GPU

Seminar Report

Submitted in partial fulfillment of the requirements
for the degree of

Master of Technology

by

Hitesh Kumar Sahu
(Roll No. 213079013)

Under the guidance of
Prof. Virendra Singh

Department of Electrical Engineering
Indian Institute of Technology Bombay
September 2022

Acknowledgement

I express my gratitude to my guide Prof. Virendra Singh for providing me the opportunity to work on this topic.

Hitesh Kumar Sahu
Electrical Engineering
IIT Bombay

Contents

List of Figures

Chapter 1

Introduction

Graphics Processing Unit (GPU) architectures are a high-performance parallel computing systems. GPU provides very fast and energy efficient execution for many general purpose applications which can be paralleled. GPU provides a more attractive options in terms of cost and bulkiness compared to supercomputers. Though GPUs have the resources to compute floating-point computations but suffer from some major problems that limit scalability as well as the ability to deliver the promised throughputs for a wide range of applications: high power consumption, long memory access latencies and generalizing for any scientific computations. Also due to tight coupling between L1 cache and the GPU cores, the conventional cache hierarchy system in GPU results in inefficient utilization of the valuable on-chip L1 caches.

So to overcome the above mentioned problems two methods are proposed. First an optimized two-dimensional SIMD datapath design which used paralleled chained floating point units (FPUs) with additional features of Dynamic degree prefetching and Divergence folding. The second one is to separate L1 caches from the GPU cores i.e. Decoupled L1 caches (DC-L1). Each DC-L1 cache is accessed by multiple GPU cores. Aggregating DC-L1 caches improves their individual bandwidth utilization and reduces data replication across the DC-L1s as more cores are accessing a given DC-L1 which reduces cache replication.

Chapter 2

Literature Survey

2.1 Analyzing and Leveraging Decoupled L1 Caches in GPUs

In GPU architecture the L1 cache is usually private to its own individual cores. Due to this a different cores try to fetch information from low level caches which may be already used by some different cores which is referred as cache replication. Cache replication is a big issue which decreases the performance due to latency while fetching the data. Due to the mentioned issue the L1 cache are also not completely utilised. An alternative method where L1 caches are decoupled from the cores called Decoupled L1 caches(DC-L1) so that different cores cores can access the information which may be present in one the DC-L1 cache which drastically reduces the cache replication by improving the use of DC-L1 cache. As L1 cache is decoupled from cores an extra NoC networks to be introduced between GPU cores and DC-L1 caches. As the complexity of NoCs increases the power dissipation also increases. So an optimal design is required to increase the utilization DC-L1 caches with minimal extra power consumption.

2.2 PEPSC Architecture

GPU are highly attractive to industry performing graphics processing as multiple and independent parallel instructions can be processed with very high throughput. But when it comes to scientific computations there are huge number of dependent instructions to be processed which makes GPU less reliable to use. So to counter the issue of GPU , a new architecture is implemented referred as PEPSC - Power-Efficient Processor for Scientific Computing. In this architecture instead of just implementing parallel cores , the single cores is again chained of 3 to 5 deep Floating point arithmetic (FPU) units. By implementing the chaining technique the dependent computations within the instructions can be easily handled which in terms increases the performance of the processor much higher than the generic GPUs. Even while chaining it has allowed to execute the parallel independent instructions. Dynamic degree prefetcher and divergence control method is also implemented which increases the GPU utilization with less power dissipation which enhances the Performance over power ratio.

Chapter 3

Review

3.1 Analyzing and Leveraging Decoupled L1 Caches in GPUs

3.1.1 Summary

- A DC-L1 node simply contains the DC-L1 cache , two queues to handle the traffic from/to the GPU core, and two queues to handle the traffic to/from the L2 and memory partitions. A GPU core is a lite core which is similar to the baseline GPU core but without the L1 data cache and the associated Miss Status Holding Registers (MSHR).

