snatch can alternatively reduce the package cost by about 30%.

References

[1] K. Banerjee, S. Souri, P. Kapur, and K. Saraswat, "3-D ICs: A Novel Chip Design for Improving Deep-Submicrometer Interconnect Performance and Systems-on-Chip Integration," *Proceedings of the IEEE*, 2001.

[2] S. Borkar, "3D integration for energy efficient system design," in *IEEE Design Automation Conference*, Jun. 2011.

[3] B. Black, M. Annavaram, N. Brekelbaum, J. DeVale, L. Jiang, G. Loh, D. McCauley, P. Morrow, D. Nelson, D. Pantuso, P. Reed, J. Rupley, S. Shankar, J. Shen, and C. Webb, "Die Stacking (3D) Microarchitecture," in *IEEE International Symposium on Microarchitecture*, Dec. 2006.

[4] X. Dong, J. Zhao, and Y. Xie, "Fabrication Cost Analysis and Cost-Aware Design Space Exploration for 3-D ICs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Dec 2010.

[5] Intel Corp. http://www.intel.com/content/dam/www/public/us/en/documents/datasheets/processors-for-communications-infrastructure-datasheet-vol-1.pdf.

[6] M. Popovich, A. V. Mezhiba, and E. G. Friedman, *Power Distribution Networks with On-chip Decoupling Capacitors*. Springer, 2008.

[7] S. Chen, Y. Hu, Y. Zhang, L. Peng, J. Ardonne, S. Irving, and A. Srivastava, "Increasing off-chip bandwidth in multi-core processors with switchable pins," in *Proceedings of the 41st Annual International Symposium on Computer Architecture*, June 2014.

[8] I. Paul, W. Huang, M. Arora, and S. Yalamanchili, "Harmonia: Balancing Compute and Memory Power in High-performance GPUs," in *Proceedings of the 42nd Annual International Symposium on Computer Architecture*, 2015.

[9] W. Godycki, C. Torng, I. Bukreyev, A. Apsel, and C. Batten, "Enabling realistic fine-grain voltage scaling with reconfigurable power distribution networks," in *Proceedings of the 41st Annual IEEE/ACM International Symposium on Microarchitecture*, 2014.

[10] P. A. M. Bezerra, F. Krismer, T. M. Andersen, J. W. Kolar, A. Sridhar, T. Brunschwiler, T. Toifl, M. Jatlaoui, F. Voiron, Z. Pavlovic, N. Wang, N. Cordero, C. Rabot, and C. O. Mathuna, "Modeling and multi-objective optimization of 2.5D inductor-based Fully Integrated Voltage Regulators for microprocessor applications," in *2015 IEEE 13th Brazilian Power Electronics Conference and 1st Southern Power Electronics Conference (COBEP/SPEC)*, Nov 2015.

[11] E. A. Burton, G. Schrom, F. Paillet, J. Douglas, W. J. Lambert, K. Radhakrishnan, and M. J. Hill, "FIVR - Fully integrated voltage regulators on 4th generation Intel Core SoCs," in *Twenty-Ninth Annual IEEE Applied Power Electronics Conference and Exposition (APEC)*, March 2014.

[12] S. T. Kim, Y. C. Shih, K. Mazumdar, R. Jain, J. F. Ryan, C. Tokunaga, C. Augustine, J. P. Kulkarni, K. Ravichandran, J. W. Tschanz, M. M. Khellah, and V. De, "Enabling Wide Autonomous DVFS in a 22 nm Graphics Execution Core Using a Digitally Controlled Fully Integrated Voltage Regulator," *IEEE Journal of Solid-State Circuits*, vol. 51, no. 1, pp. 18–30, Jan 2016.

[13] N. Sturcken, E. J. O'Sullivan, N. Wang, P. Herget, B. C. Webb, L. T. Romankiw, M. Petracca, R. Davies, R. E. Fontana, G. M. Decad, I. Kymissis, A. V. Peterchev, L. P. Carloni, W. J. Gallagher, and K. L. Shepard, "A 2.5D Integrated Voltage Regulator Using Coupled-Magnetic-Core Inductors on Silicon Interposer," *IEEE Journal of Solid-State Circuits*, vol. 48, no. 1, pp. 244–254, Jan 2013.

