# Investigating Nvidia GPU Architecture Trends via Microbenchmarks

Lingqi Zhang [12] Ryan Barton [2] Peng Chen [3] Xiao Wang [4] Toshio Endo [2] Satoshi Matsuoka [12]
Mohamed Wahib [1]

[1]RIKEN Center for Computational Science [2]Tokyo Institute of Technology

[3]National Institute of Advanced Industrial Science and Technology [4]Oak Ridge National Laboratory

## Introduction

The GPU platforms included in this study are listed in Table 1.

| GPU Platform | Release Year | Arch | SM count | Freq (MHz) | FP64 (TFLOPS) | FP32 (TFLOPS) | TC FP16 (TFLOPS) | Bandwidth (GB/s) |
|---|---|---|---|---|---|---|---|---|
| Tesla/ Data Center GPUs | | | | | | | | |
| K20 | 2013 | 3.50 | 15 | 758 | 1.17 | 3.52 | - | 208 |
| P100-PCIE | 2016 | 6.00 | 56 | 1480 | 5.2 | 10.6 | - | 720 |
| V100-SMX | 2017 | 7.00 | 80 | 1530 | 7.8 | 15.4 | 125 | 900 |
| A100-PCIE | 2020 | 8.00 | 108 | 1410 | 9.7 | 19.5 | 312 | 1555 |
| GH200 | 2024 | 9.00 | 132 | 1830 | 33.5 | 66.9 | 989.4 | 4096 |
| Workstation/ Consumer GPUs | | | | | | | | |
| GTX 980TI | 2015 | 5.20 | 22 | 1000 | 0.189 | 6.06 | - | 336.6 |
| GTX 1080TI | 2017 | 6.10 | 28 | 1481 | 0.354 | 11.3 | - | 484 |
| RTX 2080TI | 2018 | 7.50 | 68 | 1350 | 0.42 | 14.2 | 130.5 | 616 |
| RTX 3090 | 2020 | 8.60 | 82 | 1395 | 0.556 | 35.6 | 142 | 936 |

Table 1. GPU platforms used in this study.

## Microbenchmarks

In this research, we introduce the microbenchmarks used to gather data. The throughput for Nvidia GPUs is documented in sources such as white papers. So, our focus is on the undocumented latency information.

Listing 1. Pseudo code for kernel: test latency of add instruction, repeat 512 times

```
__global__ void kernel1(){
  start=clock(gt0_beg);
  repeat256(p=p+q;q=p+q);
  end=clock(gt0_end);
  return q;
}
```

Listing 2. Pseudo code for kernel: test latency of add instruction, repeat 1024 times

```
__global__ void kernel1(){
  start=clock(gt1_beg);
  repeat512(p=p+q;q=p+q);
  end=clock(gt1_end);
  return q;
}
```

Listing 3. Pseudo code for cpu timer

```
cpuclock(ct0_beg);
kernel1();
syncdevice();
cpuclock(ct0_beg);
cpuclock(ct1_beg);
kernel2();
syncdevice();
cpuclock(ct1_end);
```

### Intra Stream Multiprocessor Microbenchmark

We utilize Wong's microbenchmark [2] to evaluate the intra stream multiprocessor latency in Nvidia GPUs. The specific microbenchmark is shown in Listing 1. The latency of the instruction (add) is calculated as $(\text{gt0\_end-gt0\_beg})/512$.

### Inter Stream Multiprocessor Microbenchmark

We utilize a microbenchmark previously proposed in [3] to measure inter stream multiprocessor latency in Nvidia GPUs. The microbenchmark can be found in Listing 1, Listing 2, and Listing 3. The latency of the instruction (add) is calculated as $((\text{ct1\_end}-\text{ct1\_beg})-(\text{ct0\_end}-\text{ct0\_beg}))/(1024-512)$.

