3D GPU Architecture using Cache Stacking: Performance, Cost, Power and Thermal analysis

Ahmed Al Maashri, Guangyu Sun, Xiangyu Dong, Vijay Narayanan and Yuan Xie
Department of Computer Science and Engineering, Penn State University
{maashri, gsun, xydong, vijay, yuanxie}@cse.psu.edu

Abstract—**Graphics Processing Units (GPUs) offer tremendous computational and processing power. The architecture requires high communication bandwidth and lower latency between computation units and caches. 3D die-stacking technology is a promising approach to meet such requirements. To the best of our knowledge no other study has investigated the implementation of 3D technology in GPUs. In this paper, we study the impact of stacking caches using the 3D technology on GPU performance. We also investigate the benefits of using 3D stacked MRAM on GPUs. Our work includes cost, power, and thermal analysis of the proposed architectural designs. Our results show a 53% geometric mean performance speedup for iso-cycle time architectures and about 19% for iso-cost architectures.**

I. INTRODUCTION

In the last few years, commodity Graphics Processing Units (GPUs) have enjoyed a dramatic increase in programmability as well as in computational power, which allowed them not only to make an impact on the game industry, but also to be utilized as coprocessors for general purpose applications [1]. GPU manufacturers are not oblivious to the increasing demand for these units for such a variety of diversified applications, and therefore, more attention was directed towards providing the facilities necessary to utilize GPUs in the most efficient manner. For example, CUDA [2] is a general purpose parallel computing architecture that leverages the parallel compute engine in NVIDIA graphics processing units to solve many complex computational problems.

However, GPUs do have some limitations that would affect the performance. For example, a study has shown the limitations on video memory and floating point buffer sizes prevent GPUs from reaching their theoretical capacities [3]. Another study has shown that the low cache bandwidth in prior GPUs makes matrix-matrix multiplication inefficient [4], something that needs to be investigated and verified in modern GPUs. Also, because of the smaller size of caches compared to CPU caches, a less number of blocks can be brought to the GPU cache [5], this would highly impact hit rate as workloads are becoming more memory/cache intensive. Hence, memory wall is in the arena of GPUs.

A number of studies attempted to tackle these issues [5] - [7] by proposing memory and cache models that are more suitable for scientific applications. However, as applications get more resource demanding, there are still many open questions. For example, how can we mitigate the high latency that is associated with increasing GPU cache sizes? Furthermore, as we increase the computational capabilities of GPUs, there is an increase in power consumption as well as an increase in chip footprint, which is directly related to cost. Unfortunately, very few studies have focused on the links between performance enhancement and the major constraints of cost and power.

3D integration offers a number of benefits including reduced latency in circuits, reduced wires length that results in a reduction in power consumption and a reduction in footprint. 3D technology also enables heterogeneous integration. For example, Loi et. al. [8] investigated the performance of a 3D processor-memory hierarchy, and Puttaswamy et. al [9] discussed the benefits of stacking layers of cache on processors. However, there has been no prior study on the 3D architectures for GPUs.

The focus of this work is to investigate the impact of 3D integration on GPU architectures. This study investigates thoroughly the 3D stacking of different types of caches that are found in a modern GPU with a focus on texture unit caches and Z caches. Our simulation results show that stacking these caches would potentially improve overall performance of the system by a 53% geometric mean speedup. In addition, we study the benefits of implementing these caches using MRAM technology. The exploration of such heterogeneous architectures using mixed CMOS and MRAM technologies becomes feasible with 3D integration.

Our study also evaluates the extra costs associated with 3D integration and proposes two iso-cost models, which show that 3D-enabled GPU design outperforms 2D planar design for the same bare-die cost.

To the best of our knowledge, this work is the first to study and assess the impact of 3D technology on GPUs, and the key contributions include:

- Performance evaluation of 3D stacked caches on GPUs
- Comparison between 3D stacked SRAMs and MRAMs in GPUs in terms of power consumptions
- Cost assessment of 3D GPU implementation
- Power and thermal analysis of proposed architectural designs.

The rest of the paper is organized as follows; Section II presents GPU architecture and an overview of 3D technology. Section III introduces the design methodology followed in this paper, while section IV presents the simulation setup. Section V discusses the simulation results. Finally, we conclude with a summary of the work done and future work.

