To Stack or Not To Stack

Richard Afoakwa, Lejie Lu, Hui Wu, Michael Huang
Department of Electrical and Computer Engineering
University of Rochester
Rochester, New York, USA
{richard.afoakwa, lejie.lu, hui.wu, michael.huang}@rochester.edu

***Abstract*— 3D memory technology, such as Micron's hybrid memory cube (HMC), has re-energized the architectural pursuit of computation very close to, or inside the memory chip. Such a design falls into the broader category of near-data processing (NDP). The motivation for such design is because the current Von Neumann architecture of chip-multiprocessors is thought to make data movement expensive. Current NDP work focuses on the possibility of architecting computation engines, such as accelerators, cores, or graphic processing units right below the memory layers and inside the logic layer of the HMC subsystem. However, such a stacking design does present a number of technical challenges such as heat dissipation, power supply, etc. While these challenges can certainly be overcome, and needs to be addressed, in this work, we seek to answer a related question of whether it is necessary to stack general-purpose computation engines, directly inside the memory unit, in order to achieve the performance potential of NDP system; thus, to stack or not to stack. We show that, with computing models used in current NDP designs, placing the computation engines very close to, but outside the memory system (not stacking) can provide comparable performance without significant energy costs. This can be achieved without inventing any new technology, but utilizing current state-of-the-art high-speed link design practices.**

***Index Terms*—Near-Data or In-Memory Processing, Silicon Interposer, High-Speed Links**

I. Introduction

The latency disparity between instruction execution and main memory access persisted over decades. When the memory access pattern defies conventional caching, the latency disparity becomes a significant challenge for performance improvement. It has long been hoped that some processing happens inside the memory chip, giving rise to terms such as processing-in-memory (PIM) or intelligent RAM (IRAM). Intuitively, by doing computation inside the DRAM chip, we not only ameliorate the access latency problem but also enjoy the much higher bandwidth at the DRAM bank level which is otherwise thrown away when crossing the chip boundary.

Unfortunately, the idea has not seen commercial realization. Though there are legitimate technical concerns such as heat dissipation issues, it was thought that market realities are the major causes that prevented the realization: ⓐ DRAM fabrication process does not lend itself to efficient inclusion of large-scale logic; ⓑ the system's success depends on new programming models and close collaboration among the designers of memory, processor, and software; ⓒ the commodity nature of the DRAM business ultimately discourages such expensive, non-standard products.

Today, we are seeing renewed interests of a similar concept called near-data processing (NDP), thanks to the technology of 3D-stacked memory such as Hybrid memory cube (HMC) [1] or High Bandwidth Memory (HBM) [2]. Recent studies have reported performance improvement in excess of 16x [3,4]. With diminishing returns from process technology and the looming end of Moore's law, such significant performance gains are clearly worth the attention of researchers and practitioners alike. To be sure, the 3D stacked memory technology indeed makes it easier to include high-performance processing elements directly in the same chip as the main memory, right under the memory layers. But some of the challenges facing PIM systems still exist today for NDP designs. Heat dissipation, for instance, perhaps becomes worse in the presence of 3D stacking. The question becomes: Should we embrace these design proposals now and set out to address all those technical challenges? Or can alternative designs, without 3D stacking, achieve similar benefits?

In particular, some works have compared conventional uniprocessor system with an NDP-style system, where NDP utilizes a large number of cores performing parallel processing right inside the memory cube [5,6]. How much benefit is really attributable to bringing processors into memory (and not simply due to using a large number of cores)? We thus want to perform a more direct control experiment that isolates the single factor that defines these NDP systems: stacking of processor cores under the memory layers. In other words, if we maintain the same processing configuration (many simple processing engines) and vary their location from being tightly integrated with the memory chip to being more loosely attached via some fabric, how much impact on performance and energy is there? In short: to stack or not to stack, that is the question.

Fig. 1 provides a high-level schematic of the systems under comparison. On the left, we show a typical implementation of an NDP system where a layer of logic die under the DRAM layer contains an array of NDP cores. On the right, we show a control configuration where those same cores are off the memory chip and placed next to the memory, interconnected with the memory via dedicated medium. It is the performance of these medium that separate the fully-stacked (3D) NDP system from the non-stacked (2.5D) system. The latter can access memory with an entirely different latency, bandwidth, and energy profile. Our goal is to study the impact such a difference can make on the ultimate performance of represen-

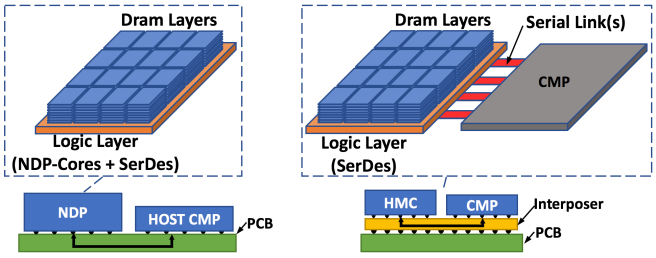


Fig. 1. Comparison between the primary system configurations. Fully stacked 3D NDP (left) and 2.5D interconnected with dedicated fabric such as interposer (right).

tative workloads.

Importantly, our goal is NOT to support one particular design point or another. Instead, we hope to help the process of articulating the vision of NDP and identifying key focus areas for future research in realizing the vision.

II. Related Work

As discussed earlier, Fig. 1 provides a high-level view of our systems under test. From this, we review some previous work on the three major system components to draw more insight into this work.

A. Near-Data or In-Memory Processing

Processing-in-memory has long been a subject of investigation for the architecture community [7]–[11]. Recent industry developments make such an architecture more feasible. Consequently, there has been a renewed interest to investigate processing in, or close to, memory. The topics of investigation include many different aspects such as system-level analysis of general-purpose processing [3,4,12]–[15]; configuration design space [16]–[18]; interconnection [19,20]; functional units [21,22]; GPU-style or other special-purpose processing [6,23]–[30]; software or runtime scheduling or coordination [31]–[38]. While new analysis and designs will continue to emerge, the general theme has been to get processing elements near the data storage so as to reduce both access latency and energy overhead. While some prior work tried such a separation on a narrower scale [39], in this paper, we focus on general-purpose processing and try to address the single question of whether "near" dictates physical stacking.

B. Silicon Carrier Systems

On the system level, there has been some work interconnecting multiple chips with silicon carriers (aka. interposer). In most of such work, the interposer substrate contains active components such as transistors for routing and networking among multiple processor chips. Given the interposer, the design space for packet-switching interconnects can be extended [40]–[43]. Indeed, the interposer system design can impact exa-scale systems, where diverse memory and processing components coexist on a singular platform [44,45]. Interposer also provides the flexibility to interconnect computing chips together with 3D memory chips [46].

C. Serial Links

The technology of serial links and in particular those embedded in interposer has seen considerable work. A silicon carrier test chip has been fabricated specifically for chip-to-chip communication, showing the capability of driving 2 cm links at 11.5 Gb/s *without equalization* [47]. Current-mode transceivers have been proposed to overcome RC-limited bandwidth of interposer channel [48]. The technique is an alternative to aggressive equalization required to compensate for high channel losses. Other works show significantly improved link speeds. For example, a 16.8 Gb/s single-ended transceiver presents a new low-power driver as well as source follower-type continuous time linear equalizer (CTLE) to compensate for channel loss [49]. Finally, a 20 Gb/s parallel interface design uses single-sided signaling, capacitive termination in the receiver side, and passive equalization in the transmitter to drastically reduce power consumption, and obtain 0.3 pJ/b energy efficiencies [50].

In short, state-of-the-art serial links can achieve 10s of Gb/s transmission per link at sub-pJ/b energy cost, at current technology nodes. Such links can be embedded into silicon carriers to provide dense 2.5D chip-to-chip interconnection.

III. Architectural Design Space

In theory, when architecting an NDP system, there is a non-trivial design space with regard to the overall structure, the connection among the components, and the individual components themselves. In practice, there are some consensus about some design decisions. Specifically, the NDP cores are generally a low-complexity core (*e.g.*, single-issue, in-order) with a simple memory hierarchy (*e.g.*, one level of caches only) [3]–[6]. Given that the current promising platform to support NDP is an HMC-like system, where the cores will be physically under the memory vaults, the design space is essentially reduced to that of the interconnection between the cores and the memory units, which is our focus in this paper.

A. Performance Impact of Interconnects

There are a number of design points worth considering. At one extreme of the spectrum, these cores are directly connected to the memory vault that is physically adjacent to the core, Fig. 2 (a). In this case, there is no extra latency, energy overhead, or throughput limit imposed by an interconnect. The constraint is that all data need to reside in the corresponding memory vault. In reality, this requires extra data shuffling and copying, perhaps orchestrated from host processor(s). By ignoring the overhead of fine-grain data shuffling, we obtain an idealized configuration, which we use for finding upper-bounds on the benefits of NDP systems.

