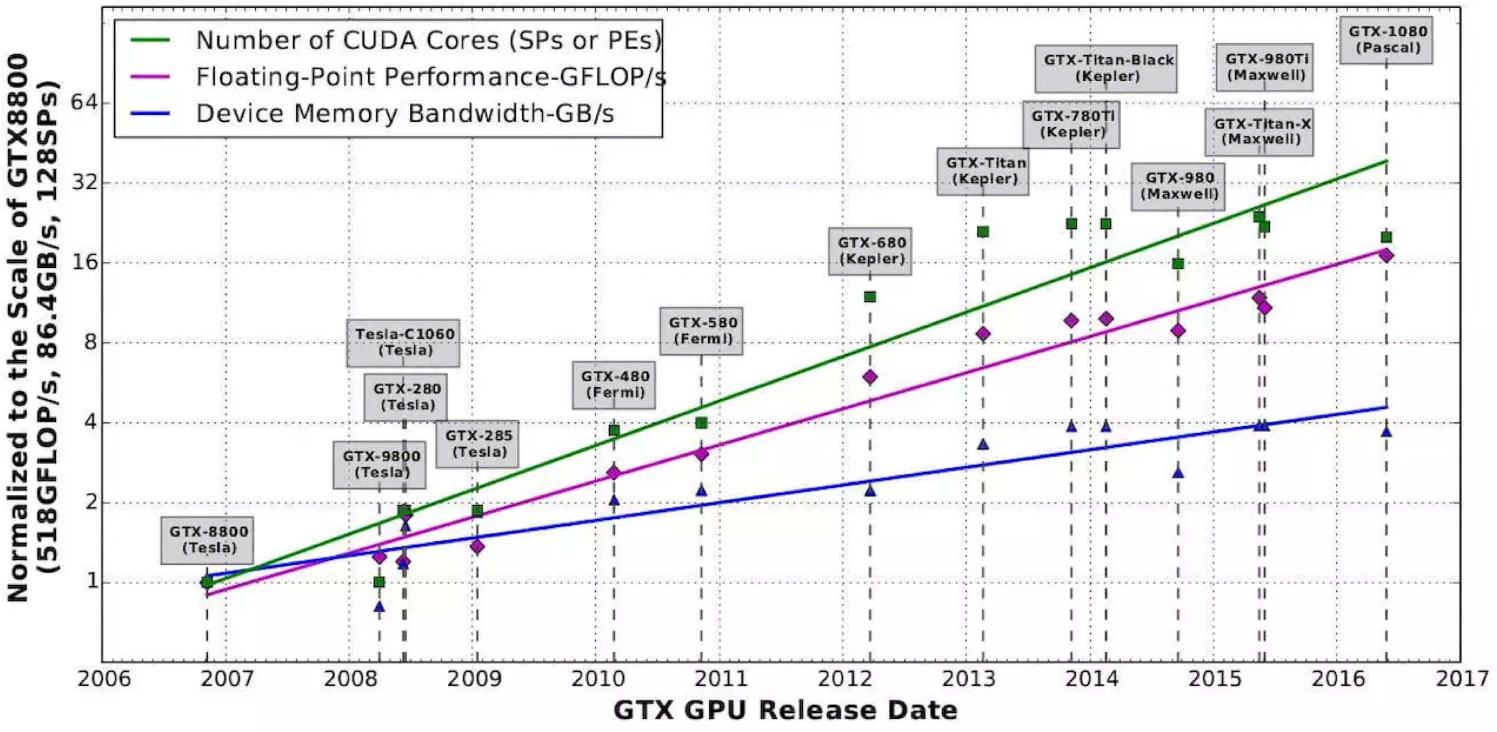
TESLA V100 GPU

Xudong Shao
Houxiang Ji
Hao Gao

The history of GPU architecture



2017
Volta architecture

Figure 1.5: The Scaling of NVIDIA GTX Products for Desktop Utilizations

Reference & Credit: Erik Lindholm,John Nickolls,Stuart Oberman,John Montrym, NVIDA Tesla: A Unified Graphics And Computing Architecture

Components of GPU

➢ host interface

- vertex work
- pixel fragments work
- compute work

➢ TPC
texture/processor clusters
numbers --> performance

➢ Unification
Starts from Tesla architecture

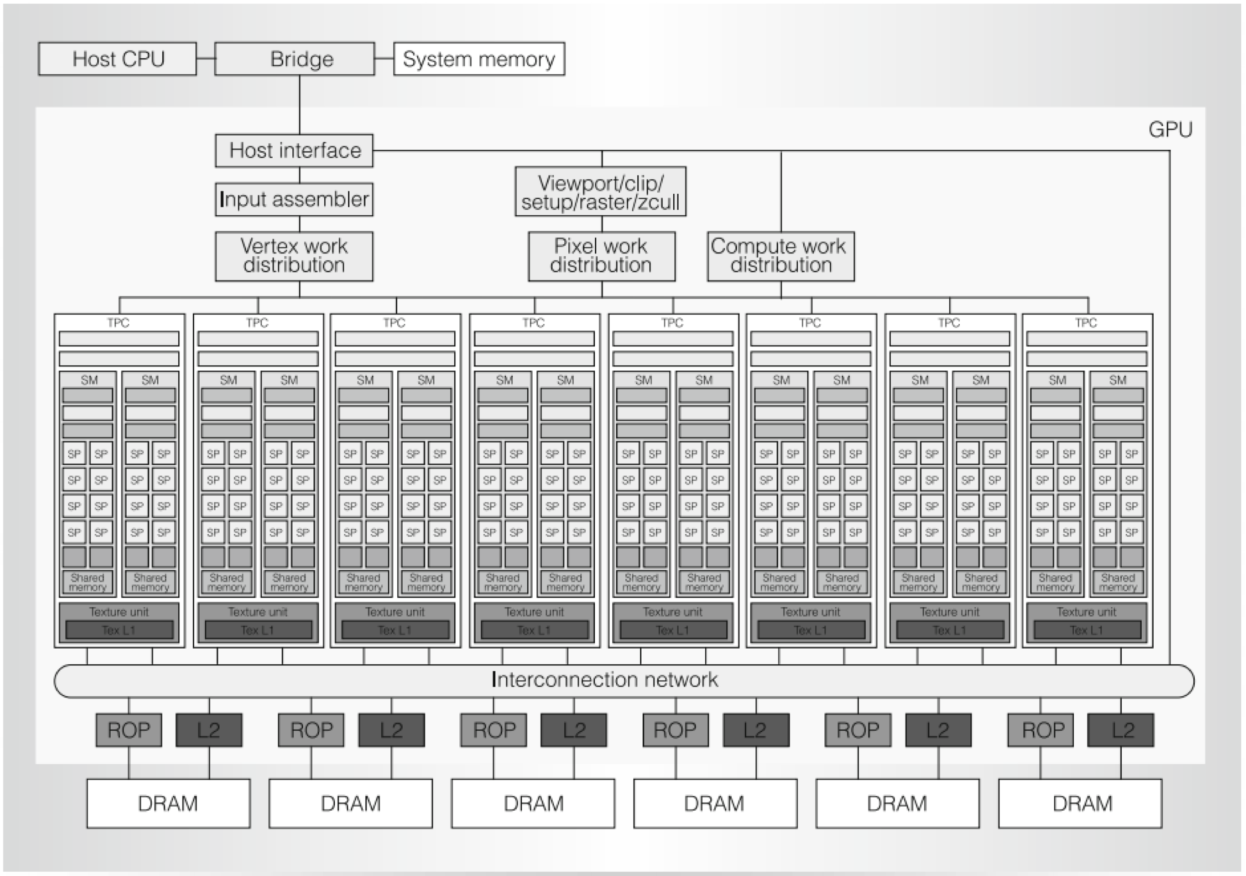


Figure 1. Tesla unified graphics and computing GPU architecture. TPC: texture/processor cluster; SM: streaming multiprocessor; SP: streaming processor; Tex: texture, ROP: raster operation processor.