- The NoC is divided into two parts. The first NoC connects the GPU cores and the DC-L1 nodes and the second NoC connects the DC-L1 nodes and the L2 memory.

Three different styles of architecture is proposed here. First one being the Private DC-L1 cache shown in Figure 1 where X GPU cores can access Y DC-L1 cache with each DC-L1 cache being accessed by N=X/Y number of GPU cores. This design requires N X 1 crossbars for GPU to DC-L1 NoC. This type of private aggregated DC-L1 design is referred as PrY.

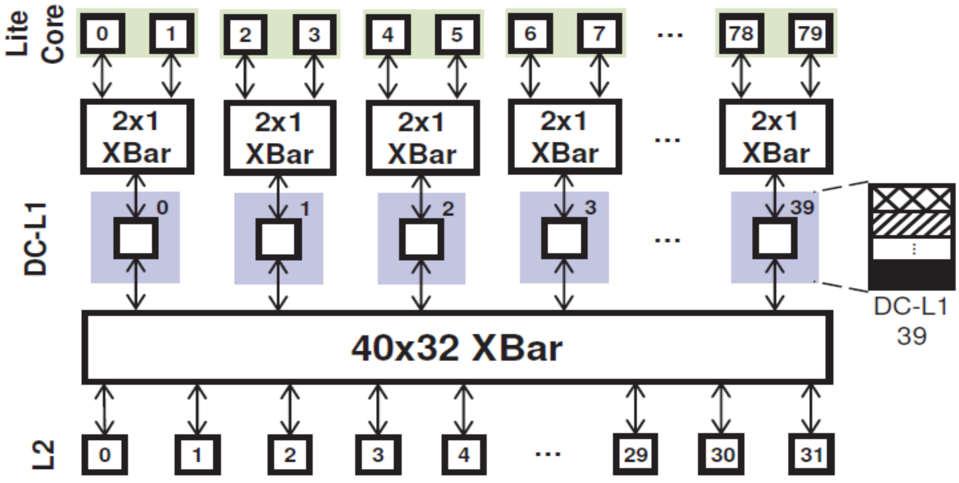


Fig. 5: Pr40 design.

Figure 1: Pr40 design

Another design is shown in Figure 2 referred to Shared DC-L1 cache design where any of the GPU cores can access the any of the DC-L1 cache. In this design the X GPUs cores can access Y DC-L1 cache using a more complex NoC network of X x Y crossbar is required. This design is referred as ShY design.