A second, more realistic configuration connects the cores through an on-chip fabric to interface with the memory vaults, Fig. 2 (b). Most recent proposals fall into this category [3,4,17]. The approach improves the programming flexibility: as long as the data reside in the memory chip, the cores can access them. Of course, the price to pay is the additional

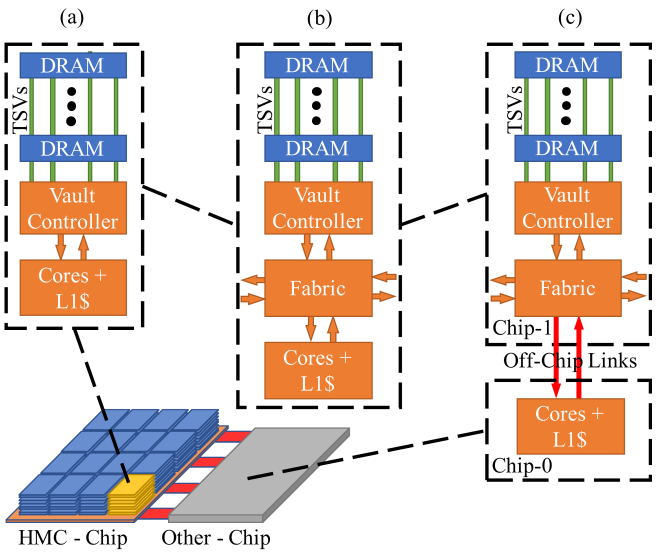


Fig. 2. NDP interconnection design points. (a) Processor cores are directly connected to the memory vault, requiring no interconnect overhead. (b) On-chip fabric connect processing core(s) to memory vaults. (c) Processing cores and memory vaults implemented on separate chips, requiring off-chip communication medium.

latency and energy overhead as well as throughput limits imposed by the fabric.

Finally, once the cores rely on a fabric to connect to memory, they do not need to be physically on the same chip as the memory unit. The fabric can allow them to be in a different die (Fig. 2 (c)). So long as said fabric imposes an insignificant overhead and/or limitation, from a performance standpoint, the cores could still be considered as processing near the data. In other words, it is the performance profile of the interconnect fabric between the cores and the memory units that determines whether the system is truly NDP or not. Physical proximity influences (perhaps significantly) the performance profile of the interconnect, but is otherwise not a requirement per se to achieve the goal of NDP. Indeed, physical proximity brings unwanted thermal coupling and related consequences, among other architectural issues.

Before we get into more detailed discussions about the performance profile of various interconnection designs, it is helpful to get a rough picture of the performance impact of some readily available configurations. We take a number of commonly used benchmarks in NDP designs and measure their execution speed under the different design points mentioned above. For simplicity, we ignore energy issues for now. To recap, the configurations are: [1]

- Ideal: memory accesses have no extra latency cost, energy cost, or throughput limit due to interconnect.
- 3-D Integration (3DI): as in recent NDP proposal, an all-to-all crossbar (with 3-cycle request and 3-cycle response time) is used between cores and memory vaults.

[1]Configuration details are explained in Section V-A

- High-performance links (HPL): The cores reside on a different chip linked to the memory chip via high-performance communication links equivalent to that envisioned for HMC-based systems [51]. Note that these links are by no means the state of the art. We refer to them as high-performance in comparison to legacy interfaces such as DDR.
- Legacy memory channels (DDR): The cores are connected to legacy (DDR4) memory banks with 4 channels. Note that in this configuration, not only is the fabric of lower performance, the DRAM accesses themselves are somewhat slower as well since we use the DDR4 timing parameters.

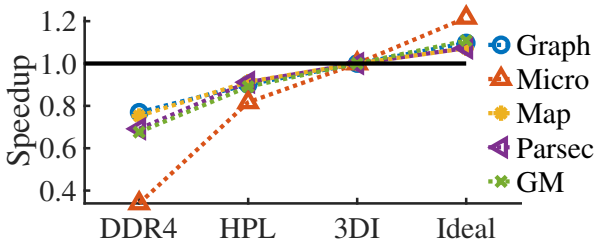


Fig. 3. Performance impact of various interconnection configurations; DDR4, HPL, 3DI, Ideal. Processing distance to memory decreases from left to right. Processor cores are on the same chip as memory in Ideal and 3DI, while in HPL and DDR4 they are on separate chips. Performance results of individual benchmarks are normalized to that of 3DI before averaged geometrically.

Fig. 3 shows the performance impact of the interconnection for a range of workloads. While the details of the setup is discussed in Section IV, we note that many workloads are commonly used in evaluating NDP systems. To this collection, we add a microbenchmark with a random access pattern without reuse to demarcate the extreme of workloads which lend themselves to NDP systems. In other words, we expect that the performance curve for realistic workloads will be less steep than that of this microbenchmark.

From this analysis a number of high-level observations can be made.

1) We can see that the general concept of processing near the data is valid. The performance of all benchmarks improves noticeably as the processing elements get nearer to the data.
2) While we recognize the performance improvements, we see that the gain from the weakest configuration (DDR4) to a more ideal access setup is on the orders of 2x. We note that while repeating similar experiments as reported in recent NDP proposals ([3,4]), we obtained similar results of about 16x performance improvement going from a conventional uni-processor to a parallel NDP design (more on that in Section V). Here, we maintain the processing elements to be the same across all configurations to isolate the contribution of interconnect and thus the importance of *nearness*.
3) Finally, note that even this 2x improvement is the result of comparing to an under-provisioned conventional

memory interface (DDR). When we model a more advanced serial link interface, the performance gap shrinks to about 10% even compared to an ideal interconnect fabric for all but the microbenchmark.

Combining these observations, we can draw a high-level conclusion of NDP with caveats: While at the fundamental level, processing near the data is superior to processing from afar, there is no significant advantage to get as close to data as to be on the same chip. There are three caveats to this conclusion. First, the conclusion applies to the tested workloads and configurations. With better partitioning of work to identify the most appropriate parts for NDP, and with newer workloads, the benefit of NDP could be more pronounced as suggested by the microbenchmark result. Second, these advanced serial links must be practical to build, indeed far more than moving the cores to the memory chip. Third, the analysis is only on a performance basis without consideration to energy. To get a better understanding of these latter two issues, we now investigate the technology of the links.

B. Design of Communication Links

1) Basic overview of link design mechanics: Whether to connect cores to memory on the same chip or separate chips, there are different underlying communication link technologies. The capabilities of the communication substrate directly translate to the logical nearness between data and processing into physical distance constraints. Therefore, we only focus on the high-performance part of the design space using technologies commonly referred to as high-speed serial links, illustrated in Fig. 4.

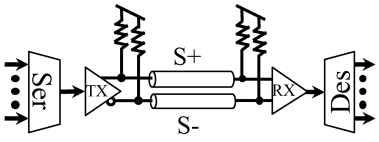


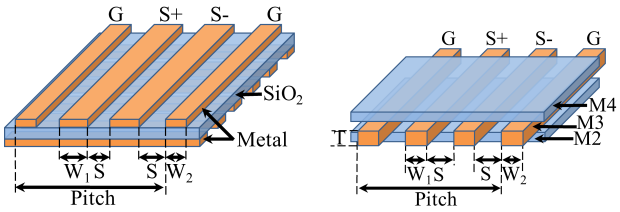
Fig. 4. Basic serial link schematic. Data transmission flows from left to right; transmitter (with serializer), channel, and receiver (with de-serializer).

Because signals are modulated at comparatively high baud rates (e.g., 10s of Gbauds) in such a link, designers take into account every aspect of the link: materials of metals, dielectric and substrate in the technology, signal attenuation due to more pronounced skin effect, and the resulting inter-symbol interference (ISI). They also control more design elements to achieve desired metrics: layout and geometry of the transmission lines; equalization circuits at the transmitter and/or receiver; and power supply mechanisms to these circuits. The design space as well as performance metric space are quite large. A comprehensive exploration is beyond the scope of this paper. Our plan is thus to take more of a middle-of-the-road approach and avoid both commodity-level configurations, as well as overly complex systems aimed for extreme performance.

Finally, some ultra-dense die-to-die communication systems implement serial links on interposer material (Sections II-B and II-C). This is therefore a readily realizable technology option for the off-chip links. Additionally, we consider serial links for the on-chip component of our fabric. This provides a singular and rather straight-forward design space compared to alternative packet-switched interconnects such as meshes.

2) Link design and analysis for this work: The circuit dynamics of the transmitter is a designer choice, mostly for high-performance, low energy, or both. Indeed, any conventional voltage- or current-mode transmitter can be adopted. Using cadence tools, we design and analyze our transmitter similar to prior work, and target high performing non-return-to-zero (NRZ) or pulse-amplitude modulation (PAM) signaling [52]–[54].

We model the characteristics of the channel component through electromagnetic (EM) simulations using Sonnet [55]. We obtain channel-specific s-parameters from such EM simulation. Based on channel characteristics, we choose a differential coplanar waveguide (CPW) structure for on-chip (shown in Fig. 5 (a)) and shielded differential strip-line structure for the off-chip interposer (shown in Fig. 5 (b)). Our geometries are optimized for maximum bandwidth density using the analysis suggested in prior work [56]. Our channel dimension parameters are shown in Table I.



(a) On-chip channel (b) Off-chip interposer channel

Fig. 5. (a) Differential coplanar waveguide topology on-chip links, and (b) differential strip-line topology off-chip links, metal layers M2 and M3 are GND.

TABLE I
CHANNEL PHYSICAL CHARACTERISTICS. LENGTHS ARE IN μM. T IS THICKNESS, W IS WIDTH (W_1 FOR SIGNAL AND W_2 FOR GROUND), S IS SPACING, AND P IS PITCH ($3S + 2W_1 + W_2$).

Channel	Material	T	W_1	W_2	S	P
On-Chip	Aluminum	4	10	5	10	55
Off-Chip	Copper	3	5	6	5	31

The on-chip and interposer off-chip channels have different geometries because of their difference in link distance, metal materials, and transmission lines structures. We account for all such properties in our EM simulations. The interposer-based chip-to-chip links are rather short-reach, often less than a few centimeters in length. We therefore assume a maximum of 2 cm channel length for these links [47,48]. For on-chips links, we assume a maximum of 8 cm channel length. We base our assumption on channel characteristics from other studies, as well as system level applications of such channels [56,57].

