Core vs. Uncore: The Heart of Darkness

Hsiang-Yun Cheng[†*], Jia Zhan[*], Jishen Zhao[‡],
Yuan Xie[*], Jack Sampson[†], Mary Jane Irwin[†]
[†]The Pennsylvania State University, {hoc5108,sampson,mji}@cse.psu.edu
[*]University of California Santa Barbara, {yuanxie,jzhan}@ece.ucsb.edu
[‡]University of California Santa Cruz, jishen.zhao@ucsc.edu

Invited Paper

ABSTRACT

Even though Moore's Law continues to provide increasing transistor counts, the rise of the utilization wall limits the number of transistors that can be powered on and results in a large region of dark silicon. Prior studies have proposed energy-efficient core designs to address the "dark silico" problem. Nevertheless, the research for addressing dark silicon challenges in uncore components, such as shared cache, on-chip interconnect, etc, that contribute significant on-chip power consumption is largely unexplored. In this paper, we first illustrate that the power consumption of uncore components cannot be ignored to meet the chip's power constraint. We then introduce techniques to design energy-efficient uncore components, including shared cache and on-chip interconnect. The design challenges and opportunities to exploit 3D techniques and non-volatile memory (NVM) in dark-silicon-aware architecture are also discussed.

1. INTRODUCTION

Recent trends in VLSI technology have led to a dark silicon era. Even though Moore's Law continues, Dennard scaling [7], which gave us near-constant chip power even with the doubling of transistors in each new process generation, has come to the end. Due to the breakdown of Dennard scaling, the fraction of silicon chip that can be operated at full frequency is dropping exponentially with each process generation to maintain a constant power envelope. Thus, large fractions of chips are effectively dark, i.e., idle, or dim silicon, i.e., under-clocked. To tackle this utilization wall [29] challenge, computer designers are looking for ways to stay on the performance curve by using multi-cores but without exceeding the chip's thermal design power (TDP).

Ensuring peak performance while staying within the power budget is clearly a very challenging task. In particular, in a multicore architecture with different types of components, such as cores, caches, on-chip network, memory controller, etc, there can be many different on-dim-dark configurations which allow one to stay within the power budget. These different configurations can exhibit significantly varying performance. Thus, an integrated approach is needed in which different components, including cores and uncores, must collaborate to maximize performance under power and thermal constraints as well as under dynamic changing program behavior and execution parameters.

There is a growing body of work on managing power budget both temporally and spatially in the dark silicon era, both at design time and at runtime. Most prior works on dark silicon [8, 25, 30, 31] are characterization studies and focus on cores. For example, a prior study [25] proposed the concept of "computational sprinting" to dynamically switch on all cores for maximum throughput and then shut down all but one core when the thermal limit is reached. Some other researchers proposed to use heterogeneous cores [30] and near-threshold circuits [31] to improve energy efficiency. Even though these hardware-based prior works target both high-end machines [8] and mobile architectures [25], they mainly focus on cores rather than uncore components.

While the research for addressing dark silicon challenges with uncore components is largely unexplored, the uncore components contribute significant power consumption and play important roles in joint performance/power/thermal optimization. In recent years, an increasing percentage of on-chip transistors are invested on the large last-level caches (LLCs) utilized to bridge the gap between fast CPU cores and slow off-chip memory accesses. Specifically, LLCs occupy as much as 50% of the chip area and consume higher than 20% of the chip's leakage power [6, 33]. Network-on-Chip (NoC) also has significant impact on the performance of many-core processors and consumes about 10% to 30% of total chip power [11, 19, 35]. Therefore, how to design the uncore components, especially shared caches and on-chip interconnect structures, is critical to tackle the challenges of multicore scaling in the dark silicon era.

Emerging technologies, such as three-dimensional integrated circuits (3D ICs) [3, 18] and non-volatile memories (NVMs) [2, 16, 24], also bring new challenges and opportunities to the design of dark-silicon-aware many-core systems. 3D integration is envisioned as a solution for future many-core design to mitigate the interconnect crisis and the "memory wall" problem [18]. In addition, the heterogeneous integration advantages of 3D ICs make it cost-efficient to integrate emerging non-volatile memories as on-chip cache or main memory to reduce the leakage power consumption. Nevertheless, architectural management techniques are required to tackle the increased power density in 3D die-stacking and the write energy/reliability issues in non-volatile memories.

