CUDA
ВВЕДЕНИЕ

Романенко А.А.
arom@ccfit.nsu.ru

Новосибирский государственный университет

Мощность вычислительных систем

Рост производительности

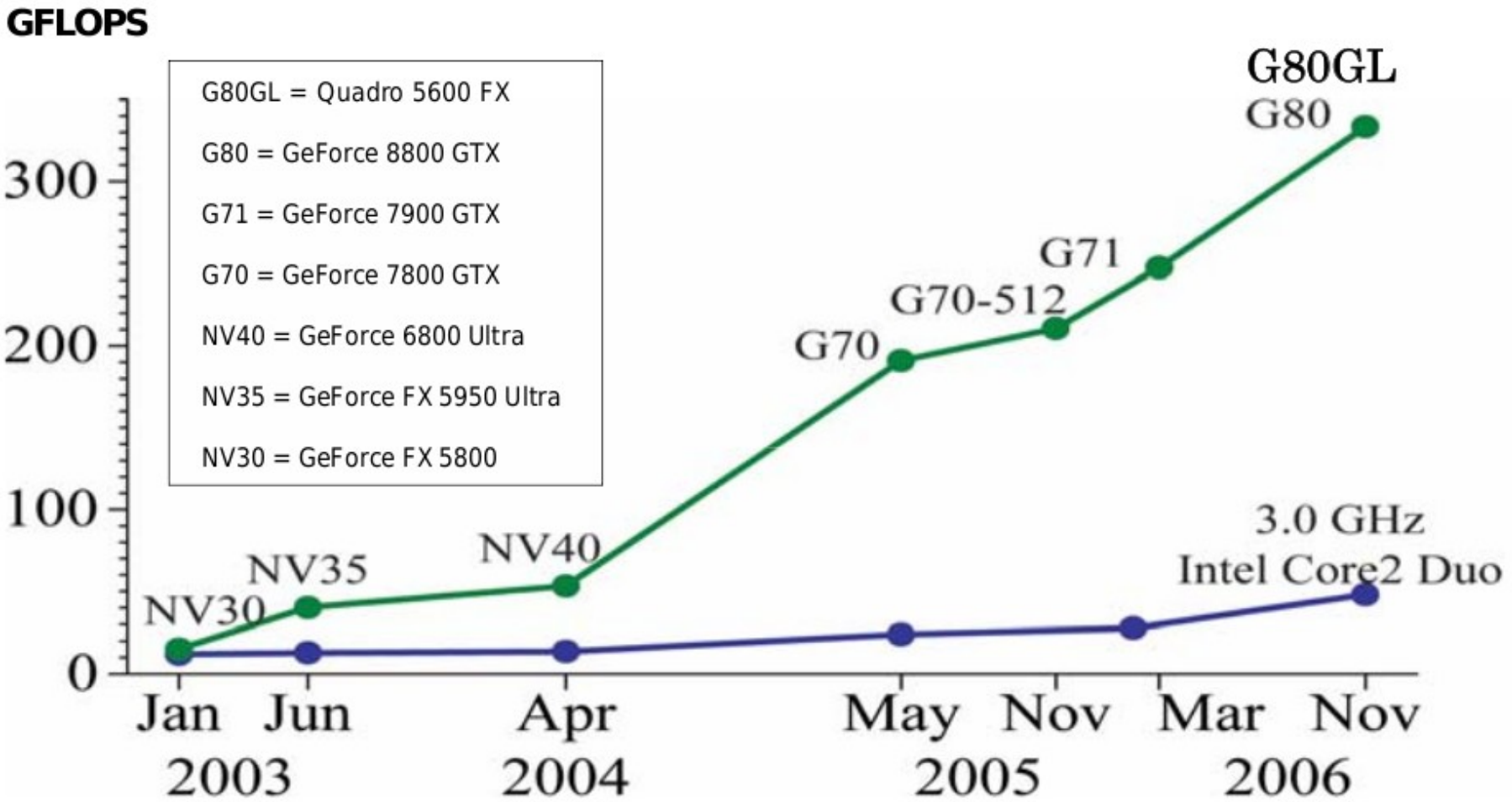
- За счет увеличения частоты процессоров
- За счет увеличения количества ядер/процессоров
- За счет усложнения архитектуры самих процессоров
  - Увеличение количества регистров
  - Изменение длинны конвейера
  - Увеличение разрядности
  - пр.