In terms of signal propagation, high-speed links offer near speed of light propagation, limited only by specific material

properties and channel length. In our analysis, we utilize FR4 material (as a conservative baseline) with material constant $\epsilon_r = 4.2$, and channel propagation latency of 6.7 ps/mm (using the fundamental propagation velocity expression: $c/\sqrt{\mu_r * \epsilon_r}$). It must be noted that off-chip interposer channels can be manufactured from much lower loss material such as ePTFE ($\epsilon_r = 1.4$) for better propagation, 3.9 ps/mm. Our worst-case end-to-end signal propagation latencies are 536 ps and 134 ps respectively for on-chip and off-chip links. This translates to less than a compute cycle in terms of the NDP system clock rate (see Table VII).

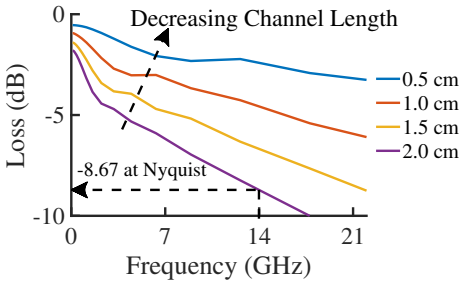


Fig. 6. Off-chip channel loss for diverse channel lengths. Worst-case channel (2 cm) loss at 14 GHz Nyquist (minimum operating) frequency is 8.7 dB.

TABLE II
INTERPOSER OFF-CHIP CHANNEL POWER BREAKDOWN BY COMPONENT AT 28 GHZ BAUD RATE. POWER VALUES ARE IN mW.

Component	Sub-Component	NRZ	PAM-4
Transmitter	Mux	10	20
	Pre-Driver	9.6	9.6
	Main-Driver	5.7	8.2
Receiver	Slicer	1	3
	Demux	8	16
Total		**34.3**	**56.8**

TABLE III
INTERPOSER OFF-CHIP CHANNEL ENERGY EFFICIENCIES.

Signaling	NRZ			PAM-4		
Baud (GHz)	28	17	12	28	17	12
Power (mW)	34.3	23.5	18.5	56.8	39.0	28.2
Energy (pJ/b)	**1.23**	**1.38**	**1.54**	**1.01**	**1.15**	**1.18**

From our transmitter design choice and channel characteristics, we analyze the performance of the complete link (Fig. 4). In our links, the channel loss and ISI are relatively low compared to long-reach backplane links. For example, our EM analysis showed that the off-chip links have 8 dB loss at 14 GHz Nyquist frequency (for 28 Gbaud/s applications). Thus, strong equalization is not required. We therefore only adopt passive equalization techniques on the receiver side to compensate for the channel loss. Fig. 6 shows our channel losses for diverse channel lengths. Passive equalization can provide better linearity and consume less power compared to conventional active continuous-time linear equalization (CTLE). On the receiver side a passive CTLE, comparator, and de-serializer are employed. The whole link operates on 1 V supply.

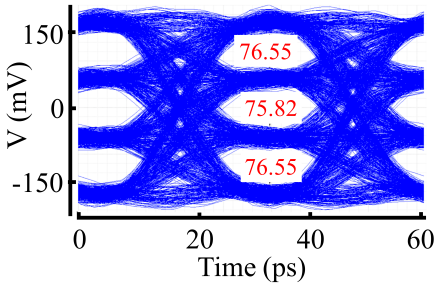


Fig. 7. Interposer off-chip link eye diagram at the receiver for PAM-4 signal rate.

To validate the end-to-end signal integrity of our design, we obtain the eye diagram at the receiver side, see Fig. 7. We observe 75 mV minimum eye opening. This is sufficient for a bit-error-rate criteria better than 10^{-15}. We perform our system level experiments at the maximum baud-rate of 28 GHz. In terms of signaling, this translates to 28 Gb/s and 56 Gb/s bit-rates for NRZ and PAM-4 respectively. We report our power consumption breakdown at the operating baud rate in Table II. Our link power accounts for the SerDes components, which represent more than 60% of total consumption. Finally, we report our link energy efficiencies in Table III, which is derived as follows; $P\,/\,B\,/M$, where P is power, B is bit rate, and M is modulation index (1 for NRZ and 2 for PAM-4). PAM-4 encodes 2 bits per symbol, and therefore requires some component duplication. This fact is evident in the higher PAM-4 power compared to NRZ (Table II). But multiple bit encoding in PAM amortizes the overall per-bit energy cost compared to NRZ.

We observe similar relative energy efficiencies across our 8 cm on-chip and 2 cm off-chip channels per baud-rate and signaling. Our design and analysis only account for the data-path of the link, and does not include a clock-path. Therefore, our system level link power expenditure is proportional to utilization.

To sum, without using the most advanced design techniques (let alone inventing new ones) we can create communication links that offer 10s of Gb/s throughput, near speed-of-light signal propagation, and low pJ/b energy costs. When evaluating the benefits of stacking NDP designs, we have to compare to technology available today. Not just legacy interfaces.

IV. Experimental Methodology

A. Workloads

We use a diverse set of multi-threaded benchmark suites: graph, map-reduce, and parsec. We choose applications to cover the range of thread-level massive parallelism as well as memory intensity. For graph and map-reduced workload,

we use an average of five sets of real-world inputs from the Stanford large network dataset collection [58], and for parsec, we use native-sized input set. We choose our workloads and input set to match previous NDP-related work [4]. Specific workload descriptions can be found in Table IV.

For each workload and input combination, threads only execute the parallel phases. We choose input sizes to achieve a minimum of 1 billion executed instructions. We reduce the impact of non-determinism typical in multi-threaded executions by collecting statistics at specific regions of interest, and we report an average from multiple runs.

TABLE IV
WORKLOAD CONFIGURATION SHOWING 22 APPLICATIONS ACROSS 4 BENCHMARKS SUITES.

Graph [59]	**apsp** - All Pairs Shortest Path, **bece** - Betweenness Centralities, **brfs** - Breadth First Search, **coco** - Connected Components, **comm** - Community Detection, **defs** - Depth Search First, **pgrk** - Page Rank, **sssp** - Single Source Shortest Path, **trct** - Triangle Counting
Map-Reduce [60]	**hist** - Histogram, **kmean** - K-Means, **lire** - Linear Regression, **stma** - String Match, **mamu** - Matrix Multiply, **woco** - Word Count
Parsec [61]	**cann** - Canneal, **flui** - Fluid Animate, **stcl** - Streamcluster
Micro [Written C++ algorithms for required access patterns]	**rand** - Random memory access over entire address range, **rdck** - Random memory access over specific chunk of address range per thread, **st01/st02** - Linear streaming memory access at specific offsets

B. System Level Modeling

We evaluate our systems using a modified version of the GEM5 simulator [62]. HMC sub-system is modeled using HMCSim which we port into GEM5, and the DRAM layers are modeled using a modified version of DRAMCtrl embedded in GEM5 [63]. Prior NDP work utilize 2 NDP sockets each with 512 cores [3]–[6]. For simulation efficiency, we only model one such chip with 128 cores. HMC hardware configuration and timing parameters are derived from literature and specification 2.1, while DDR parameters are derived from Micron DDR data-sheets [1,51,64]. Circuit-level design and simulations are performed with sonnet and cadence [55,65]. Channel s-parameters are obtained from sonnet EM simulations.

1) Memory System: We utilize DDR4 and HMC memory sub-systems in our experiments. For either system, we use 8 GB of total memory capacity. For DDR4 sub-system, we use 4 memory channels, each accessing 2 GB partition of total memory. This configuration is indicative of current CMP systems. High-level memory sub-system configurations are shown in Table V.

For HMC sub-system, we use 1 HMC cube. This is broken down as follows; there are 32-vaults, each vault accesses 8 vertical partitions, each partition has 32 MB DRAM device ($32 * 8 * 32MB = 8GB$) at 16 MB per bank.

TABLE V
MEMORY SUB-SYSTEM CONFIGURATION.

HMC	**Organization**: 8GB, 1 units (cubes), 32 vaults, 8 partitions, 32MB devices **Timing**: t_{CK} = 0.8 ns, t_{RAS} = 21.6 ns, t_{RCD} = t_{CAS} = 10.2 ns, t_{WR} = 8 ns, t_{RP} = 7.7 ns
DDR4	**Organization**: 8GB, 1600 MHz, 4 channels, 4 memory controllers **Timing**: t_{CK} = 0.833 ns, t_{RAS} = 32 ns, t_{RCD} = t_{CAS} = 14.16 ns, t_{WR} = 15 ns, t_{RP} = 14.16 ns

There exists trade-offs in the HMC design parameters. Having more vaults, improves vertical concurrency, but decreases device size. We choose our parameters for good relative performance characteristics, based on the trade-off analysis outlined in other work [66,67].

Internally, each HMC vault has a maximum 10 GB/s vertical bandwidth [51]. This translates to a total of 40 (per vault) data through silicon vias (TSVs) at 2 Gb/s bandwidth [1]. The maximum DRAM bandwidth per cube is therefore 320 GB/s ($10GB/s * 32 vaults$) supplied through 1280 TSVs. Externally, we provide each cube with up to 4 serial links in each direction, for a full duplex setup. Each link having 16 differential strip-line channels (see Fig. 5 (b), for a total of 128 channels. The total external HMC bandwidth is therefore 448 GB/s or 896 GB/s using our 28 Gb/s NRZ or 56 Gb/s PAM-4 signaling respectively (obtained as follows; $4 links * 2 duplex * 16 lanes * bitrate / 8$).

2) Conventional System: Our generic conventional system is a CMP consisting of 2 out-of-order cores, with corresponding private L1 caches, shared L2 cache, and unified L3 cache. The CMP is connected to either DDR or HMC memory sub-system. We use 2 cores for 1-to-64 ratio when compared to NDP core count, similar to prior NDP work. All conventional high-level system configurations are outlined in Table VI.

TABLE VI
HOST/CONVENTIONAL SYSTEM CONFIGURATION.