978-1-4244-5028-2/09/$25.00 ©2009 IEEE

II. OVERVIEW

In this section, we briefly discuss the GPU pipeline and highlight the main components that will be the focus of this work. Then, we take a glance at 3D technology; basics and merits.

A. Modern GPU Architecture

Graphics processors do vary in architectural specifications from one manufacturer to another. Therefore, we describe a generic architecture of a modern GPU.

It all starts when an application invokes certain Application Programming Interfaces that generate a stream of vertices defining the scene geometry. The GPU receives this stream of vertices through the "streamer unit", which has a cache associated with it to capture localities. The streamer then passes these vertices to a pool of shaders. These shaders use some attributes associated with these vertices to apply shading features to these vertices. After that, the shaders pass the shaded vertices to the "Primitive Assembly" stage, where shaded vertices are converted to triangles. These triangles need to be passed to the "Clipper stage" to perform a trivial triangle rejection test. The next stage is the "Triangle Setup", where the triangle edge and depth interpolation equations are calculated. Once these triangles are ready to be converted into fragment "pixels", they are sent to the "Fragment Generator". Depending on the depth complexity of the scene, the "Z and Stencil Test (ZST)" stage removes non-visible fragments thereby reducing the computational load in the fragment shaders. Note that each ZST stage utilizes a cache called "ZST cache". The visible fragments are then passed to the "Interpolator", which uses perspective corrected linear interpolation, to generate the fragment attributes from the triangle attributes. Once again, these interpolated fragment quads (group of 4 fragments) are fed into the shader pool.

The "shader" is a programmable stage that allows the implementation of other stream based algorithms, (i.e. allowing general purpose algorithms to run on top of the shader). A shader consists of a pipeline and a number of "Texture Units" attached to each shader. Texture units support n-dimensional and cubemap textures, mipmapping (*a technique used to increase rendering speed and reduce aliasing artifacts*), bilinear, trilinear, and anisotropic filtering. Each texture unit has a cache associated with it, which we call TU cache for short. The "Color Write" stage comes after that where the fragments are blended and then passed to the frame buffer for display. Each color write unit has its own cache. Figure 1 shows the GPU pipeline architecture assumed in this work.

B. 3D Integration Technology

3D-die stacking technology is an emerging technology that comes with a number of benefits including reduced latency in circuits, reduced wires length that results in a reduction in power consumption and a reduction in footprint. 3D technology also enables heterogeneous integration.

In 3D technology, individual stacked layers consist of substrate, active devices, and multiple layers of metal routing. After fabrication, each wafer is thinned and then Through-Silicon Vias (TSVs) are etched through the substrate. Thermo-compression is then used to bond individual layers together to form the 3D stack. There are a number of techniques for stacking dies of which Wafer-to-Wafer (W2W) and Die-to-Wafer (D2W) techniques are the most common. Unlike W2W, D2W allows for stacking individual dies to another wafer resulting in higher flexibility and higher yield. However, this also results in higher costs because of the additional tests that are required to verify functionality (e.g. Known-Good-Die testing).

III. DESIGN METHODOLOGY

As stated earlier, the objective of this study is to explore the benefits of enabling 3D integration technology in GPUs. We begin by establishing our design space.

A. Design Space Exploration

As observed by Fatahalian et. al [4], caches do pose significant performance limitations in some of the GPU workloads. However, this observation was pertaining to prior GPUs. In this work we attempt to study the impact of increasing the cache bandwidth on the overall system. This increase in bandwidth can be achieved using 3D technology, where the vertical interconnects (i.e. TSVs) are not bounded by 2D topology limitations of using only horizontal interconnects. Stacking caches on GPUs would allow us to utilize TSVs to increase the bandwidth and hence analyze the potential improvement in performance.

Our design space exploration covered the following caches; Streamer caches, TU caches, ZST caches and Color Write caches. Our preliminary simulations show that increasing the bandwidth of all caches will improve the performance by only 1%. These results didn't improve much even when data compression techniques were used. Based on these results, we conclude that the cache bandwidth seems not to be a potential bottleneck to workloads used .