In this paper, we first analyze the power consumption of cores and uncore components in multi-core processors, and illustrate that developing power management techniques for uncore components is critical to meet the chip's power budget. We then introduce some techniques for design-

DAC '15, June 07–11, 2015, San Francisco, CA, USA

DOI: http://dx.doi.org/10.1145/2744769.2647916.

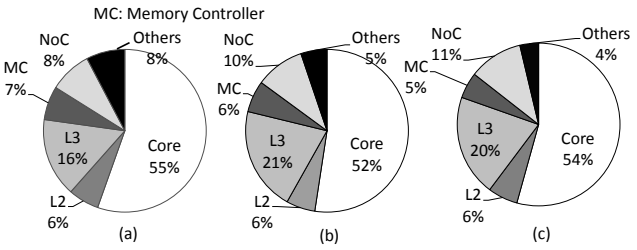


Figure 1: Power beakdown of a (a) 8-core system with 8MB LLC, (b) 16-core system with 16MB LLC, and (c) 32-core system with 32MB LLC, evaluated by McPAT.

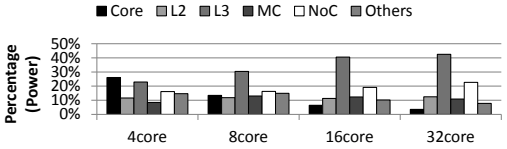


Figure 2: Chip power breakdown during nominal operation (single active core) in sprinting-based multicores, assuming idle cores are gated-off while uncore components stay active or idle.

ing energy-efficient uncore components, including LLCs and NoCs. We also discuss the design challenges and opportunities to exploit 3D techniques and NVM in dark-silicon-aware architecture. Our purpose is to draw researchers' attention on uncore-component designs that exploit different technologies and architectural techniques to improve the multicore performance scaling under the TDP constraint.

This paper presents recent research outcome of the NSF ASKS (Architecture Support for darK Silicon) project [1], with a summary of preliminary results [6, 34, 35]. The paper is part of the DAC special session on "Dark Silicon: No Way Out?". Other papers in this session are: "New Trends in Dark Silicon" [10], and "Approximate Computing and the Quest for Computing Efficiency" [28].

2. CORE VS. UNCORE

Even though most of the prior studies on dark silicon focus on power-efficient core design, the uncore components contribute significant on-chip power consumption. Figure 1 illustrates the power breakdown of multi-core systems when scaling the number of cores based on Niagara2 [21] processor with an additional shared L3 as the LLC. We evaluate the power dissipation with McPAT [17] for cores and uncore components, including L2/L3 caches, memory controllers (MCs), NoC, and others (PCIe controllers, etc). Results show that uncore components contribute about half of the chip power consumption, and a large portion of power is consumed by the LLC and NoC. In the 16-core and 32-core systems, LLC and NoC consume higher than 20% and 10% of on-chip power. Thus, designing power management techniques for uncore components, especially the shared LLC and NoC, is necessary to tackle the dark silicon challenge.

Figure 2 shows that the power consumption of uncore components become even more critical when *computational sprinting* [25] is applied on cores. Sprinting-based multi-cores activate a single core during nominal operation whereas the rest are turned off. Extra transistors are leveraged to support intense sprinting only when performance really counts, by using special phase change materials as a heat storage. We assume idle cores can be gated-off while other on-chip resources stay active or idle. As shown in

[1]project website:
`http://www.ece.ucsb.edu/~yuanxie/projects/ASKS`

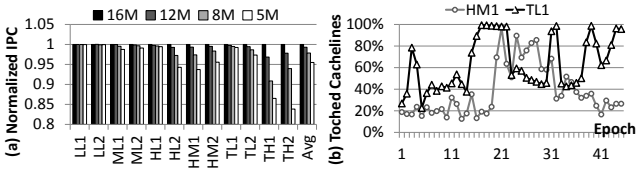


Figure 3: (a) Performance under different L3 sizes relative to 16MB L3 and (b) phase-dependent memory footprints of workloads composed of SPEC2006 [6].

Figure 2, increasing portion of on-chip power is consumed by uncore components, especially the LLC and NoC, as the number of idle cores increases. In contrast, the power ratio of the single active core keeps decreasing as the "dark silicon" grows. Therefore, in this scenario, it is inappropriate to only consider the power consumption of cores when power budget is the design limitation.

3. SHARED CACHE MANAGEMENT

In many-cores, the energy consumption of caches is increasing due to higher leakage in storage cells as a result of shrinking feature sizes, larger cache capacity, and lower supply voltages. Therefore, it is necessary to develop energy-efficient cache designs to stay within the many-core's power/thermal budget.