Processing Node	28 nm, 2 core(s), out-of-order, 4-wide, 192 ROB, 96 LSQ, 128INT/128FP PRF, 4INT/ 2MEM/ 4FP FUs
Operating Points	1V supply, 3GHz freq.
L1 Cache(s)	Split Private I/D, 32 KB, 4-way, 64B blocks, 3 ports, 1 ns hit latency, 4 MSHRs
L2 Cache(s)	256 KB Bank, 8-way, 64B blocks, 1 port, 10 ns hit latency, 20 MSHRs
L3 Cache(s)	20MB, 16-way, 64B blocks

3) 3D-Integrated NDP System: Our baseline NDP system consists of a single chip with 128 single-issue in-order cores implemented in the HMC logic layer, at 28 nm technology node. The cores operate on 1 V supply at 1 GHz frequency. Each core has corresponding privately split instruction and data first level caches. The caches are 32 KB, 2-way associative, and 64 B blocks. We group 4 cores into a node, and each node accesses a single vault controller. We connect the nodes with a 3-cycle fabric adapted from the crossbar designs of prior work [19,20]. The NDP chip is interconnected to a conventional CMP (Section IV-B2) for initialization and synchronization, similar to prior work.

4) Interposer-Based System: An alternative to 3D integrated NDP systems is to use two separate chips one being an HMC chip, the other essentially a CMP with all the (in-order) computing cores. The two chips are connected via an interposer fabric. Fig. 8 shows the high-level schematic of such a system. The processor chip utilizes cores similar to the NDP system. 4 cores are grouped into a node, and an on-chip fabric connects all the nodes as well as interposer transceiver points (ITP). These ITPs are connected to the memory chip via dense interposer-based off-chip serial links, such as Fig. 5 (b). The off-chip links, CMP, and HMC have the same number of communication pins as per HMC hardware specification 1.2 [51]. The corresponding link configuration parameters are outlined in Table VII.

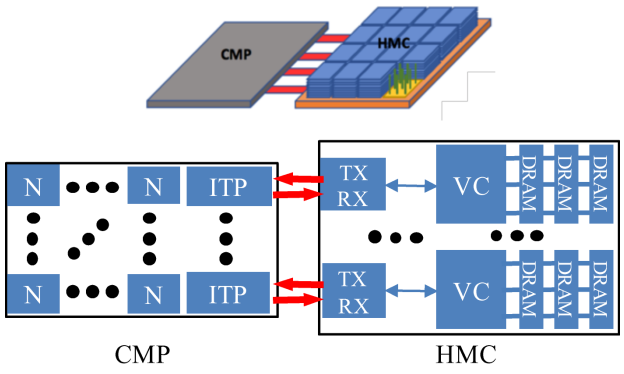


Fig. 8. Interposer system architecture. Dense state-of-the-art off-chip high-speed links are used to connect HMC and CMP chips. Additional on-chip links (not shown) are used for intra CMP communication, and connects processing nodes (N) to interposer transceiver points (ITPs) for memory access.

The CMP's on-chip fabric also uses serial links as a simple, shared, point-to-point medium (called *transmission line link buses* or TLLBs) [68]. These TLLBs offer competitive throughput at a fraction of the energy cost compared to conventional packet-relayed interconnects [68]–[70]. The TLLBs can be used for on-chip communication between cores, thereby eliminating the need for synchronization by a dedicated host processor, as well as other coherence purposes.

Just like a conventional CMP, when a computing core needs off-chip data, it creates a request packet and arbitrates for one of the on-chip links. Here, we adopt the centralized arbitration scheme from prior work [69]. Upon receiving a grant signal, the node's local transmission circuitry transmits the packet to the designated ITP, which is addressed interleaved. The ITP then forwards buffered packet(s) on its off-chip interposer serial links to the HMC. Conversely, on receiving an off-chip packet from the HMC, the ITP can arbitrate for one of the on-chip TLLB links and subsequently forward the packet to the designated node.

TABLE VII
HIGH-SPEED LINK SUB-SYSTEM CONFIGURATION.

On-chip (Serial)	1.75 GHz clock, 8 B flit (1 flit meta packet, 9 flits data packets), 9 bidirectional links - 4 channels per link, 28 Gb/s or 56 Gb/s bit rates, 1 cycle propagation latency, 4 cycle overhead (2 cycle SerDes, 1 cycle request, 1 cycle grant/wakeup)
Off-Chip (Serial)	1.75 GHz clock, 72 B payload, 8 links (4 each direction) - 16 channels per link, 28 Gb/s or 56 Gb/s bit rates
Off-Chip (Parallel) [2]	500 MHz clock, 72 B payload, 4 links (bi-directional) - 128 channels per link, 2 Gb/s bit rates

In this configuration, packet latencies are a function of queueing, serialization, and propagation delays, all of which are modeled faithfully. Queueing delay varies and is modeled in the execution driven simulation. Serialization delay is dependent on data rates. Therefore, for the off-chip links a 72 byte data payload (64 B cache line plus 8 B header) can be serialized in 2 cycles at moderate 28 Gb/s NRZ or 1 cycle at high 56 Gb/s PAM-4 as follows;

$$\left\lceil \frac{72\,B}{16\,lanes * bitrate\,/\,8} \right\rceil = \left\lceil \frac{36}{bitrate} \right\rceil \quad (1)$$

Finally, as discussed earlier (Section III-B2), signal propagation delay (134 ps or 0.134 ns) is less than the NDP-core compute cycle (1 ns). We account for propagation by incurring a 1 cycle delay in our simulations. Essentially, the minimum (unloaded) latency of a data packet from the ITP to the HMC is 3 or 2 cycles respectively for NRZ or PAM-4 signaling. The corresponding latency details for the on-chip interconnect component is modeled similar to other work [57].

C. Power and Area Models

We model power and area characteristics of the cores using a version of McPAT [71], while memory is modelled with DRAMCtrl. Our NDP and interposer systems use high-performance and energy efficient cores such as the ARM Cortex-A5 [72], similar to previous literature. Based on cacti analysis (embedded in McPAT), at 28nm technology node, we obtain an approximate 0.70 mm^2 area per core. Additionally, the logic area of a single HMC cube with 4 full-duplex serial links is estimated to be no more than 90 mm^2 (34 mm package size [51]). We estimate 100 mW core power based on previous ARM energy characterization for similar processor type [73].

On the memory side, our 896 GB/s maximum aggregate bandwidth at PAM-4 transmission rates (56 Gb/s) have

1.01 pJ/b maximum efficiency (see Table III). The peak power of our links is computed in Eq. 2 as follows:

$$\frac{896}{2}\frac{GB}{s} * 1.01\frac{pJ}{b} = 448 * 8\frac{Tb}{s} * 1.01\frac{pW}{s^2 b} = 3.62\,W \quad (2)$$

We assume $2.89\,W$ peak power for the logic layer [3]. Our $6.51\,W$ total HMC logic layer power is approximately half of previous work. Additionally, at 3.7 pJ/b memory access energy, the peak power at the maximum 320 GB/s (2.56 Tb/s) bandwidth is 9.47 W ($2.56 * 3.7$). To match previous work, we attribute 10% of this total power to background activity [3].

On the processor side, our on-chip TLLB is designed to transmit a maximum 64 byte cache lines. At our PAM-4 transmission rates, it provides 3.6 TB/s bandwidth at 1.19 pJ/b maximum energy, and 34.3 W peak power.

D. Thermal Models

We model temperature and leakage characteristics using a version of the HotSpot simulator [74]. We produced a single high-level floor plan for the processor (cores and caches), and a separate high-level floor plans for each of the HMC layers. For our non-stacked systems, we evaluate the processor and HMC thermal characteristics separately. While for fully-stacked system, we evaluate a 3D structure with processor layers below HMC layers. All configurations have appropriate heat sinks, as well as thermal insulators between layers. Additionally, power traces required for analysis are obtained from simulations.

V. Experimental Analysis

In this section, we analyze the performance and energy benefits that are attributable to physical proximity based on our experimental results. There is a diverse range of parameters used in our experiments (or for that matter similar prior experiments). Each come with its own estimation errors and uncertainties. A particularly dangerous source of errors is mixing parameters derived from first principles with product specifications. This paper is no different. Quantifying these uncertainties is beyond the scope of our effort. Consequently, we would present (over-)simplified first-principle analyses together with simulation results.

A. Performance Analysis

Consistency check: First, as a consistency check that we are experimenting with a design similar to recent NDP system proposals, we compare performance of a NDP system to a conventional general-purpose architecture. The NDP setup utilizes an HMC system technology, while the conventional setup utilizes DDR, similar to prior studies [3,4].

In Fig. 9, we see that most applications enjoy more than 10x speedup when executing on an NDP system. We see variation among the groups of applications. The graph applications enjoy a more consistent gain, while the map-reduce workloads see less gain. Overall, the geometric mean speedup is about 15x. This observation is largely in agreement with earlier studies [3,4]. However, this significant performance gain is not entirely due to the architectural feature of stacking memory on top of processors for maximum proximity. In fact, much of the difference is attributable to the large number of cores in the NDP configuration providing a processing throughput advantage over the single-core conventional system. Accessing profile does play a non-trivial role as we will see next.

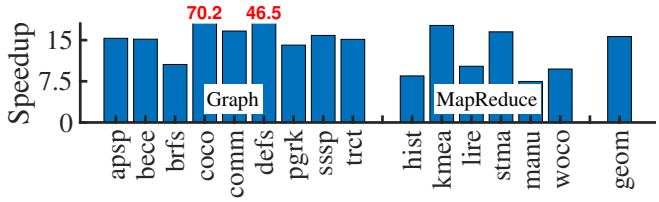


Fig. 9. Performance advantage of NDP system (single-issue in-order cores) over conventional out-of-order cores system. NDP has 64 times more cores than conventional.