Next, we focus on the organization of the caches. We start by investigating the effects of changing the organization of the GPU caches on the hit rate. Our simulation results show negligible impact on the hit rate for all the caches, except for the TU and the ZST caches, as shown in figures 2 and 3, respectively.

Figure 2 shows that the hit rate in the texture cache has improved by exploiting the temporal locality by increasing the number of sets and/or associativity. On the other hand, Figure 3 shows that ZST caches take advantage of the spatial locality because of the very nature of the depth buffer where neighboring fragments are more likely to be fetched in an X-Y frame grid. This improvement in the hit rate motivated our work to further investigate the impact of this improvement on the overall performance.

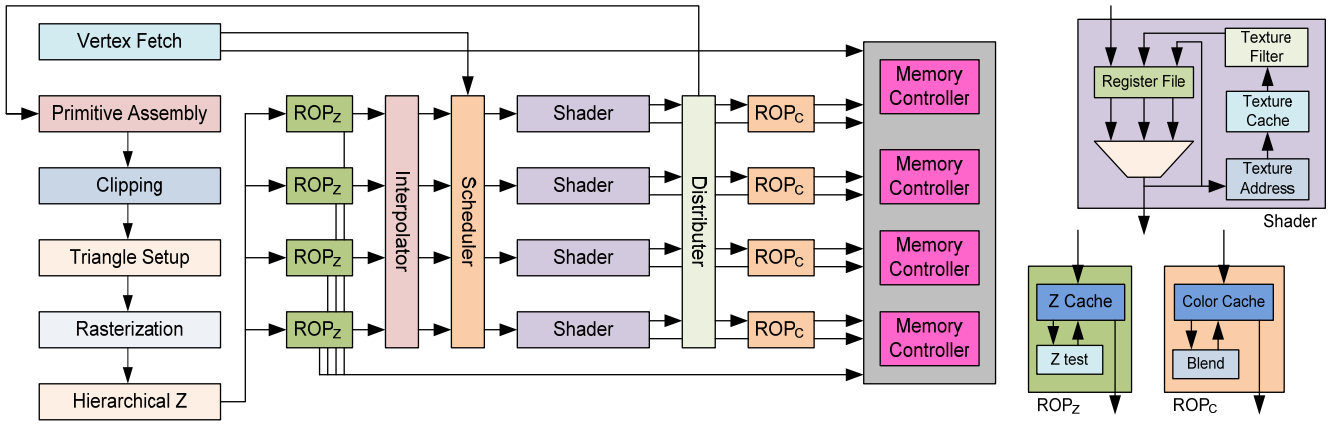

Figure 1: GPU pipeline architecture with unified shaders [10]

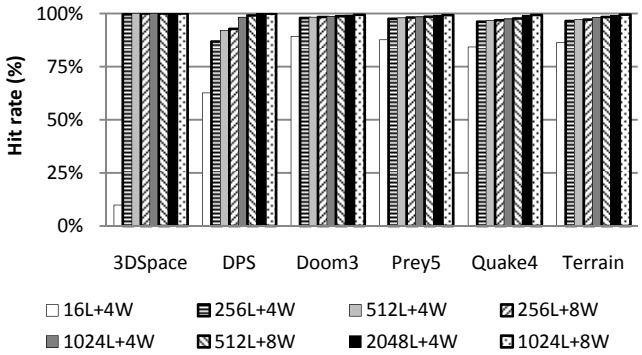

Figure 2: Hit rate of different TU cache configurations. The notion *N*L+*M*W means *N* lines and *M*-way associativity. Increasing number of lines increases hit rate dramatically.

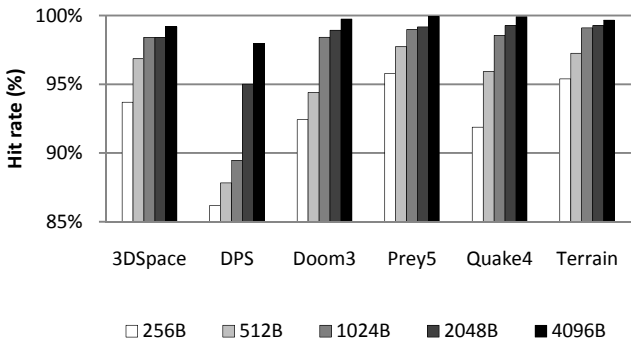

Figure 3: Hit rate of different ZST cache configurations. The notion NB means N-Byte block size. Increasing the block size results in higher hit rates in ZST caches

Therefore, we limit our design space to these two caches (namely, TU and ZST caches), and investigate the effect of modifying the organization of these two caches on the performance.

