



Неграфические вычисления  
на графических процессорах с  
использованием технологии

NVIDIA CUDA

Выполнил студент группы ВМ-42 Завалов Р.Е.

# От транспьютеров к 3D ускорителям



Транспьютер INMOS  
IMSB008  
1980-е

Видеоускоритель NVIDIA  
GeForce GTX 480  
2010



GPGPU – General-Purpose computation on GPUs  
Вычисления общего назначения на графических процессорах

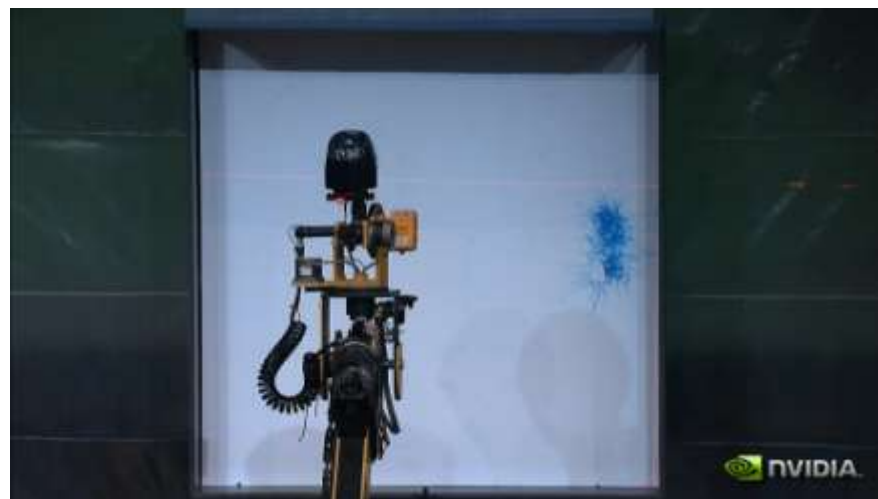
Неспециализированные API:

- Direct3D (DirectX)
- OpenGL

API для вычислений общего назначения:

- NVIDIA CUDA
- AMD Stream
- OpenCL
- DirectCompute (DirectX)

CPU

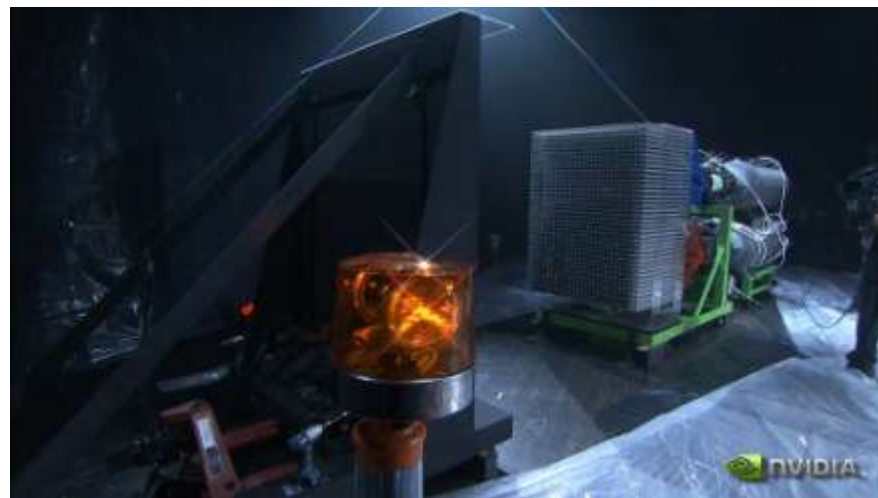




## Особенности вычислений на CPU:

- Один поток команд на процессор
- Один поток данных на процессор (до четырех при использовании SSE)
- Произвольный доступ к памяти
- Большое число ветвлений

