Architecting Waferscale Processors - A GPU Case Study

Saptadeep Pal*, Daniel Petrisko†, Matthew Tomei†, Puneet Gupta*, Subramanian S. Iyer*, and Rakesh Kumar†
*Department of Electrical and Computer Engineering, University of California, Los Angeles
†Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign
{saptadeep,puneetg,s.s.iyer}@ucla.edu, {petrisk2,tomei2,rakeshk}@illinois.edu

***Abstract*—Increasing communication overheads are already threatening computer system scaling. One approach to dramatically reduce communication overheads is waferscale processing. However, waferscale processors [1], [2], [3] have been historically deemed impractical due to yield issues [1], [4] inherent to conventional integration technology. Emerging integration technologies such as Silicon-Interconnection Fabric (Si-IF) [5], [6], [7], where pre-manufactured dies are directly bonded on to a silicon wafer, may enable one to build a waferscale system without the corresponding yield issues. As such, waferscalar architectures need to be revisited. In this paper, we study if it is feasible and useful to build today's architectures at waferscale. Using a waferscale GPU as a case study, we show that while a 300 mm wafer can house about 100 GPU modules (GPM), only a much scaled down GPU architecture with about 40 GPMs can be built when physical concerns are considered. We also study the performance and energy implications of waferscale architectures. We show that waferscale GPUs can provide significant performance and energy efficiency advantages (up to 18.9x speedup and 143x EDP benefit compared against equivalent MCM-GPU based implementation on PCB) without any change in the programming model. We also develop thread scheduling and data placement policies for waferscale GPU architectures. Our policies outperform state-of-art scheduling and data placement policies by up to 2.88x (average 1.4x) and 1.62x (average 1.11x) for 24 GPM and 40 GPM cases respectively. Finally, we build the first Si-IF prototype with interconnected dies. We observe 100% of the inter-die interconnects to be successfully connected in our prototype. Coupled with the high yield reported previously for bonding of dies on Si-IF, this demonstrates the technological readiness for building a waferscale GPU architecture.**

***Keywords*—Waferscale Processors, GPU, Silicon Interconnect Fabric**

I. Introduction

With the emergence of new applications [8], [9], [10], [11], [12], new business models [13], [14]), and new data processing techniques (e.g., deep learning), the need for parallel hardware has never been stronger [15], [16], [17]. Unfortunately, there is an equally strong countervailing trend. Parallel hardware necessitates low overhead communication between different computing nodes. However, the overhead of communication has been increasing at an alarming pace. The area taken by IO circuitry to support chip-to-chip communication already exceeds 25% on some of today's processors [18]. Power overhead of such IO exceeds 30% on some processors. These overheads are expected to be worse in future as communication energy, latency, and bandwidth scale much worse than computation [19].

A large body of recent work targets the communication bottleneck. Focus areas include energy-proportional communication fabrics [20], [21], [22], in-memory and near-memory computational models [23], [24], [25], communication avoiding algorithms [26], and novel packaging techniques such as 2.5D and 3D integration [27], [28]. We explore a different approach - *waferscale processors*. Conventionally, many copies of a single processor die are manufactured simultaneously on a single wafer - largest size of the die determined by yield. Post-manufacturing, the wafer is diced into individual processor dies which are then packaged and integrated into a parallel system using IO links that connect these packaged processors on a printed circuit board (PCB). In a waferscale processor, on the other hand, the wafer is the processor, i.e., either a monolithic processor is designed to be as large as an entire wafer or a set of processors are designed that continue to reside on the wafer and the processor die are connected on the wafer itself using a low cost, on-wafer interconnect.

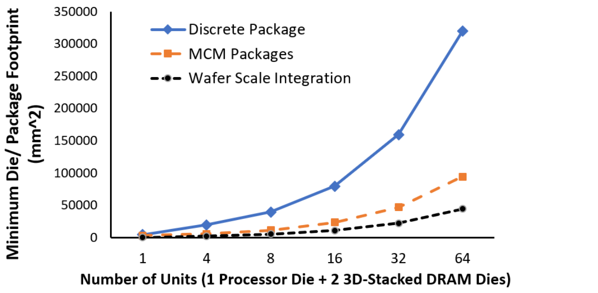
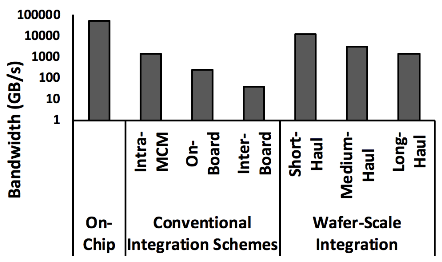
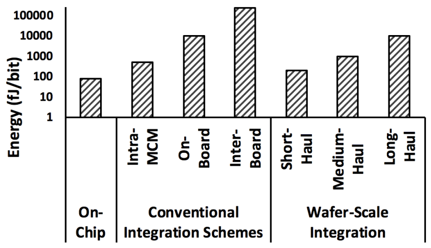


Fig. 1: Minimum die/package footprint for different integration schemes are shown for increasing number of processor dies per system

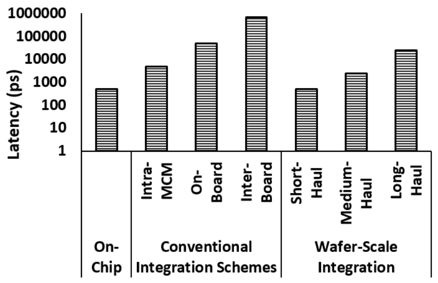
A waferscale processor can have considerable advantages over conventional systems in terms of area, performance and energy efficiency. Figure 1 shows the total area footprint of the compute dies in multiple scenarios: each die is placed in a discrete package, 4 units (each unit consists of a processor die and two 3D-stacked DRAM dies) inside an multi-chip module (MCM) package and, waferscale integration. The area overhead of packages is usually quite high and for high performance systems, the package to die ratio can be more than 10:1 [29]. While MCM packages help decrease the package footprint per die, packageless waferscale integration would provide substantial benefits in terms of compute per area. Figure 2 compares the available communication bandwidth, latency, and energy per bit for waferscale integration versus conventional integration schemes and on-chip interconnects. In waferscale integration, the interconnects would have similar pitch as that of on-chip interconnects. In conventional integration schemes, while intra-die connections inside an MCM package can have fine wire pitch, the PCB traces as well as between-PCB links are I/O limited. This constrains the total bandwidth. Therefore, waferscale integration enables much larger bandwidth than what conventional integration schemes can provide. Since the waferscale links are smaller and high density, simple parallel communication protocol can be used where a massive number of links run at a relatively lower frequency [6]. This helps eliminate SerDes circuitry and thus



(a) Bandwidth



(b) Energy Per Bit



(c) Latency

Fig. 2: Comparison of communication link bandwidth, energy per bit and latency for on-chip links and different types of links in conventional integration schemes and waferscale integration scheme.

reduces energy and latency of communication. There are also significant advantages in terms of test and packaging costs [30].

Unsurprisingly, waferscale processors were studied heavily in the 80s. There were also several commercial attempts at building waferscale processors [1], [4]. Unfortunately, in spite of the promise, such processors could not find success in the mainstream due to yield concerns. In general, the larger the size of the processor, the lower the yield - yield at waferscale in those days was debilitating [4].

We argue that considerable advances in manufacturing and packaging technology have been made since then and that it may be time to revisit the feasibility of waferscale processors. In particular, it is now possible to reliably bond pre-manufactured dies directly on to the wafer [31], [32], [5]. So, it may be possible to build a high yield waferscale processor by bonding small, high yielding dies on to the wafer and connecting them using a low-cost wafer-level interconnect (Silicon Interconnect Fabric, $Si-IF$). Coupled with the fact that potential benefits from waferscale processing may be much larger now considering the high (and increasing) communication overheads today, we would like to better understand the benefits and challenges of building a waferscale processor today.

Previous work in [7] has proposed packageless processors based on the Si-IF based substrate and shown significant performance improvement coming from bandwidth, thermal and area benefits of removing packages. However, that work only focused on a conventionally-sized single-die processor system ($\sim$600 mm^2, 150W). This paper, on the other hand, focuses on the architecture of a GPU system that is as large as an entire 300mm wafer, i.e., 70,000mm^2 of available area.

This paper makes the following contributions:

- This is the first paper that studies if it is feasible and useful to build a waferscale GPU system. We show that while a 300 mm wafer employing an emerging integration technology can house about 100 GPU modules (GPM), only a much scaled down GPU architecture with about 40 GPMs can be built when physical concerns are considered.
- We perform an architectural exploration for waferscale GPUs under different physical constraints. We find that waferscale GPUs are area-constrained due to power delivery network overheads, *not* thermally-constrained. We show that a 24 GPM architecture is possible on a 300mm wafer for a junction temperature constraint of 105°C. A 41 GPM architecture is enabled when 4-module voltage stacks are allowed with each GPM running at lowered voltage and frequency. We also find that waferscale GPU architectures can be supported with ring, mesh, or 1D/2D torus topologies - more connected topologies such as crossbars are not feasible to build due to wiring limitations on such large processors.
- We show that waferscale GPU architectures have considerable performance and energy efficiency benefits for many GPU applications compared to equivalent interconnected discrete GPUs or even interconnected MCM-GPUs. E.g., *color* [33] has 10.9x and 17.8x speedup for 24-GPM and 40 GPM waferscale GPUs over equivalent interconnected MCM-GPU-based systems. Average performance and energy efficiency benefits of a 40-GPM system across all our workloads are 5.14x and 22.5x respectively.
- We study the impact of thread block scheduling and data placement on wafer-scale GPU architectures. Our techniques for thread group scheduling and data partitioning coupled with our placement strategy can provide up to 2.88x performance benefit (average 1.4x) over state-of-art [34] in large GPU scheduling. Average benefit in terms of EDP is 49% and 20% for 24 and 40 GPM systems respectively.
- Finally, we present the first Si-IF prototype with interconnected dies. The 100% successful interconnection between dies we observed for our 100mm wafer Si-IF prototype with 10 interconnected 4mm^2 dies, coupled with high yield reported previously for bonding of dies on Si-IF, demonstrates the technological readiness for building waferscale GPU architecture.

II. Background and Technology Readiness

Several recent integration technologies have been aimed at building larger systems. In particular, 2.5D integration technologies such as TSMC CoWoS (interposer based solution) [35] and Intel's EMIB [20] allow building larger systems by integrating multiple high yielding dies on a high bandwidth, low latency interconnect substrate. However, these technologies have size limitations. Interposers use thinned silicon and are, therefore, fragile. As such, the size of interposer-based systems is usually limited to the size of the reticle. Beyond reticle size, interposers are built by stitching multiple reticles, which is a costly and complex process and has low yield [36], [37]. As a result, the largest commercial interposer today [38] is about 1230 mm^2 in size and accommodates only one GPU and 4 memory stacks. Even Intel's EMIB technology can integrate only about 5-10 dies on the interconnect substrate [20].