However, increasing the cache size may increase access time latency. Therefore, we use the 3DCacti simulator [11] in order to determine the extra cycles incurred due to size increase. Figures 4 and 5 compare the access time latency (cycle time is based on a 1.3GHz clock) incurred by the increase in cache size for a single-layer, 2-layer, and 4-layer stacked caches, for TU and ZST caches, respectively. These 2-layer and 4-layer caches were die-stacked by dividing the word lines. As we can see from the figures, increasing the cache size increases the latency; however, dividing the caches into a number of layers has reduced latency. Figure 6 summarizes the design space that was explored in this study. Based on the simulation results, we conclude that 3D technology can also be used in GPU architecture to maintain access time latency while increasing the cache size, hence improving performance.

Stacking caches on GPUs is associated with a number of constraints. For example, so far we allowed the size of stacked dies to be mismatched. However, in real implementations, these stacked dies needs to be matched. Therefore, we need to discuss a number of scenarios and assess the cost incurred by 3D technology. Thus the need for a cost model emerges, which is discussed next.

B. 3D Cost Model

Our cost model is based on that developed in [12]. This cost model allows us to estimate the extra cost incurred or saved by using 3D technology. The cost of the bare die depends on the bonding technology that we use. We assume a bonding cost of $150 per wafer [13], and we assume a KGD test cost of 1.5X cost of bare die for a yield of 90%. If using (W2W or D2W), then the cost is computed as follows:

$$C_{W2W} = \frac{\sum_{i=1}^{N} C_{die_i} + (N-1) * C_{bonding}}{(\prod_{i=1}^{N} Y_{die_i}) * Y_{bonding}^{N-1}} \quad (1)$$

$$C_{D2W} = \frac{\sum_{i=1}^{N} \frac{C_{die_i}+C_{KGDtest}}{Y_{die_i}} + (N-1) * C_{bonding}}{Y_{bonding}^{N-1}} \quad (2)$$

Based on these equations, the cost is mainly determined by the following factors:

- Die Cost: Cost of fabricating a single die before 3D bonding

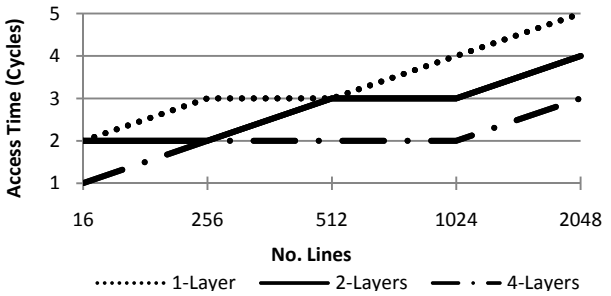

Figure 4: TU cache: Access time (in cycles) vs. number of lines. The figure shows that latency decreases when partitioning cache into two or four layers.

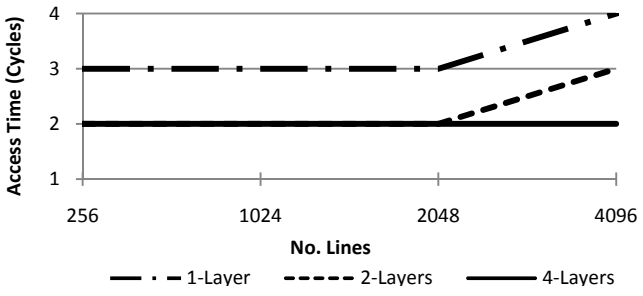

Figure 5: ZST cache: Access time (in cycles) vs. Block size (in bytes). The figure shows that latency decreases when partitioning cache into two or four layers.