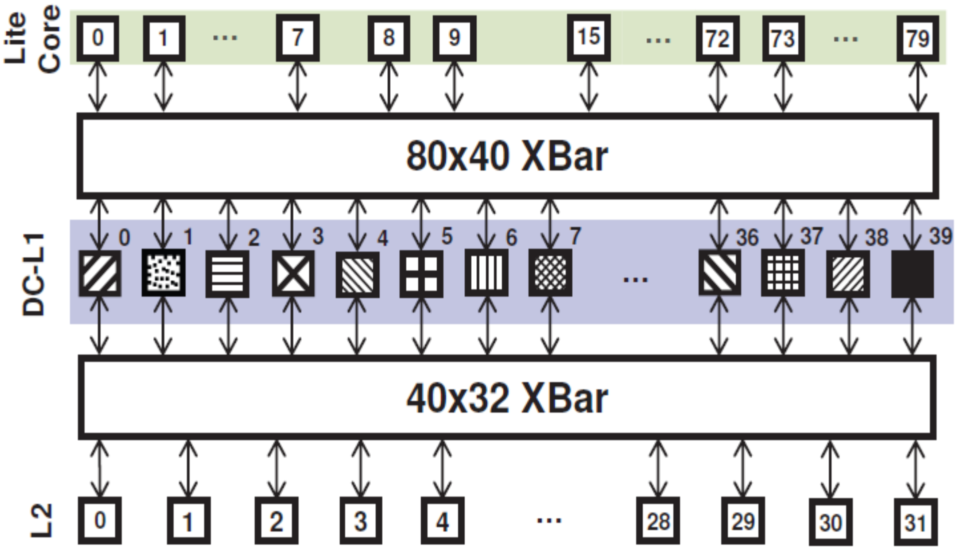


Figure 2: Sh40 design

The last style of design is called Clustered DC-L1 design shown in Figure 3 a cluster of M DC-L1 cache is accessed by N GPU cores via N X M crossbars NoCs. This design is referred as ShY +CZ, where Y is the total number of DC-L1 nodes and Z is the number of clusters. The L2 slices and that each DC-L1 within a cluster is assigned a unique address range due to the shared nature of both L1 and L2 caches, a given DC-L1 will communicate only with a few L2 slices. Therefore, instead of using a full Y × L crossbar in NoC L1 to L2 to connect the Y DC-L1 nodes to the L L2 slices (L = M, L mod M = 0), a given DC-L1 will communicate only with O = L/M L2 slices via an Z × O crossbar in NoC L1 to L2.

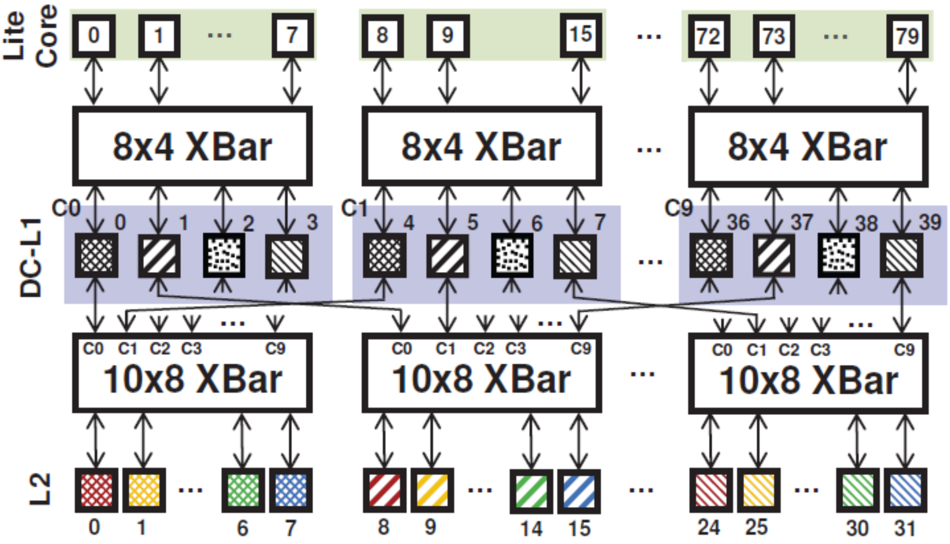


Figure 3: Sh40+C10 design

3.1.2 Strengths

1. Due to DC-L1 techniques the tight coupling between L1 cache and GPU cores is eliminated which reduces the cache line replication.
2. High bandwidth utilization due reduction in cache line replication.
3. Decreased latency due to memory access.

3.1.3 Weakness

1. Due to Decoupled L1 cache the NoC is introduced between cores and DC-L1 which is power consuming.
2. Optimal requirement of number of DC-L1 caches to be chosen for a single cluster which may induces a fair share in the power budget if number of DC-L1 caches within a cluster increases.

3.2 PEPSC: A Power-Efficient Processor for Scientific Computing

3.2.1 Summary

The PEPSC architecture in figure 4 has presented an two-dimensional design that extracts power efficiency from both the width and the depth of a SIMD datapath, Fine-grain control of the SIMD datapath to mitigate the cost of control divergence, a dynamically adjusting prefetcher to mitigate memory latency. and an integrated reduction floating-point adder tree for fast, parallel accumulation with low hardware overhead.