GPU





GPU

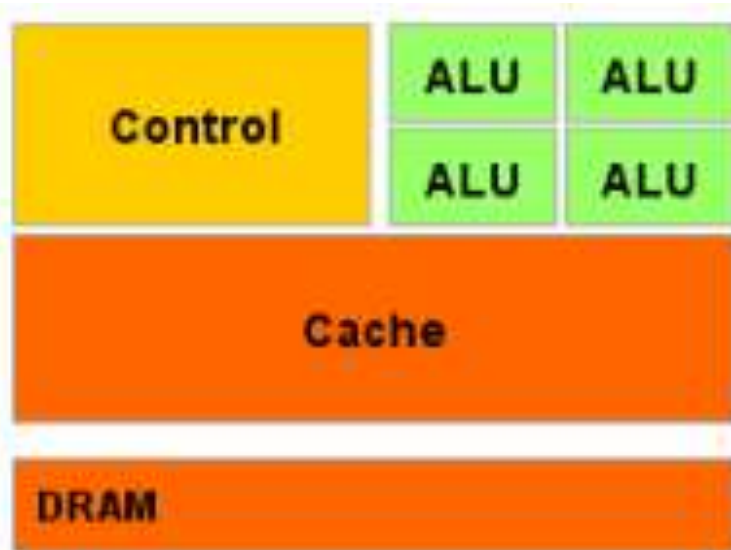


## Особенности вычислений на GPU:

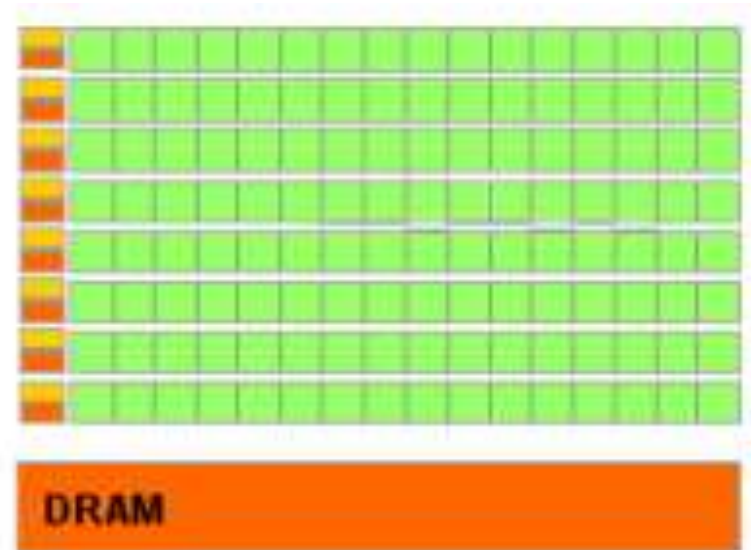
- Один поток команд на процессор
- Большое число данных на процессор
- Линейный доступ к памяти
- Малое число ветвлений