Control experiment: To tease out the performance difference attributable to 3D stacking, we keep the processing elements (single-issue in-order cores) and memory (hybrid memory cubes) the same across all configurations and compare different interfaces between them. Starting from the configuration where processing is nearest to data, we gradually move the processing away:

- Ideal: This is an idealized configuration where memory access takes place directly after the cache miss, without going through any intermediate fabric. We normalize all results to that of this configuration.
- 3DI: This is the stereotypical NDP configuration where processor cores are on the same chip as the memory units and are connected via an on-chip fabric such as a crossbar.
- HPL & VHPL: The next two configurations use separate chips for processing cores and for memory vaults. They are connected with high-speed serial links. Depending on the materials and circuit design, these links can offer quite different performance profiles. We thus use two design points to better characterize the space. The HPL (high-performance links) configuration represents a more basic variety assuming NRZ modulation on relatively lossy transmission lines (*e.g.*, over FR4-based channels). This is an approximation of what can be achieved via commodity components today and provides 28 Gb/s throughput per channel. The VHPL (very-high-performance link) configuration models a more advanced point with low loss material such as LTCC [75] and PAM-4 modulation (56 Gb/s).

Fig. 10 shows the detailed results for performance comparison of various fabric alternatives. From this more detailed set of results we can better understand the performance implication of the interconnection between memory units and the cores. Additionally, we evaluate a parallel link fabric configuration (PAR). PAR is equivalent to the interposer-based link interface used in High-Bandwidth Memory systems [2]. The parallel link parameters are given in Table VII under Off-Chip (Parallel) configuration.

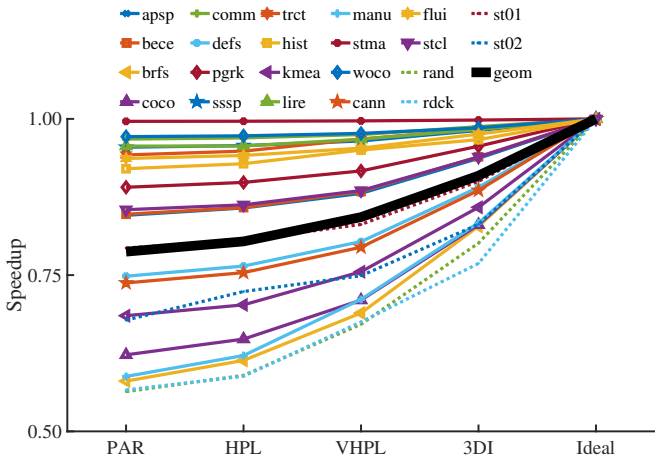


Fig. 10. Performance comparison of different interconnections between processing elements and memory, showing speedups for all applications, normalized to an Ideal interconnection fabric.

- The most extreme performance benefit is seen from the microbenchmark with random access pattern (`rand` and `rdck`). The difference between the two extreme configurations (Ideal vs PAR) is a factor of 1.53x. This is the "maximum dynamic range" that is due to interconnection, or in other words, the upper bound on the performance impact that can be attributable to the decision of stacking or not.
- Note that not all workloads have a sensitivity indicated by this upper bound. At one extreme, one benchmark (`stma`) is utterly unaffected by the fabric. In fact, excluding the microbenchmarks, the range of performance difference narrows to at most 1.33x.
- The bold line represents geometric mean results of our benchmark suite and is thus a reasonable approximation of performance profiles of these design points in general. VHPL and 3DI represents two realistic design points differing only in the aspect of whether to stack or not. The performance difference is about 1.08x.

To summarize, over non-optimized conventional design, stacking processors in memory chips can bring a significant performance gain (1.53x), but only for a very special class of applications. For the workloads evaluated, and using a more realistic alternative for non-stacking design, the performance benefit is about 8%.

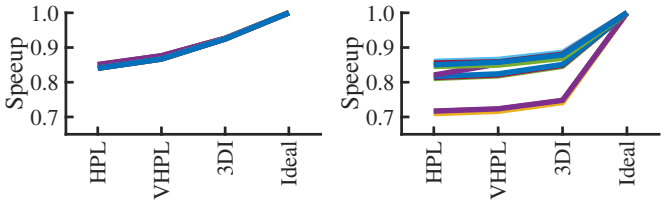
From a simplified first-principle analysis, the average memory access latency after LLC is about 33 cycles. The round trip latency using VHPL adds less than 4 processor cycles. The median cycles-per-instruction (CPI) for these codes is about 3.6 with a median of 8.1 LLC misses per kilo instructions (MPKI). In other words, a memory access happens about once every 440 cycles for each core. The direct latency overhead is in the neighborhood of 1%. Indirect overhead such as queuing delay contributes to the rest of the performance gap. Using the median again, the throughput demand for 128 cores is about 19 GB/s. This is equivalent to what 3 wires in the highest performing configuration can carry.

Performance Sensitivity: To understand whether the conclusion is sensitive to processor configuration, we vary core type, count, cache sizes, as well as frequency. For out-of-order core systems, we reduce the baseline in-order core count by a factor of 4 for rough area equivalence. We summarize our sensitivity options in Table VIII.

TABLE VIII
PERFORMANCE SENSITIVITY CONFIGURATION OPTIONS.

Variable	Options
#Cores	128 (in-order), 32 (out-of-order)
Freq. (GHz)	1, 2, 3
L1I/D Size (kB)	16, 32, 64, 128
L1I/D Assoc.	1, 2, 4, 8

Out of all possible combination of configurations, we repeat our prior performance analysis. For each configuration, only the suite-wide geometric mean curve is shown. Fig. 11 shows the final results. In both figures, the maximum performance gains going from non-stacked (VHPL) to fully-stacked (3DI) system is no more than 6%. Additionally, there is noticeable performance groups with out-of-order cores in Fig. 11b, which is attributable to core frequency.



(a) 128-Core IO (b) 32-Core OO

Fig. 11. Performance sensitivity of architectural design points. IO and OO represent in-order and out-of-order cores respectively. Each line represents the geometric mean of all applications, and across the different systems, with 82 configurations per figure.

Caveats: Finally, it must be noted that our performance analysis in no way argues against bring processing closer to data. It is only isolating the last step of the process of bringing processing closer to memory, namely stacking the processors directly under the memory units. From a performance stand point, our results show that comparable performance can be obtained even if processors and memory are not stacked together. As long as a suitable, carefully designed interconnecting fabric (such as state-of-the-art high-speed links) are utilized. Furthermore, our results are obtained when modeling a single, current generation HMC cube-like device. If future memory devices support orders of magnitude higher internal bandwidths, the scalability of high-performance fabric may deserved a more careful study at that point.

B. Energy Analysis

We next turn to the consideration of energy. Stacking improves energy in two ways. One is the reduction of commu-

nication needed to bring data from memory to the processor. The other is the savings of cycles and thus fixed-cost energy overhead such as clock and leakage. For this analysis, we first ignore leakage energy and thus the effect of elevated temperatures on the system and return to this later.

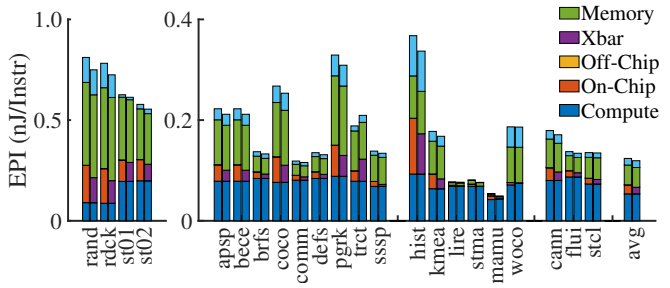


Fig. 12. Energy per instruction (EPI) for VHPL (left bars for each application) and 3DI (right bars). EPI for micro-benchmarks, shown on left y-axis. EPI for real benchmarks on right y-axis due to difference in scale. Energy is broken down into different components from bottom up: compute, various interconnects (on-chip, off-chip, crossbar), DRAM access, and other fixed costs. Note: Off-chip energy is shown but not visible in plot due to the small magnitude relative to other components.

Fig. 12 shows the breakdown of energy per instruction for the designs labeled VHPL and 3DI. Excluding micro-benchmarks (left group of plots in the figure), energy per instruction increases by 3.5% on average from stacked design (3DI) to non-stacked design (VHPL). Keep in mind that in reality, stacking will increase DRAM operating temperature and may increase background energy, which is not accounted for in this plot.

Let us return to the simplified first-principle analysis. We note that our circuit models show an energy on the orders of 1 pJ/b for communication over off-chip high-speed links. This is more conservative than reported values from real test chips [50]. In our simulations, excluding DRAM itself, energy per instruction is on average 188 pJ comparable to measurement results of a similarly-configured processor [73]. Per bit energy consumption of DRAM access in the HMC is about 4 pJ/b. On average, our workload shows about 10 bits DRAM access per instruction, though there is non-trivial variations among different applications (the median is 4 bits per instruction). So from first-order approximation, the direct overhead of going over an additional high-speed link is not significant. It represents less than 10% of the DRAM's access energy, and about 2% of overall energy cost per instructions.

It is sometimes noted by researchers that data movement is extremely expensive energy-wise, especially compared to bare ALU operations. There are two factors to keep in mind. First, communication is not inherently expensive, but can be so depending on circuit design and load. Repeated digital wires, for example, are mainly attractive for their simplicity. We should not extrapolate their energy consumption over long distances. Second, general-purpose processors have significant instruction processing overhead on top of bare functional execution. Specialized accelerators may be a solution, but moving the same general-purpose design closer to main memory does not address this overhead.