- Bonding Cost: Cost incurred due to bonding dies; this cost may impact the cost-performance factor of 3D designs.
- Die Yield: The die area is inversely proportional to the die yield. Dividing the cache die reduces the footprint hence increasing the yield and consequently reduces the cost.
- Bonding Yield

Estimating the cost requires an estimation of the design area. We perform area estimation for a generic GPU architecture using the following approach. For the caches and memory structures, we utilize estimates of the 3DCacti simulator. For other GPU units/stages, we identify the number and type of components in each module. These are fed to the model presented in [14] for different logic components to estimate the gate count. Finally, we translate the gate count to gate area using the following scaling parameters: $2.6x10^{-12}$ ($1.3x10^{-12}$) m^2 for 65nm (45nm) technology [15]. The area estimation model is also used in estimating power consumption as discussed next.

C. *Power/Thermal Analysis*

One of the issues related to die stacking is the increase in power density which leads to an increase in chip temperature [9]. Unlike microprocessors, there are no architectural-level power estimation tools available for modern GPUs. Therefore, our power consumption model is based on the area model developed above. For the device layer, we obtain the capacitive load per gate and nominal voltage for the 65nm and 45nm technologies from ITRS [15]. Using these parameters, along with the estimated gate count from our area model and assuming an operating frequency of 1.3 GHz, the device layer dynamic power is estimated. For interconnect layers, we estimate the dynamic power consumption using the power index from ITRS. We multiply the number of metal layers and area of the design with the power index. The power index used is 16000 (18000) W/GHz*m^2 for the 65nm (45nm) technology [15]. Leakage power is estimated based on its relative value to overall power consumption [16]. Then, we use hotspot simulation tool [17] to analyze the thermal profile of the proposed 3D GPU design.

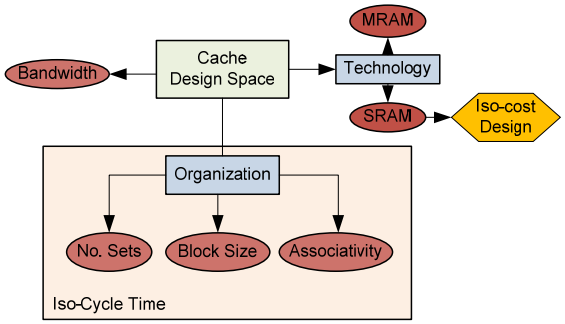

Figure 6: Domains of the design space explored in this work. More focus was on modifying cache organization since it offers higher performance.

Since leakage power is an important component of power consumption, we consider the impact of utilizing non-volatile Magnetic Random Access Memory (MRAM) [18] that has zero standby power as a candidate for implementing caches.

IV. SIMULATION SETUP

The baseline architecture used in our simulations consists of 64 shader units. We also use a texture unit cache, a ZST cache, and a Color Write cache of 16KB each. The organization of these caches is as follows; 16 lines, 4-ways set associative and 256KB line size. In this work we use ATTILA GPU simulator [10]. The workloads used in the simulations [19] are diverse in their characteristics (e.g. depth complexity, texture intensiveness, etc…). Table 1 summarizes the workloads used in the simulations.

V. RESULTS

In this section, we discuss the potential performance improvement by stacking caches on GPU units, assuming iso-cycle time of 0.75 ns. This cycle time captures typical frequency ranges used in current GPUs [20]. Then, we present two iso-cost scenarios where we compare the performance of two systems, one in 2D planar design and the other implemented using 3D technology.

A. *Iso-Cycle Time Results*

While we explored a number of cache organizations, we present a representative set of the obtained results for brevity. Figure 7 shows the speedup achieved by increasing the number of layers in the TU cache.

Table 1: Workloads used in simulations

Workload	Resolution	No. frames
3Dspace	640x480	128
DPS	788x540	200
Doom3	320x240	40
Prey5	320x240	40
Quake4	320x240	40
Terrain	788x540	15

We observe that the 2-layer and 4-layer caches outperform a single-layer cache due to reduced number of access cycles. The 2-layer cache offers up to 22% speedup, whereas the 4-layer cache offers up to 55% speedup. Figure 8 shows similar trends for the ZST cache where the 2-layer (4-layer) cache implementation improves performance by up to 15 (37)%. Merging the 4-layer stacked TU and ZST caches into a single design results in a 53% geometric mean speedup.

