

CUDA. Танилцуулга

F.CS306 ПАРАЛЛЕЛ ПРОГРАММЧЛАЛ – Лекц 11

Г.ГАНБАТ
ganbatg@must.edu.mn

Хичээлийн агуулга

- ❖ GPU, GPGPU, CUDA
- ❖ CUDA Hello world
- ❖ CUDA-Enabled архитектур
- ❖ VRAM, bandwith
- ❖ Тооцоолох нөөцийн зохион байгуулалт

Хураангуй

- ❖ Graphics Processing Unit (GPU)
 - ❖ 80-аад оны дунд: GPU-ын тооцоолох боломжийн хөгжил нь илүү комплекс дэлгэцийг харуулах болсноор 3D тоглоомууд дэлгэрсэн.
 - ❖ 80-аад оноос хойш: 3D дэлгэцийг үр ашигтай дүрслэхийн тулд геометрийн үйлдлүүдийн боловсруулалтыг дэмждэг болсон.
 - ❖ 00 оноос хойш: Shader languages гарсан (hard-wired).
- ❖ GPGPU – General-purpose computing on GPUs
 - ❖ GPU – ийн хөгжиж буй тооцоолох чадлыг Shader languages-ээр бичсэн ерөнхий зориулалтын алгоритмуудад хэрэглэх.
- ❖ CUDA – Compute Unified Device Architecture, 2007, NVIDIA
 - ❖ C- or FORTRAN төрлийн хэл, энгийн код,
 - ❖ single-threaded CPU-тэй харьцуулахад two orders-of-magnitude

CUDA Hello World

- ❖ CPU дээр ажиллах үндсэн функц
- ❖ GPU дээр ажиллах hello_kernel функц

```
#include <stdio.h>
__global__ void hello_kernel() {
    // calculate global thread identifier, note blockIdx.x=0 here
    const int thid = blockDim.x * blockIdx.x + threadIdx.x;
    // print a greeting message
    printf("Hello from thread %d!\n", thid);
}
int main(int argc, char * argv[]) {
    // set the ID of the CUDA device
    cudaSetDevice(0);
    // invoke kernel using 4 threads executed in 1 thread block
    hello_kernel << < 1, 4 >>> ();
    // synchronize the GPU preventing premature termination
    cudaDeviceSynchronize();
}
```

CUDA Hello World

- ❖ cudaSetDevice: CUDA – capable GPU-г сонгоно
- ❖ hello_kernel «« () »» () : GPU дээр параллел ажиллах 4 thread бүхий 1 блокийг дуудна
- ❖ cudaDeviceSynchronize() : Кернел нь GPU (device) дээр, код нь CPU (host) дээр тус тусдаа ажиллана. Device дээрх кернелийг ажиллаж дуустал host-ийг хүлээлгэнэ.
- ❖ global : host-oos дуудагддаг ба device дээр ажилладаг функц.
 - ❖ host : host-oos дуудагддаг ба host дээр ажилладаг,
 - ❖ device : device-aac дуудагддаг ба device дээр ажилладаг

CUDA Hello World

- ❖ Дараах командаар алгоритмыг ажиллуулах :

```
nvcc hello_world.cu -O2 -o hello_world
```

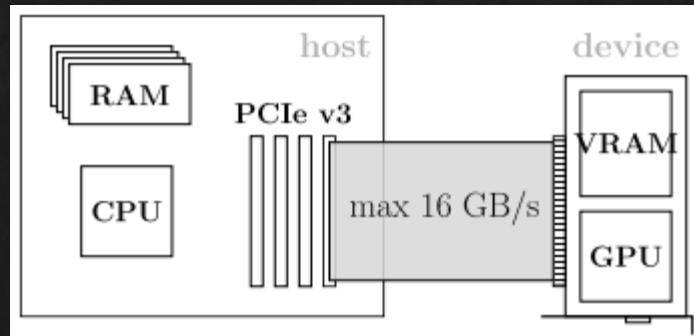
- ❖ Yr дүн

```
Hello from thread 0!  
Hello from thread 1!  
Hello from thread 2!  
Hello from thread 3!
```

- ❖ Thread-үүдийн **warp** : Хэвлэгчид нь thread блокт дараалсан 32 thread бүхий багцуудад цуварсан хэлбэрээр байдаг болохоор дараалсан юм шиг хэвлэсэн

CUDA-Enabled GPUs архитектур

- ❖ host болон device хоорондын мэдээлэл солилцоо
 - ❖ 16 GB/s нь хурдан юм шиг боловч CUDA программуудын хувьд bottleneck үүсэх үндсэн асуудал.
 - ❖ NVLink : 80 GB/s хүртлэх bandwidth боломж
- ❖ host-ын RAM болон device-ын VRAM (video memory) тусдаа байдаг
 - ❖ GPU дээр боловсруулагдах өгөгдлийг host-ooc device-руу зөөнө.
 - ❖ VRAM дээр тооцоологдсон үр дүнг буцууаж дискрүү хуулна.