Reference & Credit: Erik Lindholm,John Nickolls,Stuart Oberman,John Montrym, NVIDA Tesla: A Unified Graphics And Computing Architecture

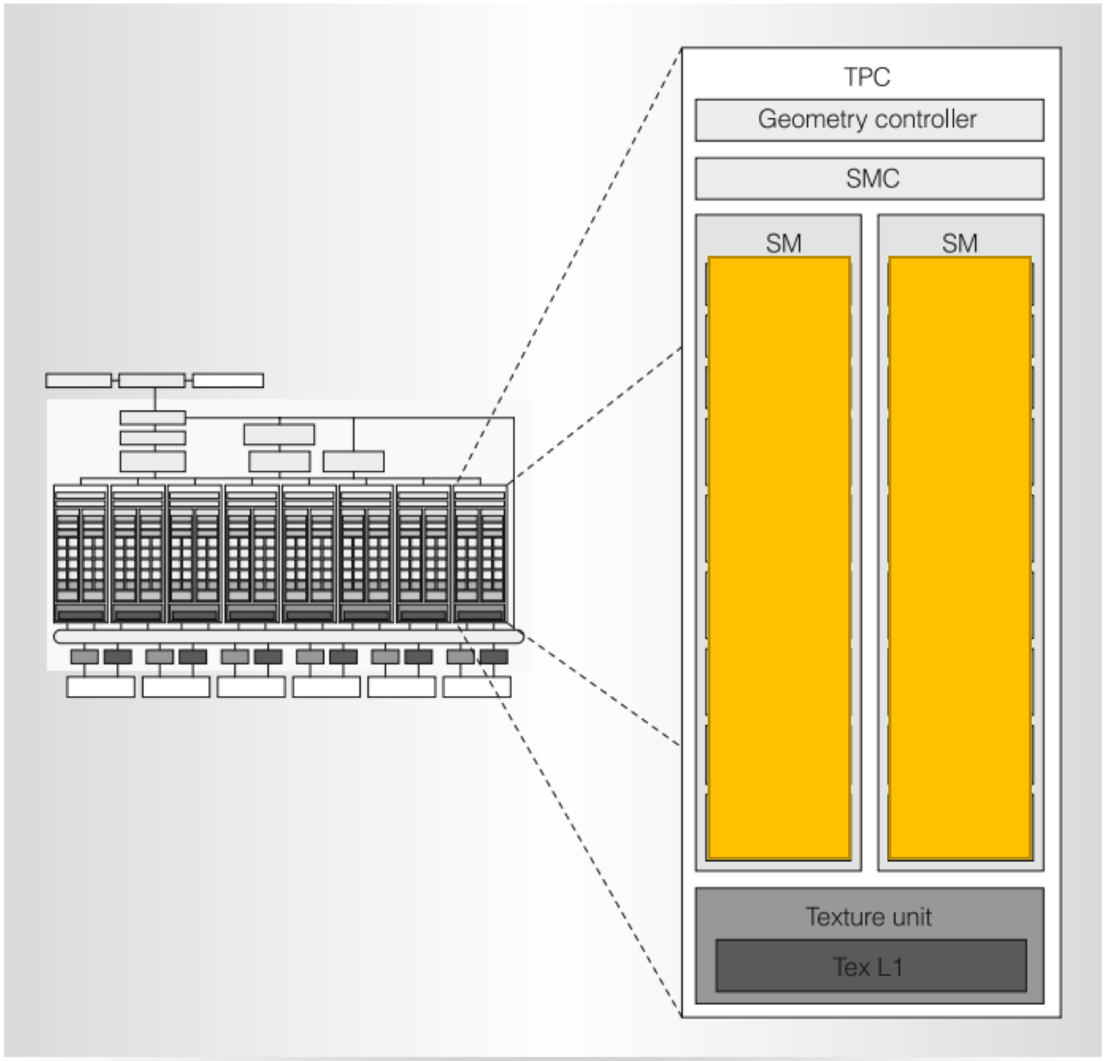
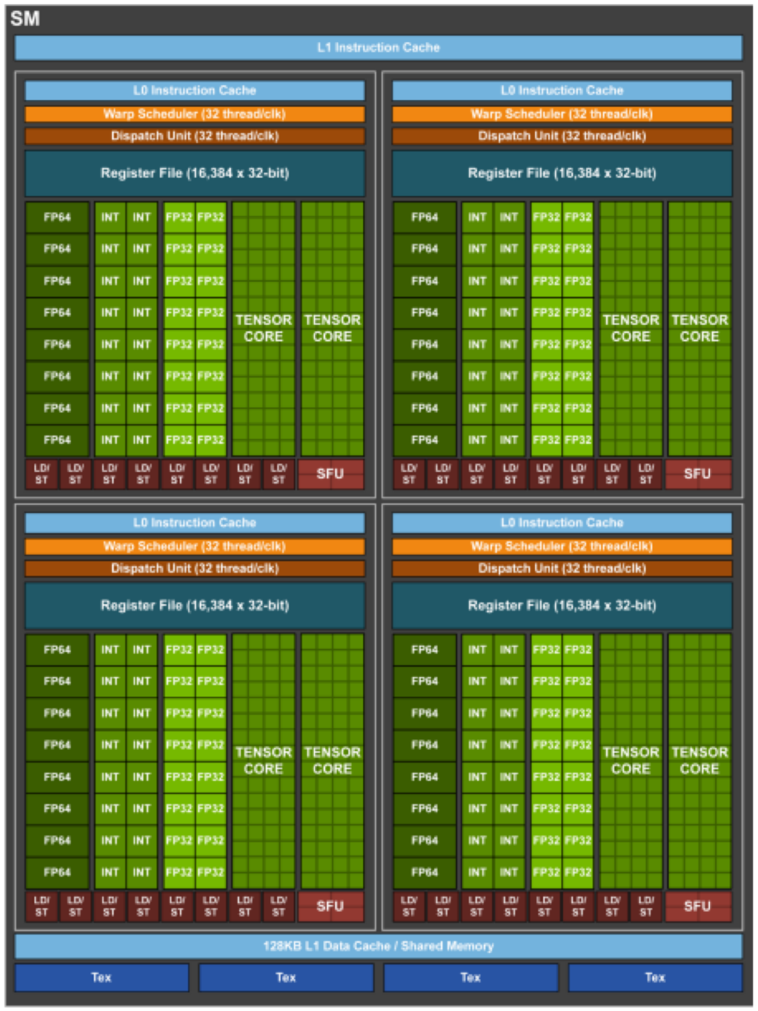


Figure 2. Texture/processor cluster (TPC).