The performance improvement from increasing cache capacity varies significantly for different applications. While some applications use as much cache capacity as they are given, others need only a small amount and do not benefit from a larger capacity. Figure 3 illustrates that the fraction of required cache varies over time within an application as well as among applications. When the required cache size is smaller, some parts of the LLC can be disabled to reduce leakage power. In Figure 3 (a), for example, if a 5% performance degradation is acceptable, we can shutdown more than half of the LLC to save power in all but two workloads.

Similarly, when some of the cores are powered down, applications running on the remaining cores receive an increase in their effective shared cache capacity. However, these applications may not be able to utilize this additional cache capacity effectively. If so, instead of using more cache space to improve performance, it might be better (i.e., more energy efficient) to turn off some parts of the shared cache and save energy without impacting performance.

3.1 Techniques and Granularity

Prior studies have proposed circuit-level techniques, including drowsy cache [9] and the gated-Vdd approach [23], to reduce the leakage power of on-chip caches. Drowsy circuits, however need two supply voltages for each cache line, which incurs design overheads. Thus, in this paper, we focus on using the gated-Vdd technique to power-off portions of the LLC. Based on the circuit-level techniques, several architectural approaches have been proposed to power-off portions of the LLC at different granularities [12, 13, 20]. Some techniques [12, 20] attempt to disable useless cache ways. Others [13] managed the LLC at finer granularity and disabled cache lines that are not likely to be reused. These way-based or cache-line-based approaches incur large area overhead to shutdown cache due to additional wire routing in each subarray. Therefore, in this paper, we focus on slice-based cache organization [6] that forces a subset of ways into a single subarray and shutdowns several ways, i.e., a single slice, at the same time to reduce the circuit overhead.

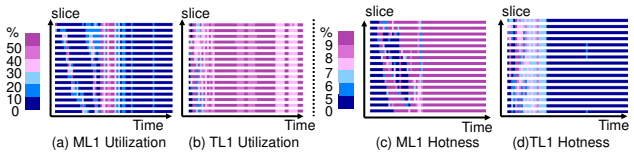


Figure 4: Utilization and hotness of ML1(with small active footprint but frequently reused data) and TL1(with large active footprint but rarely reused data) [6].

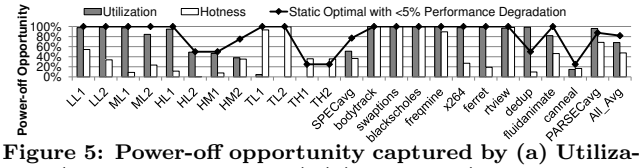


Figure 5: Power-off opportunity captured by (a) Utilization (utilization<0.3 slices) (b) Hotness (Hotness<0.075 slices) and (c) Static Optimal (statically select minimum cache size that incurs <5% performance degradation) [6].

3.2 Metrics of Interest

In order to save energy by turning off cache slices, we need to dynamically capture the cache size requirements of different workloads. There are three main factors that can be utilized to make turn-on/turn-off decisions: utilization, hotness, and the distribution of dirty cache lines.

3.2.1 Utilization

Ideally, the cache capacity should be large enough to fit the active cache footprint of workloads. Figure 4(a) and (b) show the utilization, i.e., the percentage of cache lines that are referenced in a time epoch in each slice, of two types of workloads. As shown in the figure, the utilization of ML1 is low, while TL1 has high utilization. In addition, the utilization varies across different time epochs and cache slices. Low-utilization slices represent potential power-off opportunities, as only few additional cache misses would be incurred when powering off these slices. Utilization alone can capture the power-off opportunity of most of the workloads, as illustrated in Figure 5. Nevertheless, if the data in the slices are seldom reused, such as in TL1, TL2, TH1, and TH2, it misses some power-off opportunities. This observation motivates us to consider additional metrics.

3.2.2 Hotness

In addition to the utilization, the access frequency also helps to capture the power saving opportunity. Powering off a frequently accessed slice would incur more cache misses than shutting down a seldom reused slice. We define the hotness of a slice as the number of hits to the slice divided by the total number of LLC misses in a time epoch. Thus, the hotness implies the increase in the cache miss rate if the slice is powered off. Figure 4(c) and (d) illustrate the hotness of two type of workloads. At ML1, only few cache lines are accessed, but these referenced data are highly reused. Shutting down only cold slices for ML1 would lose considerable power-off opportunities provided by the small active footprint. On the other hand, TL1 has large active footprint, but the referenced data are seldom reused. Thus, powering off cold slices for TL1 may save more power than shutting down low-utilization slices. Figure 5 shows that the hotness of slices can better capture the power-off opportunity than utilization for workloads with large but seldom reused cache footprint, such as TL1, TL2, TH1, and TH2.