Обработка графики

- Работа с векторами
- Работа с маленькими матрицами
- Фильтры/post-processing
- Вычисление проекций
- пр.

- Однотипные операции над большим количеством данных

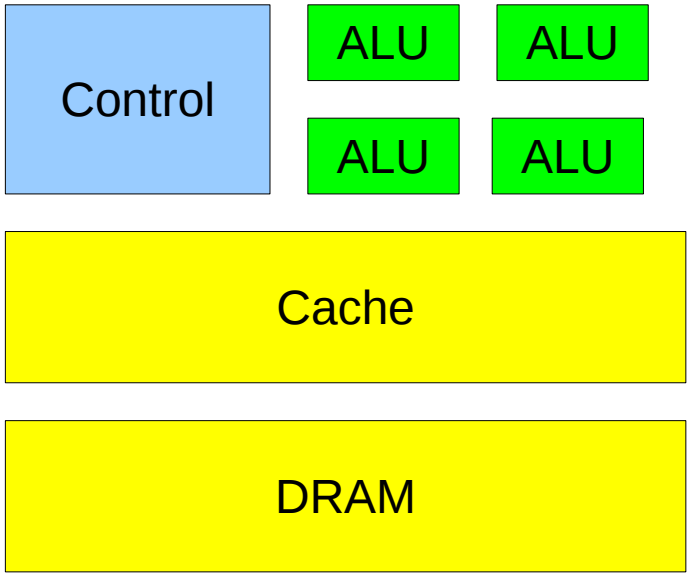
Производительность видеокарт



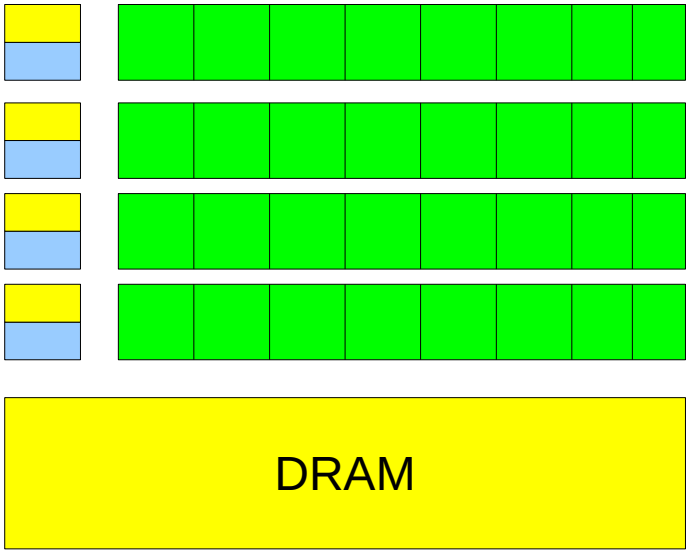
GPU — Graphical Processing Unit

- GPU — процессор на видеокарте. Имеет свою архитектуру
- Программа на GPU не может общаться с хостом
- Программа на GPU не может писать в память хоста
- Загрузка и выгрузка данных на видеокарту производятся через шину PCI Express 1 (2). Передача данных инициируется на стороне хоста
- Видеокарта - сопроцессор