- Geometry controller
- SMC

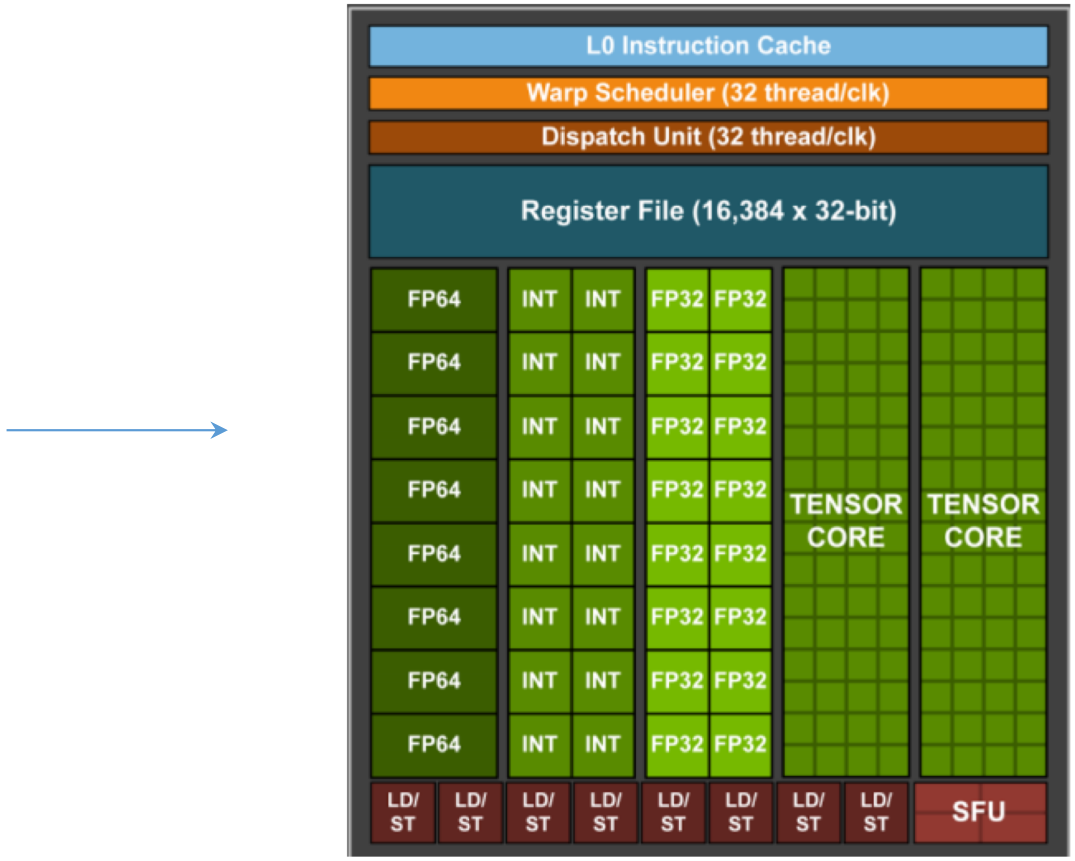
  Streaming multiprocessor controller
- Texture unit

Reference & Credit: Erik Lindholm,John Nickolls,Stuart Oberman,John Montrym, NVIDA Tesla: A Unified Graphics And Computing Architecture



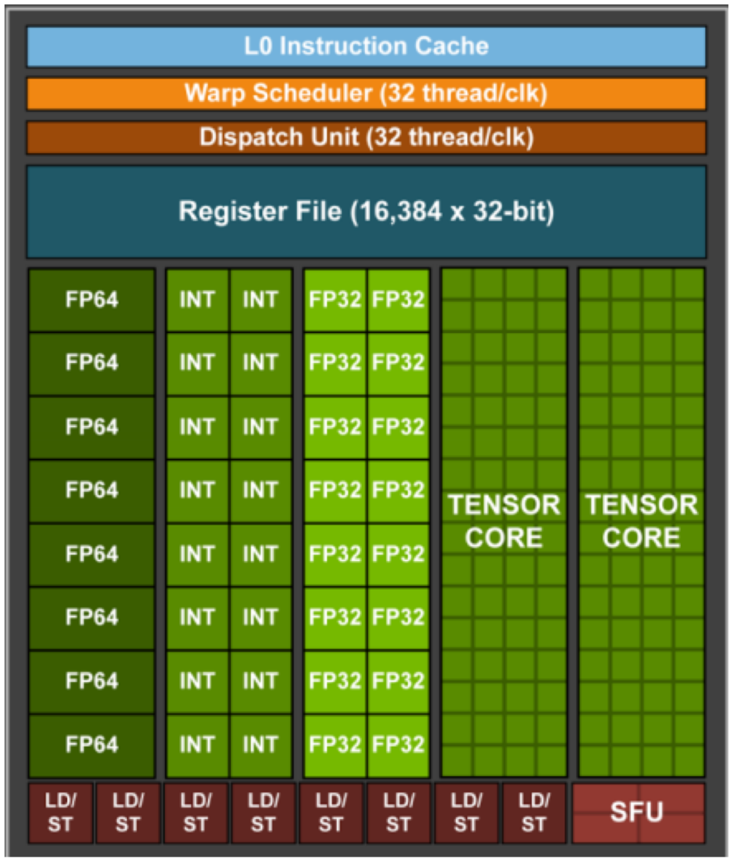
SM
L1 Instruction Cache
L0 Instruction Cache
Warp Scheduler (32 thread/clk)
Dispatch Unit (32 thread/clk)
Register File (16,384 x 32-bit)
FP64
INT
FP32
TENSOR CORE
LD/ ST
SFU
128KB L1 Data Cache / Shared Memory
Tex

Figure 5. Volta GV100 Streaming Multiprocessor (SM)