3.2.3 Writeback of Dirty Data

When a slice in the LLC is powered off, the dirty data need to be written back to the main memory. Since the slice can only be powered doen once all the dirty data have been written back, shutting down a slice with a higher number of dirty cache lines would reduce power savings. Moreover, these write-backs of dirty data would frequently fill up the write-buffer in the memory controller and could delay critical reads. Therefore, when deciding which slice should be turned off, a slice with less dirty data should be chosen among the slices with the same level of utilization or hotness.

In summary, the spatial acccess behavior can be represented by cache utilization, while cache hotness indicates temporal access behavior. Low-utilization slices can be turned off to capture most of the power saving opportunities. To further save the leakage power, cold slices can be powered off when the stored data are seldom reused. Also, the number of dirty lines should play an important role when choosing which slices to power off. The discussion above suggests that an ideal LLC turn-off/turn-on strategy should consider all these metrics.

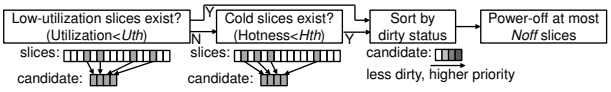


Figure 6: Flow chart of the power-off policy [6].

3.3 Power Management Policies

Based on the above mentioned metrics of interest, we can design power-off, power-on, and data migration policies accordingly. We can use set-sampling to monitor the utilization, hotness, and dirty status of cache slices with small hardware overhead. By analyzing these workload behavior, the cache controller dynamically determines whether to power-on or power-off slices at every time epoch.

3.3.1 Power-off Policy

Considering all the three metrics of interest, we design a power-off policy to save as much power as possible without significantly degrading system performance, as illustrated in Figure 6. We first select the slices with utilization less than a threshold (*Uth*) as the power-off candidates. If there is no low-utilization slice, we instead select the slices with less than *Hth* hotness, as the hotness characteristic can help to identify seldom reused slices. Among the candidates, at most *Noff* slices with fewer dirty cache lines are chosen to be powered off. The threshold settings can be determined empirically based on the system configuration.

3.3.2 Power-on Policy

After some slices are turned off to save power, the cache misses may increase due to the smaller LLC size. To avoid significant performance degradation, we need to determine whether the workload would benefit from a larger LLC at each time epoch, as the cache access behavior changes in different program phases. We keep the whole tag array powered-on for monitoring the potential hits to the dark portion of the LLC. The tag arrays of the dark slices are called victim tags, and store the evicted cache lines from the active slices. During each hit to the victim tags, a potential hit counter is increased by one. At the end of each time epoch, if the hit rate to the victim tags is higher than a threshold, *Noff* slices are turned on to improve performance.

3.3.3 Data Migration Policy

Before turning off the victim slices to save power, the dirty and clean blocks should be written back/discarded, in order to guarantee data coherency. The loss of data in the powered-off slices may incur additional miss penalty when these data are reused. One possible solution is to migrate

useful data to other active slices. During each migration, a replacement victim is selected from the active slices. If the evicted replacement victim is reused later, the migration of a cache line may incur an additional conflict miss. Thus, we choose to migrate only the clean blocks in hot-clean slices, and dirty blocks in hot-dirty slices.

3.4 Discussion

In this section, we have introduced an architectural approach to shutdown portions of the LLC to save leakage power. The technique is useful when the capacity demand to the LLC is smaller, as the running workloads are compute-intensive or some running cores are power-gated. By saving the leakage power from the LLC, the system would become more power-efficient, or the saved power can be utilized to power-on darkened cores for performance improvement. Based on this power management technique, we can further develop methods to coordinately manage the power state of each individual core and different uncore components to maximize system performance under the TDP constraint.

4. NOC MANAGEMENT

One key research question about NoC management is how to best support power-efficient sprinting. In this section, we analyze the effectiveness of computational sprinting using a variety of workloads with representative communication characteristics.

4.1 Fine-Grained Computational Sprinting

A naive method to perform computational sprinting is to transiently activate the dark cores all at once. But doing so fails to exploit the sporadic workload parallelism. As a result, this scheme can significantly increase power dissipation yet does not reasonably improve system performance. This is especially true for multithreaded applications with various scales. As an example, we evaluate the performance of PARSEC 2.1 [4] benchmarks with various core counts using gem5 [5]. Note that Figure 7 only shows a set of selected results that can represent various workload characteristics.