VRAM болон боломжит Bandwidth

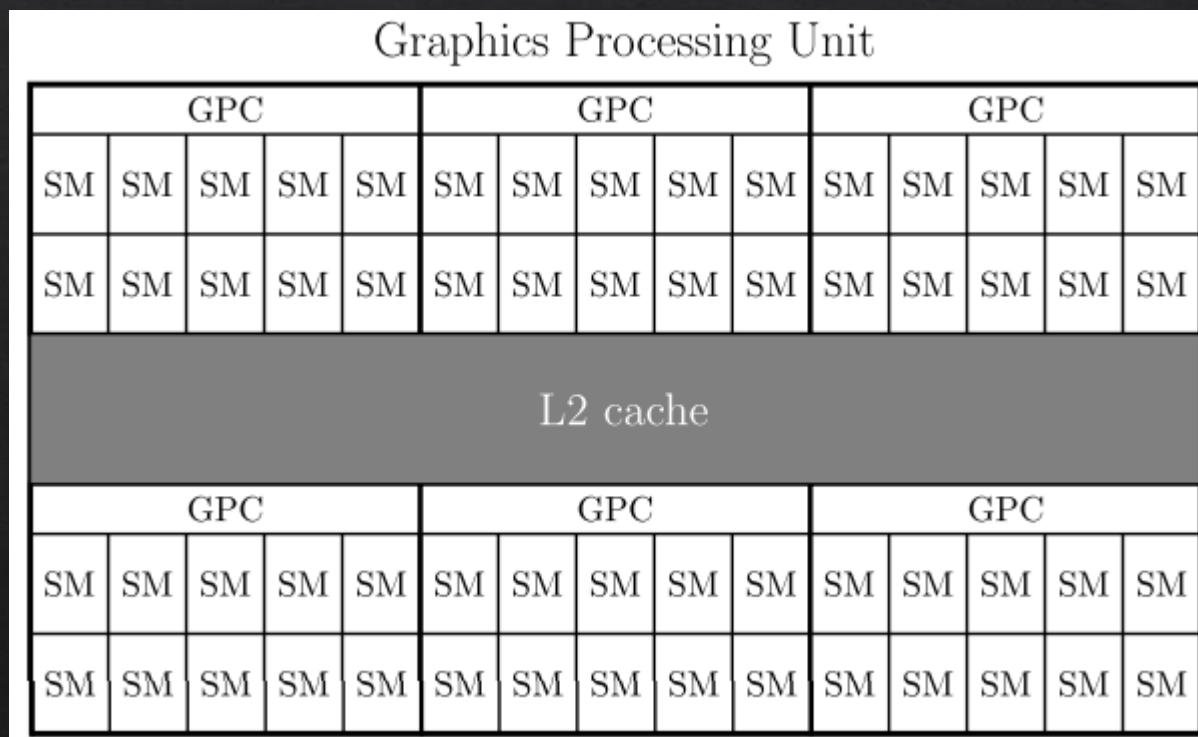
- ❖ График картийг өөрийн тооцоолох нэгж, санах ойтой болохоор бие даасан тооцоолох платформ гэж үзнэ. Үндсэн санах ойтой харьцуулахад хурдан
 - ❖ Xeon workstations **100 GB/s** бага бол 12 GB of GDDR5X DRAM үзүүлэлттэй Pascal-based Titan X **480 GB/s**. Tesla P100 ба Tesla V100 720 болон **900 GB/s**
- ❖ TB/s шахам хурдаар хандах боломжтой байгаа ч санах ойн шатлалыг анхаарах хэрэгтэй.
 - ❖ Жнь single-precision floating point (FP32) утгуудыг VRAM-д уншин нэмэгдүүлж, үр дүнг нь буцааж бичихэд 1 Flop/8 В *compute-to-global-memory-access* харьцаатай байхад Tesla P100 картын хувьд боломжит bandwidth 1% байна. Энд 125 Gflop/s тооцоололох хурд, 1TB/s зөөх хурд