Another promising integration technology that can enable larger systems is *Silicon Interconnect Fabric, (Si-IF)* [5], [6], [7]. Si-IF replaces the organic printed circuit board (PCB) with

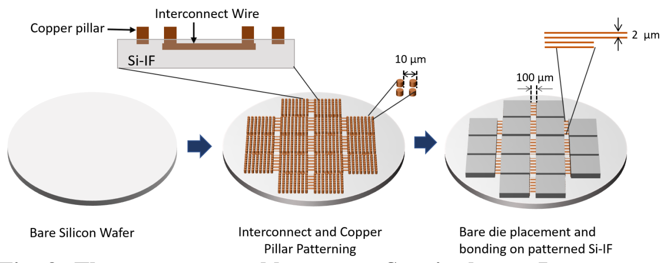

Copper pillar
Interconnect Wire
Si-IF
10 µm
100 µm
2 µm
Bare Silicon Wafer
Interconnect and Copper Pillar Patterning
Bare die placement and bonding on patterned Si-IF

Fig. 3: The system assembly process flow is shown. Interconnect layers and copper pillars are made by processing the bare silicon wafer. Bare dies are then bonded on the wafer using TCB

a silicon substrate and allows placing and bonding bare silicon dies directly on to the thick silicon wafer using copper pillar based I/O pins. The smaller, high yield dies are interconnected on the passive interconnect substrate (wafer) using mature fabrication technologies to ensure high yield. Different system components such as processor, memory dies alongside non-compute dies such as peripherals, VRM, and even passives (inductors and capacitors) can be directly bonded on the Si-IF without the requirement of packages for individual components [19], [5], [7]. Figure 3 shows the process flow of a system assembly on Si-IF. Direct bonding of silicon dies on a silicon wafer using copper pillar coupled with short channels between the dies due to absence of packages allows Si-IF to achieve significantly more performant and energy efficient communication (Figure 3). [1]

Si-IF is an obvious candidate for waferscale integration since silicon dies (and other components) are directly integrated on to a silicon wafer. However, to enable waferscale processing, Si-IF must provide high system yield. Below we argue that Si-IF will provide the high system yield needed for waferscale integration.

There are three components to the yield of a Si-IF-based waferscale system - yield of the die, yield of the copper pillar bonds, and yield of the Si-IF substrate. High yield (>99%) can be ensured for the dies by using known-good-die (KGD) testing techniques [39], [40] to pre-select the dies to be used for assembly on the Si-IF. The yield of the copper pillar bonds is also expected to be close to 99%. Note that the primary mode of copper pillar based I/O failures is opens. Copper pillars are not prone to extrusion unlike solder based connections, so shorts are not possible. Also, since both the substrate and the dies are made out of silicon, there is no co-efficienct of thermal expansion mismatch between them to cause large stresses on the copper pillar bonds due to large temperature fluctuations. Moreover, the misalignment in the place and bond tool used for the prototypes is <1 μm while the pillar spacing is 5 μm. Therefore, any I/O failures due to shorts or misalignment are unlikely. In fact, previous work indeed observed copper pillar yield higher than 99% [5], [7]. Furthermore, since fine pitch copper pillars ($<10\mu m$) allow at least 25x more I/Os than today's integration schemes with solder based connections ($>50\mu m$ pitch), redundancy can be employed to improve system yield i.e., multiple pillars per logical I/O. Moreover, network level resiliency techniques [41], [42] can be employed to route data around faulty dies and interconnects on the wafer to enhance system yield.

[1]Other bonding technologies such as solder-based micro bumps can also be used. Tradeoff of using solder-based microbumps with Si-IF instead of using copper pillars include coarser pitch (25 μ vs 5 μm, higher electrical resistivity (11-13 $\mu\Omega$-cm vs 1.7 $\mu\Omega$-cm), higher likelihood of intermetallics and fatigue related failures, and easier debondability (at 220-230$^\circ C$ vs 1000$^\circ C$).

Finally, the yield of the Si-IF substrate will be high ($>90\%$) since it is a passive wafer with only thick interconnect wires (2um width, 4um pitch) and no active devices[2]. We calculate the expected yield of an Si-IF substrate (shown in Table I) for different number of metal layers and metal layer utilization with $2\mu m$ wire width and spacing using industry standard yield modeling equations 1 and 2. [3] The calculated yield values for the substrate for small number of metal layers is high. [4]

$$Yield = (1 + \frac{D_0 \times F_{crit} \times Area}{\alpha})^{-\alpha} [44],[43] \tag{1}$$

$$F_{crit}^{open} = \int_0^\infty (2r - p/2) * \frac{r_c^2}{r^3} dr = F_{crit}^{short} [45] \tag{2}$$

TABLE I: Yield of Si-IF for different number of metal layers

Si-IF Metal Layer Utilization (%)	Number of Layers		
	1	2	4
1	99.6	99.19	98.39
10	96.05	92.26	85.11
20	92.29	85.18	72.56

Previous Si-IF prototypes had not established connectivity across multiple dies. Therefore, there was no measurement of the yield of the Si-IF substrate or the system yield for an Si-IF-based system with interconnected dies. To assess viability of inter-die interconnect on Si-IF, we built a prototype where we bonded connectivity testing dielets on a 100mm waferscale Si-IF. Copper pillars are connected in a serpentine fashion within and across dielets as shown in Figure 4. Figure 5 shows the micrograph of the prototype. Each row on a dielet with dimensions 2mm×2mm has 200 copper pillars in a row (40,000 pillars in total) which are connected using the serpentine structure. We connect an array of 5x2 dielets, each 4 mm^2. We tested the electrical connectivity across the dies to check whether Si-IF can indeed provide connectivity across many dielets at high yields.

Our electrical tests show that 100% of the interconnects in this prototype were connected illustrating very high yield of the copper pillars as well as inter-die interconnect on Si-IF. Post-bonding thermal cycling tests were done from -40°C to 125°C to test the impact of temperature change on copper pillar bonds and the results demonstrated that all the copper pillars and interconnects withstood the thermal cycles without any noticeable degradation in bond contact resistance. The high yield we observed for this prototype, coupled with high yield reported previously for bonding of dies on Si-IF, demonstrates the technological readiness for building waferscale systems. In subsequent sections, we use a GPU case study to a) explore the space of feasible waferscale architectures, b) understand and maximize the performance benefits from waferscale architectures, and c) develop data placement and thread scheduling strategies for waferscale GPUs.

[2]Vast majority of the yield loss in semiconductor manufacturing happens in transistor layers (front-end-of-the-line) and first few metal layers with small pitch ($\leq$200nm).

[3]D_0 is the defect density per mm^2, α is the defect clustering factor and we use the ITRS value of 2200 and 2 [43] respectively for the calculations. r_c is the critical defect size, p is the interconnect pitch, F_{crit} is the fraction of critical area prone to faults.

[4]Since the inter-die interconnect would only run between the tightly spaced dies, the amount of interconnect area is expected to be less than 10% of the entire wafer area.

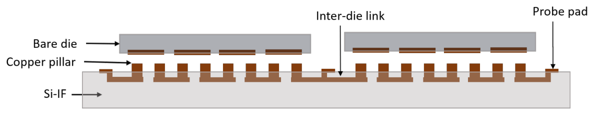


Fig. 4: Schematic of the prototype with interconnection between the serpentine structure of two different dies is shown

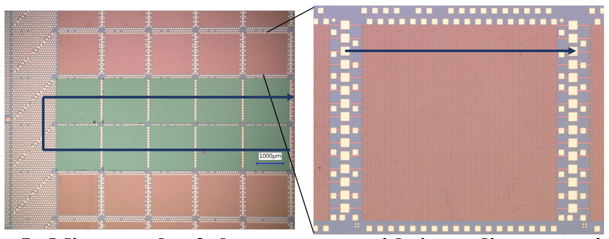


Fig. 5: Micrograph of the prototype with inter-die connectivity is shown. Ten 4 mm^2 dies are bonded and tested for continuity of a signal across the dies. Each die has rows of serpentine structure as shown in the schematic. A zoomed-in picture of the Si-IF is also shown with 40,000 copper pillars.

III. A Case for waferscale GPU Architecture

GPU applications tend to have large amounts of parallelism [46], [47]. As such, GPU hardware parallelism keeps increasing [34], [48], limited only by cooling and yield. In fact, a large class of applications from the domain of physics simulations, linear algebra and machine learning routinely run across 100s of GPUs [5] - such applications will greatly benefit from increasing the effective size of a GPU since multi-GPU approaches have programmability [52] and communication overheads [34]. Proposals such as MCM-GPU [34] do attempt to increase the effective size of the GPU, but are limited by the size limit of conventional integration technologies.

TABLE II: GPU Topologies

	ScaleOut SCM-GPU	ScaleOut MCM-GPU	Waferscale GPU
CUs per GPM	64	64	64
L2 Cache per GPM	4 MB	4 MB	4 MB
DRAM (HBM)	1.5 TB/s 100 ns 6 pJ/bit	1.5 TB/s 100 ns 6 pJ/bit	1.5 TB/s 100 ns 6 pJ/bit
GPMs per package	1	4 (Ringbus)	All (Mesh)
GPM Interconnect	None	1.5 TB/s 56 ns 0.54 pJ/bit	1.5 TB/s 20 ns 1.0 pJ/bit
Package Topology	Mesh	Mesh	Overall System Package
Package Interconnect	256 GB/s 96 ns 10 pJ/bit	256 GB/s 96 ns 10 pJ/bit	None

We evaluated three constructions of a highly parallel GPU system, described in Table II. We consider the smallest hardware unit in these constructions to be a GPM (GPU Module), roughly equivalent to a large sized GPU available today combined with a 3D-DRAM die. Each GPM has a TDP of 200W and area of 500mm^2 for the GPU die, plus 70W and 200mm^2 for two 3D-stacked DRAM dies. Note that the inter-GPM communication energy for waferscale case is higher than on-package inter-GPM communication energy in an MCM-GPU. This is because the GPM dies in the floorplan we considered (see Section IV-D) are separated by DRAM and voltage regulator modules (VRM) on the wafer and so the inter GPM distance is about ∼20 mm versus 2-5 mm in MCM package (where VRM are usually off-package or on the package periphery and not in between the dies)

[5]E.g.NWChem, PSDNS, MILC, NAMD, QMCPAK, Chroma, GAMESS, MELD, AMBER etc. routinely run across the 3072 GPUs on Blue Waters [49], [50], [51]; Nvidia also provides HPC containers for warehouse scale GPU applications

The first construction we consider is ScaleOut SCM-GPU (single-chip module GPU), where each GPM is contained in its own package. GPMs are placed in a 2D mesh on a traditional PCB, connecting via an inter-package link with bandwidth, latency and energy characteristics similar to QPI. The second is ScaleOut MCM-GPU, an extension of MCM-GPU where MCM-GPU units are placed in a 2D mesh on a traditional PCB connected with a QPI-like link. The last architecture we evaluate is a hypothetical waferscale GPU (i.e., we do not consider thermal or power delivery constraints) - a single wafer containing a 2D mesh of GPMs connected via Si-IF [6]. The GPMs constitute a single logical GPU from the perspective of the programmer.

Figures 6, 7 show the potential advantages of a waferscale GPU over the ScaleOut SCM-GPU or ScaleOut MCM-GPU approach for two benchmarks, SRAD and Backprop, both from the Rodinia benchmark suite [53]. These two applications were chosen to be representative of medical imaging and machine learning, both fields expected to substantially benefit from waferscale processing. Our simulations are performed by using gem5-GPU [54] to generate memory traces and activity profiles, which we feed into our own trace-based GPU simulator. We expand upon our experimental methodology in Section VI.

For Backprop, we observe a 47.54x speedup for 64 GPM waferscale GPU over a single GPM system. Speedup is 20.8x and 21.13x over the highest performing ScaleOut SCM-GPU and ScaleOut MCM-GPU configurations respectively. The speedups are eventually limited by the memory transfer latency. Note that these speedups are achieved without requiring changes to the programming model unlike speedups achieved by other ScaleOut system integration schemes.