[14] W. Kim, M. S. Gupta, G. Y. Wei, and D. Brooks, "System level analysis of fast, per-core DVFS using on-chip switching regulators," in *IEEE 14th International Symposium on High Performance Computer Architecture*, Feb 2008, pp. 123–134.

[15] M. Rodriguez, L. Corradini, C. Olalla, and D. Maksimovic, "Average current-mode control of boost converters with bidirectional power transfer capabilities," in *IEEE 13th Workshop on Control and Modeling for Power Electronics (COMPEL)*, June 2012, pp. 1–7.

[16] E. A. Burton, "Package and platform view of Intel's Fully Integrated Voltage Regulators (FIVR)," Tech. Rep., 2015, [Online] http://www.apec-conf.org/wp-content/uploads/IS-8.7-A-Package-and-Platform-View-of-Intels-Fully-Integrated-Coltage-Regulator.pdf.

[17] N. Kurd, M. Chowdhury, E. Burton, T. P. Thomas, C. Mozak, B. Boswell, P. Mosalikanti, M. Neidengard, A. Deval, A. Khanna, N. Chowdhury, R. Rajwar, T. M. Wilson, and R. Kumar, "Haswell: A Family of IA 22 nm Processors," *IEEE Journal of Solid-State Circuits*, vol. 50, no. 1, pp. 49–58, Jan 2015.

[18] *Coilcraft 0603HP Datasheet*, Coilcraft, Inc., [Online] Available at www.coilcraft.com/0603hp.cfm.

[19] P. Hazucha, G. Schrom, J. Hahn, B. A. Bloechel, P. Hack, G. E. Dermer, S. Narendra, D. Gardner, T. Karnik, V. De, and S. Borkar, "A 233-MHz 80%-87% efficient four-phase DC-DC converter utilizing air-core inductors on package," *IEEE Journal of Solid-State Circuits*, vol. 40, no. 4, pp. 838–845, April 2005.

[20] G. Schrom, P. Hazucha, F. Paillet, D. S. Gardner, S. T. Moon, and T. Karnik, "Optimal Design of Monolithic Integrated DC-DC Converters," in *IEEE International Conference on Integrated Circuit Design and Technology*, 2006, pp. 1–3.

[21] X. Huang, T. Yu, V. Sukharev, and S. X.-D. Tan, "Physics-based electromigration assessment for power grid networks," in *Design Automation Conference*, 2014, pp. 80:1–80:6.

[22] J. R. Black, "Electromigration: A brief survey and some recent results," *IEEE Transactions on Electron Devices*, vol. 16, no. 4, pp. 338–347, Apr 1969.

[23] W. J. Choi, E. C. C. Yeh, and K. N. Tu, "Mean-time-to-failure study of flip chip solder joints on Cu/Ni(V)/Al thin-film under-bump-metallization," *Journal of Applied Physics*, vol. 94, no. 9, pp. 5665–5671, 2003.

[24] R. Zhang, K. Wang, B. Meyer, M. Stan, and K. Skadron, "Architecture implications of pads as a scarce resource," in *International Symposium on Computer Architecture (ISCA)*, June 2014, pp. 373–384.

[25] Texas Instruments, "An-1205 electrical performance of packages," Tech. Rep., 2004. [Online]. Available: http://www.ti.com/lit/an/snoa405a/snoa405a.pdf

[26] K. Yao, "High-Frequency and High-Performance VRM Design for the Next Generations of Processors," Ph.D. dissertation, (http://theses.lib.vt.edu/theses/available/etd-04272004-215842/) Department of Electrical Engineering and Computer Science Virginia Polytechnic Institute and State University, 2004.

[27] J. Renau, B. Fraguela, J. Tuck, W. Liu, M. Prvulovic, L. Ceze, K. Strauss, S. Sarangi, P. Sack, and P. Montesinos, "SESC Simulator," January 2005, http://sesc.sourceforge.net.