B. Iso-Cost Design: Scenario I

We consider two iso-cost designs: a 2D GPU and a 3D-stacked cache GPU. Both GPUs contain 128 shaders, and both utilize 65nm technology. The first layer in the 3D GPU contains the GPU processing units, while the other two layers contain the partitioned ZST and TU caches as illustrated in Figure 9(a). The cost of each of these configurations was matched by adjusting the architectural parameters such as the size of queues that interface the caches. Further, we use the D2W stacking since W2W technology costs more than D2W due to deteriorating yield with increasing die area [12] (the GPU die size is relatively large (~ 170 mm^2) in our case). Figure 10 shows that for the iso-cost scenario, the 3D architecture achieves up to 45% speed up over the 2D planar architecture.

C. Iso-Cost Design: Scenario II: Heterogeneous Integration

As discussed earlier, one of the benefits that 3D technology offers is the ability to integrate heterogeneous technologies. This subsection presents a design that integrates two different technologies. In the first layer of the 3D design, we implement the GPU units in 65nm technology. However, the second layer uses 45nm technology as illustrated in Figure 9(b). Working with smaller feature sizes allows us to cram all the caches into one layer saving cost incurred due to bonding. Figure 11 shows that 3D design outperformed 2D by a 19% geometric mean speedup.

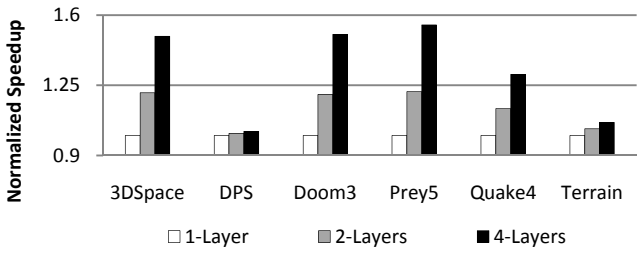

Figure 7: TU cache: Execution time speedup with increasing number of layers. All values are normalized to a one-layer design. No. of cache lines = 2048. A 2MB 4-way associative cache with 256 byte word size was used.

D. Power and Thermal Analysis

1) Iso-Cost Designs: Based on the power estimation model developed earlier, we estimate the total power consumption of Scenarios I and II to be 106.4W and 82.1W, respectively. Furthermore, using hotspot simulation tool [17], we found the maximum temperature for scenarios I and II to be 121.55°C and 82.24°C, respectively.

2) MRAM vs. SRAM Results: We replace SRAM by MRAM caches in our GPU and evaluate the power and performance impact. Figure 12 shows the simulation output. For caches with a less number of writes compared to reads, we observed a performance gain. However, due to the slow write times of the MRAM, compared to SRAM, when the number of writes is large, there is performance degradation. Consequently, we can conclude that adopting MRAMs in GPU caches is not necessarily beneficial for higher performance. However, it is clear from Figure 13 that the power benefits of MRAM over SRAM makes the former more appealing for power-conserving applications.

VI. CONCLUSIONS AND FUTURE WORK

This paper represents a first step towards the implementation of a fully 3D die-stacked GPU. We started this study by narrowing our design space exploration by choosing the caches that would potentially improve performance. Then, by modifying the organization of caches, and partitioning these caches into multiple layers, we were able to achieve higher hit rate while maintaining a reasonable access time. We further assessed these implementations taking into consideration the cost incurred when implementing 3D technology. The cost model was used to propose two design scenarios where 2D and 3D designs are compared to each other using iso-cost case. We show that our 3D design provides significant performance gains including iso-cost case. Based on the results of this work, it would be interesting to extend this work further and design other GPU functional units (e.g. shaders) in 3D technology and analyze the performance gain.

ACKNOWLEDGMENT

The authors would like to thank the reviewers for their comments. Special thanks to Victor Moya for his assistance with ATTILA. This work was supported in part by NSF grants 0903432, 0702617, CAREER 0643902 and a grant from SRC.