## Little's Law

We are interested in identifying the minimum sufficient parallelism for Nvidia GPUs. To achieve this, we use Little's Law, which has been employed in previous studies [1]. Little's Law uses latency $\mathbb{L}$ and throughput $\mathbb{THR}$ to infer the concurrency $\mathbb{C}$ of a hardware system:

$$\mathbb{C} = \mathbb{L} \times \mathbb{THR} \quad (1)$$

- Lower concurrency levels, easier device saturation simpler programming
- Higher concurrency levels, harder device saturation more complex programming

## Computation

The evolution of GPU compute throughput and latency is depicted in Figure 1. The results show that there has been a significant increase in peak throughput over the past decade, as well as a noticeable reduction in latency.

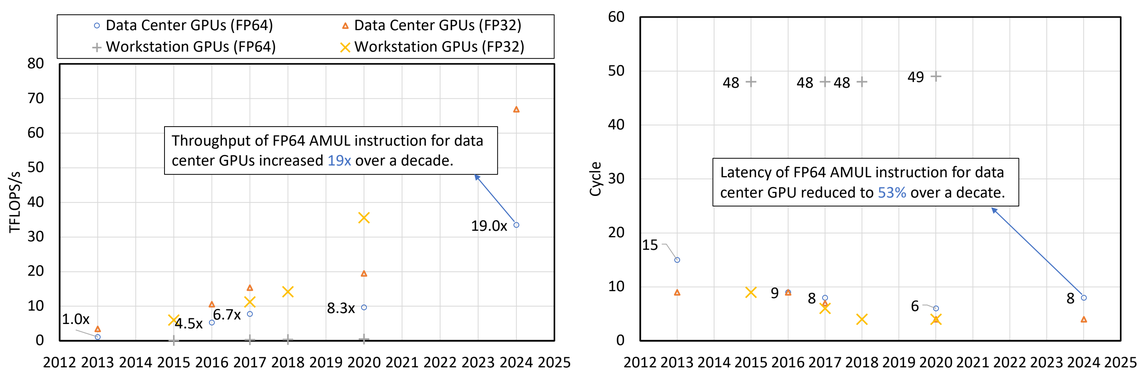


Figure 1. Compute instruction (FP64 & FP32) is optimized over time (w/o tensor core).

## Memory

The evolution of GPU memory bandwidth and memory access latency is shown in Figure 2. The figure displays a significant increase in memory bandwidth over the past decade and a slight reduction in memory access latency.

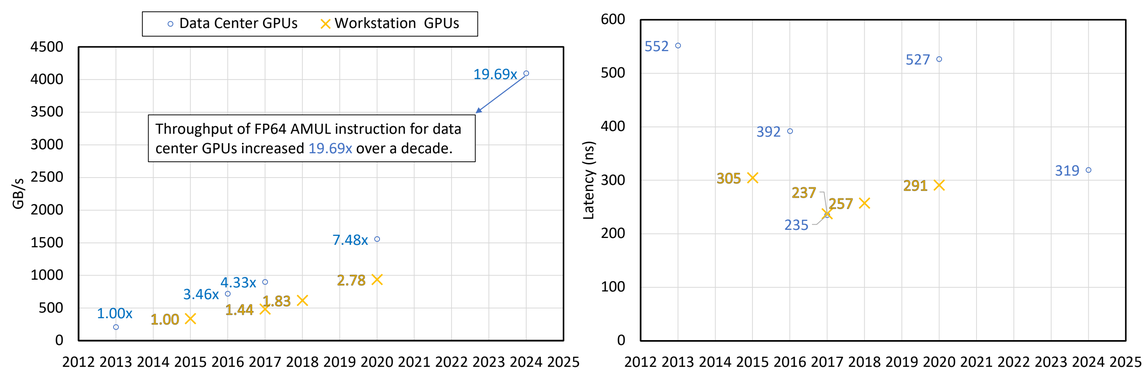


Figure 2. Memory features are also optimized over the evolution of the GPU.

## Concurrency

We use information on throughput (bandwidth) and latency to calculate the concurrency based on Little's Law, as shown in Figure 3.

