Билет 30. Архитектурные особенности графических процессоров, направленные на массивно-параллельные вычисления.

CPU - небольшое число мощных, независимых ядер, 3-х уровневый кэш
GPU - вместо ядер потоковый мультипроцессор (streaming multiprocessor, SM):

1. 32 скалярных ядра CUDA cores, ~1.5ГГц
2. 2 Warp Scheduler (warp - 32 ядра)
3. Файл регистров, 128 КБ
4. 2-х уровневый кэш
5. Текстурные юниты
6. 16 Special Function Unit - интерполяция ~ трансцендентная математика один. Точности

Много простых ядер, работающих на небольшой частоте. Небольшие кэши на GPU: 32 ядра делят 64 КБ L1 кэша, общий L2 кэш, L3 нет.
Оперативная память с высокой пропускной способностью и **высокой латентностью**, оптимизированная для коллективного доступа. Так как память с высокой латентностью, то чтобы эффективнее загружать ядра на много нитей и за счет быстрого переключения контекста перекрывать обращения одних нитей вычислениями в других имеется следующая иерархия:
Нить -> warp (32 нити) -> Block (до 32 warp) -> Grid (решетка из блоков)
Block и Grid - 3d, каждая нить, блок получают свои координаты.
На уровне нитей - SIMD - все нити выполняют одно и то же действие с разными данными. Warp-ы выполняются независимо, называется все - SIMT.

Глобальная память

Расположение в DRAM GPU, до 4ГБ, можно выделять как с хоста, так и с устройства.
Работа оптимизирована так, чтобы отдать максимальное число данных за одно обращение.
Транзакция - загрузка из глобальной памяти пула в 128 байт, начиная с адреса, кратного 128. Взаимодействие идет через кэш. Размер кэш линии L1 - 128 байт, L2 - 32 байта, можно работать не через L1 кэш, тогда транзакции по 32 байта.
Данные хранятся по строкам, чтобы добиться выравнивания по 128 байт, можно воспользоваться CudaMallocPitch - передаем, сколько у нас будет входных данных, возвращаем с учетом padding.