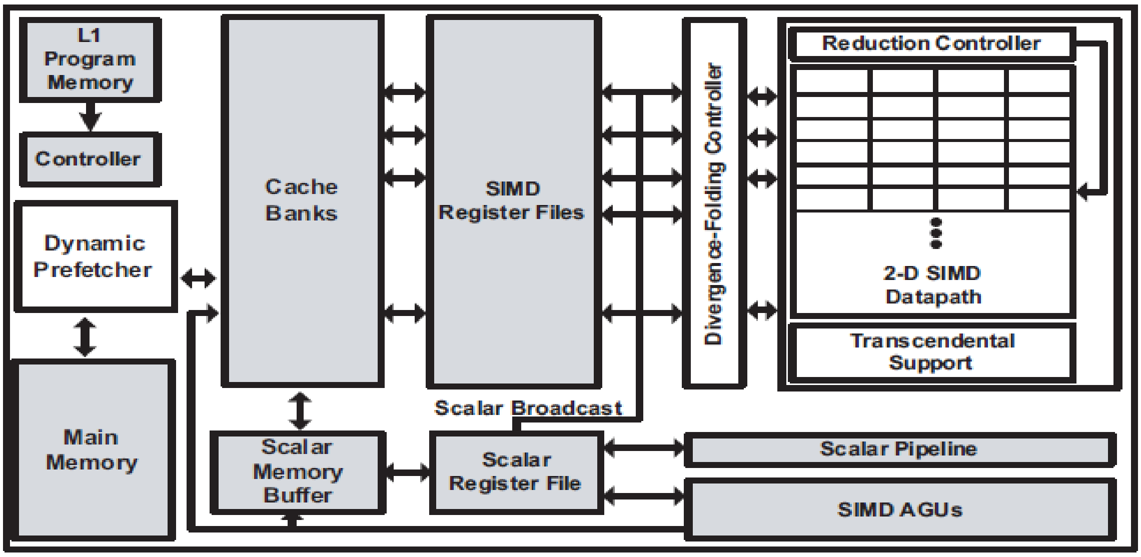


Figure 4: PEPSC architecture

The two dimensional SIMD Datapath has multiple SM cores as one dimension and 3-5 level depth chained FPU units shown in figure 5. As scientific applications are usually back to back FPU operations so by chaining improves power efficiency and performance. For independent operations the chained is modified to support execution in parallel. It also realizes a dynamic degree prefetcher(DDF) shown in figure 6 which changes the degree of prefetching for the different length of iterations. The PEPSC architecture is also modified for the branch statement which usually decreases the GPU utilizations due the repeated execution for different branch conditions. Here a predicate bit assigned for each of the instructions for two different branches taken. Complementary predicates executes parallely with some additional modifications. The modifications is done at compiler to allow the gaurding of operation by predicates and the control logic is modified to select which of the two concurrently executing subgraphs should write their values to the RF based on a given SIMD lane's predicate mask bit simultaneously.