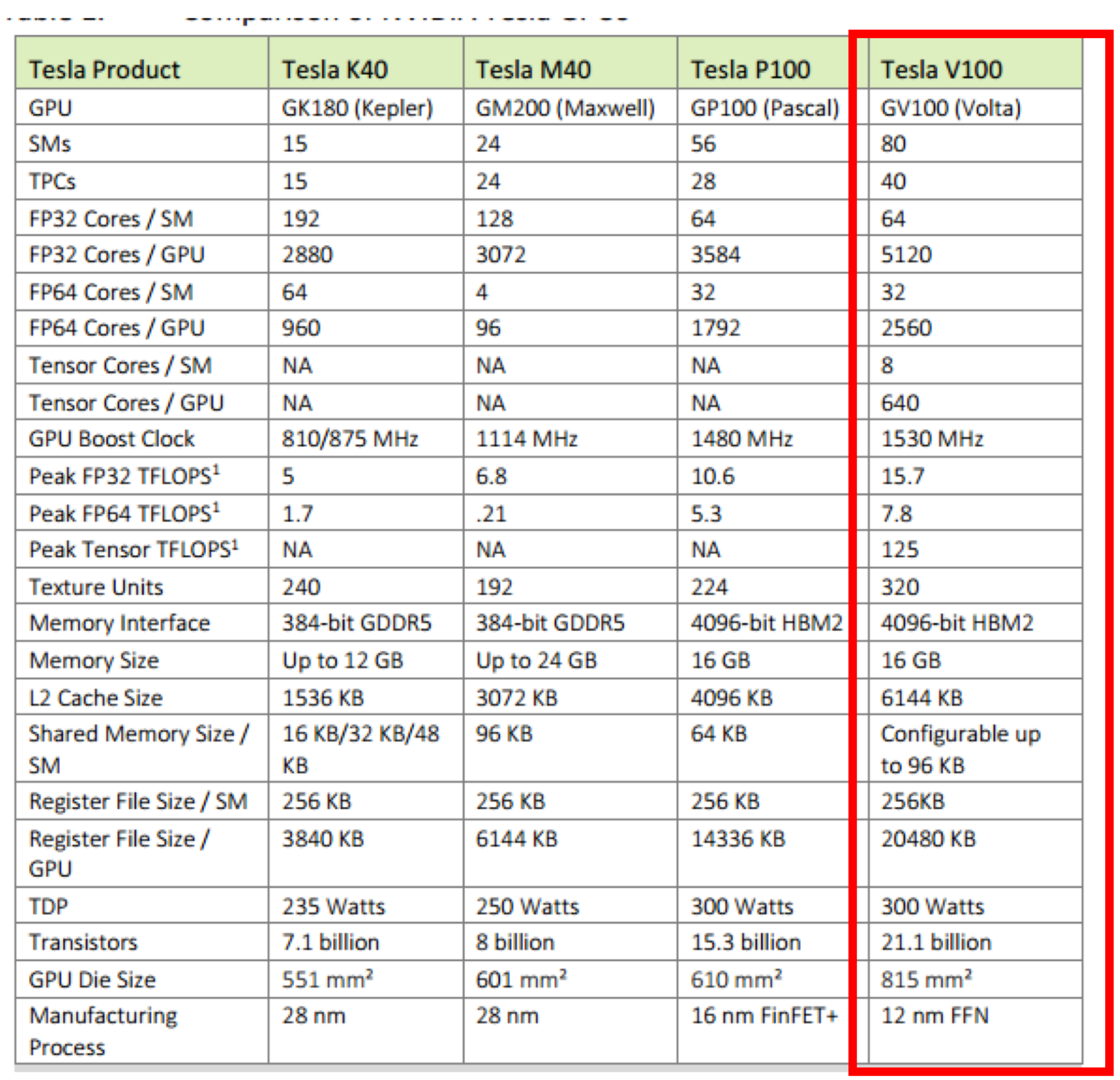
L0 Instruction Cache
Warp Scheduler (32 thread/clk)
Dispatch Unit (32 thread/clk)
Register File (16,384 x 32-bit)
FP64
INT
FP32
TENSOR CORE
LD/ ST
SFU

Reference & Credit: Nvidia Tesla V100 GPU Architecture, The World's Most Advanced Data Center GPU. NVIDIA Corporation, 2017


L0 Instruction Cache
Warp Scheduler (32 thread/clk)
Dispatch Unit (32 thread/clk)
Register File (16,384 x 32-bit)
FP64
INT
FP32
TENSOR CORE
LD/ST
SFU

- FP64 cores
- FP32 cores
- INT32 cores
- LD/ST
- Register File
- SFU

Special-Function-Unit (sin,cos,etc)
- Cache,memory,tensor core (introduced later)
- Warp Scheduler



Tesla Product	Tesla K40	Tesla M40	Tesla P100	Tesla V100
GPU	GK180 (Kepler)	GM200 (Maxwell)	GP100 (Pascal)	GV100 (Volta)
SMs	15	24	56	80
TPCs	15	24	28	40
FP32 Cores / SM	192	128	64	64
FP32 Cores / GPU	2880	3072	3584	5120
FP64 Cores / SM	64	4	32	32
FP64 Cores / GPU	960	96	1792	2560
Tensor Cores / SM	NA	NA	NA	8
Tensor Cores / GPU	NA	NA	NA	640
GPU Boost Clock	810/875 MHz	1114 MHz	1480 MHz	1530 MHz
Peak FP32 TFLOPS[1]	5	6.8	10.6	15.7
Peak FP64 TFLOPS[1]	1.7	.21	5.3	7.8
Peak Tensor TFLOPS[1]	NA	NA	NA	125
Texture Units	240	192	224	320
Memory Interface	384-bit GDDR5	384-bit GDDR5	4096-bit HBM2	4096-bit HBM2
Memory Size	Up to 12 GB	Up to 24 GB	16 GB	16 GB
L2 Cache Size	1536 KB	3072 KB	4096 KB	6144 KB
Shared Memory Size / SM	16 KB/32 KB/48 KB	96 KB	64 KB	Configurable up to 96 KB
Register File Size / SM	256 KB	256 KB	256 KB	256KB
Register File Size / GPU	3840 KB	6144 KB	14336 KB	20480 KB
TDP	235 Watts	250 Watts	300 Watts	300 Watts
Transistors	7.1 billion	8 billion	15.3 billion	21.1 billion
GPU Die Size	551 mm²	601 mm²	610 mm²	815 mm²
Manufacturing Process	28 nm	28 nm	16 nm FinFET+	12 nm FFN

Reference & Credit: Nvidia Tesla V100 GPU Architecture, The World's Most Advanced Data Center GPU. NVIDIA Corporation, 2017

SM multithreading

- ➢ single-instruction multiple-thread (SIMT)
- thread block
- warp (32 threads)
- active mask
- its own instruction address and register state
- select a warp and issue the next instruction

- ➢ Independent thread scheduling for volta architecture

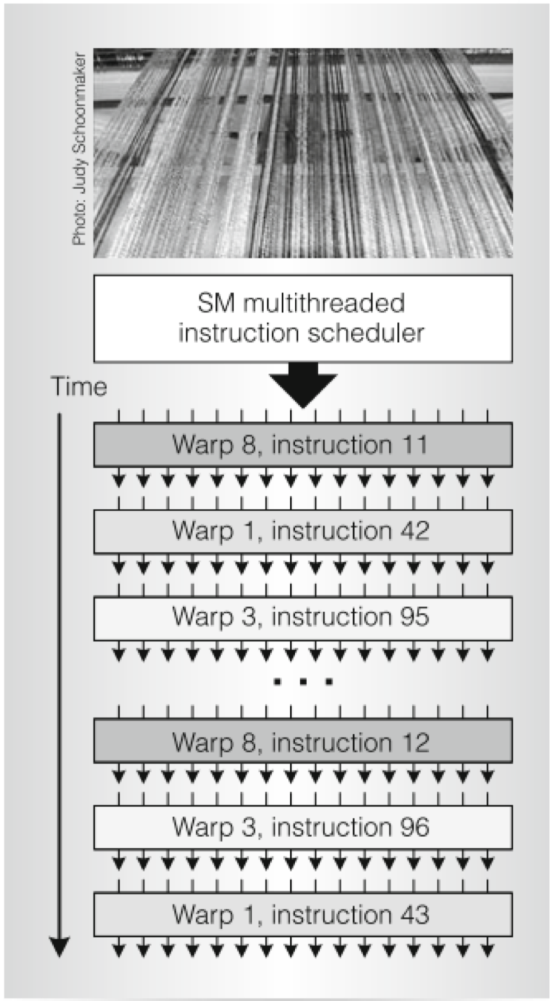
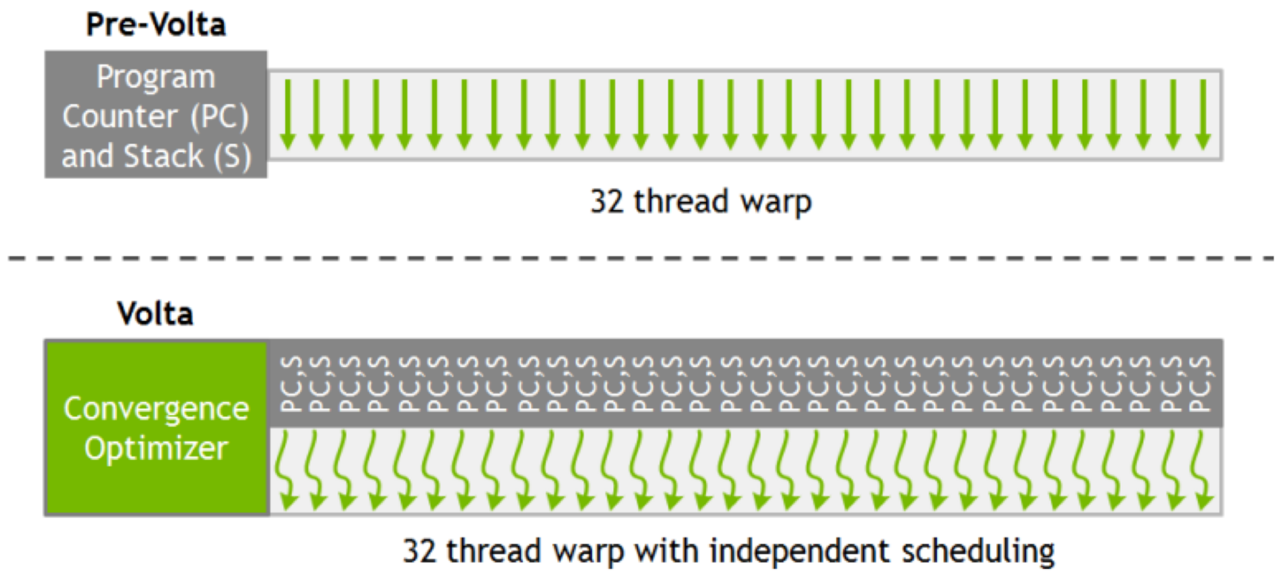