Figure 7 shows that the naive method can speedup *blackscholes* and *bodytrack* significantly, when we increase the number of cores. Performance of *freqmine* stays quite stable regardless of the change of core count. A couple of other benchmarks such as *vips* and *swaptions* illustrate interesting results, where system performance seems to suffer when we increase the number of cores. The rationale behind is that further adding cores beyond application parallelism can incur substantial performance overheads, including thread scheduling, synchronization, and long interconnect delay due to the spread of computation resources.

To better serve the sporadic workload characteristics, which will quickly react to intense computation and determine the optimal number of cores that should be offered for instantaneous responsiveness, we design a fine-grained computational sprinting method. We explore how to best design on-chip network under such circumstances[2].

4.2 Irregular Topological Sprinting

Nominal operation only allows a single core (namely *master core*) to remain active. We may place the master core at various locations. Without loss of generality, we choose the top-left corner node (Node 0 in Figure 8a) as the master node, because the location is the closest to the memory controller.

[2]We assume that these application parallelism can be learnt in advance or monitored during run-time execution.

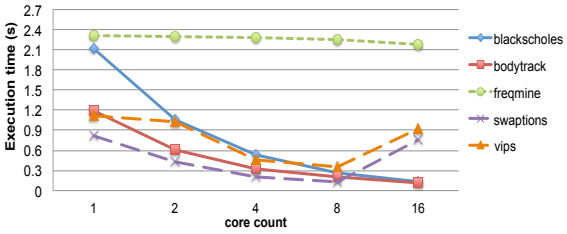
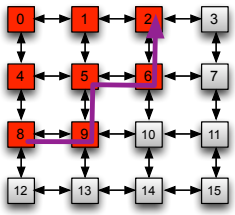
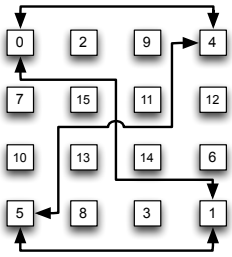


Figure 7: Performance of PARSEC 2.1 benchmarks when we increase the number of cores from one to sixteen [35].



(a) Logical connection of a 16-node mesh network. The irregular topology and convex DOR routing.



(b) Physical allocation for the original network in (a). Only links for 4 nodes are shown for clarity.

Figure 8: Topology, routing, and floorplan for fine-grained sprinting [35].

A number of cores will be activated and keep running for a short duration after the system transfers to the sprinting mode. We propose to start from the master node, and connect other nodes to the network in ascending order of their Euclidean distances to the master node. For instance, the red nodes (Figure 8a) demonstrate the topology of a 8-core sprinting. Here we use Euclidean distances instead of Hamming distances. While Hamming distances can incur a shortest routing distance between the newly-added node to the master node, they will generate longer inter-node communication to other nodes as well. For example, both cases could choose node 0, 1, and 4 as 3-core sprinting; if 4-core sprinting is triggered, the method with Hamming distance may choose node 2 – less optimal as node 5, which the Euclidean distance is likely to generate. Algorithm 1 illustrates how to generate an order of N nodes used for topological sprinting.

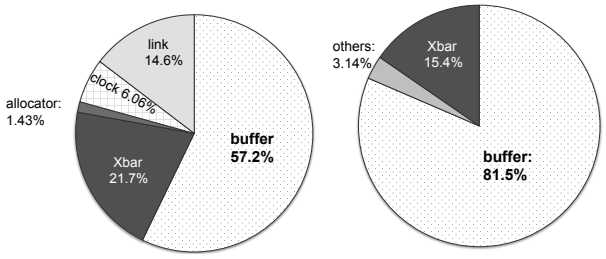
Algorithm 1: Irregular Topological Sprinting

Result: A linked-list L of routers to be activated
Initialize: $D[i] = 0, i = 0, 1, 2...N-1$. The coordinate for R_k is (x_k, y_k).
for $i \leftarrow 1$ **to** $N-1$ **do**
  $D[i] = \sqrt{(x_i - x_0)^2 + (y_i - y_0)^2}$;
end
Sort $R[i](i = 0, 1...N-1)$ in ascending order of $D[i](i = 0, 1...N-1)$ and put them into a linked-list L. Break ties according to the order of indexes.