C. Thermal Analysis

Stacking involves significant thermal coupling between the cores and the DRAM stacks. This can create elevated temperature and the concomitant higher leakage. The exact detail of the thermal impact depends on many aspects of the design that are hard to estimate accurately. The following analysis is thus a very crude estimate based on public-domain tools.

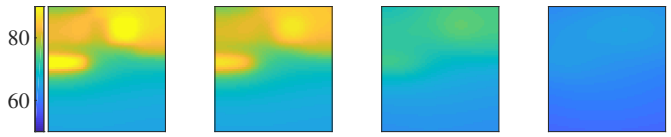


Fig. 13. 3D NDP stacked system temperature profile, showing layers from bottom to top. Logic layer (a) contains the computing cores, bottom layer (b) is DRAM layers closest to logic, middle layer (c) is 4th DRAM layer from bottom, and top layer (d) is layer closest to heat sink.

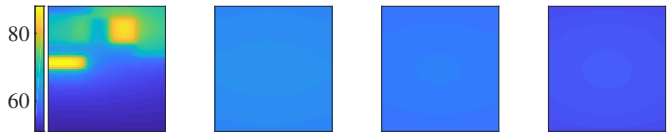


Fig. 14. 2.5D non-stacked temperature profile. HMC DRAM layers are shown from bottom to top. DRAM-middle is 4th DRAM layer from bottom.

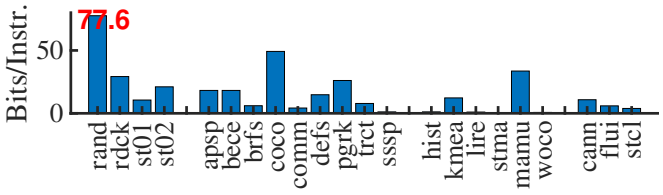
Fig. 13 and Fig. 14 show the temperature profiles for stacked 3D and non-stacked 2.5D systems respectively. As expected, stacking computing cores under memory changes the thermal distribution property of both. We observe that; ① the DRAM layer closest to the cores see at least 9°C rise in temperature, ② the overall temperature of the cores in the 3D stack also increases by last 15°C compared to 2.5D counterpart. Temperature dependent leakage models show that a quadratic relational model in the DRAM operating range (55°C to 90°C) provides more than 99% accuracy [76,77]. Utilizing similar model in our HotSpot analysis, we observe that a 10°C rise in temperature incurs 13% leakage overhead [78].

The exact leakage power depends on many design decisions. If we take one case where 10% of the transistors used in the processors are high-performance transistors and ignore the leakage from the rest of the chip, then the average energy per instruction for 3DI and VHPL would be 175 pJ and 167 pJ, respectively. In this case, the improved energy efficiency for transport is more than offset by the leakage increase due to higher temperature.

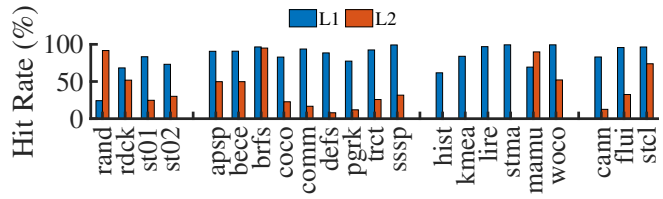
D. Workload Analysis

The analysis so far clearly depends on the workload characteristics. Our workloads include graph applications and

map-reduce workloads. These are often used for evaluating NDP designs. From the memory access statistic shown in Fig. 15, we can see that there is significant variations among applications. For instance, the number of bits accessed per instruction (Fig. 15 (a)) ranges from 0.1 to 26.1 excluding micro-benchmarks, while the micro-benchmark accesses 77 bits per instruction. But overall, these workloads are not very different from general-purpose parallel workloads in their memory access behaviors. For example, most of them have high L1 hit rates (Fig. 15 (b)). Note in the figure that our MapReduce applications have significantly low L2 hit rates; `hist` - 0.002, `kmean` - 0.036, `lire` - 0.119, and `stma` - 0.0267.



(a) Intensity measured as bits per instructions.



(b) Hit rates at L1 and L2 cache levels. MapReduce applications have very low L2 hit rates.

Fig. 15. Memory access profile for different workloads.

Given such memory access behavior, intensity, and DRAM products' access capabilities, the access bandwidth needed for a current generation product is well matched by the bandwidth capabilities using today's high-speed link technologies. The use of silicon carrier technology provides reasonable future scalability. In other words, there is no reason to believe high-speed link will quickly become throughput bottleneck in the near future.

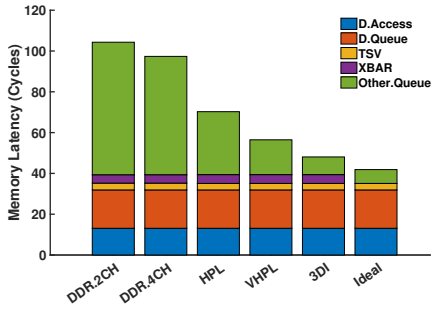


Fig. 16. Memory access latency components; D.Access, D.Queue, TSV, XBAR, and Other. Queue specify DRAM access, DRAM queuing, through silicon via, HMC crossbar, and vault queuing latencies respectively.

Fig. 16 compares the memory access latency breakdown among the several configurations discussed before. We report a controlled experiment where the only change is the processor to memory interconnecting fabric. The processor cores are all the same and the memory structure is HMC. Note that DDR.4CH and DDR.2CH utilize DDR channel timing with 4 or 2 channels respectively. As we move from right to left, the interconnect fabric becomes "weaker" (having degrading performance characteristics), and the queuing delay gradually increases.

Clearly, processing "nearer" data is better. However, we can also see that the effect is a gradual one. Physical stacking represent but a small, incremental step. It is all about the performance characteristics of the fabric connecting data and memory. Additionally, even for NDP workloads, caches still provide substantial filtering such that the latency difference seen in Fig. 16 is only experienced relatively infrequently. Therefore, the step of stacking processor directly under the memory unit is but another step in terms of the performance (and energy) effect. Utilizing high-speed links and avoiding stacking altogether is a valid design point.

VI. Conclusions

Processing-in-memory has long been considered an attractive model to improve execution efficiency. With the increase in data set sizes, it is intuitively even more attractive today. Recent industry development of 3D-stacked memory products gives rise to the notion of finally allowing the embrace of a similar architecture. Dubbed Near-Data Processing, the new proposals advocate for the use of simpler and more energy-efficient cores placed directly under the stacks of memory layers. However, embedding a large number of processing cores and indeed stacking them under the memory layer does present a number of challenges. In this paper, we investigate the performance and energy impact of the single decision in the architecture design space of NDP: that of stacking. Stacking allows the distance between processors and memory to be shrunk to the ultimate limit. However, we have shown that physical stacking is not essential. Indeed if we only apply the state of the art in high-speed link design practice, we can provide an interconnection fabric that virtually places the connected processor near enough to their memory banks. Such a fabric adds insignificant latency to the access path and provides sufficient bandwidth to today's data-intensive applications. As a result, for the parallel sections of a set of data-intensive applications, the overall performance impact, at 1.08x is relatively small. In terms of energy, such a fabric imposes about 4% overhead, if we neglect leakage. If we account for leakage, 3D stacking may end up costing more depending on the transistor type used. Overall, while moving processing closer to the memory is a sensible strategy, the final step of stacking does not appear essential to performance benefits.

REFERENCES

[1] J. Jeddeloh and B. Keeth, "Hybrid memory cube new dram architecture increases density and performance," in *2012 Symposium on VLSI Technology (VLSIT)*, June 2012, pp. 87–88.

[2] D. U. Lee, K. W. Kim, K. W. Kim, H. Kim, J. Y. Kim, Y. J. Park, J. H. Kim, D. S. Kim, H. B. Park, J. W. Shin, J. H. Cho, K. H. Kwon, M. J. Kim, J. Lee, K. W. Park, B. Chung, and S. Hong, "25.2a 1.2v 8gb 8-channel 128gb/s high-bandwidth memory (hbm) stacked dram with effective microbump i/o test methods using 29nm process and tsv," in *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, Feb 2014, pp. 432–433.

[3] S. H. Pugsley, J. Jestes, H. Zhang, R. Balasubramonian, V. Srinivasan, A. Buyuktosunoglu, A. Davis, and F. Li, "Ndc: Analyzing the impact of 3d-stacked memory+logic devices on mapreduce workloads," in *2014 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, March 2014, pp. 190–200.

[4] M. Gao, G. Ayers, and C. Kozyrakis, "Practical near-data processing for in-memory analytics frameworks," in *2015 International Conference on Parallel Architecture and Compilation (PACT)*, Oct 2015, pp. 113–124.

[5] A. Farmahini-Farahani, J. H. Ahn, K. Morrow, and N. S. Kim, "Nda: Near-dram acceleration architecture leveraging commodity dram devices and standard memory modules," in *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)*, Feb 2015, pp. 283–295.

[6] J. Ahn, S. Hong, S. Yoo, O. Mutlu, and K. Choi, "A scalable processing-in-memory accelerator for parallel graph processing," in *2015 ACM/IEEE 42nd Annual International Symposium on Computer Architecture (ISCA)*, June 2015, pp. 105–117.

[7] D. Patterson, T. Anderson, N. Cardwell, R. Fromm, K. Keeton, C. Kozyrakis, R. Thomas, and K. Yelick, "A Case for Intelligent RAM," *Micro, IEEE*, vol. 17, no. 2, pp. 34–44, Mar./Apr. 1997.

[8] M. Hall, P. Kogge, J. Koller, P. Diniz, J. Chame, J. Draper, J. LaCoss, J. Granacki, J. Brockman, A. Srivastava *et al.*, "Mapping Irregular Applications to DIVA, a PIM-Based Data-Intensive Architecture," in *Proceedings of the Supercomputing*, Nov. 1999, p. 57.