CPU vs. GPU



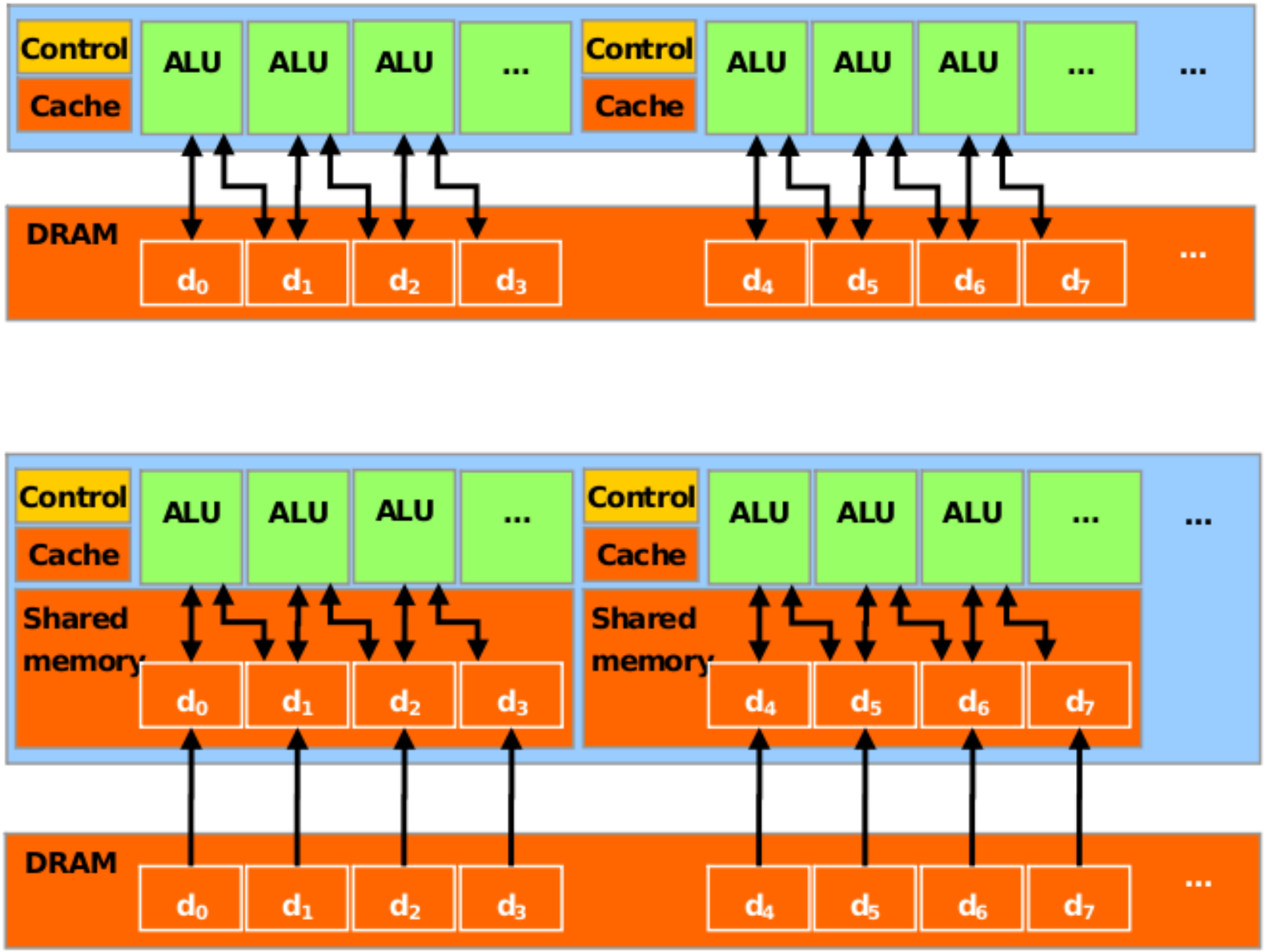
CPU



GPU