Moreover, we slightly modify the conventional dimension-order routing in order to detect the dark nodes and re-route. Specifically, we leverage two connectivity bits to indicate whether a router is connected to its western or eastern neighbors. The NoC will then route messages from the current router to the destination router, according to the offsets of coordinates and the two connectivity bits per router.

4.3 Thermal-Aware Floorplanning

One key design constraint of *fine-grained sprinting* is the thermal design power (TDP). To offer better thermal distribution to avoid hot spots, we propose a design-time floor-



(a) Dynamic power **(b)** Leakage power

Figure 9: Dynamic and leakage power breakdown of a virtual-channel based router [34].

planning algorithm that can be seamlessly integrated with the topological sprinting process.

Figure 8a demonstrates our design with a 4-core sprinting in a 16-node mesh network. We may opt to choose the top-left four nodes to achieve better performance, but alternatively can prefer the four scattered corner nodes to minimize thermal impact. To address this dilemma, *we maintain the original logic connectivity of the mesh network in consistent of the topological sprinting process, and propose a heuristic algorithm to reallocate the physical location of each node.*. Our floorplanning algorithm interprets the 2D mesh network as a graph, and allocates the nodes iteratively based on the list generated from Algorithm 1. At each iteration, it chooses an unallocated node from the list, and maps the node to another node with the maximum weighted sum of Euclidean distances to all allocated nodes.

With such floorplanning algorithm, we effectively obviate the thermal issue of the sprinting process and routing algorithm. Only logical connectivity of mesh network remain to be considered during topological sprinting. Figure 8b shows the final floorplan of the physical network and only links for four-core sprinting. Note that the floorplanning algorithm will increase the wiring complexity and generate long links. A standard method of reducing delay of long wires is to insert repeaters in the wire at regular intervals. Recently, Krishna *et al.* [15] have validated such clockless repeated wires that allow multi-hop traversals to be completed in a single clock cycle.

4.4 Network Power Gating

Network power gating scheme can be easily built on top of our fine-grained sprinting method. *Because the topological sprinting algorithm activates a subset of routers and links to connect the active cores, we turn off other network components as shown in the shaded nodes of Figure 8a* .Moreover, our routing algorithm routes packets within the active network and thus avoids unnecessary wakeup of intermediate routers for packet forwarding. Doing so further increases the idle period of the dark region for longer power gating.

4.5 STT-RAM Based Buffer Design

To improve system power efficiency, we redesign the internal NoC components. We simulate a classic wormhole router with DSENT [27]. Figure 9a and Figure 9b show the breakdown of dynamic and leakage power consumption in various router components (in 32nm technology with a supply voltage of 0.9v). An important observation is that buffer power (especially leakage power) contributes to a significant portion of router power.

In order to reduce the NoC leakage power dissipation, we propose an NVM-based router buffer design. NVM technologies (PCM, STT-RAM, ReRAM) promise much lower leakage power than conventional SRAM. In particular, STT-RAM combines the benign effects including near-SRAM performance, near-DRAM density, non-volatility, and low leakage power. In addition, STT-RAM promises the best endurance characteristic among a variety of NVM technologies, making it an attractive candidate for implementing on-chip buffer which typically need to tolerate frequent access.

High write latency and high write energy used to be an issue of various NVM technologies. But recent designs have demonstrated that STT-RAM can have much lower write latency of 2 - 4 ns [14, 22], i.e., 2 -4 cycles at 1 GHz clock frequency. Some previous STT-RAM technologies may incur higher write latency and write energy than SRAM. In this case, we can mitigate the write overhead by relaxing the non-volatility of STT-RAM [26].

5. OPPORTUNITIES IN 3D AND NVM

We have illustrated that traditional SRAM caches and NoCs need to be redesigned to tackle the dark silicon challenge. In this section, we show that emerging 3D die-stacking technology and non-volatile memories also bring new challenges and opportunities to the design of dark-silicon-aware many-core systems.

5.1 3D Die-stacking Technology

The integration of multiple cores on a single die is expected to accentuate the already daunting memory-bandwidth problem. Supplying enough data to a many-core chip will become a major challenge for performance scalability. Three-dimensional integrated circuits (3D ICs) [3, 18] are attractive options for overcoming the barriers in interconnect scaling, thereby offering an opportunity to continue performance improvements using CMOS technology. The manufacturing/process techniques for 3D integrations are nearly mature. Consequently, 3D integration is envisioned as a solution for future many-core design to tackle the memory wall problem, and it provides new opportunities for future many-core designs with its heterogeneous integration capability. With 3D die-stacking, cores and uncore components manufactured in different technologies can be integrated into a single package to reduce wire lengths, mitigate the pin count problem, and improve performance.