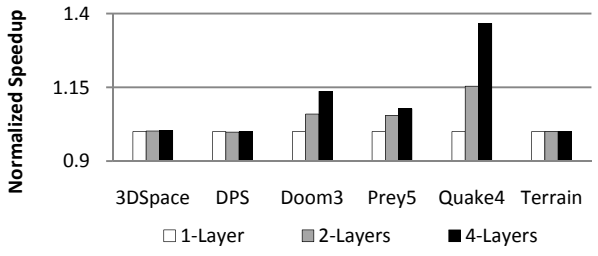

Figure 8: ZST cache: Execution time speedup with increasing number of layers. All values are normalized to a one-layer design. A 256KB 4-way associative cache with 4096 byte word size was used.

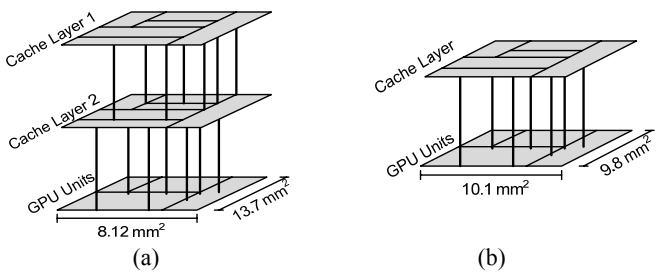

Figure 9: Iso-Cost designs: (a) cache is divided into two layers and stacked to GPU unit layer. (b) Using 45nm to fabricate caches into a single layer.

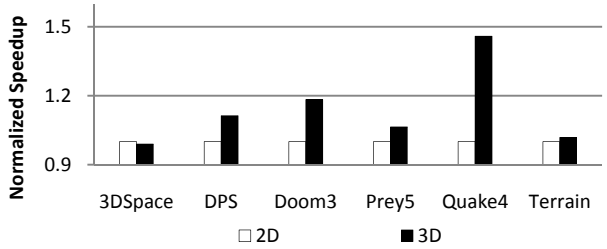

Figure 10: Scenario I: 2D vs. 3D-cache stacked GPU (2-layer cache + 1 layer GPU units). Results are normalized to 2D design.

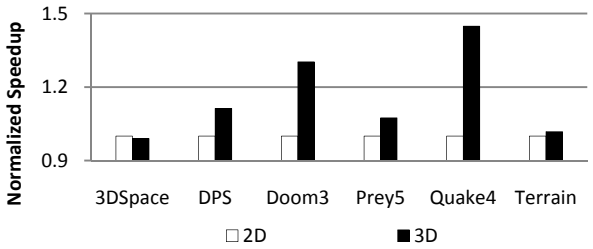

Figure 11: Scenario II: Heterogeneous Integrations Design (1 layer cache, 1 layer GPU). Results are normalized to 2D design.

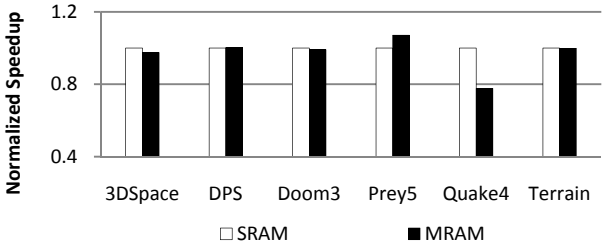

Figure 12: Comparing system-level improvement using 4-layer SRAM and MRAM caches (Normalized to SRAM per set). TU cache is 2MB 4-way associative and ZST cache is 256KB 4-way associative

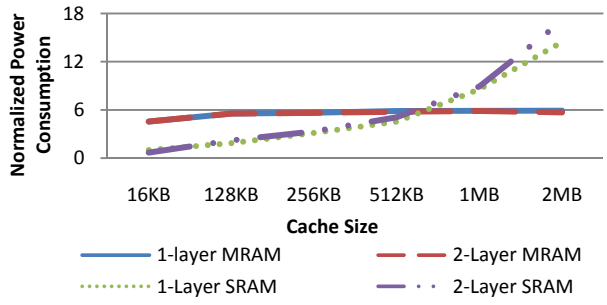

Figure 13: Power consumption of MRAM and SRAM. As cache size increases, MRAM consumes less power than SRAM cache.