[9] Y. Kang, W. Huang, S. Yoo, D. Keen, Z. Ge, V. Lam, P. Pattnaik, and J. Torrellas, "FlexRAM: Toward an Advanced Intelligent Memory System," in *Proceedings of the International Conference on Computer Design*, Oct. 1999, pp. 192–201.

[10] P. M. Kogge, "EXECUBE-A New Architecture for Scaleable MPPs," in *Proceedings of the International Conference on Parallel Processing*, Aug. 1994, pp. 77–84.

[11] M. Oskin, F. T. Chong, and T. Sherwood, "Active PAGES: A Computation Model for Intelligent Memory," in *Proceedings of the International Symposium on Computer Architecture*, Jun.–Jul. 1998, pp. 192–203.

[12] R. Nair, S. F. Antao, C. Bertolli, P. Bose, J. R. Brunheroto, T. Chen, C. Y. Cher, C. H. A. Costa, J. Doi, C. Evangelinos, B. M. Fleischer, T. W. Fox, D. S. Gallo, L. Grinberg, J. A. Gunnels, A. C. Jacob, P. Jacob, H. M. Jacobson, T. Karkhanis, C. Kim, J. H. Moreno, J. K. O'Brien, M. Ohmacht, Y. Park, D. A. Prener, B. S. Rosenburg, K. D. Ryu, O. Sallenave, M. J. Serrano, P. D. M. Siegl, K. Sugavanam, and Z. Sura, "Active memory cube: A processing-in-memory architecture for exascale systems," *IBM Journal of Research and Development*, vol. 59, no. 2/3, pp. 17:1–17:14, March 2015.

[13] S. Khoram, Y. Zha, J. Zhang, and J. Li, "Challenges and Opportunities: From Near-memory Computing to In-memory Computing," in *Proceedings of the 2017 ACM on International Symposium on Physical Design*, ser. ISPD '17. New York, NY, USA: ACM, 2017, pp. 43–46. [Online]. Available: http://doi.acm.org/10.1145/3036669.3038242

[14] M. Zhang, Y. Zhuo, C. Wang, M. Gao, Y. Wu, K. Chen, C. Kozyrakis, and X. Qian, "Graphp: Reducing communication for pim-based graph processing with efficient data partition," in *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, Feb 2018, pp. 544–557.

[15] C. Zhang, T. Meng, and G. Sun, "Pm3: Power modeling and power management for processing-in-memory," in *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, Feb 2018, pp. 558–570.

[16] M. Gao and C. Kozyrakis, "HRL: Efficient and flexible reconfigurable logic for near-data processing," in *2016 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, March 2016, pp. 126–137.

[17] yed Minhaj Hassan, S. Yalamanchili, and S. Mukhopadhyay, "Near Data Processing: Impact and Optimization of 3D Memory System Architecture on the Uncore," in *Proceedings of the 2015 International Symposium on Memory Systems*, ser. MEMSYS '15. New York, NY, USA: ACM, 2015, pp. 11–21. [Online]. Available: http://doi.acm.org/10.1145/2818950.2818952

[18] M. Scrbak, M. Islam, K. M. Kavi, M. Ignatowski, and N. Jayasena, "Exploring the Processing-in-Memory Design Space," *J. Syst. Archit.*, vol. 75, no. C, pp. 59–67, Apr. 2017. [Online]. Available: https://doi.org/10.1016/j.sysarc.2016.08.001

[19] E. Azarkhish, D. Rossi, I. Loi, and L. Benini, "High performance AXI-4.0 based interconnect for extensible smart memory cubes," in *2015 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2015, pp. 1317–1322.

[20] E. Azarkhish, C. Pfister, D. Rossi, I. Loi, and L. Benini, "Logic-Base Interconnect Design for Near Memory Computing in the Smart Memory Cube," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 1, pp. 210–223, Jan 2017.

[21] H. A. D. Nguyen, L. Xie, M. Taouil, R. Nane, S. Hamdioui, and K. Bertels, "On the Implementation of Computation-in-Memory Parallel Adder," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 8, pp. 2206–2219, Aug 2017.

[22] M. Imani, Y. Kim, and T. Rosing, "MPIM: Multi-purpose in-memory processing using configurable resistive memory," in *2017 22nd Asia and South Pacific Design Automation Conference (ASP-DAC)*, Jan 2017, pp. 757–763.

[23] R. Panda, Y. Eckert, N. Jayasena, O. Kayiran, M. Boyer, and L. K. John, "Prefetching Techniques for Near-memory Throughput Processors," in *Proceedings of the 2016 International Conference on Supercomputing*, ser. ICS '16. New York, NY, USA: ACM, 2016, pp. 40:1–40:14. [Online]. Available: http://doi.acm.org/10.1145/2925426.2926282

[24] B. Hong, G. Kim, J. H. Ahn, Y. Kwon, H. Kim, and J. Kim, "Accelerating Linked-list Traversal Through Near-Data Processing," in *Proceedings of the 2016 International Conference on Parallel Architectures and Compilation*, ser. PACT '16. New York, NY, USA: ACM, 2016, pp. 113–124. [Online]. Available: http://doi.acm.org/10.1145/2967938.2967958

[25] S. H. Pugsley, A. Deb, R. Balasubramonian, and F. Li, "Fixed-function hardware sorting accelerators for near data mapreduce execution," in *2015 33rd IEEE International Conference on Computer Design (ICCD)*, Oct 2015, pp. 439–442.

[26] W. Wen, J. Yang, and Y. Zhang, "Optimizing Power Efficiency for 3D Stacked GPU-in-memory Architecture," *Microprocessors and Microsystems*, vol. 49, no. C, pp. 44–53, Mar. 2017. [Online]. Available: https://doi.org/10.1016/j.micpro.2017.01.005

[27] P. C. Santos, G. F. Oliveira, D. G. Tom, M. A. Z. Alves, E. C. Almeida, and L. Carro, "Operand size reconfiguration for big data processing in memory," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2017*, March 2017, pp. 710–715.

[28] C. Xie, S. L. Song, J. Wang, W. Zhang, and X. Fu, "Processing-in-memory enabled graphics processors for 3d rendering," in *2017 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, Feb 2017, pp. 637–648.

[29] P. Chi, S. Li, C. Xu, T. Zhang, J. Zhao, Y. Liu, Y. Wang, and Y. Xie, "Prime: A novel processing-in-memory architecture for neural network computation in reram-based main memory," in *Proceedings of the 43rd International Symposium on Computer Architecture*, ser. ISCA '16. Piscataway, NJ, USA: IEEE Press, 2016, pp. 27–39. [Online]. Available: https://doi.org/10.1109/ISCA.2016.13

[30] L. Song, Y. Zhuo, X. Qian, H. Li, and Y. Chen, "Graphr: Accelerating graph processing using reram," in *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, Feb 2018, pp. 531–543.

[31] D. Xu, Y. Liao, Y. Wang, H. Li, and X. Li, "Selective Off-loading to Memory: Task Partitioning and Mapping for PIM-enabled Heterogeneous Systems," in *Proceedings of the Computing Frontiers Conference*, ser. CF'17. New York, NY, USA: ACM, 2017, pp. 255–258. [Online]. Available: http://doi.acm.org/10.1145/3075564.3075584

[32] S. Lee, H. Sim, Y. Kim, and S. S. Vazhkudai, "AnalyzeThat: A Programmable Shared-Memory System for an Array of Processing-In-Memory Devices," in *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, May 2017, pp. 619–624.

[33] A. Boroumand, S. Ghose, M. Patel, H. Hassan, B. Lucia, K. Hsieh, K. T. Malladi, H. Zheng, and O. Mutlu, "LazyPIM: An Efficient Cache

Coherence Mechanism for Processing-in-Memory," *IEEE Computer Architecture Letters*, vol. 16, no. 1, pp. 46–50, Jan 2017.

[34] J. Ahn, S. Yoo, O. Mutlu, and K. Choi, "Pim-enabled instructions: A low-overhead, locality-aware processing-in-memory architecture," in *2015 ACM/IEEE 42nd Annual International Symposium on Computer Architecture (ISCA)*, June 2015, pp. 336–348.

[35] K. Hsieh, E. Ebrahim, G. Kim, N. Chatterjee, M. O'Connor, N. Vijaykumar, O. Mutlu, and S. W. Keckler, "Transparent offloading and mapping (tom): Enabling programmer-transparent near-data processing in gpu systems," in *Proceedings of the 43rd International Symposium on Computer Architecture*, ser. ISCA '16. Piscataway, NJ, USA: IEEE Press, 2016, pp. 204–216. [Online]. Available: https://doi.org/10.1109/ISCA.2016.27

[36] A. Pattnaik, X. Tang, A. Jog, O. Kayiran, A. K. Mishra, M. T. Kandemir, O. Mutlu, and C. R. Das, "Scheduling techniques for gpu architectures with processing-in-memory capabilities," in *2016 International Conference on Parallel Architecture and Compilation Techniques (PACT)*, Sept 2016, pp. 31–44.

[37] Z. Liu, I. Calciu, M. Herlihy, and O. Mutlu, "Concurrent data structures for near-memory computing," in *Proceedings of the 29th ACM Symposium on Parallelism in Algorithms and Architectures*, ser. SPAA '17. New York, NY, USA: ACM, 2017, pp. 235–245. [Online]. Available: http://doi.acm.org/10.1145/3087556.3087582

[38] Y. Wang, M. Zhang, and J. Yang, "Towards memory-efficient processing-in-memory architecture for convolutional neural networks," in *Proceedings of the 18th ACM SIGPLAN/SIGBED Conference on Languages, Compilers, and Tools for Embedded Systems*, ser. LCTES 2017. New York, NY, USA: ACM, 2017, pp. 81–90. [Online]. Available: http://doi.acm.org/10.1145/3078633.3081032

[39] S. F. Yitbarek, T. Yang, R. Das, and T. Austin, "Exploring specialized near-memory processing for data intensive operations," in *2016 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2016, pp. 1449–1452.