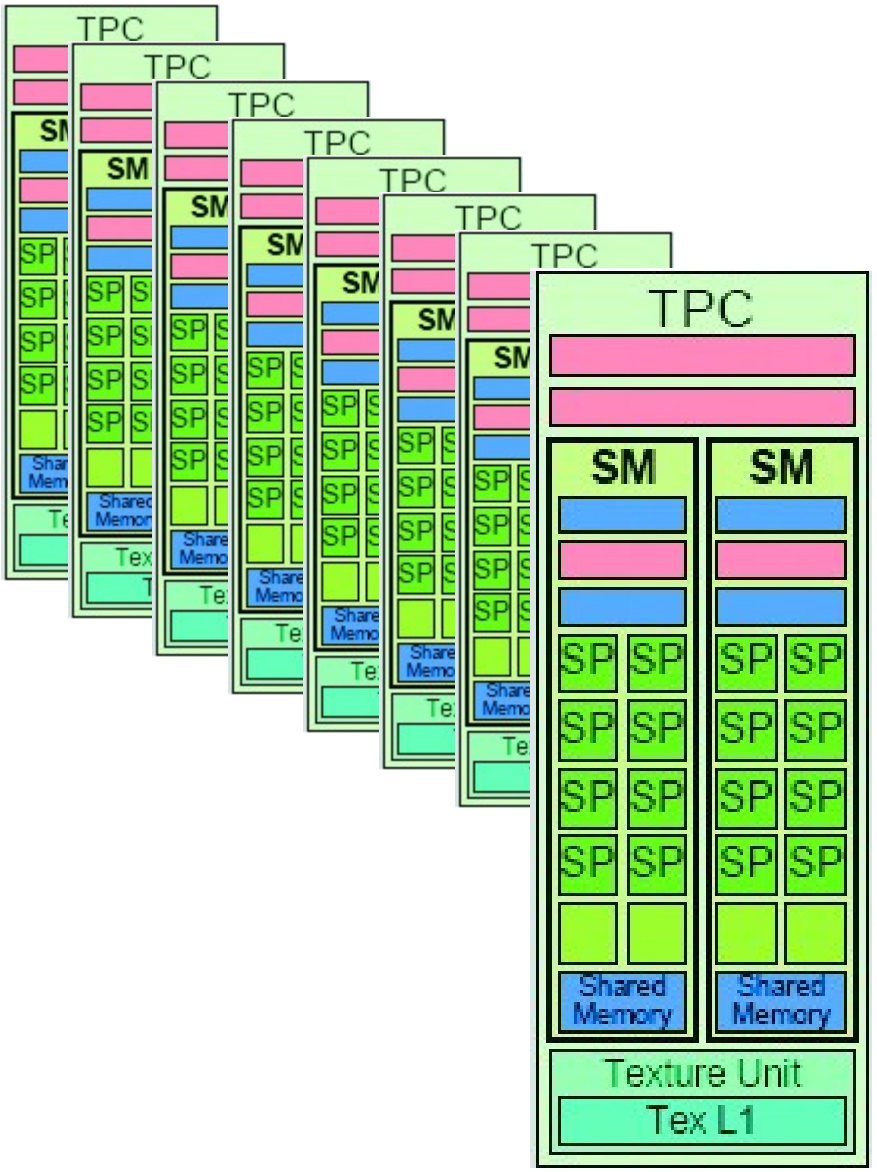
- Меньше транзисторов на управление и кэш
- Больше на АЛУ