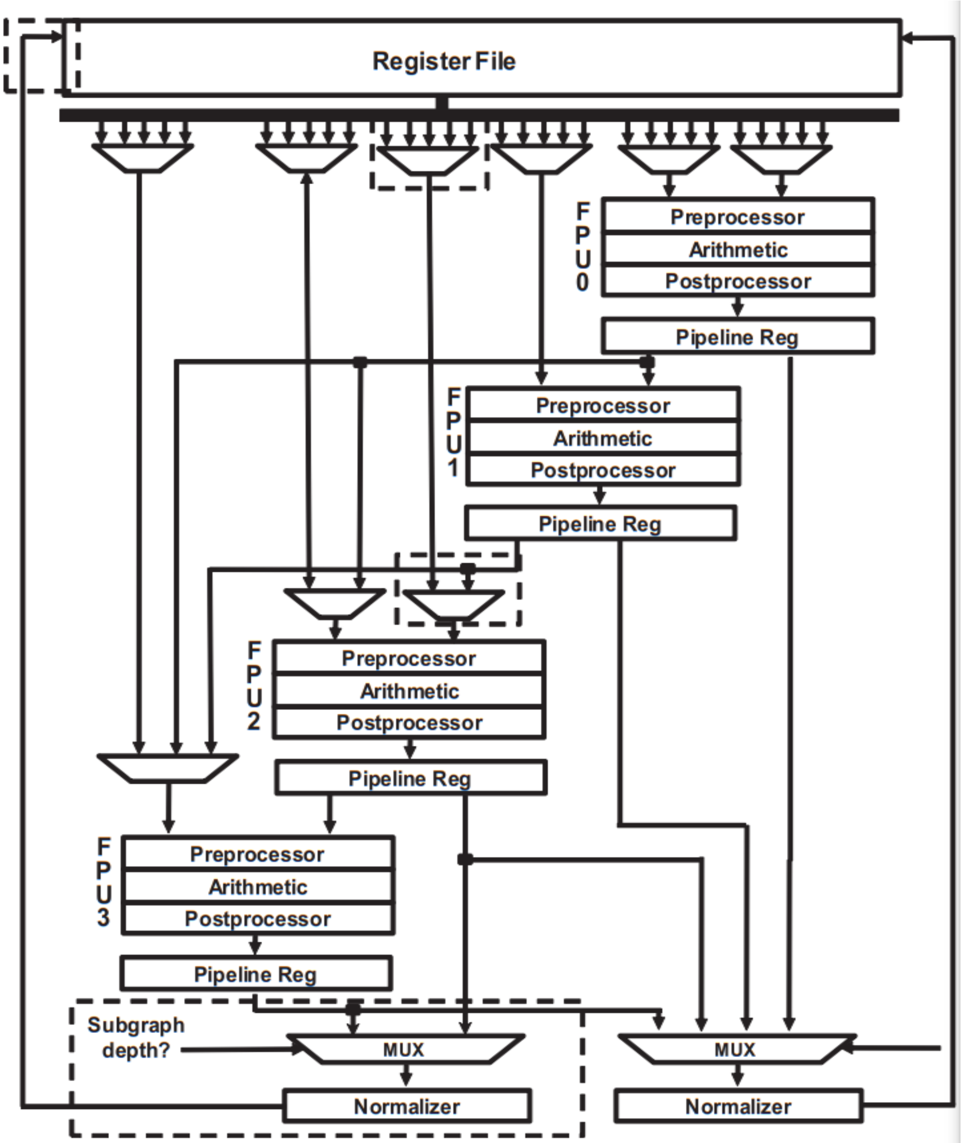


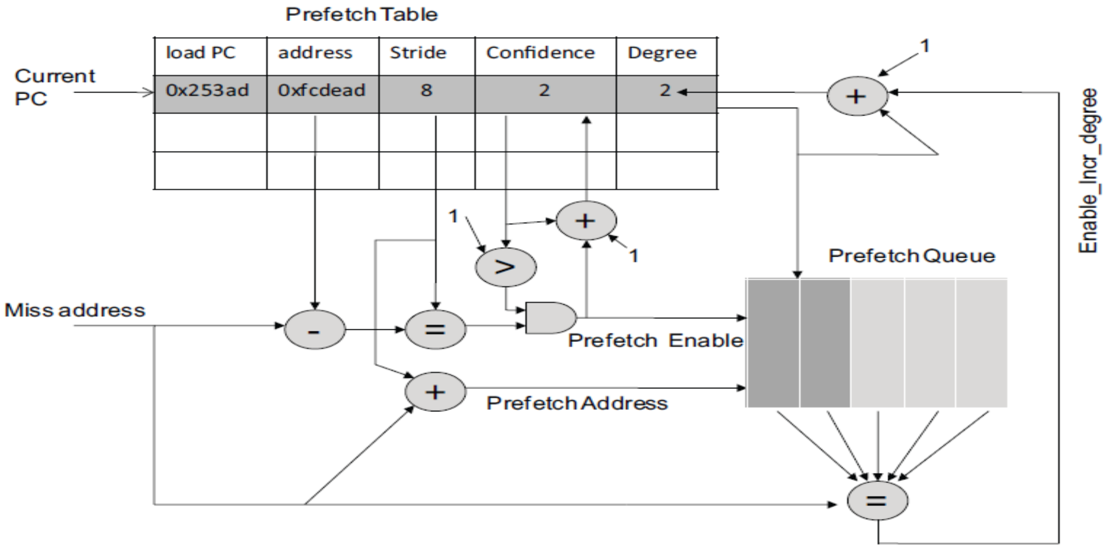
Figure 5: Chained FPU design

3.2.2 Strengths:

1. Good power efficiency and performance improve compared to generic GPU available in market.

2. Higher GPU utilization due improvised PEPSC by adding dynamic degree prefetching and divergence control mechanism.

3. Due to chain coalescence independent threads can also be executed.



load PC	address	Stride	Confidence	Degree
0x253ad	0xfcdead	8	2	2

Figure 6: Dynamic degree prefetcher

3.2.3 Weakness:

1. Due to addition divergence control, compiler needs to be modified which requires upgrading the available compiler.

2. Due to high depth of chaining , generalizing for graphics processing needs to be take care.

3. Due to chaining the ports to access the register banks has increased which in terms of cost is a big hit.

Chapter 4

Conclusions:

4.1 Analyzing and Leveraging Decoupled L1 Caches in GPUs

It has introduced an L1 cache decoupled from the GPU core i.e. Decoupled L1 (DC-L1) cache. Due to decoupling GPU cores can access information within DC-L1 cache which reduces the cache line replication and enhances bandwidth utilization of the L1s. A clustered-based DC-L1 cache organization was introduced , where a cluster of GPU cores access a cluster of shared DC-L1s.

4.2 PEPSC: A Power-Efficient Processor for Scientific Computing