The benefits of waferscale GPU over ScaleOut SCM-GPU and ScaleOut MCM-GPU are more apparent when considering EDP. 64 GPM Waferscale GPU has a 31.54x reduction in EDP compared to a single GPM and trends downwards, whereas both ScaleOut MCM-GPU and ScaleOut SCM-GPM systems increase in EDP past 9 GPMs.

For SRAD, we observe a 42.56x speedup for 64 GPM waferscale GPU over a single GPM system. This is compared to ScaleOut SCM-GPU and ScaleOut MCM-GPU which saturate at 3.57x and 3.65x speedup, respectively. Additionally, by avoiding costly inter-package communication, the 64 GPM waferscale GPU manages a 24.88x reduction in EDP, a sharp contrast to ScaleOut MCM-GPU and ScaleOut SCM-GPU, where additional GPMs actually increase EDP.

The above results show that GPU architectures are a good fit for building at waferscale as performance and energy efficiency scaling of GPU applications is much stronger on a waferscale GPU than equivalent interconnected discrete GPUs or even interconnected MCM-GPUs. In the next section, we explore the space of feasible waferscale GPU architectures.

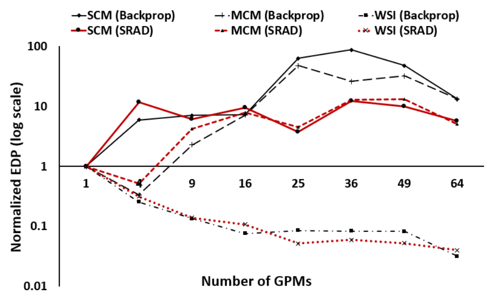


Fig. 6: Normalized EDP for Backprop and SRAD

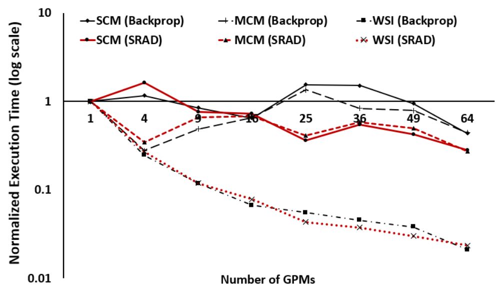


Fig. 7: Normalized Execution Time for Backprop and SRAD

IV. Architecting a Waferscale GPU

Architecting a waferscale GPU is a unique problem due to the physical constraints of a waferscale processor. A waferscale GPU architecture will need to operate at kilowatts of power; the corresponding architecture must be feasible in presence of the associated thermal and power delivery concerns. Similarly, a waferscale GPU will need enormous interconnection resources (due to the need for connectivity among the GPMs). A waferscale GPU architecture must support network topologies that are realizable at the waferscale level.

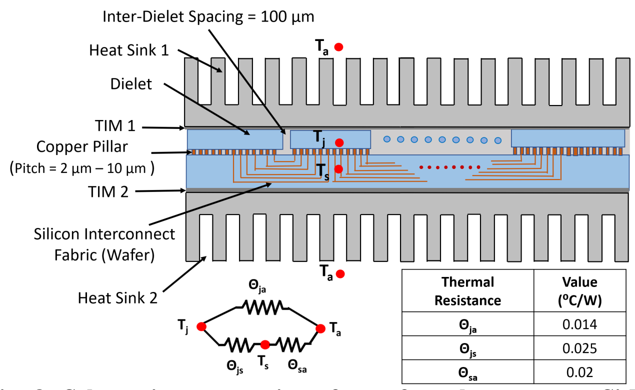
In this section, we attempt to identify feasible waferscale GPU architectures in presence of thermal, power delivery, and connectivity constraints. Our analysis considers a GPM module consisting of a 500 mm^2 GPU die and 200 mm^2 DRAM dies with TDP of 200W and 70W respectively[6].

A. *Waferscale GPU Architecture under Thermal Constraints*

In this subsection, we ask the question: Given a target maximum junction temperature and forced air cooling, what is the maximum number of GPMs that can be accommodated on the 300 mm wafer?

To determine the maximum allowable TDP, we assume that system would be cooled using one or two square heat sinks covering the round 300 mm wafer with forced air convection cooling. Figure 8 shows the schematic of two heat sinks attached to the wafer, one directly on top of the dies, and the other on the back side of the wafer. The secondary heat sink not only provides mechanical support for the wafer, it also helps increase the heat extraction efficiency. The thermal resistance model is shown in Figure 8. The thermal modeling and analysis is performed using a commercial CFD-based thermal modelling tool from R-tools [55] - CFD is known to provide more accurate results than simple spice based models used in tools such as HotSpot [56].

We evaluated both the cases, one heat sink and two heat sinks for three different junction temperatures (T_j). Conventionally, 85°C [57], [17] and 105°C [58] are used as reliable T_j. Since, we were not able to simulate a very up-to-date heat sink solution, for fair comparison, we performed thermal simulation of a recently published multi-GPM system (MCM-GPU) described in [34] using our framework. It resulted in a junction temperature of 121°C considering a heat-sink of the size of the package 77mm×77mm with ambient temperature of 25°C. As a result we also analyze for T_j=120°.



Thermal Resistance	Value (°C/W)
Θ_{ja}	0.014
Θ_{js}	0.025
Θ_{sa}	0.02

Fig. 8: Schematic cross-section of a waferscale system on Si-IF, alongside the thermal resistance model. Ta, Tj , Ts denote the ambient, chip-junction and silicon substrate temperatures respectively. Two heat sinks one directly attached to the dies and another backside heatsink covering the Si-IF substrate is shown. The thermal resistance values for a 300 mm^2 waferscale system with heatsinks are also shown [55]

We consider that 20000 mm^2 out of the 70000 mm^2 would be used for external connections and other interfacing dies[7]. Therefore, 50000 mm^2 would be available on the wafer for placing the GPMs and point-of-load voltage regulator modules. For the maximum TDP estimation, we consider that multiple heat sources (GPMs and DRAMs) are generating heat on a surface of size 50000 mm^2.

In Table III, we show the sustainable TDP for the various scenarios described above. We also present the total number of GPMs within that thermal budget with and without voltage regulator modules (VRMs). When no VRM is considered, the only heat sources are the GPM modules. In case VRMs are placed on the wafer, there would be additional heat loss due to VRM inefficiency. Here, we assume on-Si-IF VRM to have an efficiency of about 85% [59]. Therefore, effectively a VRM would lead to an additional power dissipation of 48W per GPM.

Our analysis shows that while 50000 mm^2 of area is available on the wafer for computational purposes ($\sim$ 71 GPMs), thermal limitations constrain the maximum number of GPMs that can be placed on the Si-IF to be much lower. Considering the dual heat sink solution, up to 34 GPMs can be supported if no power loss from VRM is assumed, else the number reduces to 29 GPMs.

B. *Waferscale GPU Architecture considering Power Delivery*

A waferscale GPU system is constrained by the heat sink technology to a total TDP of up to about 9.3 kW. Considering

[6]3D stacked memory is not a necessity for waferscale. Using a planar memory dies would decrease the capacity and bandwidth per unit area.

[7]A waferscale GPU can interface to the external world using multiple (e.g., PCIe) ports connected to one or more root complexes.

TABLE III: No of supportable GPMs for different junction temperatures

Target Junction Temperature (°C)	Dual Heat Sink			Single Heat Sink		
	Power (W)	Num GPMs w/o VRM	Num GPMs with VRM	Power (W)	Num GPMs w/o VRM	Num GPMs with VRM
120	9300	34	29	6900	25	21
105	7600	28	24	5400	20	17
85	5850	21	18	4350	16	14

the rated TDP to be 0.75 times [60], [61] the peak power of the system, the power distribution network (PDN) must be able to provide up to 12.5 kW of power (compared to 1-2kW for a modern server board [62], [63], [64]) with reasonable efficiency even at peak power.

We explore external power supply alternatives of 48V, 12V, 3.3V and 1.2V to the wafer with a point of load (POL) power conversion using efficient buck converters for every GPM i.e, there would be one VRM per GPM. In general, the higher the input voltage, the larger the PDN circuitry overhead, but also the fewer the number of layers required to supply power (also lower resistive loss), as shown in Tables IV and V.

TABLE IV: Number of Layers Required vs Supply Voltage to the Wafer. 10μm thick metal is available in most technologies which support RF

Input Voltage(V)	I^2R Loss (W)	No of Layers		
		Thickness = 10μm	Thickness = 6μm	Thickness = 2μm
1	500	42	68	202
3.3	200	10	16	44
	500	6	8	18
12	100	2	4	10
	200	2	2	4
48	50	2	2	2
	100	2	2	2

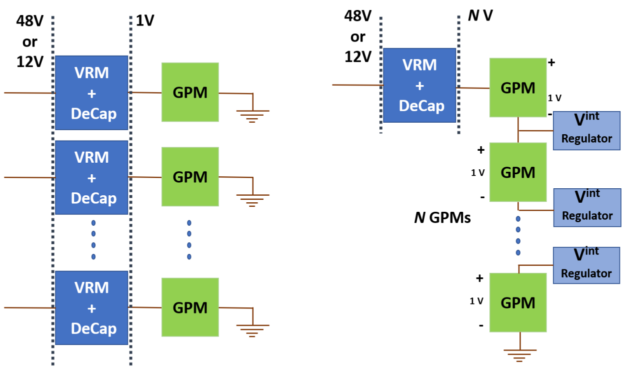
TABLE V: VRM & Decap Overhead Per GPM

Input Voltage (V)	VRM+DeCap Area Per GPM (mm^2)			Number of GPMs		
	No Stack	2-Stack	4-Stack	No Stack	2-Stack	4-Stack
1	300	-	-	50	-	-
3.3	1020	610	-	29	38	-
12	1380	790	495	24	33	41
48	2460	1330	765	15	24	34

We use the power distribution mesh sizing models given in [65] to determine the minimum area and thus the *number of layers required for power distribution*. As shown in Table IV, we find that the number of metal layers required for 1V and 3.3V supply to the wafer is very high, even for a very large I^2R loss. Moreover, more than 4 metal layers for power delivery is undesirable due to cost and manufacturability reasons. Therefore, the only viable options are 12V or 48V external power input.

To compare the two external power input options (12V and 48V), we recognize that the size of VRMs for DC-DC conversion and regulation can be very large due to the required inductance and capacitance. Area efficiency of state-of-art 48V-to-1V converters at reasonably high power conversion efficiency (>90%) is in the 1W/10mm^2 - 1W/5mm^2 (including the VRM inductor) [66], [59] range when implemented using PCB based integration. We conservatively assume 1W/6mm^2 area overhead of VRM for a 48V-to-1V power conversion.

This means that to support a GPM with TDP 200W alongside 70W 3D stacked local DRAM (i.e., peak power of 360W), the area of the VRM for the 48V-to-1V option would be approximately 2160 mm^2 (360×6)! We also calculated the



(a) One VRM per PDN **(b) Stacked voltage supply**

Fig. 9: PDN schemes

area overhead of surface mount decoupling capacitors required to compensate for current load variation of about 50A with a frequency of 1 MHz [67] to be ∼300 mm^2. Therefore, for a 48V-to-1V conversion strategy, the total number of GPUs that can be accommodated in the usable 50,000mm^2 area on the wafer would be only about 15. Also, in this case, the total peak power draw from the external source would be ∼ 6.5 kW i.e., a TDP of 4.9 kW assuming 85% VRM efficiency and accounting for I^2R losses. Thus, *the system is area constrained and not TDP constrained (since the maximum allowable TDP is about 9.3 kW)*. This is a salient result since it suggests that much higher performance and energy efficiency may be possible from waferscale computing simply through improving the area efficiency of voltage conversion.