# Различия в топологии

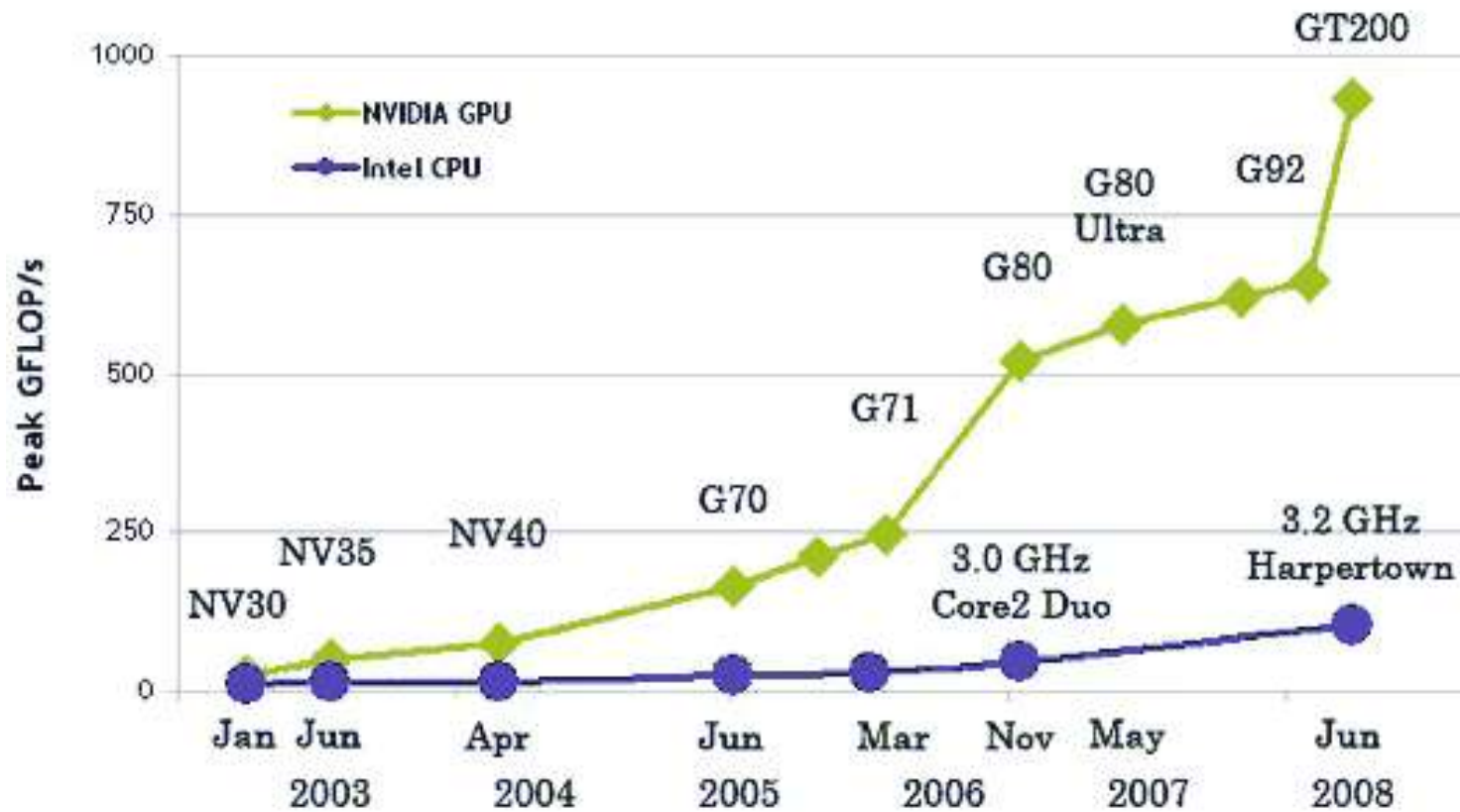


CPU




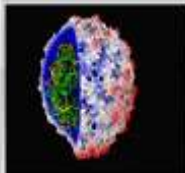

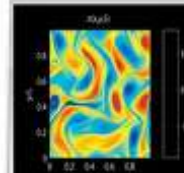

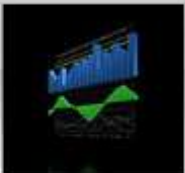



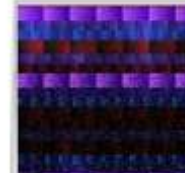
GPU

# Рост производительности CPU и GPU



# Ускорение за счет GPU

## Speedups Using GPU vs CPU

				
<b>146X</b>	<b>36X</b>	<b>18X</b>	<b>17X</b>	<b>100X</b>
Interactive visualization of volumetric white matter connectivity <sup>1</sup>	Ionic placement for molecular dynamics simulation on GPU <sup>2</sup>	Transcoding HD video stream to H.264 for portable video <sup>3</sup>	Simulation in Matlab using .mex file CUDA function <sup>4</sup>	Astrophysics N-body simulation <sup>5</sup>
				
<b>149X</b>	<b>47X</b>	<b>20X</b>	<b>24X</b>	<b>30X</b>
Financial simulation of LIBOR model with swaptions <sup>6</sup>	GLAME@lab: M-script API for linear Algebra operations on GPU <sup>7</sup>	Ultrasound medical imaging for cancer diagnostics <sup>8</sup>	Highly optimized object oriented molecular dynamics <sup>9</sup>	Cmatch exact string matching - find similar proteins & gene sequences <sup>10</sup>