PEPSC architecture is power efficient computing architecture with efficient chained-operation datapath. It reduces register file accesses and computation latency. It has significant improvement over 10X over current GPUs.Extra hardware components used while in some of the hardware in some benchmarks being underutilized. Even register file accesses decreases , port to access those register increases due to chaining. Compiler is changed to tackle divergence control problem.

Chapter 5

Future Works

5.1 Conference Papers to be Follow Through:

- W. W. L. Fung, I. Sham, G. Yuan and T. M. Aamodt, "Dynamic Warp Formation and Scheduling for Efficient GPU Control Flow," 40th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO 2007), 2007, pp. 407-420, doi: 10.1109/MICRO.2007.30.
- Adwait Jog, Onur Kayiran, Nachiappan Chidambaram, ASit K. Mishra, Mahmut T. Kandemir, Onur Mutlu, Ravi R. Iyer and Chita R. Das, "OWL: Cooperative Thread Array Aware Scheduling Techniques for Improving GPGPU Performance, ASPLOS 2013
- A. Sethia, D. A. Jamshidi and S. Mahlke, "Mascar: Speeding up GPU warps by reducing memory pitstops," 2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA), 2015, pp. 174-185, doi: 10.1109/HPCA.2015.7056031.

Chapter 6

References

1. M. A. Ibrahim, O. Kayiran, Y. Eckert, G. H. Loh and A. Jog, "Analyzing and Leveraging Decoupled L1 Caches in GPUs," 2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA), 2021, pp. 467-478, doi: 10.1109/HPCA51647.2021.00047.

2. G. Dasika, A. Sethia, T. Mudge and S. Mahlke, "PEPSC: A Power-Efficient Processor for Scientific Computing," 2011 International Conference on Parallel Architectures and Compilation Techniques, 2011, pp. 101-110, doi: 10.1109/PACT.2011.16.