For the 12V option, the size of the VRM is expected to be smaller (∼ 1W/3mm^2). Therefore, about 24 GPMs could now be accommodated which would amount to a total of approximately 10.3 kW of maximum power (7.8 kW TDP). However, we still cannot accommodate the maximum number of 29 GPMs, as dictated by the thermal limit at 120°C T_j. Therefore, as was the case for 48V supply, the system is area constrained, not thermally constrained.

An alternate power distribution strategy is voltage stacking of multiple GPUs [68], [69], [70], as shown in Figure 9. If N GPMs are stacked, the supply voltage to the stack should be N times the supplied voltage required for one GPM and the same current flows through the stacked GPMs. As shown in the previous work [70], this approach is viable in the GPM context since neighboring GPMs are expected to have similar activity and power draw at any time interval (good data placement and scheduling policy can also help). Now, instead of using N voltage regulators (one per GPM), one VRM with $1/N$ conversion ratio would be shared across N GPMs. As the conversion ratio decreases, the size of the VRM modules can be decreased for the same efficiency. Though a perfectly balanced stack would ensure stable intermediate node voltage, we anticipate use of lightweight voltage regulators (such as push-pull regulators [71] which can reduce middle-rail noise while minimizing static power dissipation) to ensure guaranteed stability of the intermediate nodes. More details regarding such intermediate voltage circuitry has been shown in [70]. Since these intermediate voltage node regulators would only be responsible for stabilizing small current demands and not voltage conversion, compact switched capacitor (SC) based

TABLE VI: Proposed PDN solutions

Target Junc. Temp. (°C)	Dual Heat Sink			Single Heat Sink		
	Thermal limit (W)	Supply Voltage (V)/ # GPMs per Stack	Maximum Number of GPMs at Nominal	Thermal limit (W)	Supply Voltage (V)/ # GPMs per Stack	Maximum Number of GPMs at Nominal
120	9300	48/4 or 12/2	29	6900	48/2 or 12/1	21
105	7600	48/2 or 12/1	24	5400	48/2 or 12/1	17
85	5850	48/2 or 12/1	18	4350	48/1	14

TABLE VII: Operating Voltage and Frequency for the 41 GPMs with 12 V supply and 4-GPMs per stack

Target Junc. Temp. (°C)	Dual Heat Sink			Single Heat Sink		
	GPM Power (W)	Operating Voltage (mV)	Operating Frequency (MHz)	GPM Power (W)	Operating Voltage (mV)	Operating Frequency (MHz)
120	125.75	877	469.6	71.75	752	364.2
105	92	805	408.2	44.75	664	291.4
85	51.5	689	311.7	24.5	570	216.2

regulators or linear drop-out (LDO) regulators can be used. Our conservative estimates (based on experience and prior work) suggest that these intermediate regulators would have an area footprint of around 200 mm^2. Due to this reduced per GPM footprint, we find that 34 GPMs can now be accommodated with 4 GPMs per stack and 48V power supply to the wafer, while 41 GPMs can be accommodated with 12 V supply and 4 GPMs in a stack. These results show that voltage stacking is a promising technique to enable scalable waferscale GPU architectures. Table VI shows the different proposed PDN design choices and corresponding number of supportable GPMs.

While up to 41 GPMs can be supported using voltage stacking, recall that thermal limits only allow us to pack 29 GPMs (with VRMs considered) running at nominal operating conditions. We, therefore explore the opportunity to further maximize the number of GPMs by decreasing the supply voltage and operating frequency of each GPM. To accommodate 41 GPMs, we find the operating voltage and frequency of these GPMs such that the total power is within the maximum thermal power budget. We only considered scaling the GPM voltage while maintaining the same DRAM voltage. The scaled operating voltage and frequency for GPMs is presented in Table VII. Notice that to support the decreased voltage per GPM, the down conversion ratio of the VRM needs to be enhanced. Earlier, we conservatively estimated the size of the 12V/4V VRM to be that of 12V/1V conversion ratio, this increase in down conversion ratio (up to 12V/2.4V) can be easily handled by the VRM of this size.

C. Allowable Network Architectures for a Waferscale GPU

The above analysis does not consider interconnection between the GPM modules. For a GPM die size of 500 mm^2 (90 mm perimeter), wire pitch of 4 μm and effective signalling rate of 2.2 GHz per wire [6] (ground-signal-ground with 4.4GHz signal speed), the total bandwidth available per layer is ∼6TBps. Increasing the number of layers would result in increased inter-GPM and DRAM bandwidth, however, it would lower yield[8]. As shown in [34], increasing the local DRAM bandwidth to the GPMs beyond 1.5TBps results in only a very small performance improvement, but lowering the bandwidth results in significant performance loss. With this DRAM bandwidth in mind (1.5TB/s), we analyze a few realizable inter-GPM network topologies for different signal metal layer count on Si-IF (see Table VIII). Here, we only consider yield loss due to shorts and opens of the signalling wires on different metal layers[9]. One should note that increasing the DRAM bandwidth has much smaller effect on the Si-IF interconnect yield than increasing the inter-GPM bandwidth as the GPM to local DRAM spacing is 100-500 μm spacing while the inter-GPM distance for a 5×5 GPM array would be about 16 mm. We assume KGD and that the copper pillar redundancy scheme would take care of the yield loss due to bonding failure.

Three topologies: ring, mesh and connected 1D Torus can be realized using one layer. Note that 2D Torus cannot be realized using one metal layer without major design and signalling efforts as some links would have to be routed around the GPM array.

Moving to a two layer solution, we can see that a ring network would be over provisioned with inter-GPM as well as DRAM bandwidth. Increasing the signal layer count to three layers enables more balanced 2D torus network, however at an expense of yield which can now be as low as 73.4%.

To summarize, yield concerns constrain the total number of metal layers on a GPU and this, in turn, limits the allowable network topology configurations on a waferscale GPU. We are now ready to select the viable waferscale GPU architectures that maximize performance and energy while satisfying all the physical constraints.

D. Overall System Architecture

We consider two configurations at the target junction temperature of 105°C; one with 24 GPMs running at nominal voltage of 1V and 575 MHz, second with 40 GPMs running at reduced voltage of 805mV and 469 MHz. For the former, we consider power supply at 12V and no stacking, while for the latter case, we consider 12V power supply with 4 GPMs in a stack. We show the floorplans for both these options in Figures 11 and 12. We show the floorplans with 25 GPMs and 42 GPMs considering redundancies of 1 GPM and 2 GPMs respectively.

Our inter-GPM network is a mesh network that uses two metal layer. The local DRAM bandwidth as well as inter-GPM bandwidth for each GPM is considered to be 1.5 TBps. Though 6 TBps of DRAM bandwidth per GPM can be supported with one layer, with 6pJ/bit memory access energy, the total DRAM power would be quite high (200 W) [73]. Hence, we considered DRAM bandwidth of 1.5TBps as it would result in about 72W of DRAM power. This is almost equal to the aggregate TDP of 2 3D-stacked DRAM modules [74] that we have assumed per GPM.

For the first case, when no stacking is used, every GPM has a set of 2 local 3D stacked DRAM chips, a VRM and decoupling capacitors, forming a tile of dimensions 42mm×49.5mm. The goal is to floorplan a total of 25 such regular tiles. A 5×5 array floorplan is difficult to realize since the size of the largest square that can be inscribed in a round 300mm wafer is only about 45000 mm^2 (∼ 21 tiles). Hence, one possible floorplan to lay out these 25 tiles is as shown in Figure 11. This floorplan closely resembles a mesh architecture

[8]Increasing the number of layers increases process complexity as well as the amount of critical area susceptible to particle defects [72], [45]

[9]Yield is calculated using the industry-standard negative binomial yield model [45], [44], [43] described in section II. We use inverse cubic defect density distribution [72], defect density value as per ITRS [43] and metal pitch of 4μm.

TABLE VIII: Inter-GPM Network Topologies

Num of Layers	Topology	Memory Bandwidth (TBps)	Inter-GPM Bandwidth (TBps)	Yield (%)	Diameter	Average Hop Length	Bisection Bandwidth (TBps)
1	**Ring**	3	1.5	95.9	15	7.5	3
	Mesh	3	0.75	95.9	10	4	3.75
	Connected 1D Torus	3	0.5	94.1	8	3	3.75
2	**Ring**	6	3	91.9	15	7.5	6
	Ring	3	4.5	88.6	15	7.5	9
	Mesh	6	1.5	91.9	10	4	7.5
	Mesh	3	2.25	88.6	10	4	11.25
	Connected 1D Torus	3	1.5	84.3	8	~3	11.25
	2D Torus	3	1.125	79.6	5	~2.6	11.25
3	**2D Torus**	6	1.5	77.0	5	~2.6	15
	2D Torus	3	1.875	73.4	5	~2.6	18.75

without the corner tiles. We also considered the area for $System+I/O$ which would house many system level blocks for external interfaces (CPU-GPU, drivers), oscillators etc.

Similarly for the floorplan with voltage stacking, we place 42 dies with one VRM + DeCap every voltage stack of 4 GPMs and three intermediate node voltage (V^{int}) regulators. We were able to place 32 GPMs in an array while 10 other GPMs and $System+I/O$ are placed on the top and bottom side of the array. A mesh network connects the GPMs. Considering KGD GPM, DRAM and VRM die with 99% average yield per I/O and 4 pillars per I/O, the estimated bond yield for the 25 and 42 GPM systems are about 98% and 96.6% respectively. The Si-IF substrate yields which depend on the interconnect length are 92.3% and 95% respectively, this is because the inter-GPM wires are smaller in the 40-GPM floorplan. Therefore, overall yield is expected to be about 90.5% and 91.8% for the two cases respectively. Note that while the floorplans have 25 and 42 GPMs respectively, maximum number of operating GPMs would be limited to 25 and 40 due to thermal budgets. The extra GPMs can be used as spare GPMs to improve system yield in case of one/two GPMs become faulty.

Note also that even larger GPU systems could be built by tiling multiple wafer-scale GPUs. Given we have 940 mm of wafer edge (300 mm wafer diameter) and 20,000 mm^2 of area left for external connectors, about 20 PCIe socket connectors could be accommodated at the periphery assuming half the periphery is used to deliver power. Using PCIe 5.x which supports 128 GBps per x16 link, a total of 2.5 TBps of off-wafer bandwidth can be supported.

System Integration: One system integration strategy (Figure 10) involves using a primary heat sink and an optional secondary heat sink to cover the wafer assembly (which is passivated to protect against humidity, etc). In case the secondary heat sink is not used, a backside rigid metal plate would be used to encase the system. In case of installation into a chassis (servers/ desktops etc.), the complete system can be inserted using normal plug connectors or low force insertion sockets [10]. Alternatively, the external metal heat sinks can be used to bolt to the chassis. We estimate that a row in a standard 19in wide / 36in deep cabinet can house two 300mm (12in) waferscale processors, including heatsinks. A 42U cabinet can house up to 6 rows (i.e., 12 WS-GPUs).

[10]Since silicon is a much more robust material than FR4 material used to build PCBs (compressive strength of 3.2-3.4 GPa vs 370-400 MPa) and with backside support (using heat sink or plate), a 700 μm to 1 mm thick wafer can easily bear the normal insertion force of plug connectors (few 10s of MPa); connections can also be wire-bonded to the system I/O pads, if needed.