Figure 4. Single-instruction, multiple-thread (SIMT) warp scheduling.

Reference & Credit: Erik Lindholm,John Nickolls,Stuart Oberman,John Montrym, NVIDA Tesla: A Unified Graphics And Computing Architecture

➢Independent thread scheduling for volta architecture


Pre-Volta
Program Counter (PC) and Stack (S)
32 thread warp
Volta
Convergence Optimizer
PC,S
32 thread warp with independent scheduling

Volta (bottom) independent thread scheduling architecture block diagram compared to Pascal and earlier architectures (top). Volta maintains per-thread scheduling resources such as program counter (PC) and call stack (S), while earlier architectures maintained these resources per warp.

Figure 21. Volta Warp with Per-Thread Program Counter and Call Stack

Reference & Credit: Nvidia Tesla V100 GPU Architecture, The World's Most Advanced Data Center GPU. NVIDIA Corporation, 2017

Its own program counter and call stack.

GPU Memory Hierarchy

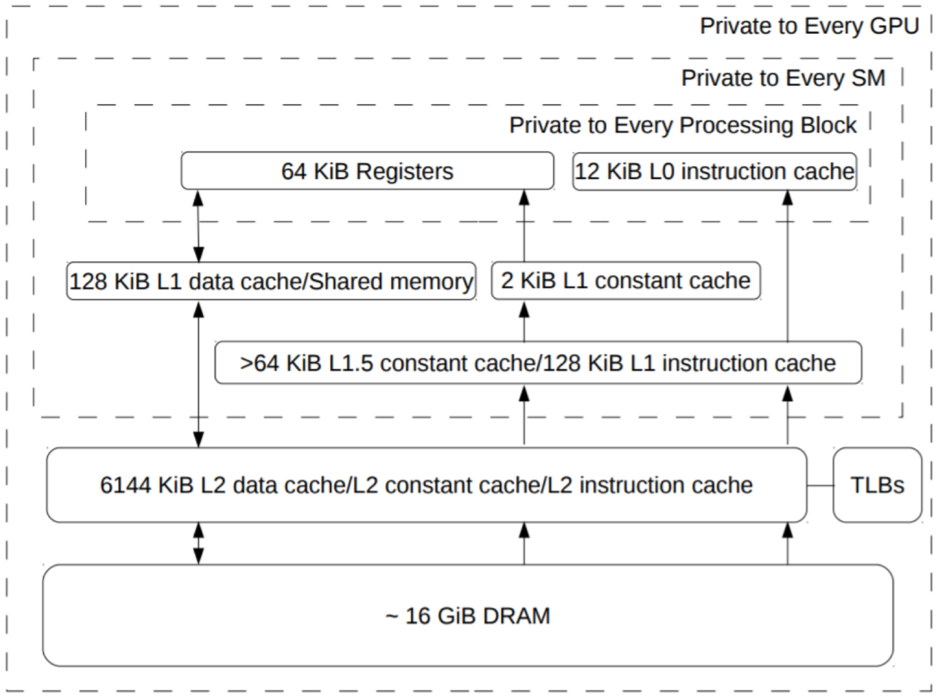

Private to Every GPU
Private to Every SM
Private to Every Processing Block
64 KiB Registers
12 KiB L0 instruction cache
128 KiB L1 data cache/Shared memory
2 KiB L1 constant cache
>64 KiB L1.5 constant cache/128 KiB L1 instruction cache
6144 KiB L2 data cache/L2 constant cache/L2 instruction cache
TLBs
~ 16 GiB DRAM

Figure 3.1: Memory hierarchy of the Volta V100 GPU (GV100).

Reference & Credit: Jia, Z., Maggioni, M., Staiger, B., & Scarpazza, D. P. (2018). Dissecting the NVIDIA Volta GPU Architecture via Microbenchmarking. arXiv preprint arXiv:1804.06826.

Where can we get information?

- Published by Nvidia: official but limited

  [1] Nvidia Tesla V100 GPU Architecture, The World's Most Advanced Data Center GPU. NVIDIA Corporation, 2017.

  [2] Pascal GP100 Whitepaper. NVIDIA Corporation, 2016.

  [3] Lindholm, E., Nickolls, J., Oberman, S., & Montrym, J. (2008). NVIDIA Tesla: A unified graphics and computing architecture. IEEE micro, 28(2).

  [4] CUDA C Programming Guide, NVIDIA Corporation, 2018.

  [5] CUDA C Best Practices Guide, NVIDIA Corporation, 2018.

- Microbenchmarking

  [6]: X. Mei and X. Chu, "Dissecting GPU memory hierarchy through microbenchmarking," IEEE Transactions on Parallel and Distributed Systems, vol. 28, no. 1, pp. 72–86, Jan 2017.

  [7]: Jia, Z., Maggioni, M., Staiger, B., & Scarpazza, D. P. (2018). Dissecting the NVIDIA Volta GPU Architecture via Microbenchmarking. arXiv preprint arXiv:1804.06826.

Registers

- Virtual Registers

  Two levels of assembly: PTX and SASS. Difference?

  Sample PTX and SASS for vector addition

  The intermediate language (PTX) use virtual registers. Why?