5.2 Emerging Non-volatile Memories

In conventional computer architecture design, SRAM and DRAM are the common memory embodiments at different levels in the memory hierarchy. As technology scales, increasing leakage power dissipation and significant degradation of the reliability of SRAM and DRAM are of increasing concern. In recent years, we have seen a lot of effort to address the research and development of some emerging non-volatile memory (NVM) technologies, e.g., Phase-Change RAM (PCRAM) [16, 24] and Spin-Transfer Torque RAM (STT-RAM) [2]. By combining the speed of SRAM, the density of DRAM, the non-volatility of Flash memory, and low leakage power, these emerging NVM memory technologies have a great potential to be the universal memories of the future. It is anticipated that the emerging NVM technologies will break important ground and move closer to the market in the near future.

5.3 Design Challenges

With 3D die-stacking, the thermal profile of the package becomes even more challenging due to the increased power density. Furthermore, when eDRAM or DRAM is stacked on top of cores as cache or main memory, the heat generated by

the core-layer can significantly aggravate the refresh power of eDRAM/DRAM layers. Thus, the power consumption due to refresh needs to be considered when designing the power management policy for stacked DRAM memory or eDRAM cache.

Non-volatile memories can provide larger capacity with lower leakage consumption when being employed in on-chip caches or memory. However, the write energy of NVMs is much higher than in traditional SRAM and DRAM. Thus, management policies, such as data placement, replacement policy, etc, need to be redesigned for NVM-based memory systems. Prior studies have proposed data placement and bypass policies to avoid frequently writing data to NVM [1, 32]. These management policies may need to be redesigned when portions of the cores or uncore components are shutdown to meet the power constraint.

5.4 Opportunities

Stacking caches in 3D offers an opportunity to provide and exploit cache heterogeneity in two dimensions at the same time, namely heterogeneity in technology and heterogeneity in organization. With 3D stacking, multiple technologies can tightly integrate in a single design. Even in the same technology, 3D stacking allows us to pursue heterogeneity in cache organization, with different block sizes, associativity, prefetching mechanisms, etc. for each cache layer. Different cache layers meet different workload demands, and during nominal operation, idle cache layers can be shutdown to stay within the TDP. New 3D NoC design is required to flexibly connect the desired amount of resource and provide efficient communication for high performance.

The non-volatility and low leakage power of NVMs make it become promising to replace traditional SRAM and DRAM in uncore components. The non-volatile feature of NVMs can enable fast instant-on/off for on-chip cache/memory, as the data stored in the powered off portions remains in the cache. The low leakage characteristic of NVMs can not only help to reduce the power consumption of caches and memory, but also reduce the power consumption of NoCs, as illustrated in Section 4.

6. CONCLUSION

This paper illustrates that uncore components contribute significant power consumption in many-core systems. Thus, power management policies are required for these uncore components, especially last-level caches and NoCs, to maximize system performance under the TDP constraint. We introduce techniques to design energy-efficient LLCs and NoCs, and discuss the design challenges and opportunities to exploit 3D die-stacking and NVMs in dark-silicon-aware many-core systems. Our illustration and discussion show that there are many opportunities to explore when designing uncore components in dark silicon era.

7. ACKNOWLEDGMENTS

This research is funded by NSF grants 1500848, 1461698, 1213052, 1017277, and Department of Energy under Award Number DE-SC0005026. Detail information about this project can be found at `http://www.ece.ucsb.edu/~yuanxie/projects/ASKS/`.

8. REFERENCES

[1] J. Ahn et al. DASCA: Dead write prediction assisted STT-RAM cache architecture. In *HPCA*, 2014.

[2] D. Apalkov et al. Spin-transfer torque magnetic random access memory (STT-MRAM). *J. Emerg. Technol. Comput. Syst.*, 9(2):13:1–13:35, May 2013.

[3] K. Bernstein et al. Interconnects in the third dimension: Design challenges for 3D ICs. In *DAC*, 2007.

[4] C. Bienia. *Benchmarking Modern Multiprocessors*. PhD thesis, Princeton University, 2011.

[5] N. Binkert et al. The gem5 simulator. *ACM SIGARCH Computer Architecture News*, 39(2):1–7, 2011.

[6] H.-Y. Cheng et al. EECache: Exploiting design choices in energy-efficient last-level caches for chip multiprocessors. In *ISLPED*, 2014.