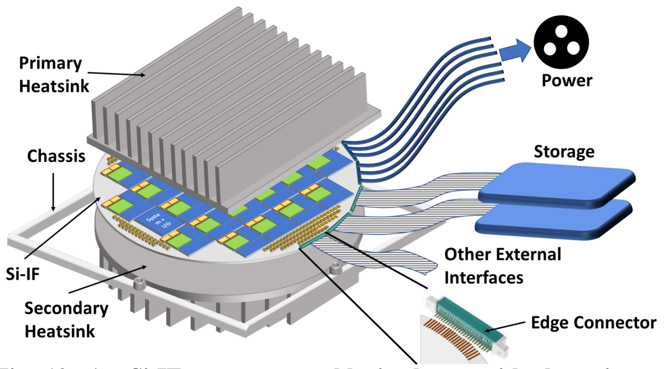


Fig. 10: An Si-IF system assembly is shown with the primary and backside secondary heat sinks. The whole system is bolted to a chassis. The host CPU could either be connected externally or reside on the wafer itself.

V. Thread Block Scheduling and Data Placement

Performance and EDP of a waferscale GPU architecture would also depend upon how compute and data is distributed across the waferscale system. Conventionally, thread blocks (TB) in a GPU during kernel execution are dispatched by a centralized controller to the compute units (CUs) in a round-robin order based on CU availability. However, such a fine-grained scheduling policy could place TBs of a kernel across multiple GPMs. Often, consecutive TBs benefit from data locality, and therefore such a policy could destroy the performance and energy benefits of waferscale integration as a large number of memory accesses would now need to be made across the inter-GPM network. Therefore, we use distributed scheduling instead of centralized scheduling.

In this distributed policy, a group of contiguous TBs of a kernel are assigned to each GPU so that spatial data locality between TBs can be utilized. Such a policy was used for the MCM-GPU presented in [34]. Consecutive groups of TBs were placed on the GPM array starting from a corner GPM and moving row first. Data placement is first-touch, i.e., when the first memory access to a particular page is done, the page is moved to the local DRAM of the GPM from which the memory reference was made. However, strong spatial data locality may still exist between non-neighboring TBs which cannot be exploited by this online policy. In a waferscale GPU with a large number of GPMs, because communication between non-neighboring GPMs can result in high multi-hop latency, this can lead to poor performance. Therefore, we need policies that allow TBs which share a large amount of data to be placed on neighboring GPMs so as to minimize data access latency as well as total network bandwidth utilization.

To solve this problem, we developed an offline partitioning and placement framework where the goal is to find the

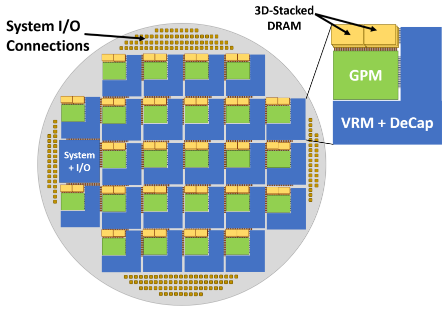


Fig. 11: Waferscale GPU with 25 GPM units (1 redundant unit) comprising of two 3D-stacked DRAM per unit, VRM unit and decoupling capacitors.

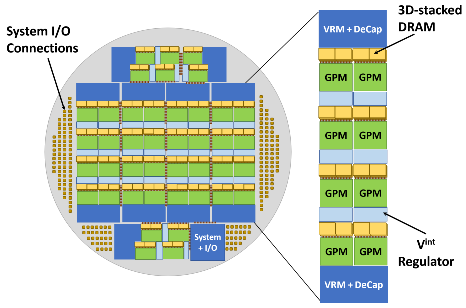


Fig. 12: Waferscale GPU with 42 GPM units (2 redundant units) comprising of two 3D-stacked DRAM per unit, VRM unit and decoupling capacitors.

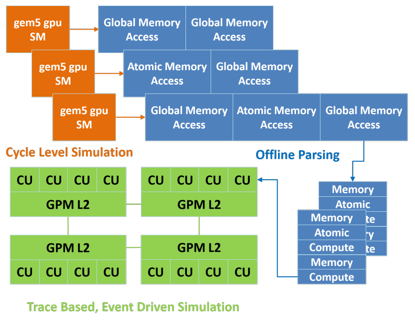


Fig. 13: Simulator Workflow

schedule and GPM allocation for TBs and DRAM pages that minimizes remote memory accesses. Our framework is fully automated and takes in a TB - DRAM page (TB-DP) access graph as input and outputs TB to GPM mapping as well as data placement (shown in Figure 15). Nodes in the TB-DP access graph represents either a TB or a DRAM page, and an edge between a TB and a DRAM page signifies that the particular TB accesses the DRAM page. The edge weight corresponds to the total number of accesses. Given this graph, the aim is to partition the graph in to k partitions such that the total weight of the edges crossing the partition boundaries is minimized. We solve this partitioning problem using an iterative form of Fiduccia-Matthessey (FM) partitioning algorithm [75] where in each iteration of the algorithm, we extract one partition with N/k nodes. In our implementation, we allow the size ratio to drift by up to $\pm 2\%$ to minimize partition cut further. This generates a TB schedule as well as the corresponding data placement for an application which minimizes data accesses across the partitions. Partitioning, however doesn't solve the problem of minimizing overall load on the inter-GPM network and if a few but very remote (many-hops) accesses still remain, latency issues can affect the overall performance and energy efficiency. Therefore, given the network topology and number of GPMs, the next step is to allocate the TB-DP clusters to these GPMs, this is the *cluster placement problem*.

For the placement problem, we consider minimization of a *remote access cost* metric which is the summation of product of number of accesses and distance between the source and destination of the access. For example, let's consider a grid of 5x5 GPMs and 5 accesses made between GPMs at locations (1,1) and (3,5). The minimum Manhattan hop distance between the locations is 6 hops. Therefore the cost we consider is 30. The total cost is indicative of the total bandwidth utilization of the inter-GPM network and minimizing number of hops essentially minimizes latency of accesses. We use simulated annealing based placement to map the clusters to the appropriate GPMs in the GPM array. In Figure 14, we compare this cost for network topologies with baseline runtime dynamic scheduling and first touch page placement (discussed below) for the 40 GPM system. Our offline policy reduces the cost of accesses by up to 57%.

Note that, the partitioning and placement policy has been driven by spatial access patterns. A policy based on spatio-temporal access patterns would be able to provide better optimizations but we leave it for future work. Moreover, in this partitioning scheme, we didn't consider load balancing explicitly. The partitioning algorithm divides the TB-DP graph into k nearly equal partitions, therefore this scheme does not necessarily load balance the TBs. We therefore, also use a runtime load balancing scheme on top of the static partitioning, where if there are TBs waiting (to be allocated to a CU) in the queue in a GPM and there are idle GPM(s), the queued TBs are migrated to the nearest idle GPM.

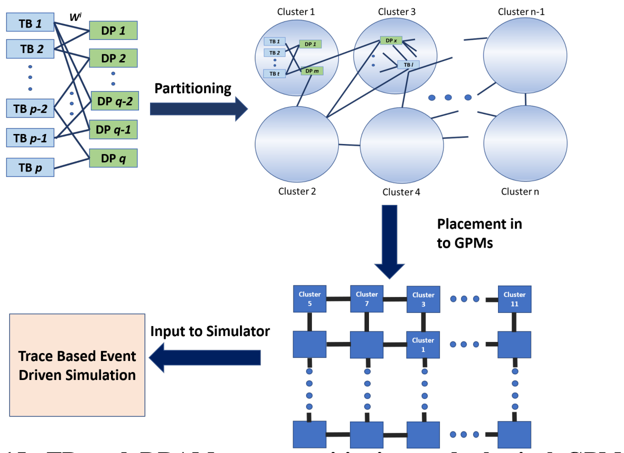


Fig. 15: TB and DRAM page partitioning and physical GPM mapping flow

Other Policies: We also evaluated an online locality aware placement policy where the first group of TBs is placed in the centre GPM and the subsequent groups are assigned to the GPMs spirally out of the central GPM. This policy showed performance within $\pm 3\%$ compared to the simple placement policy of starting from a corner GPM and moving row first. For offline policies, we considered other access cost metrics such as summation of $\#access^2 * hop$ (this allows the most connected TB clusters to be placed closest) and $hop^2 * \#access$ (this minimizes maximum latency of data accesses). However, placements generated from these metrics have 2% poorer performance on average compared to the $\#access * hop$ metric, except 7% benefit when using $\#access * hop^2$ on 24-GPU system for *color* which is an irregular application and is network latency bound. Therefore, in Section VII, we only discuss and analyze the online policy with simple placement and offline policy with $data * hop$ metric for cluster placement.

TABLE IX: Benchmarks

Benchmark	Suite	Domain
backprop	Rodinia	Machine Learning
hotspot	Rodinia	Physics Simulation
lud	Rodinia	Linear Algebra
particlefilter naive	Rodinia	Medical Imaging
srad	Rodinia	Medical Imaging
color	Pannotia	Graph Coloring
bc	Pannotia	Social Media

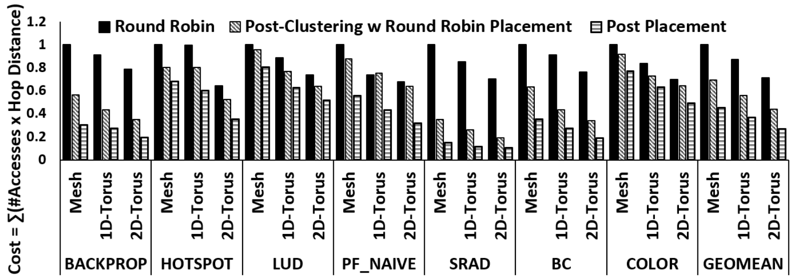


Fig. 14: Improvement in the access cost metric from offline partitioning and GPM placement is shown. Baseline is data locality aware distributed scheduling and first touch data placement

VI. Methodology

Current architectural simulators such as gem5 [76] and GPGPU-Sim [77] were designed to study systems with at most dozen of compute units: they are simply unable to simulate waferscale processors in a reasonable time-frame. In order to study the scalability of GPU applications on a waferscale processor, a more abstract simulation model is necessary.

Figure 13 shows our simulation methodology. The first step in modelling a waferscale GPU is to collect memory traces from which to extract the application behavior. We create an 8 compute unit (CU) GPU in gem5-gpu Syscall Emulation (SE) mode, placing a memory probe on the LSQ (Load-Store Queue) of each CU. Next, we run each benchmark in fast-forward through Linux boot until the beginning of the ROI (region of interest), where we take a checkpoint[11]. Finally, we run the entire ROI of the application in detailed mode, collecting a memory trace of every global read, write, and atomic operation and their associated threadblock. The files are fed into our trace-based simulator.

The trace-based simulator first parses the traces, gathering the relative timing, virtual address, type of operation (read/write/atomic) and size of the memory access, maintaining the block id of the access, but clearing any affinity to a particular compute unit. Private compute time is estimated as the time spent between non-consecutive memory accesses multiplied by the duty cycle of the compute unit which originally executed the request. Note that this private compute time is not simply raw computation, but also includes accessing block shared memory. In the view of the simulator, there is no difference between the two operations. These compute requests are grouped along with global memory accesses into thread blocks. When executing a thread block, compute requests must conservatively wait until all outstanding memory requests have completed. Conversely, new memory requests must wait until there are no outstanding compute requests to proceed. This assumption is based on the in-order execution of warps within a thread block. In reality, the local warp scheduler will overlap computation and memory accesses by switching out warps upon cache misses.