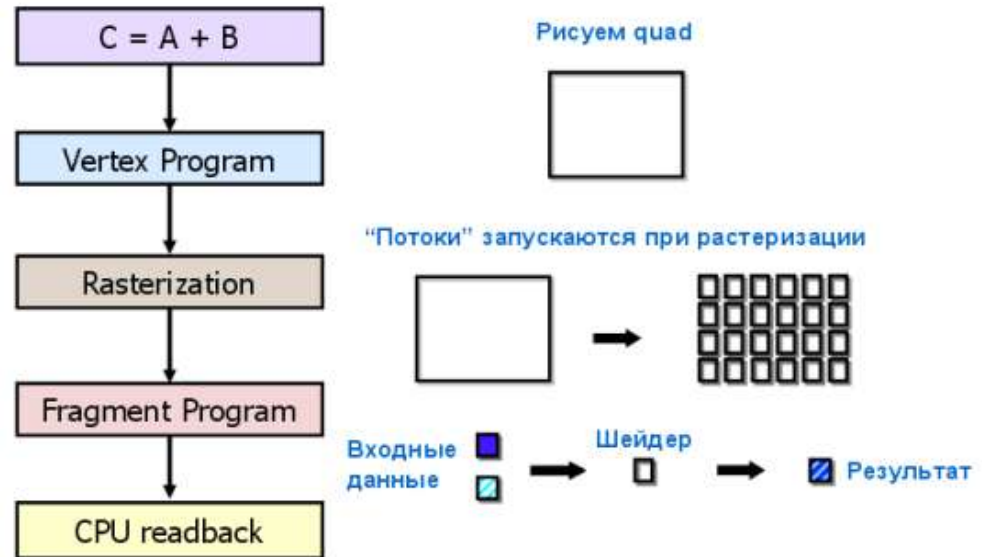


## CUDA – Compute Unified Device Architecture

- унифицированное программно-аппаратное решение для параллельных вычислений на видеочипах NVIDIA;
- большой набор поддерживаемых решений: от мобильных до мультичиповых;
- стандартный язык программирования Си;
- стандартные библиотеки численного анализа FFT (быстрое преобразование Фурье) и BLAS (линейная алгебра);
- оптимизированный обмен данными между CPU и GPU;
- взаимодействие с графическими API OpenGL и DirectX;
- поддержка 32- и 64-битных операционных систем: Windows XP, Windows Vista, Linux и MacOS X;
- возможность разработки на низком уровне.

# Порядок работы конвейера GPU

- Подготовка геометрии
- Заполнение текстур данными
- Выполнение вершинного шейдера
- Растеризация
- Выполнение фрагментного (пиксельного) шейдера
- Чтение из буфера кадра в основную память



# Преимущества CUDA

- интерфейс программирования приложений CUDA основан на стандартном языке программирования Си с расширениями, что упрощает процесс изучения и внедрения архитектуры CUDA;
- CUDA обеспечивает доступ к разделяемой между потоками памяти размером в 16 Кб на мультипроцессор, которая может быть использована для организации кэша с широкой полосой пропускания, по сравнению с текстурными выборками;
- более эффективная передача данных между системной и видеопамью;
- отсутствие необходимости в графических API с избыточностью и накладными расходами;
- линейная адресация памяти, и gather и scatter, возможность записи по произвольным адресам;
- аппаратная поддержка целочисленных и битовых операций.



# Ограничения CUDA

- отсутствие поддержки рекурсии для выполняемых функций;
- минимальная ширина блока в 32 потока;
- закрытая архитектура CUDA, принадлежащая nVIDIA.