- Size of Register Files

  In GV100, register file is 256KB/SM * 80SMs = 20480KB

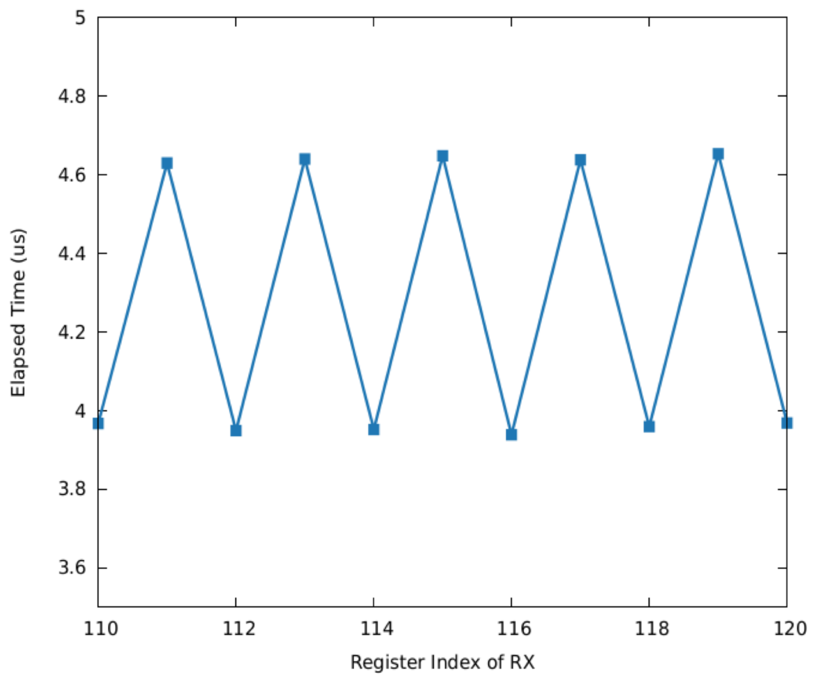
  In comparison, L2 caches only 6144KB

  Why so many registers?

  avoid register spilling

Registers

- The register file is divided into 2 banks, each bank 64 bits
  Use microbenchmark “FFMA R6, R97, R99, RX”.



Caches

- Data Cache Structure
  L1 cache on each SM
  L2 cache shared among all SMs
- Latency
  L1 cache hit: 28 cycles
  L2 cache hit: 193 cycles
  L2 cache miss with TLB hit: 375 cycles
  L2 cache miss with TLB miss: 1029 cycles
- L1 Cache
  Volta architecture features combined L1 data cache and shared memory
  (difference between L1 cache and shared memory?)

Caches

- L1 Cache (continued)

  Replacement policy: Not simply LRU.

  The same four cache lines from 4 cache set have lowest preservation priority.
- L2 Cache

  total size 6144KB; 16-way set-associative cache; cache line size 64B
- TLBs

  L1 data cache is indexed by virtual addresses;

  L2 data cache is indexed by physical addresses

  Two levels of TLB:

  L1 TLB: 2M page entries, 32M of coverage

  L2 TLB: ~8192MB coverage.

Shared Memory

- Shared within a threadblock
- Specified explicitly by programmer

```
__global__ void kernel(...)
{
    __shared__ float shared_memory[1024];
    load global memory into shared memory
    __syncthreads();
    actual computation
}
```

- configurable, up to 96KB
- shared memory bank

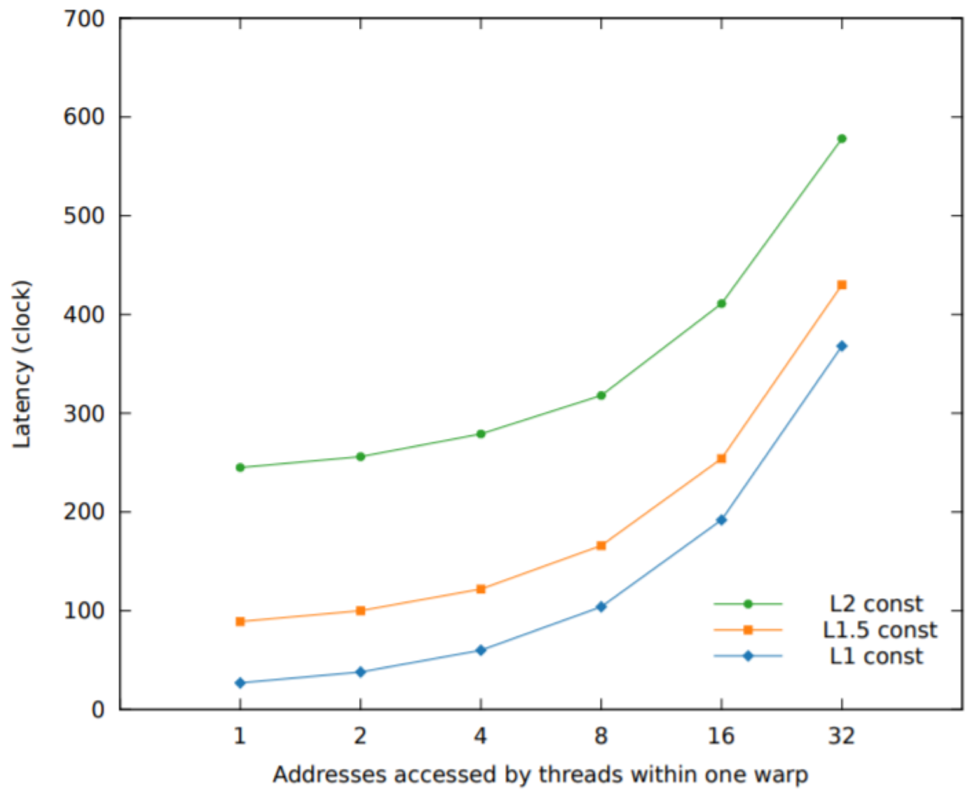
Constant Memory

- Resides on device memory but cached in the constant cache
- Cache hit -> throughput of constant cache

  Cache miss -> throughput of device memory
- Constant memory supports broadcasting: when all threads in a warp access

  the same location -> simultaneous

  diverging addresses -> serialized



Global Memory

- Memory Coalescing:
  Memory accesses from the same warp coalesced into fewer memory block accesses. (fall in the same block, meet alignment criteria)
- HBM2 Memory
  2.5D design
  better bandwidth, but slower
  energy efficient
  smaller form factor