We validated our trace-based simulator against gem5-gpu for a small number of compute units. Figure 16 compares normalized performance across the two simulators for different number of CUs.[12] Figure 17 compares normalized performance across the two simulators for different DRAM bandwidth values. We observe a geometric mean of 5% and maximum error of 28% across the two simulators when number of CUs is scaled[13]. We observe a geometric mean of 7% and maximum error of 26% across the two simulators when DRAM bandwidth is scaled. The results suggest the validity of our simulation approach (relative to gem5-gpu).

As an additional validation step, we created roofline plots [78] as visual representations of the interplay of bandwidth, data locality and computation resources in the two simulators. Visual inspection of Figure 18, roofline plots of an 8 CU system, show the same general characteristics and application positioning between gem5-gpu and our trace-based simulator. This builds further confidence that application characteristics, such as compute to memory ratio, data locality and bandwidth bottlenecks are preserved in our trace-based methodology.

For the results in Section VII, we evaluate a 24 GPM waferscale GPU running at 575 MHz and a 40 GPM waferscale GPU running at 408.2 MHz, following from the VFS scaling explored in Section IV. We compare against a single MCM-GPU, a 6 package, 24 GPM MCM-GPU and a 10 package, 40 GPM MCM-GPU. Our baseline scheduler is dynamic scheduling similar to the one proposed in [34], [79], i.e., round-robin within GPM first and first touch page placement, where a page is mapped to the GPM from which it is first accessed. Our systems are evaluated on the five benchmarks from Rodinia [53] and two irregular workloads from Pannotia suite [33] which we could run successfully on gem5-gpu (Table IX). We use a mesh network topology for the waferscale inter-GPM networks and for scale-out MCM-GPU, we consider the MCM packages interconnected using an on-board mesh network.

VII. Results

To quantify performance and energy benefits of waferscale GPU architectures, we compare the benefits of our proposed waferscale architectures against 24 and 40 GPM systems built using multiple MCM-GPUs where each MCM-GPU houses 4 GPMs, alongside the local DRAMs. The MCM-GPUs are assumed to be integrated using conventional PCB-based integration technology. Comparisons are done based on two scheduling and data-placement policies. One is online distributed scheduling of thread group (round robin within

[11]For each application, the ROI is a single contiguous code section, run with a large enough input size such that the trace produces around 20,000 threadblocks.

[12]We were unable to generate validation data for *bc* and *color* as the workload datasets were too large to finish on our gem5-GPU setup

[13]Coherency, operating system effects, threadblock scheduling runtime and sophisticated memory coalescing schemes are not modelled accurately in the trace simulator leading to errors. Also, optimizations such as using local shared memory or warp scheduling techniques to minimize memory latency are not modelled.

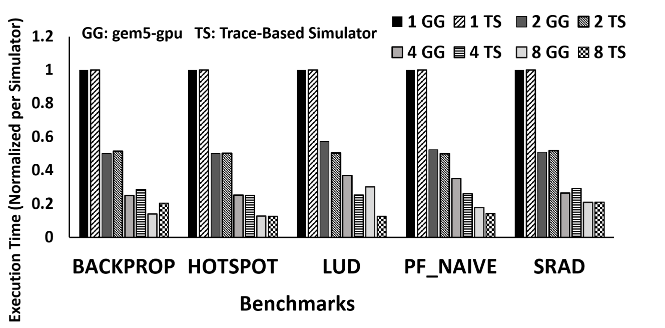


Fig. 16: CU Scaling for Gem5-GPU and Trace-based Simulator

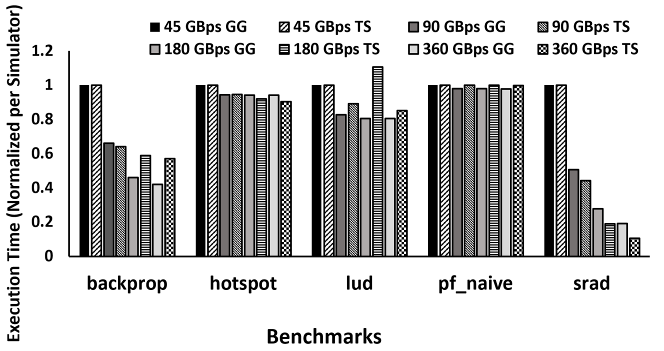


Fig. 17: DRAM Bandwidth Scaling for Gem5-GPU and Trace-based Simulator for 8 CUs

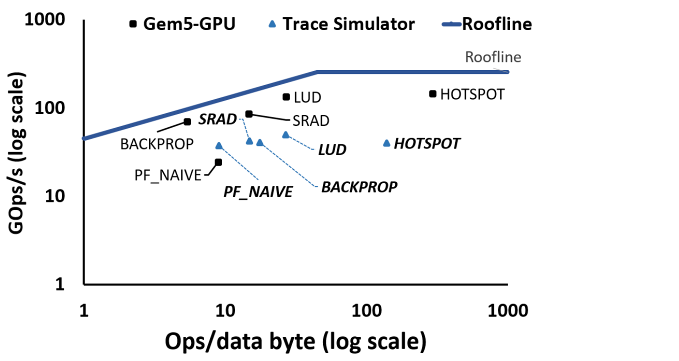


Fig. 18: Roofline Plot Comparison between Gem5-GPU and our Trace Simulator

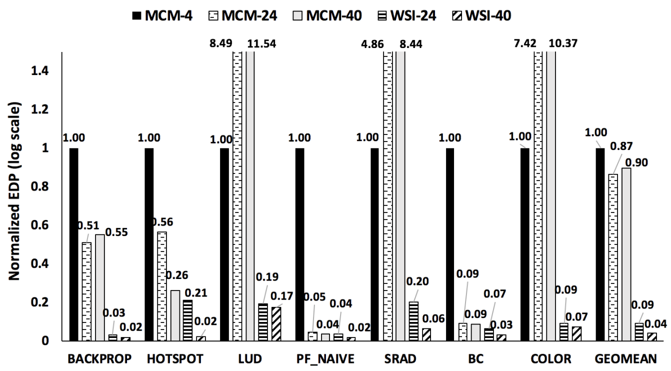


Fig. 20: EDPs for waferscale GPUs vs MCM package based conventional systems

GPM first) coupled with first touch page placement as proposed in [34] (RR-FT) and the other is our offline partitioning and placement approach (MC-DP) (Section V).

We first compare the results using MC-DP. Figures 19 and 20 show comparative raw performance and EDP results for different configurations across a variety of benchmarks when using MC-DP. We see that applications such as *backprop, hotspot and particlefilter_naive* show performance benefit, up to 4.9x and 6.1x speedup, from 24-GPM and 40-GPM systems respectively implemented using MCM-GPU over a single MCM-GPU (4-GPM) . However, applications like *lud and color* show performance degradation when run on either MCM-24 or MCM-40 systems. This is because, large memory footprint and irregular access patterns of these applications cause significant inter-MCM data transfers which ultimately dominate performance scaling.

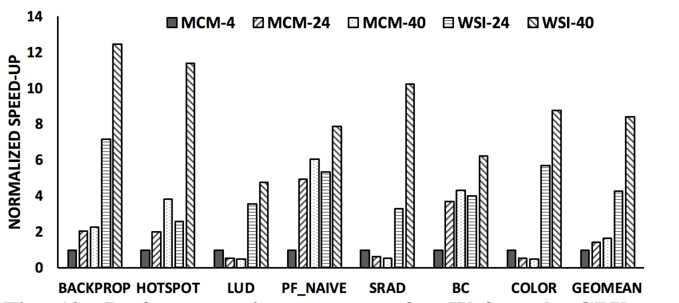


Fig. 19: Performance improvement for Waferscale GPUs vs MCM package based conventional systems

While scheduling on multi-MCM GPUs was first performed within a GPM in a round robin fashion, larger kernels were still split and spread across multiple MCM modules for load balancing. Also, as the number of MCM modules in the network increases, percentage of multi-hop communication increases, which further gets bottle-necked by the inter-MCM on-board bandwidth.

On the other hand, all the applications see substantial speedup from both 24-GPM and 40-GPM waferscale GPU architectures, up to 10.9x (2.97x, on average) and 18.9x (5.2x, on average) respectively when compared to comparable MCM-GPU based architectures. Such large speedups are because of the very high inter-GPM bandwidth available across the entire wafer. Similarly, average EDP benefit of 9.3x and 22.5x can be obtained from waferscale integration. This is because of the major reduction in execution time as well as 10x better energy efficiency of inter-GPM communication.

Moving on to the RR-FT policy, it is seen that the performance gap between waferscale systems and MCM-GPU based systems is about 2x higher when using RR-FT over MC-DP. This reduction in performance gap between MCM-GPU and waferscale system when going from RR-FT to MC-DP means that scale-out MCM-GPU based systems benefit more than waferscale based systems when using MC-DP. This is because in MCM-GPU based systems, the cost of inter-MCM on-board communication is much higher than that in waferscale systems. As a result, our offline policy, which helps to reduce this communication through intelligent scheduling and data placement, brings significant improvement in the performance of the scale-out MCM-GPU systems. Thus, our offline policy is well suited for scale-out MCM-GPU systems as well.

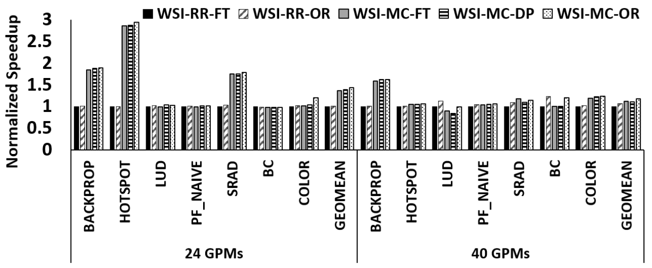


Fig. 21: Performance of different scheduling and data placement policies

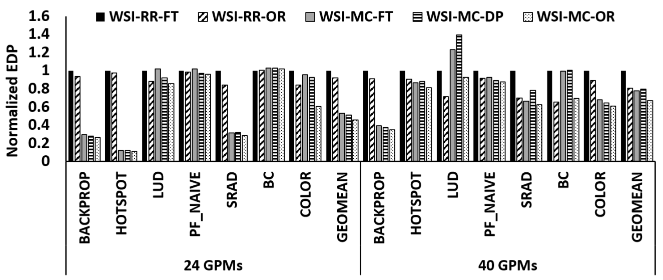


Fig. 22: EDP of different scheduling and data placement policies

We now perform a more detailed comparison of our offline partitioning and placement approach with the baseline locality-aware distributed scheduling policy for waferscale based systems. We evaluate the baseline policy with a realistic first touch page placement (RR-FT policy) as well as with an oracular data placement (RR-OR) that guarantees either that the remote accesses have no overhead or that there are no remote accesses. We simulated RR-OR by putting all DRAM pages in all the GPMs' local DRAM in our simulations. Similarly, we considered three variants of our offline approach. In the first case (MC-FT), only the thread block schedules were used from the offline partitioning results alongside a first-touch page placement policy. In the second case (MC-DP), we considered the data placement output from our partitioning and placement framework. In the third case (MC-OR), we considered the maximum speedup possible using these thread block schedules, i.e., again all DRAM pages were placed in all the GPMs' local DRAM.