# NVIDIA Tesla

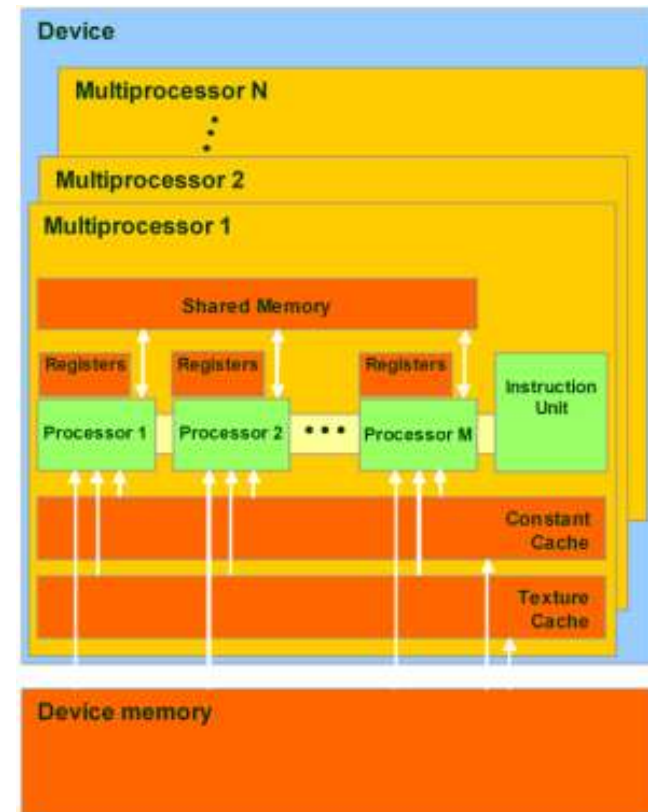
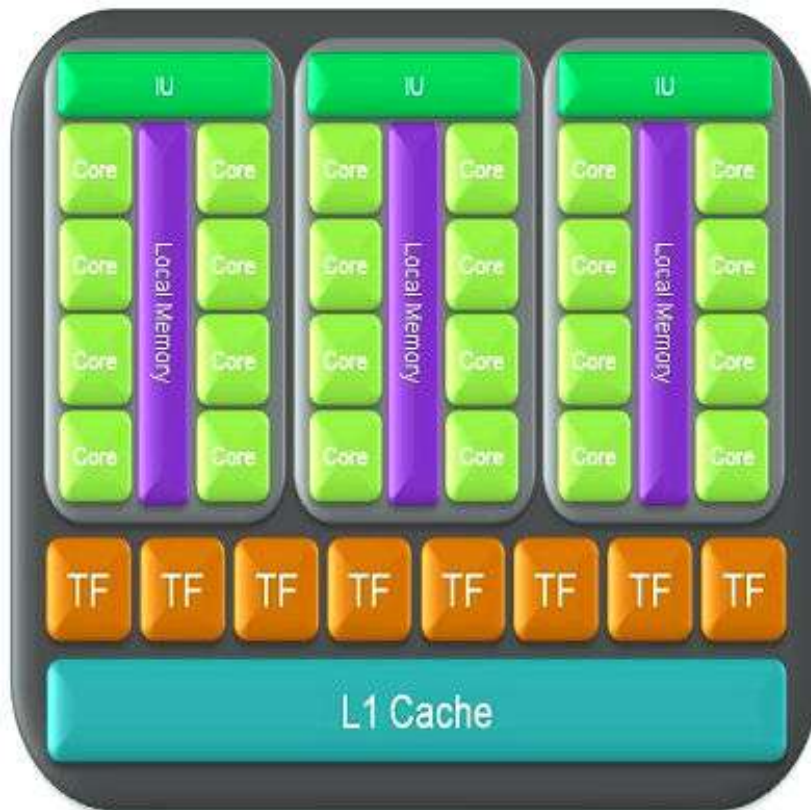
Tesla C1060