Тооцооллын нөөцийн байгуулалт

- ❖ Multi-core CPU нь комплекс заавартай хэдэн арван цул боловсруулалтын нэгжтэй бол GPU нь хязгаарлагдмал заавар бүхий их хэмжээний хөнгөн боловсруулалтын нэгжийн массив. Орчин үеийн GPU нь мод бүтэцтэй.
1. *Mapping Thread Blocks Onto SMs.* Тус бүр нь арван орчим Streaming Multiprocessors (SMs) агуулсан цөөн тооны Graphics Processing Clusters (GPCs).
 2. *Mapping Threads Into Warps.* SM бүр нь олон тооцоолох болон зааврын нэгжүүдэд задардаг.

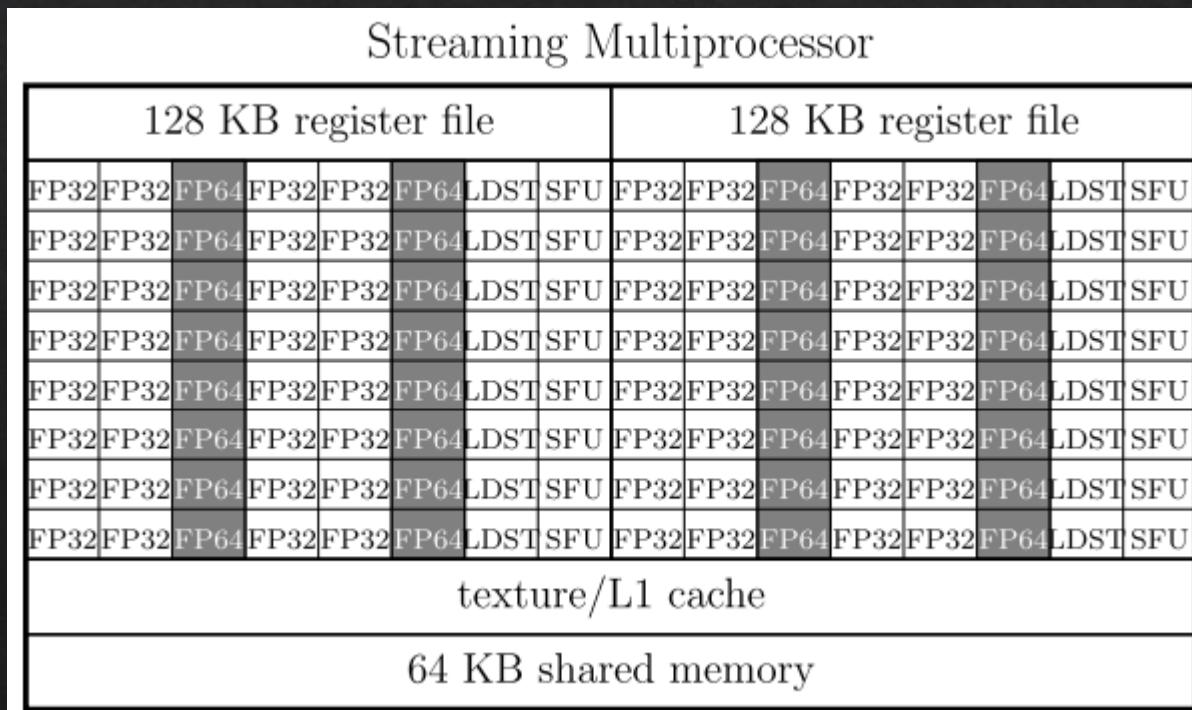
Mapping Thread Blocks Onto SMs

- ❖ Tesla P40 картын GP102 GPU-ын бүтэц. Тус бүр нь 10ш SM удирддаг GPU нь 6ш GPC д хуваагдсан. SM нь ижил L2 кэш ба глобал санах ойг дундаа хэрэлдэг.



Mapping Threads Into Warps.

- ❖ Тус бүр 32ш single-precision (FP32) болон 16ш double-precision (FP64) тооцооллын, 8ш load and store units(LDST), 8 special function units (SFU)-ээс бүрдсэн хоёр блокыг удирддаг. Блок бүр хувьсагчид зориулсан 32,76832-bit регистрт хандана(Нийт 256 KB). SM-үүд L1 кэш, 64 KB санах ойг дундаа хэрэглэгэнэ.



Баярлалаа.