As shown in Figure 21, we observe that a runtime RR-FT policy performs on an average 7% worse compared to RR-OR policy; this is similar to the observation made in [34] in context of MCM-GPUs. We also observe that our partitioning and GPM placement policy significantly outperforms RR based policies (both FT and oracular). For applications such as *backprop, hotspot, srad*, large performance benefits of up to 2.88x are attainable over RR-FT for the 24 GPM case. Benefits are upto 1.62x in 40 GPM case. Such large benefits come from the fact that our partitioning scheme clusters together the thread blocks which touch the same DRAM pages. This minimizes the total number of remote accesses made; this also makes caching more effective because data locality within GPM is strongly preserved and therefore cache hit rate increases. Overall, on an average, our offline policy (MC-DP) results in 1.4x (24 GPMs) and 1.11x (40 GPMs) benefits over the baseline RR-FT policy and comes within 16% of maximum speed up possible (MC-OR). This results in an average EDP benefit of 49% and 20% respectively for 24 and 40 GPM systems respectively (shown in Figure 22).This also indicates that online scheduling optimizations based on programmer and compiler hints [80], [81] can help achieve better performance on WS-GPU than the RR-FT online policy. Note that the relative benefit of MC-DP over RR-FT is smaller for the 40 GPM system compared to 24 GPM system. This can be attributed to the fact that TBs get distributed across more number of GPMs as the system size grows and therefore the benefit of caching decreases.

As mentioned before, we use 575 MHz as the operating frequency of the GPMs. At higher GPM frequencies, WSI-GPU benefits increase since communication is more of a bottleneck when the GPMs are running faster as frequency of data access requests increase. E.g., on average, WS-GPU-24 outperforms MCM-24 by an additional 7% at 1GHz vs 575MHz.

For the topology with 40 GPMs, we also considered non-stacked configuration. In a non-stacked configuration, GPMs need to be run at even lower voltage and frequency (0.71V/360MHz). The resulting system has 14% lower performance on average compared to the stacked configuration.

Finally, our thermal analysis assumed efficient forced air cooling model. Liquid or phase change cooling solutions can increase the sustainable TDP, enabling even higher compute density [82]. We estimate that a 2X increase thermal budget from liquid cooling can increase performance of WS-GPU-40 by additional 20-30% compared to baseline MCM-40.

VIII. Summary and Conclusion

Waferscale processors can dramatically reduce communication overheads. However, they have had unacceptable yield. Emerging integration technologies such as a Silicon-Interconnection Fabric (Si-IF)-based integration [5], [6], where pre-manufactured dies are directly bonded on to a silicon wafer, may enable waferscale processors without the corresponding yield issues. Therefore, time is ripe to revisit waferscale architectures. In this paper, we showed that it is feasible and useful to architect a modern day waferscale system. Using a waferscale GPU as a case study, we showed that while a 300 mm wafer can house about 100 GPU modules (GPM), only a much scaled down GPU architecture with about 40 GPMs can be built when physical concerns are considered. We also studied the performance and energy implications of waferscale architectures. We showed that waferscale GPUs can provide significant performance and energy efficiency advantages (up to 18.9x speedup and 143x EDP benefit compared against equivalent MCM-GPU based implementation on PCB) without any change in the programming model. We also developed thread scheduling and data placement policies for waferscale GPU architectures. Our policies outperformed state-of-art scheduling and data placement policies by 2.88x (average 1.4x) and 1.62x (average 1.11x) for 24 GPM and 40 GPM cases respectively. Finally, we built the first Si-IF prototype with interconnected dies - the 100% yield we observe for our prototype, coupled with high bond yield reported for previous Si-IF prorotypes, demonstrates the technological readiness for building waferscale GPU architecture.

IX. Acknowledgement

This work was supported in part by the Defense Advanced Research Projects Agency (DARPA) through ONR grant N00014-16-1-263, UCOP through grant MRP-17-454999, Futurewei Technologies, Samsung Electronics, and the UCLA CHIPS Consortium. The authors would like to thank SivaChandra Jangam and Adeel A. Bajwa for helping with the Si-IF prototype in the paper, and the anonymous reviewers for their helpful feedback and suggestions.

References

[1] "Trilogy Systems." https://en.wikipedia.org/wiki/Trilogy_Systems, (accessed Nov 20, 2017).

[2] R. M. Lea, "WASP: a wafer-scale massively parallel processor," in *1990 Proceedings. International Conference on Wafer Scale Integration*, pp. 36–42, Jan 1990.

[3] D. Landis and J. Yoder and D. Whittaker and T. Dobbins, "A wafer scale programmable systolic data processor," in *Proceedings Ninth Biennial University/Government/Industry Microelectronics Symposium*, pp. 252–256, Jun 1991.

[4] J. F. McDonald, E. H. Rogers, K. Rose, and A. J. Steckl, "The trials of wafer-scale integration: Although major technical problems have been overcome since WSI was first tried in the 1960s, commercial companies can't yet make it fly," *IEEE Spectrum*, vol. 21, pp. 32–39, Oct 1984.

[5] A. Bajwa and S. Jangam and S. Pal and N. Marathe and T. Bai and T. Fukushima and M. Goorsky and S. S. Iyer, "Heterogeneous Integration at Fine Pitch ($\leq 10 \mu$m) Using Thermal Compression Bonding," in *2017 IEEE 67th Electronic Components and Technology Conference (ECTC)*, pp. 1276–1284, May 2017.

[6] S. Jangam and S. Pal and A. Bajwa and S. Pamarti and P. Gupta and S. S. Iyer, "Latency, Bandwidth and Power Benefits of the SuperCHIPS Integration Scheme," in *2017 IEEE 67th Electronic Components and Technology Conference (ECTC)*, pp. 86–94, May 2017.

[7] S. Pal, D. Petrisko, A. A. Bajwa, P. Gupta, S. S. Iyer, and R. Kumar, "A case for packageless processors," in *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pp. 466–479, Feb 2018.

[8] M. Panahiazar and V. Taslimitehrani and A. Jadhav and J. Pathak, "Empowering personalized medicine with big data and semantic web technology: Promises, challenges, and use cases," in *2014 IEEE International Conference on Big Data (Big Data)*, pp. 790–795, Oct 2014.

[9] "NVIDIA Powers Virtual Reality." http://www.nvidia.com/object/virtual-reality.html, (accessed Nov 21, 2017).

[10] "NVIDIA Tesla Imaging and Computer Vision." http://www.nvidia.com/object/imaging_comp_vision.html, (accessed Nov 21, 2017).

[11] Karen Simonyan and Andrew Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *CoRR*, vol. abs/1409.1556, 2014.

[12] "Nvidia says its new supercomputer will enable the highest level of automated driving." https://www.theverge.com/2017/10/10/16449416/nvidia-pegasus-self-driving-car-ai-robotaxi, (accessed Nov 21, 2017).

[13] "An Introduction to High Performance Computing on AWS (White paper)." https://d0.awsstatic.com/whitepapers/Intro_to_HPC_on_AWS.pdf, (accessed Nov 21, 2017).

[14] "Microsoft Azure," (accessed Nov 21, 2017).

[15] "TOP500 Shows Growing Momentum for Accelerators." https://insidehpc.com/2015/11/top500-shows-growing-momentum-for-accelerators/, (accessed Nov 21, 2017).

[16] "HPC Application Support for GPU Computing." https://insidehpc.com/2015/11/top500-shows-growing-momentum-for-accelerators/, (accessed Nov 21, 2017).

[17] T. Vijayaraghavan and Y. Eckert and G. H. Loh and M. J. Schulte and M. Ignatowski and B. M. Beckmann and W. C. Brantley and J. L. Greathouse and W. Huang and A. Karunanithi and O. Kayiran and M. Meswani and I. Paul and M. Poremba and S. Raasch and S. K. Reinhardt and 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)*, pp. 85–96, Feb 2017.

[18] "Sandy Bridge (client) - Microarchitectures - Intel (Wikichip)." https://en.wikichip.org/wiki/intel/microarchitectures/sandy_bridge_(client), (accessed Nov 20, 2017).

[19] S. S. Iyer, "Heterogeneous Integration for Performance and Scaling," *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 6, pp. 973–982, July 2016.

[20] R. Mahajan and R. Sankman and N. Patel and D. W. Kim and K. Aygun and Z. Qian and Y. Mekonnen and I. Salama and S. Sharan and D. Iyengar and D. Mallik, "Embedded Multi-die Interconnect Bridge (EMIB) – A High Density, High Bandwidth Packaging Interconnect," in *2016 IEEE 66th Electronic Components and Technology Conference (ECTC)*, pp. 557–565, May 2016.

[21] J. W. Poulton and W. J. Dally and X. Chen and J. G. Eyles and T. H. Greer and S. G. Tell and J. M. Wilson and C. T. Gray, "A 0.54 pJ/b 20 Gb/s Ground-Referenced Single-Ended Short-Reach Serial Link in 28 nm CMOS for Advanced Packaging Applications," *IEEE Journal of Solid-State Circuits*, vol. 48, pp. 3206–3218, Dec 2013.

[22] Abts, Dennis and Marty, Michael R. and Wells, Philip M. and Klausler, Peter and Liu, Hong, "Energy Proportional Datacenter Networks," in *Proceedings of the 37th Annual International Symposium on Computer Architecture*, ISCA '10, (New York, NY, USA), pp. 338–347, ACM, 2010.

[23] Y. Huang and Y. Yesha and M. Halem and Y. Yesha and S. Zhou, "YinMem: A distributed parallel indexed in-memory computation system for large scale data analytics," in *2016 IEEE International Conference on Big Data (Big Data)*, pp. 214–222, Dec 2016.

[24] 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)*, pp. 105–117, June 2015.

[25] E. Azarkhish and C. Pfister and D. Rossi and 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, pp. 210–223, Jan 2017.

[26] K. Murthy and J. Mellor-Crummey, "Communication avoiding algorithms: Analysis and code generation for parallel systems," in *2015 International Conference on Parallel Architecture and Compilation (PACT)*, pp. 150–162, Oct 2015.

[27] T. G. Lenihan, L. Matthew, and E. J. Vardaman, "Developments in 2.5D: The role of silicon interposers," in *2013 IEEE 15th Electronics Packaging Technology Conference (EPTC 2013)*, pp. 53–55, Dec 2013.

[28] V. Kumar and A. Naeemi, "An overview of 3d integrated circuits," in *2017 IEEE MTT-S International Conference on Numerical Electromagnetic and Multiphysics Modeling and Optimization for RF, Microwave, and Terahertz Applications (NEMO)*, pp. 311–313, May 2017.

[29] A. Sodani, R. Gramunt, J. Corbal, H. S. Kim, K. Vinod, S. Chinthamani, S. Hutsell, R. Agarwal, and Y. C. Liu, "Knights Landing: Second-Generation Intel Xeon Phi Product," *IEEE Micro*, vol. 36, pp. 34–46, Mar 2016.

[30] James B. Brinton and J. Robert Lineback, *Packaging is becoming biggest cost in assembly, passing capital equipment*. EE Times [Online], 1999.

[31] L. D. Cioccio, P. Gueguen, R. Taibi, T. Signamarcheix, L. Bally, L. Vandroux, M. Zussy, S. Verrun, J. Dechamp, P. Leduc, M. Assous, D. Bouchu, F. de Crecy, L. Chapelon, and L. Clavelier, "An innovative die to wafer 3d integration scheme: Die to wafer oxide or copper direct bonding with planarised oxide inter-die filling," in *2009 IEEE International Conference on 3D System Integration*, pp. 1–4, Sept 2009.

[32] A. Sigl, S. Pargfrieder, C. Pichler, C. Scheiring, and P. Kettner, "Advanced chip to wafer bonding: A flip chip to wafer bonding technology for high volume 3dic production providing lowest cost of ownership," in *2009 International Conference on Electronic Packaging Technology High Density Packaging*, pp. 932–936, Aug 2009.