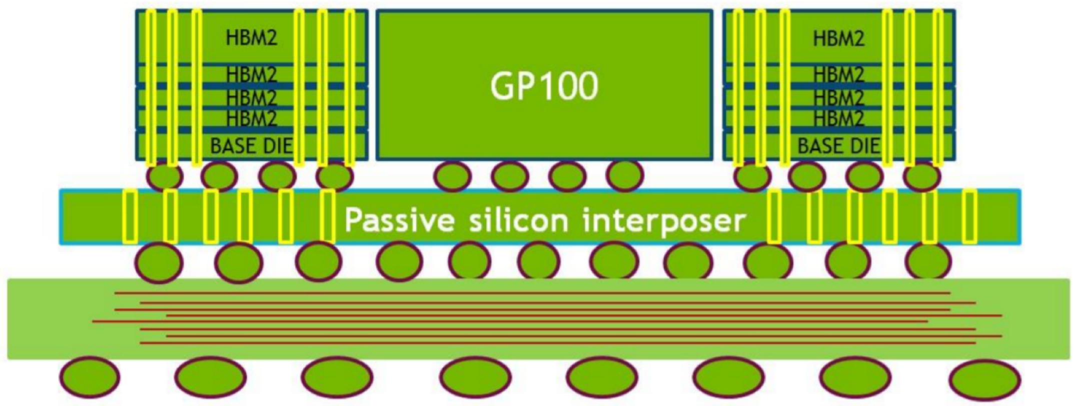
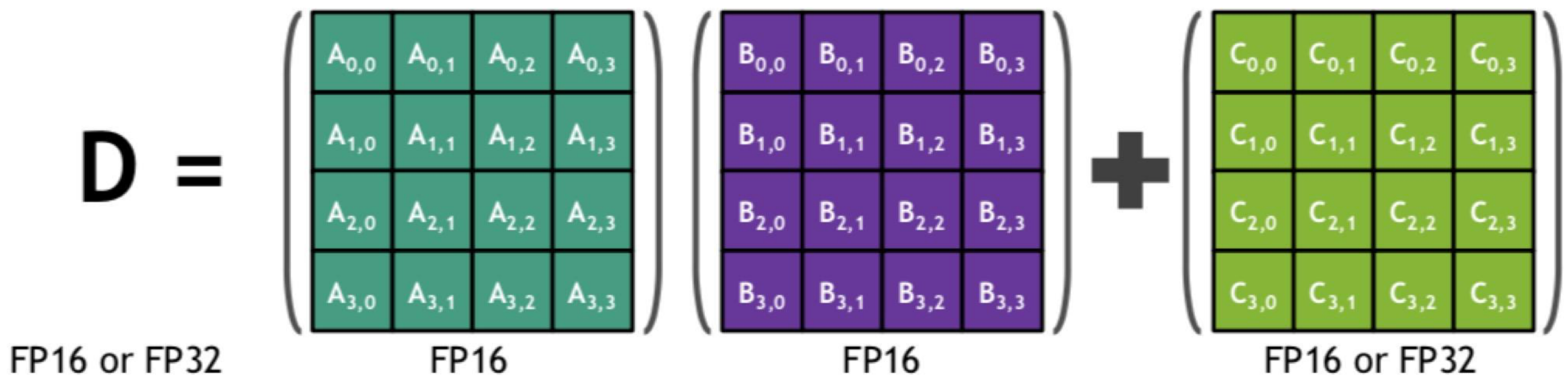


Figure 9. Cross-section Illustrating GP100 adjacent HBM2 stacks

What’s Tensor Core

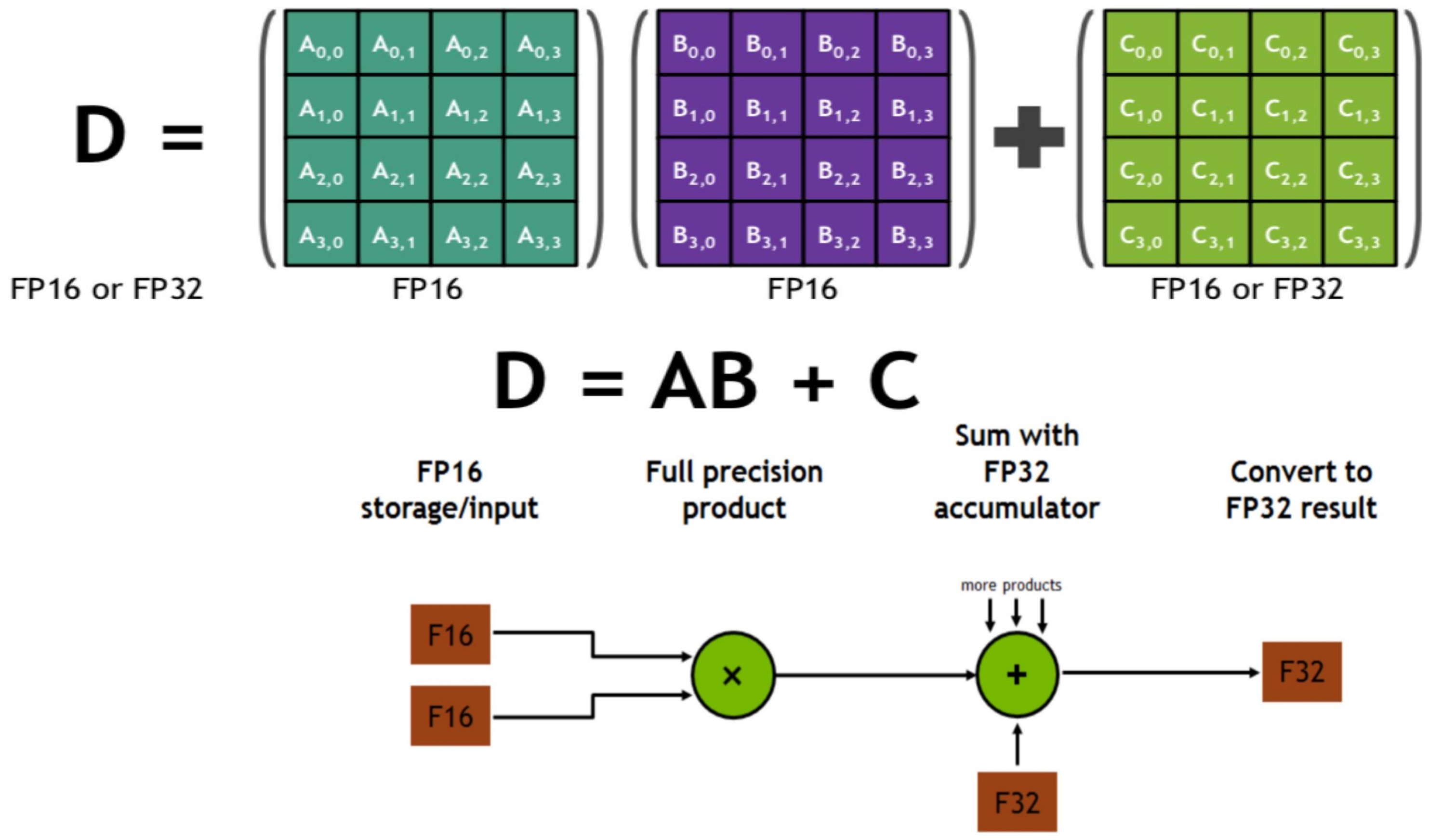
4x4x4 Warp Matrix Multiply and Accumulate (WMMA)