[40] A. Kannan, N. E. Jerger, and G. H. Loh, "Enabling interposer-based disintegration of multi-core processors," in *2015 48th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, Dec 2015, pp. 546–558.

[41] N. Jerger, A. Kannan, Z. Li, and G. H. Loh, "NoC Architectures for Silicon Interposer Systems: Why Pay for more Wires when you Can Get them (from your interposer) for Free?" in *2014 47th Annual IEEE/ACM International Symposium on Microarchitecture*, Dec 2014, pp. 458–470.

[42] A. Kannan, N. E. Jerger, and G. H. Loh, "Exploiting interposer technologies to disintegrate and reintegrate multicore processors," *IEEE Micro*, vol. 36, no. 3, pp. 84–93, May 2016.

[43] I. Akgun, J. Zhan, Y. Wang, and Y. Xie, "Scalable memory fabric for silicon interposer-based multi-core systems," in *2016 IEEE 34th International Conference on Computer Design (ICCD)*, Oct 2016, pp. 33–40.

[44] M. J. Schulte, M. Ignatowski, G. H. Loh, B. M. Beckmann, W. C. Brantley, S. Gurumurthi, N. Jayasena, I. Paul, S. K. Reinhardt, and G. Rodgers, "Achieving exascale capabilities through heterogeneous computing," *IEEE Micro*, vol. 35, no. 4, pp. 26–36, July 2015.

[45] T. Vijayaraghavan, Y. Eckert, G. H. Loh, M. J. Schulte, M. Ignatowski, B. M. Beckmann, W. C. Brantley, J. L. Greathouse, W. Huang, A. Karunanithi, O. Kayiran, M. Meswani, I. Paul, M. Poremba, S. Raasch, S. K. Reinhardt, G. Sadowski, and V. Sridharan, "Design and Analysis of an APU for Exascale Computing," in *2017 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, Feb 2017, pp. 85–96.

[46] G. H. Loh, N. E. Jerger, A. Kannan, and Y. Eckert, "Interconnect-memory challenges for multi-chip, silicon interposer systems," in *Proceedings of the 2015 International Symposium on Memory Systems*, ser. MEMSYS '15. New York, NY, USA: ACM, 2015, pp. 3–10. [Online]. Available: http://doi.acm.org/10.1145/2818950.2818951

[47] X. Gu, L. Turlapati, B. Dang, C. K. Tsang, P. S. Andry, T. O. Dickson, M. P. Beakes, J. U. Knickerbocker, and D. J. Friedman, "High-density silicon carrier transmission line design for chip-to-chip interconnects," in *2011 IEEE 20th Conference on Electrical Performance of Electronic Packaging and Systems*, Oct 2011, pp. 27–30.

[48] S. H. Lee, S. K. Lee, B. Kim, H. J. Park, and J. Y. Sim, "Current-mode transceiver for silicon interposer channel," *IEEE Journal of Solid-State Circuits*, vol. 49, no. 9, pp. 2044–2053, Sept 2014.

[49] H. Lee, T. Song, S. Byeon, K. Lee, I. Jung, S. Kang, O. Kwon, K. Cheon, D. Seol, J. Kang, G. Park, and Y. Kim, "A 16.8gbps/channel single-ended transceiver in 65nm cmos for sip based dram interface on si-carrier channel," in *2014 IEEE Asian Solid-State Circuits Conference (A-SSCC)*, Nov 2014, pp. 125–128.

[50] B. Dehlaghi and A. C. Carusone, "A 0.3 pj/bit 20 gb/s/wire parallel interface for die-to-die communication," *IEEE Journal of Solid-State Circuits*, vol. 51, no. 11, pp. 2690–2701, Nov 2016.

[51] "Hybrid memory cube specification 2.1," 2014, hybrid Memory Cube Consortium, Tech. Rep.

[52] A. K. Joy, H. Mair, H. C. Lee, A. Feldman, C. Portmann, N. Bulman, E. C. Crespo, P. Hearne, P. Huang, B. Kerr, P. Khandelwal, F. Kuhlmann, S. Lytollis, J. Machado, C. Morrison, S. Morrison, S. Rabii, D. Rajapaksha, V. Ravinuthula, and G. Surace, "Analog-DFE-Based 16Gb/s SerDes in 40nm CMOS That Operates Across 34dB Loss Channels at Nyquist with a Baud Rate CDR and 1.2Vpp Voltage-Mode Driver," in *Proceedings of the IEEE International Solid-State Circuits Conference*, 2011, pp. 350–351.

[53] S. K. Lee, S. H. Lee, D. Sylvester, D. Blaauw, and J. Y. Sim, "A 95fJ/b Current-Mode Transceiver for 10mm On-Chip Interconnect," in *Proceedings of the IEEE International Solid-State Circuits Conference*, 2013, pp. 262–263.

[54] L. Lu and H. Wu, "An energy-efficient high-swing pam-4 voltage-mode transmitter," in *2018 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*, July 2018.

[55] http://www.sonnetsoftware.com/.

[56] Y. Wang and H. Wu, "Design high bandwidth-density, low latency and energy efficient on-chip interconnect," in *2017 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*, July 2017, pp. 1–6.

[57] R. Afoakwa, L. Lu, Y. Wang, H. Wu, and M. Huang, "High swing pulse-amplitude modulation of transmission line links for on-chip communication," in *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2018, pp. 1–5.

[58] J. Leskovec and A. Krevl, "SNAP Datasets: Stanford large network dataset collection," http://snap.stanford.edu/data, Jun. 2014.

[59] M. Ahmad, F. Hijaz, Q. Shi, and O. Khan, "Crono: A benchmark suite for multithreaded graph algorithms executing on futuristic multicores," in *Proceedings of IEEE International Symposium on Workload Characterization*, 2015, pp. 44–55.

[60] T. Liu, C. Curtsinger, and E. D. Berger, "Dthreads: Efficient deterministic multithreading," in *Proceedings of the 23rd ACM Symposium on Operating Systems Principles*, ser. SOSP '11. ACM, 2011.

[61] C. Bienia, S. Kumar, J. Singh, and K. Li, "The PARSEC Benchmark Suite: Characterization and Architectural Implications," in *Proceedings of the International Conference on Parallel Architecture and Compilation Techniques*, Sep. 2008.

[62] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M. D. Hill, and D. A. Wood, "The gem5 simulator," *ACM SIGARCH Computer Architecture News*, vol. 39, no. 2, pp. 1–7, 2011.

[63] J. D. Leidel and Y. Chen, "Hmc-sim: A simulation framework for hybrid memory cube devices," in *2014 IEEE International Parallel Distributed Processing Symposium Workshops*, May 2014, pp. 1465–1474.

[64] Micron, "Ddr4 sdram," 62015, https://www.micron.com/resource-details/cf8c7ab8-15b4-4052-a39f-179488615ad1.

[65] https://www.cadence.com/.

[66] P. Rosenfeld, E. Cooper-Balis, T. Farrell, D. Resnick, and B. Jacob, "Peering over the memory wall: Design space and performance analysis of the hybrid memory cube," 2012.

[67] P. Rosenfeld, "Performance exploration of the hybrid memory cube," 2014.

[68] A. Carpenter, J. Hu, J. Xu, M. Huang, and H. Wu, "A Case for Globally Shared-Medium On-Chip Interconnect," in *Proceedings of the International Symposium on Computer Architecture*, Jun. 2011.

[69] A. Carpenter, J. Hu, O. Kocabas, M. Huang, and H. Wu, "Enhancing Effective Throughput for Transmission Line-Based Bus," in *Proceedings of the International Symposium on Computer Architecture*, Jun. 2012, pp. 165–176.

[70] J. Hu, J. Xu, M. Huang, and H. Wu, "A 25-gbps 8-ps/mm transmission line based interconnect for on-chip communications in multi-core chips," in *2013 IEEE MTT-S International Microwave Symposium Digest (MTT)*, June 2013, pp. 1–4.

[71] 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 *Proceedings of the International Symposium on Microarchitecture*, Dec. 2009.

[72] A. Developer, "Cortex-a5 processors," 2016, http://infocenter.arm.com/help/topic/com.arm.doc.ddi0433c/DDI0433C_cortex_a5_trm.pdf.

[73] E. Vasilaki", ""an instruction level energy characterization of arm processors"," "2015", "Tech. Rep. FORTH-ICS/TR-450".

[74] W. Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, K. Skadron, and M. R. Stan, "Hotspot: a compact thermal modeling methodology for early-stage vlsi design," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 14, no. 5, pp. 501–513, May 2006.

[75] J. Zhou, "Towards rational design of low-temperature co-fired ceramic (ltcc) materials," *Journal of Advanced Ceramics*, vol. 1, no. 2, pp. 89–99, Jun 2012. [Online]. Available: https://doi.org/10.1007/s40145-012-0011-3

[76] Y. Liu, R. P. Dick, L. Shang, and H. Yang, "Accurate temperature-dependent integrated circuit leakage power estimation is easy," in *2007 Design, Automation Test in Europe Conference Exhibition*, April 2007, pp. 1–6.

[77] H. Sultan, S. Varshney, and S. R. Sarangi, "Is leakage power a linear function of temperature?" *CoRR*, vol. abs/1809.03147, 2018.

[78] G. Singla, G. Kaur, A. K. Unver, and U. Y. Ogras, "Predictive dynamic thermal and power management for heterogeneous mobile platforms," in *2015 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2015, pp. 960–965.