[28] M. Halpern, Y. Zhu, and V. J. Reddi, "Mobile CPU's rise to power: Quantifying the impact of generational mobile CPU design trends on performance, energy, and user satisfaction," in *2016 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, March 2016.

[29] Anandtech. http://www.anandtech.com/show/8718/the-samsung-galaxy-note-4-exynos-review/6.

[30] W. El-Essawy and D. H. Albonesi, "Mitigating inductive noise in SMT processors," in *International Symposium on Low Power Electronics and Design (ISLPED)*, August 2004, pp. 332–337.

[31] M. S. Gupta, J. Oatley, R. Joseph, G.-Y. Wei, and D. Brooks, "Understanding voltage variations in chip multiprocessors using a distributed power-delivery network," in *Design Automation and Test in Europe (DATE)*, April 2007, pp. 624–629.

[32] J. Leng, Y. Zu, and V. Reddi, "GPU voltage noise: Characterization and hierarchical smoothing of spatial and temporal voltage noise interference in GPU architectures," in *International Symposium on High Performance Computer Architecture (HPCA)*, February 2015, pp. 161–173.

[33] J. Leng, Y. Zu, M. Rhu, M. Gupta, and V. J. Reddi, "GPUVolt: Modeling and characterizing voltage noise in GPU architectures," in *International Symposium on Low Power Electronics and Design (ISLPED)*. ACM, 2014, pp. 141–146.

[34] X. Zhang, T. Tong, S. Kanev, S. K. Lee, G.-Y. Wei, and D. Brooks, "Characterizing and evaluating voltage noise in multi-core near-threshold processors," in *Proceedings of the 2013 International Symposium on Low Power Electronics and Design*, 2013, pp. 82–87.

[35] D. Herrell and B. Beker, "Modeling of power distribution systems for high-performance microprocessors," *IEEE Transactions on Advanced Packaging*, vol. 22, no. 3, pp. 240–248, 1999.

[36] T.-H. Chen and C. Chen, "Efficient large-scale power grid analysis based on preconditioned Krylov-subspace iterative methods," in *Design Automation Conference*, 2001, pp. 559–562.

[37] P. Rosenfeld, E. Cooper-Balis, and B. Jacob, "DRAMSim2: A Cycle Accurate Memory System Simulator," *Computer Architecture Letters*, 2011.

[38] Wide I/O SDR Standard. (2011) http://www.jedec.org/standards-documents/results/jesd229.

[39] Wide I/O 2 Standard. (2014) http://www.jedec.org/standards-documents/results/jesd229-2.

[40] S. Li, J. H. Ahn, R. D. Strong, J. B. Brockman, D. M. Tullsen, and N. P. Jouppi, "McPAT: An Integrated Power, Area, and Timing Modeling Framework for Multicore and Manycore Architectures," in *International Symposium on Microarchitecture (MICRO)*, December 2009, pp. 469–480.

[41] G. Faust, R. Zhang, K. Skadron, M. Stan, and B. Meyer, "ArchFP: Rapid prototyping of pre-RTL floorplans," in *IEEE International Conference on VLSI and System-on-Chip*, 2012.

[42] J. Meng, K. Kawakami, and A. Coskun, "Optimizing energy efficiency of 3-D multicore systems with stacked DRAM under power and thermal constraints," in *IEEE Design Automation Conference*, 2012.

[43] S. C. Woo, M. Ohara, E. Torrie, J. P. Singh, and A. Gupta, "The SPLASH-2 programs: characterization and methodological considerations," in *International Symposium on Computer Architecture*, Jun. 1995.

[44] NAS Parallel Benchmarks. http://www.nas.nasa.gov/publications/npb.html.

[45] SPEC-CPU2006. https://www.spec.org/cpu2006/.

[46] R. Zhang, B. H. Meyer, K. Wang, M. R. Stan, and K. Skadron, "Tolerating the Consequences of Multiple EM-Induced C4 Bump Failures," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, no. 99, pp. 1–10, 2015.