- Before 2020, concurrency requirements was decreasing.
- After introducing the A100 and H100, the concurrency requirements is increasing.
- It is easier for compute-bound kernels to saturate the device.

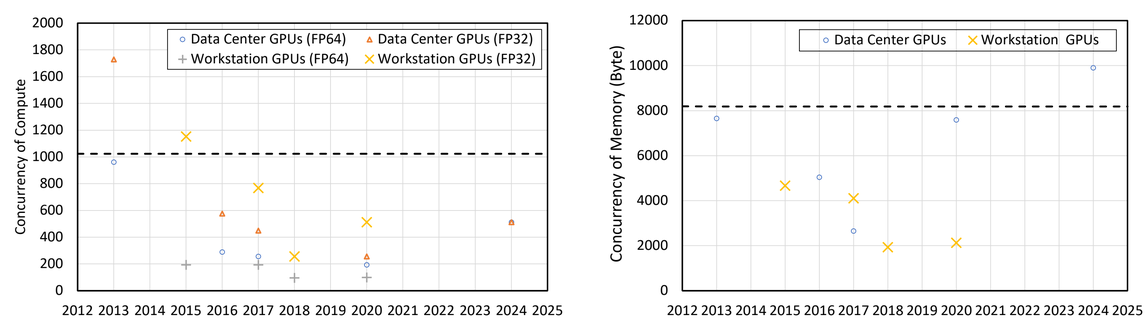


Figure 3. The concurrency trends over the evolution of GPUs.

## Balance

We plot the machine balance of different GPUs over time, as shown in Figure 4.

- The Workstation GPUs appear increasingly imbalanced over time.
- The balance of Data Center GPUs remains relatively static.

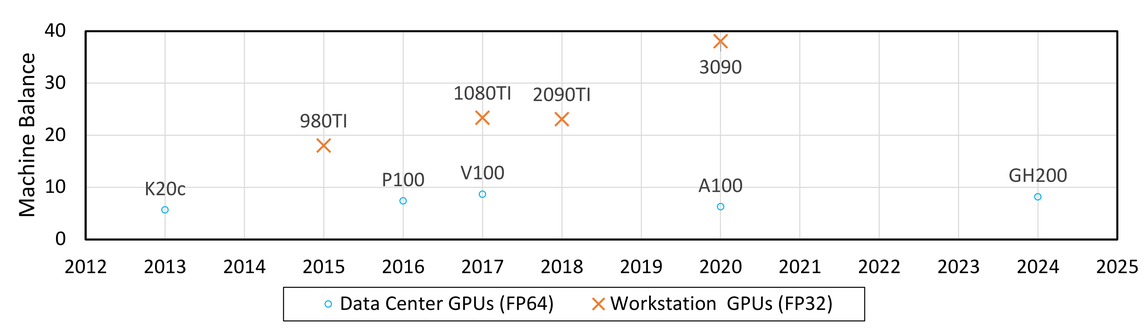


Figure 4. The balance trends over the evolution of GPUs.

## Future Work

- Tensor Core
- Optimization Effectiveness
  - Rely on hardware assumptions
  - Accomplished by hardware optimizations

## References

[1] Vasily Volkov.
Better performance at lower occupancy.
In *Proceedings of the GPU technology conference, GTC*, volume 10, page 16. San Jose, CA, 2010.

[2] Henry Wong, Misel-Myrto Papadopoulou, Maryam Sadooghi-Alvandi, and Andreas Moshovos.
Demystifying gpu microarchitecture through microbenchmarking.
In *Performance Analysis of Systems & Software (ISPASS), 2010 IEEE International Symposium on*, pages 235–246. IEEE, 2010.

[3] Lingqi Zhang, Mohamed Wahib, Haoyu Zhang, and Satoshi Matsuoka.
A study of single and multi-device synchronization methods in nvidia gpus.
In *2020 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 483–493. IEEE, 2020.