$$D = AB + C$$

Tensor Core

Mixed-precision Operation



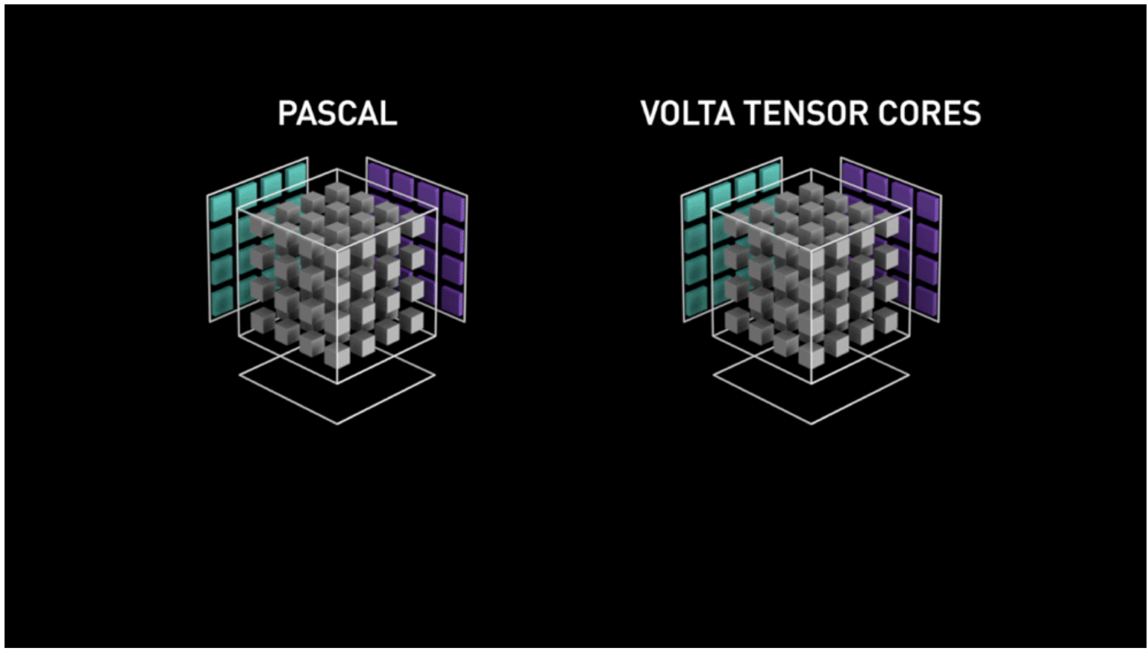
Power of Tensor Core

640 Tensor Cores on V100

64 FP FMA per Core per Cycle

125 Tensor TFLOPS for DL

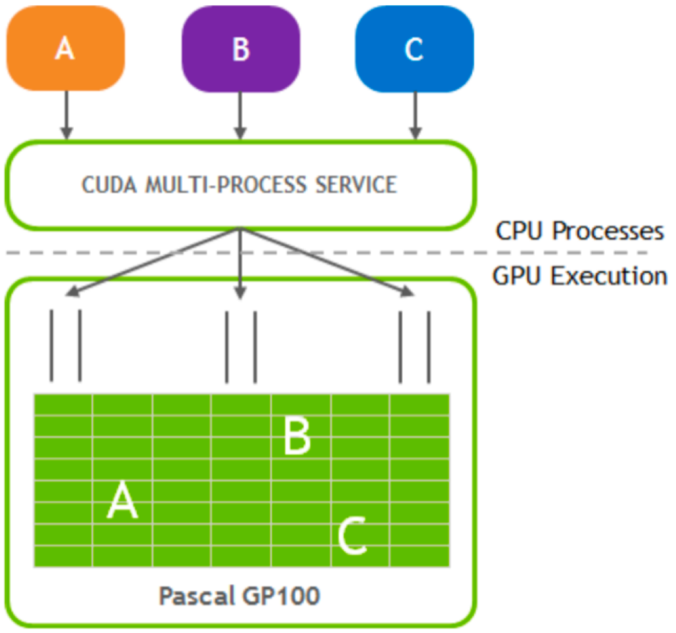
12x throughput over Pascal

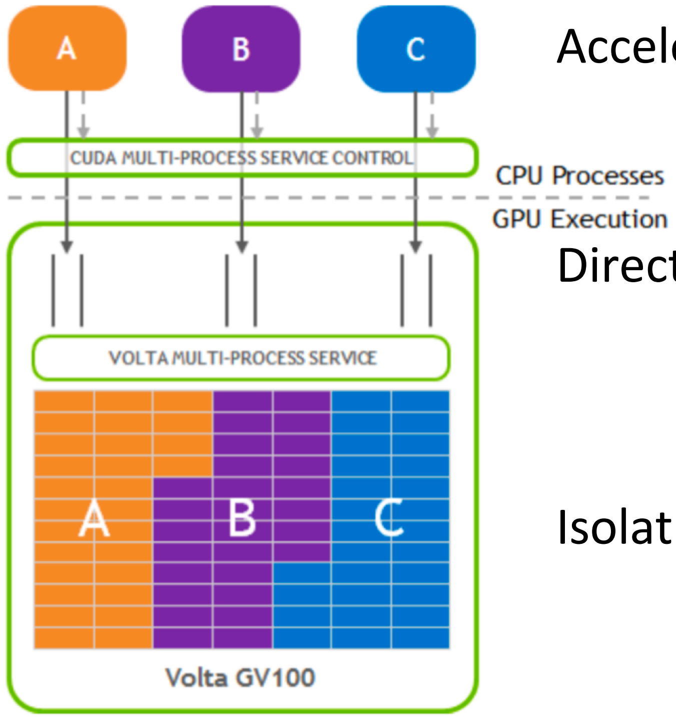


Multi-Process Service (MPS)

Software-based

Intermediary




A
B
C
CUDA MULTI-PROCESS SERVICE CONTROL
CPU Processes
GPU Execution
VOLTA MULTI-PROCESS SERVICE
A
B
C
Volta GV100

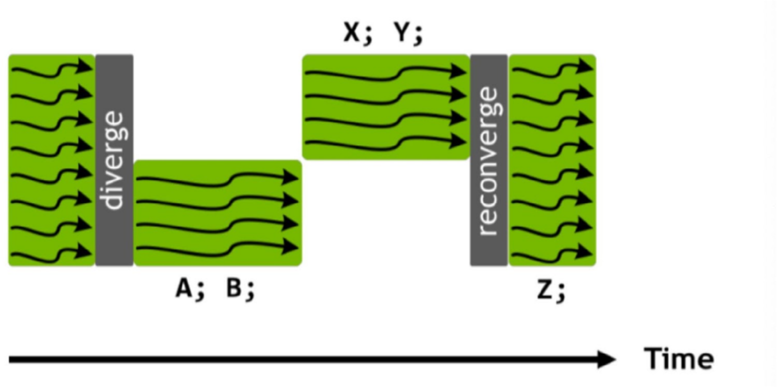
Hardware Accelerated

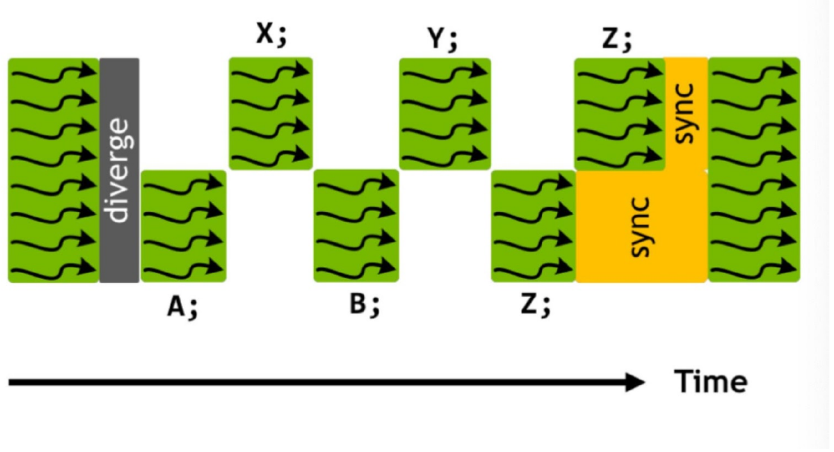
Direct Submission

Isolation

Independent Thread Scheduling

```
if (threadIdx.x < 4) {
    A;
    B;
} else {
    X;
    Y;
}
Z;
__syncwarp()
```


X; Y;
diverge
reconverge
A; B;
Z;
Time


X;
Y;
Z;
diverge
sync
sync
A;
B;
Z;
Time