Доступ к памяти



Аппаратная архитектура GPU

Streaming Processor Array



- TPC - Texture Processor Cluster
- SM — Streaming Multiprocessor
  - Multi-threaded processor core
  - Fundamental processing unit for CUDA thread block
- SP — Streaming Processor
  - Scalar ALU for a single CUDA thread

	Количество SM
GeForce 8800 GTX	16
GeForce 8800 GTS	12
Tesla D870	2x16
Tesla S870	4x16
Tesla C1060, GT200, Tesla T10	30
Tesla S1070	4x30

Tesla C1060

1 TFlops

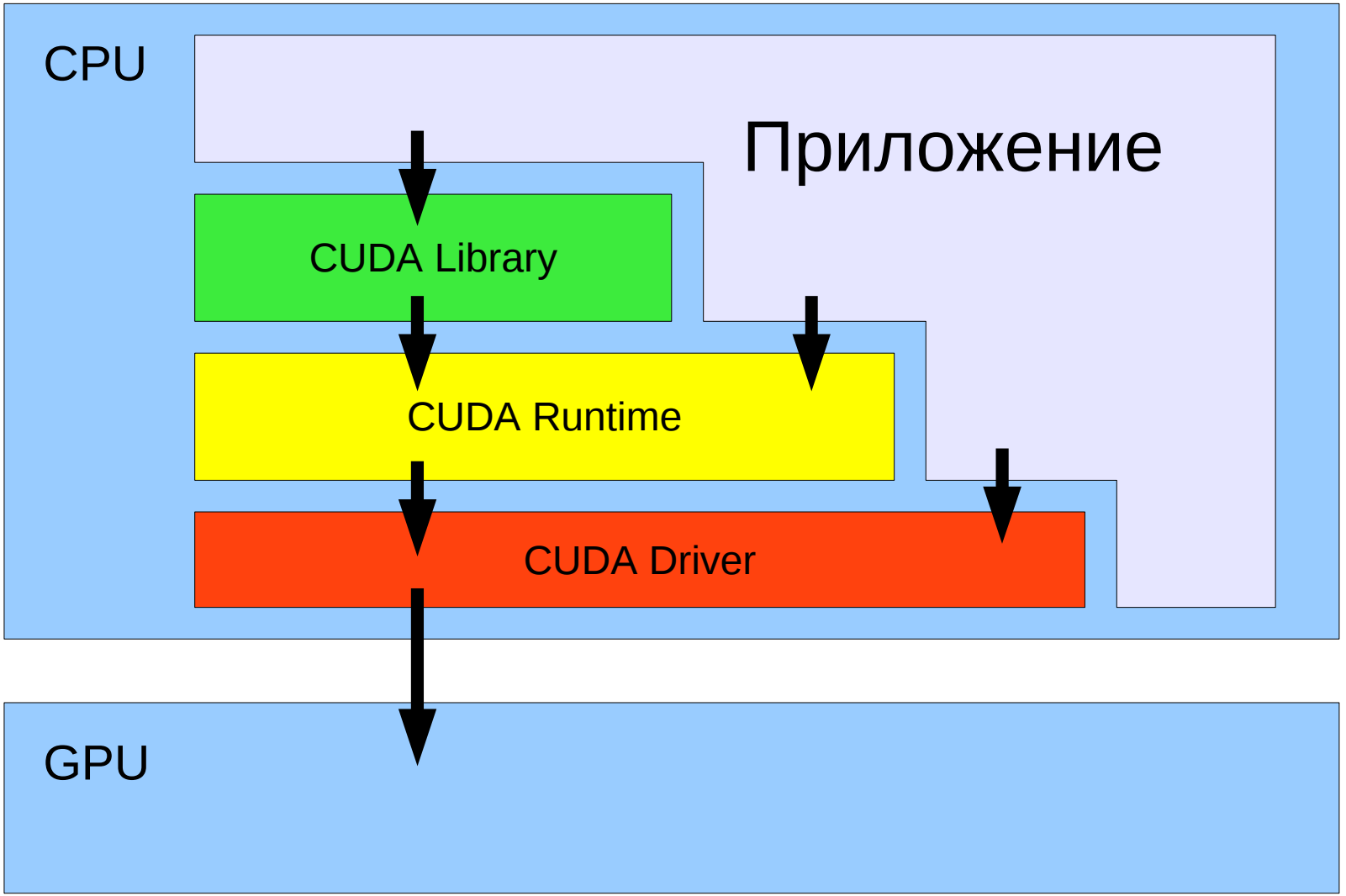
Tesla S1070

4 TFlops

Производительность для различных приложений

Example Applications	URL	Application Speedup
Seismic Database	http://www.headwave.com	66x to 100x
Mobile Phone Antenna Simulation	http://www.acceleware.com	45x
Molecular Dynamics	http://www.ks.uiuc.edu/Research/vmd	21x to 100x
Neuron Simulation	http://www.evolvedmachines.com	100x
MRI processing	http://bic-test.beckman.uiuc.edu	245x to 415x
Atmospheric Cloud Simulation	http://www.cs.clemson.edu/~jesteel/clouds.	50x