[33] S. Che, B. M. Beckmann, S. K. Reinhardt, and K. Skadron, "Pannotia: Understanding irregular gpgpu graph applications," in *2013 IEEE International Symposium on Workload Characterization (IISWC)*, pp. 185–195, Sept 2013.

[34] Arunkumar, Akhil and Bolotin, Evgeny and Cho, Benjamin and Milic, Ugljesa and Ebrahimi, Eiman and Villa, Oreste and Jaleel, Aamer and Wu, Carole-Jean and Nellans, David, "MCM-GPU: Multi-Chip-Module GPUs for Continued Performance Scalability," in *Proceedings of the 44th Annual International Symposium on Computer Architecture*, ISCA '17, (New York, NY, USA), pp. 320–332, ACM, 2017.

[35] Y. L. Chuang and C. S. Yuan and J. J. Chen and C. F. Chen and C. S. Yang and W. P. Changchien and C. C. C. Liu and F. Lee, "Unified methodology for heterogeneous integration with CoWoS technology," in *2013 IEEE 63rd Electronic Components and Technology Conference*, pp. 852–859, May 2013.

[36] C. L. Lai and H. Y. Li and A. Chen and T. Lu, "Silicon Interposer Warpage Study for 2.5D IC without TSV Utilizing Glass Carrier CTE and Passivation Thickness Tuning," in *2016 IEEE 66th Electronic Components and Technology Conference (ECTC)*, pp. 310–315, May 2016.

[37] Ying-Wen Chou and Po-Yuan Chen and Mincent Lee and C. W. Wu, "Cost modeling and analysis for interposer-based three-dimensional IC," in *2012 IEEE 30th VLSI Test Symposium (VTS)*, pp. 108–113, April 2012.

[38] John Hu, "System level co-cptimizations of 2.5D/3D hybrid integration for high performance computing system," in *Semicon West 2016*, 2016.

[39] R. Arnold and S. M. Menon and B. Brackett and R. Richmond, "Test methods used to produce highly reliable known good die (KGD)," in *Proceedings. 1998 International Conference on Multichip Modules and High Density Packaging (Cat. No.98EX154)*, pp. 374–382, Apr 1998.

[40] H. D. Thacker and M. S. Bakir and D. C. Keezer and K. P. Martin and J. D. Meindl, "Compliant probe substrates for testing high pin-count chip scale packages," in *52nd Electronic Components and Technology Conference 2002. (Cat. No.02CH37345)*, pp. 1188–1193, 2002.

[41] E. Wachter, A. Erichsen, A. Amory, and F. Moraes, "Topology-agnostic fault-tolerant noc routing method," in *Proceedings of the Conference on Design, Automation and Test in Europe*, DATE '13, (San Jose, CA, USA), pp. 1595–1600, EDA Consortium, 2013.

[42] D. Fick, A. DeOrio, G. Chen, V. Bertacco, D. Sylvester, and D. Blaauw, "A highly resilient routing algorithm for fault-tolerant nocs," in *2009 Design, Automation Test in Europe Conference Exhibition*, pp. 21–26, April 2009.

[43] ITRS, "Yield enhancement," 2007 Edition.

[44] A. Kannan and 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)*, pp. 546–558, Dec 2015.

[45] I. Tirkel, "Yield Learning Curve Models in Semiconductor Manufacturing," *IEEE Transactions on Semiconductor Manufacturing*, vol. 26, pp. 564–571, Nov 2013.

[46] J. Wang and S. Yalamanchili, "Characterization and analysis of dynamic parallelism in unstructured gpu applications," in *2014 IEEE International Symposium on Workload Characterization (IISWC)*, pp. 51–60, Oct 2014.

[47] J. Sim, A. Dasgupta, H. Kim, and R. Vuduc, "A performance analysis framework for identifying potential benefits in gpgpu applications," in *Proceedings of the 17th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, PPoPP '12, (New York, NY, USA), pp. 11–22, ACM, 2012.

[48] NVIDIA, "NVIDIA Updates GPU Roadmap; Announces Pascal," (2015).

[49] "Blue Waters." http://www.ncsa.illinois.edu/enabling/bluewaters, (accessed July 1, 2018).

[50] "GPU Supercomputing in Blue Waters." http://developer.download.nvidia.com/compute/academia/whitepapers/Achievement2013_UIUC.pdf, (accessed July 1, 2018).

[51] "Workload Analysis of Blue Waters." https://arxiv.org/ftp/arxiv/papers/1703/1703.00924.pdf, (accessed July 1, 2018).

[52] M. Viñas, Z. Bozkus, and B. B. Fraguela, "Exploiting heterogeneous parallelism with the heterogeneous programming library," *J. Parallel Distrib. Comput.*, vol. 73, pp. 1627–1638, Dec. 2013.

[53] Che, Shuai and Boyer, Michael and Meng, Jiayuan and Tarjan, David and Sheaffer, Jeremy W and Lee, Sang-Ha and Skadron, Kevin, "Rodinia: A benchmark suite for heterogeneous computing," in *Workload Characterization, 2009. IISWC 2009. IEEE International Symposium on*, pp. 44–54, Ieee, 2009.

[54] J. Power, J. Hestness, M. S. Orr, M. D. Hill, and D. A. Wood, "gem5-gpu: A heterogeneous cpu-gpu simulator," *IEEE Computer Architecture Letters*, vol. 14, pp. 34–36, Jan 2015.

[55] "R-tools 3-d heat sink thermal modelling." http://www.r-tools.com/.

[56] 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, pp. 501–513, May 2006.

[57] J. Liu, B. Jaiyen, R. Veras, and O. Mutlu, "Raidr: Retention-aware intelligent dram refresh," in *Proceedings of the 39th Annual International Symposium on Computer Architecture*, ISCA '12, (Washington, DC, USA), pp. 1–12, IEEE Computer Society, 2012.

[58] "NVIDIA GPU maximum operating temperature and overheating." https://http://nvidia.custhelp.com/app/answers/detail/a_id/2752/~/nvidia-gpu-maximum-operating-temperature-and-overheating, (accessed Nov 21, 2017).

[59] M. Ahmed and C. Fei and F. C. Lee and Q. Li, "High-efficiency high-power-density 48/1V sigma converter voltage regulator module," in *2017 IEEE Applied Power Electronics Conference and Exposition (APEC)*, pp. 2207–2212, March 2017.

[60] V. Kontorinis, A. Shayan, D. M. Tullsen, and R. Kumar, "Reducing peak power with a table-driven adaptive processor core," in *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO 42, (New York, NY, USA), pp. 189–200, ACM, 2009.

[61] Intel Corp., *Intel Pentium 4 Processor in the 423-pin Package Thermal Design Guidelines*, November 2000.

[62] F. C. Lee and Q. Li and Z. Liu and Y. Yang and C. Fei and M. Mu, "Application of GaN devices for 1 kW server power supply with integrated magnetics," *CPSS Transactions on Power Electronics and Applications*, vol. 1, pp. 3–12, Dec 2016.

[63] "Datacenter Power Delivery Architectures:Efficiency and Annual Operating Costs ." http://www.vicorpower.com/documents/whitepapers/server_efficiency_vichip.pdf.

[64] "Determining Total Cost of Ownership for Data Center and Network Room Infrastructure ." http://www.apc.com/salestools/CMRP-5T9PQG/CMRP-5T9PQG_R4_EN.pdf.

[65] P. Gupta and A. B. Kahng, "Efficient design and analysis of robust power distribution meshes," in *19th International Conference on VLSI Design (VLSID'06)*, pp. 6 pp.–, Jan 2006.

[66] M. H. Ahmed and C. Fei and F. C. Lee and Q. Li, "48-V Voltage Regulator Module With PCB Winding Matrix Transformer for Future Data Centers," *IEEE Transactions on Industrial Electronics*, vol. 64, pp. 9302–9310, Dec 2017.

[67] J. Leng and Y. Zu and V. J. Reddi, "GPU voltage noise: Characterization and hierarchical smoothing of spatial and temporal voltage noise interference in GPU architectures," in *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)*, pp. 161–173, Feb 2015.

[68] S. K. Lee, T. Tong, X. Zhang, D. Brooks, and G. Y. Wei, "A 16-core voltage-stacked system with adaptive clocking and an integrated switched-capacitor dc x2013;dc converter," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, pp. 1271–1284, April 2017.

[69] K. Mazumdar and M. Stan, "Breaking the power delivery wall using voltage stacking," in *Proceedings of the Great Lakes Symposium on VLSI*, GLSVLSI '12, (New York, NY, USA), pp. 51–54, ACM, 2012.

[70] Q. Zhang and L. Lai and M. Gottscho and P. Gupta, "Multi-story power distribution networks for GPUs," in *2016 Design, Automation Test in Europe Conference Exhibition (DATE)*, pp. 451–456, March 2016.

[71] E. Alon and M. Horowitz, "Integrated regulation for energy-efficient digital circuits," vol. 43, pp. 1795 – 1807, 09 2008.

[72] E. Papadopoulou and D. T. Lee, "Critical area computation via Voronoi diagrams," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 18, pp. 463–474, Apr 1999.

[73] N. Chatterjee and M. O'Connor and D. Lee and D. R. Johnson and S. W. Keckler and M. Rhu and W. J. Dally, "Architecting an Energy-Efficient DRAM System for GPUs," in *2017 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pp. 73–84, Feb 2017.

[74] "JEDEC Publishes HBM2 Specification as Samsung Begins Mass Production of Chips." https://www.anandtech.com/show/9969/jedec-publishes-hbm2-specification.

[75] A. E. Caldwell, A. B. Kahng, and I. L. Markov, "Design and implementation of the fiduccia-mattheyses heuristic for vlsi netlist partitioning," in *Selected Papers from the International Workshop on Algorithm Engineering and Experimentation*, ALENEX '99, (London, UK, UK), pp. 177–193, Springer-Verlag, 1999.

[76] 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," *SIGARCH Comput. Archit. News*, vol. 39, pp. 1–7, Aug. 2011.

[77] A. Bakhoda, G. L. Yuan, W. W. L. Fung, H. Wong, and T. M. Aamodt, "Analyzing cuda workloads using a detailed gpu simulator," in *2009 IEEE International Symposium on Performance Analysis of Systems and Software*, pp. 163–174, April 2009.

[78] S. Williams, A. Waterman, and D. Patterson, "Roofline: An insightful visual performance model for multicore architectures," *Commun. ACM*, vol. 52, pp. 65–76, Apr. 2009.

[79] U. Milic, O. Villa, E. Bolotin, A. Arunkumar, E. Ebrahimi, A. Jaleel, A. Ramirez, and D. Nellans, "Beyond the socket: Numa-aware gpus," in *Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO-50 '17, (New York, NY, USA), pp. 123–135, ACM, 2017.

[80] N. Vijaykumar, E. Ebrahimi, K. Hsieh, P. B. Gibbons, and O. Mutlu, "The locality descriptor: A holistic cross-layer abstraction to express data locality in gpus," in *2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture (ISCA)*, pp. 829–842, June 2018.

[81] J. Wang, N. Rubin, A. Sidelnik, and S. Yalamanchili, "Laperm: Locality aware scheduler for dynamic parallelism on gpus," in *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, pp. 583–595, June 2016.

[82] M. Skach, M. Arora, C. H. Hsu, Q. Li, D. Tullsen, L. Tang, and J. Mars, "Thermal time shifting: Leveraging phase change materials to reduce cooling costs in warehouse-scale computers," in *2015 ACM/IEEE 42nd Annual International Symposium on Computer Architecture (ISCA)*, pp. 439–449, June 2015.