[7] R. Dennard et al. Design of ion-implanted MOSFET's with very small physical dimensions. *Solid-State Circuits, IEEE Journal of*, 9(5):256–268, Oct 1974.

[8] H. Esmaeilzadeh et al. Dark silicon and the end of multicore scaling. *SIGARCH Comput. Archit. News*, 39(3):365–376, June 2011.

[9] K. Flautner et al. Drowsy caches: simple techniques for reducing leakage power. In *ISCA*, 2002.

[10] J. Henkel et al. New trends in dark silicon. In *DAC*, 2015.

[11] Y. Hoskote et al. A 5-GHz mesh interconnect for a teraflops processor. *Micro, IEEE*, 27(5):51–61, Sept 2007.

[12] D. Kadjo et al. Power gating with block migration in chip-multiprocessor last-level caches. In *ICCD*, 2013.

[13] S. Kaxiras et al. Cache decay: exploiting generational behavior to reduce cache leakage power. In *ISCA*, 2001.

[14] E. Kitagawa et al. Impact of ultra low power and fast write operation of advanced perpendicular MTJ on power reduction for high-performance mobile CPU. In *IEDM*, 2012.

[15] T. Krishna et al. Single-Cycle Multihop Asynchronous Repeated Traversal: A SMART Future for Reconfigurable On-Chip Networks. *Computer*, 46(40):48–55, 2013.

[16] B. C. Lee et al. Architecting phase change memory as a scalable dram alternative. *SIGARCH Comput. Archit. News*, 37(3):2–13, June 2009.

[17] S. Li et al. McPAT: An integrated power, area, and timing modeling framework for multicore and manycore architectures. In *MICRO*, 2009.

[18] G. Loh. 3D-stacked memory architectures for multi-core processors. In *ISCA*, 2008.

[19] T. Mattson et al. The 48-core SCC processor: the programmer's view. In *SC*, 2010.

[20] S. Mittal et al. FlexiWay: A cache energy saving technique using fine-grained cache reconfiguration. In *ICCD*, 2013.

[21] U. Nawathe et al. Implementation of an 8-core, 64-thread, power-efficient sparc server on a chip. *Solid-State Circuits, IEEE Journal of*, 43(1):6–20, Jan 2008.

[22] T. Ohsawa et al. A 1.5 nsec/2.1 nsec random read/write cycle 1Mb STT-RAM using 6T2MTJ cell with background write for nonvolatile e-memories. In *VLSIT*, 2013.

[23] M. Powell et al. Gated-Vdd: a circuit technique to reduce leakage in deep-submicron cache memories. In *ISLPED*, 2000.

[24] M. K. Qureshi et al. Scalable high performance main memory system using phase-change memory technology. *SIGARCH Comput. Archit. News*, 37(3):24–33, June 2009.

[25] A. Raghavan et al. Computational sprinting. In *HPCA*, 2012.

[26] C. W. Smullen et al. Relaxing non-volatility for fast and energy-efficient STT-RAM caches. In *HPCA*, 2011.

[27] C. Sun et al. DSENT - a tool connecting emerging photonics with electronics for opto-electronic networks-on-chip modeling. In *NoCS*, 2012.

[28] S. Venkataramani et al. Approximate computing and the quest for computing efficiency. In *DAC*, 2015.

[29] G. Venkatesh et al. Conservation Cores: Reducing the energy of mature computations. *SIGARCH Comput. Archit. News*, 38(1):205–218, Mar. 2010.

[30] G. Venkatesh et al. QsCores: Trading dark silicon for scalable energy efficiency with quasi-specific cores. In *MICRO*, 2011.

[31] L. Wang and K. Skadron. Implications of the power wall: Dim cores and reconfigurable logic. *Micro, IEEE*, 33(5):40–48, Sept 2013.

[32] Z. Wang et al. Adaptive placement and migration policy for an STT-RAM-based hybrid cache. In *HPCA*, 2014.

[33] D. Wendel et al. The implementation of POWER7TM: A highly parallel and scalable multi-core high-end server processor. In *ISSCC*, 2010.

[34] J. Zhan, J. Ouyang, F. Ge, J. Zhao, and Y. Xie. DimNoC: A dim silicon approach towards power-efficient on-chip network. In *DAC*, 2015.

[35] J. Zhan, Y. Xie, and G. Sun. NoC-Sprinting: Interconnect for fine-grained sprinting in the dark silicon era. In *DAC*, 2014.