Tesla S1070

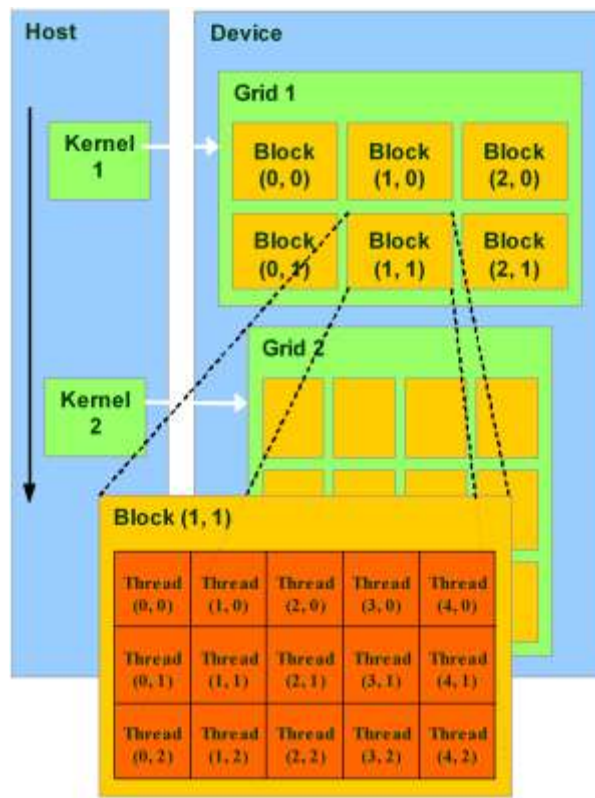


# Архитектура аппаратной составляющей

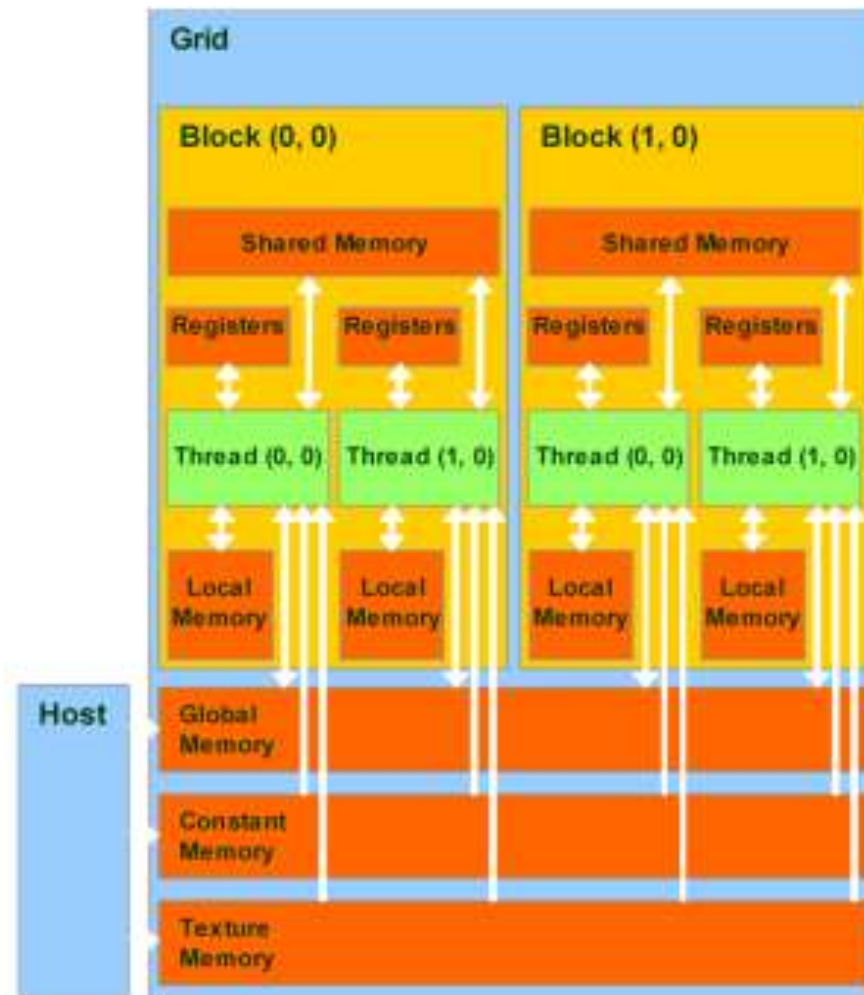




# Модель программирования



# Модель памяти



# Использование CUDA

Типичный, но не обязательный шаблон решения задач:

- задача разбивается на подзадачи;
- входные данные делятся на блоки, которые вмещаются в разделяемую память (16 КБ);
- каждый блок данных обрабатывается блоком потоков;
- блок подгружается в разделяемую память из глобальной;
- над данными в разделяемой памяти проводятся соответствующие вычисления;
- результаты копируются из разделяемой памяти обратно в глобальную.