REFERENCES

[1] General-Purpose Computation Using Graphics Hardware: available online: www.gpgpu.com

[2] Nvidia: CUDA Homepage, available online: http://www.nvidia.com/object/cuda_home.html

[3] N. Goodnight, C. Woolley, G. Lewin, D. Luebke, and G. Humphreys, "A multigrid solver for boundary value problems using programmable graphics hardware," *in Proc. SIGGRAPH/EUROGRAPHICS Conference On Graphics Hardware*, 2003, pages 102-111

[4] K. Fatahalian, J. Sugerman, and P. Hanrahan, "Understanding the Efficiency of GPU Algorithms for Matrix-Matrix Multiplication," *in Proc. SIGGRAPH*, 2004, pages 133-137

[5] N. Govindaraju, S. Larsen, J. Gray, and D. Manocha, "A Memory Model for Scientific Algorithms on Graphics Processors," *in Proc. Conference on High Performance Networking and Computing*, 2006. Article No. 89

[6] J. D. Hall, N. Carr, and J. Hart, "Cache and Bandwidth Aware Matrix Multiplication on the GPU," Technical Report UIUCDCS-R-2003-2328, University of Illinois Urbana-Champain.

[7] M. Silberstein, A. Schuster, D. Geiger, A. Patney, and J. D. Owens, "Efficient computation of sum-products on GPUs through software-managed cache," *in Proc. Inter. Conference on Supercomputing*, 2008, pp 308-318

[8] G. Luca Loi, B. Agrawal, N. Srivastava, Sheng-Chih Lin, T. Sherwood, and K. Banerjee, "A thermally-aware performance analysis of vertically integrated (3-D) processor-memory hierarchy," *in Proc. Design Automation Conference*, 2006, pages 991-996

[9] K. Puttaswamy, G. H. Loh, "Thermal Herding: Microarchitecture Techniques for Controlling Hotspots in High-Performance 3D-Integrated Processors," *in Proc. HPCA,* 2007, pp 193-204.

[10] del Barrio, V.M. Gonzalez, C. Roca, J. Fernandez, and A. Espasa E., "ATTILA: a cycle-level execution-driven simulator for modern GPU architectures," *in Proc. International Symposium on Performance Analysis of Systems and Software*, 2006

[11] Yuh-Fang Tsai, Y. Xie, N. Vijaykrishnan, and Mary Jane Irwin, "Three-Dimensional Cache Design Exploration Using 3DCacti," *in Proc. International Conference on Computer Design*, 2005, pages 519-524

[12] X. Dong, and Y. Xie, "System-level Cost Analysis and Design Exploration for 3D ICs," *in Proc. Asia and South Pacific Design Automation Conference*, 2009

[13] S. Jones, "2008 IC Economics Report," In *IC Knowledge* LLC., 2008.

[14] V. K. Kodavalla, "IP Gate Count Estimation Methodology during Micro-Architecture Phase," in IP based Electronic System, 2007

[15] ITRS, "International Technology Roadmap for Semiconductors", available online: www.itrs.net

[16] S. Rodriguez, and B. Jacob, "Energy/power breakdown of pipelined nanometer caches (90nm/65nm/45nm/32)," *in Proc. International Symposium on Low Power Electronics and Design*, 2006, pages 35-30

[17] K. Skadron, M. R. Stan, W. Velusamy, K. Sankaranarayanan, and D. Tarjan, "Temperature-Aware Microarchitecture," *in Proc. International Symposium on Computer Architecture*, 2003, pages 2-13

[18] M. Hosomi, H. Yamagishi, and T. Yamamoto, "A novel nonvolatile memory with spin torque transfer magnetization switching: spin-ram," in International Electron Devices Meeting, 2005, pages 459-462

[19] Attila Project: AttilaWiki, available online: https://attila.ac.upc.edu/wiki/index.php/Main_Page, 2008

[20] GeForce 9400 GT Specifications, available online: http://www.nvidia.com/object/product_geforce_9400gt_us.html

[21] J. Owens, "GPU architecture overview," *in Proc. International Conference on Computer Graphics and Interactive Techniques*, 2007

[22] R. Weerasekera, Li-Rong Zheng, D. Pamunuwa, and H. Tenhunen, "Extending systems-on-chip to the third dimension: performance, cost and technological tradeoffs," *in Proc. International Conference on Computer Aided Design*, 2007

[23] (Online Course Resource) K. Akeley, and P. Hanrahan, "Real-Time Graphics Architecture," CS448 Spring 2007, Stanford University