CUDA - Compute Unified Device Architecture



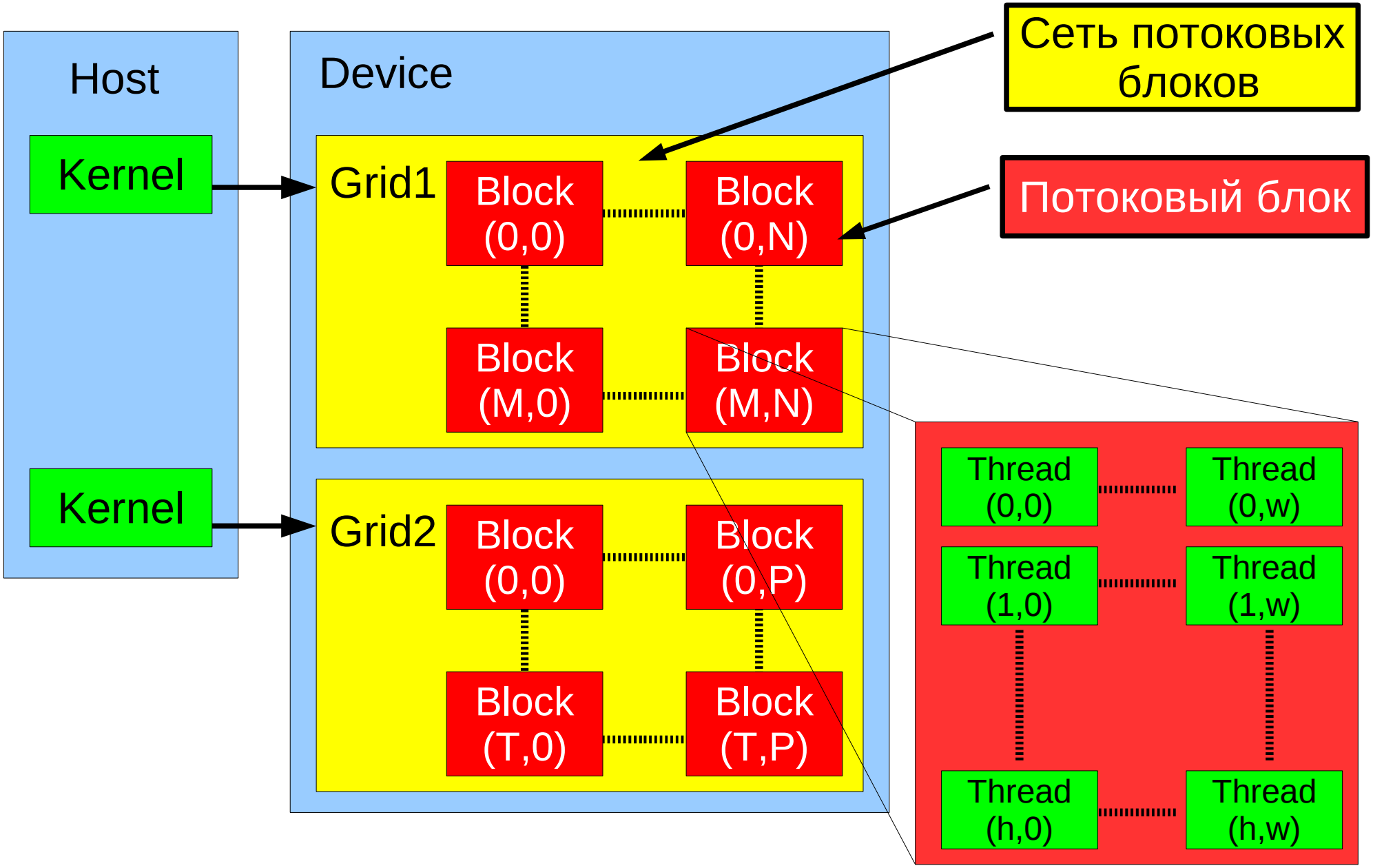
Термины

- **Поток** (Thread) — единица исполнения потока команд
- **Потоковый блок** (Thread blok) — группа связанных между собой потоков.
- **Ворп** (Warp)— группа потоков внутри потокового блока, которая исполняется физически одновременно (32 потока)
- **Сеть** (Grid) — набор блоков, который должен быть обработан прежде чем исполнение программы пойдет дальше.

Программная модель

- GPU имеет свою память
- Программа в виде потоков выполняется на SP
- SP имеет доступ только к разделяемой памяти внутри своего SM и памяти GPU
- Синхронизация потоков возможна только внутри SM
- Исполнение организовано как сеть (GRID) потоковых блоков (thread block)
- Программа выполняемая потоком — ядро (kernel)

Запуск потоков



Потоковый блок

- Каждый поток в блоке имеет свой идентификатор — threadID
- Для удобства потоки могут отражаться на одномерную, двумерную, трехмерную сетку. При этом координаты потока задаются через (x,y,z)
- Размеры области отображения задаются при запуске ядра
- Количество потоков в блоке < 512

Сеть потоковых блоков

- Каждый блок в сети имеет свой идентификатор — blockID
- Для удобства блоки могут отражаться на одномерную, двумерную, трехмерную сетку. При этом координаты блок задаются через (x,y,z)
- Размеры области отображения задаются при запуске ядра

Пример

- Пусть при запуске задана двумерная сеть из блоков размером HxW и каждый блок содержит MxK потоков
- Таким образом область моделирования разбивается на
  - H*M потоков по вертикали
  - W*K потоков по горизонтали
- координаты потока в пространстве
  - (blockID.x*M + threadID.x, blockID.y*K + threadID.y)

Модель выполнения

- Блоки выполняются на Stream Multiprocessor
  - Один блок только на одном SM
  - Последовательность исполнения блоков не определена
- Количество блоков на SM определяется количеством регистров, требуемых потоку и количеством разделяемой памяти на блок
- Исполняемый в текущий момент поток называется **активным блоком**

Модель выполнения

- Каждый активный блок разбивается на SIMD группы потоков — ворпы (**warps**). Каждый ворп содержит одинаковое количество потоков. WarpSize = 32
- Планировщик потоков периодически передает управление от одного ворпа к другому
- Распределение потоков по ворпам всегда одинаковое

Литература

- http://www.nvidia.com/object/cuda_home.html
- http://steps3d.narod.ru/tutorials/cuda-tutorial.html
- http://steps3d.narod.ru/tutorials/cuda-2-tutorial.html