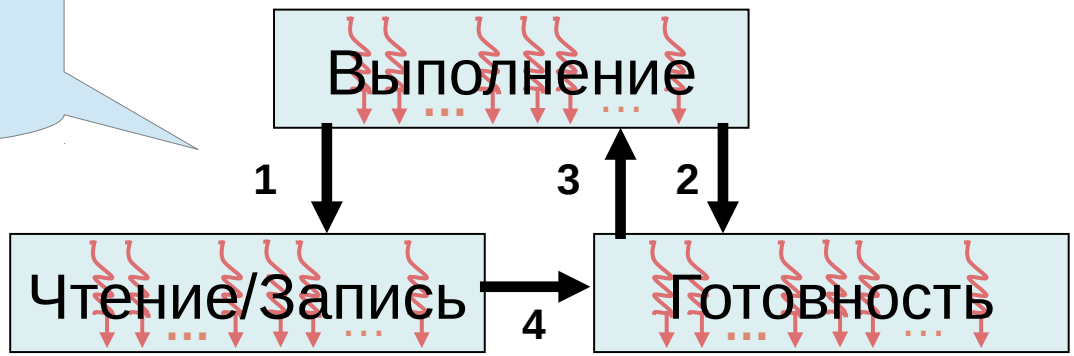
# Лекция 3

## СОДЕРЖАНИЕ

- оптимизация  нагрузки GPU;
- иерархия памяти;
- объединение запросов к глобальной памяти (coalescing);

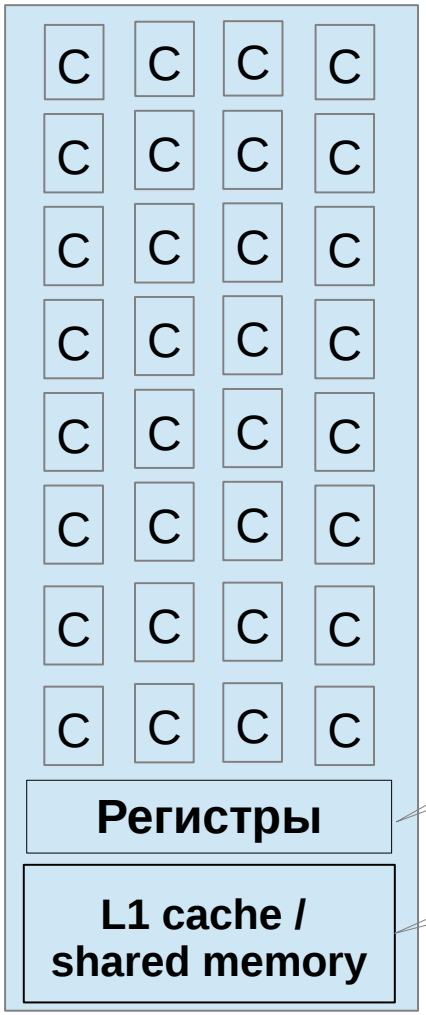
# *Оптимальное количество блоков и нитей в блоках (сокрытие латентности)*

Преимущества перекрытия/многозадачности:




Кол-во варпов % 32 ==0 && Кол-во варпов / 32 > 1
Кол-во блоков >= Кол-во мультипроцессоров

# *Оптимальное количество блоков и нитей в блоках (заполняемость мультипроцессоров)*




Maximum number of threads per multiprocessor: 2048

Max number of warps/MP: 2048/32=64

Max number of blocks/MP: 8

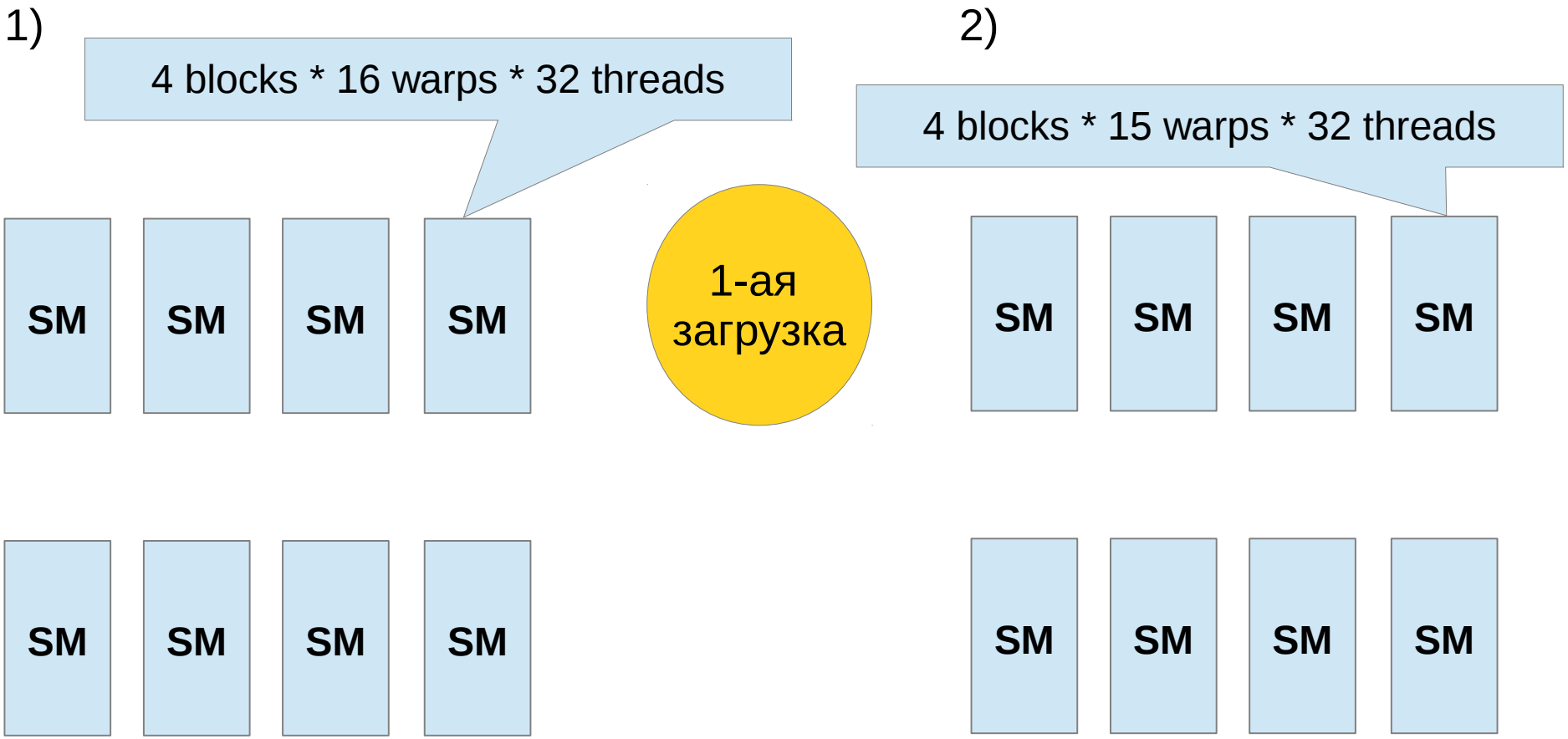
**Оптимальное кол-во нитей в блоке:** 2048/8=256

32 K

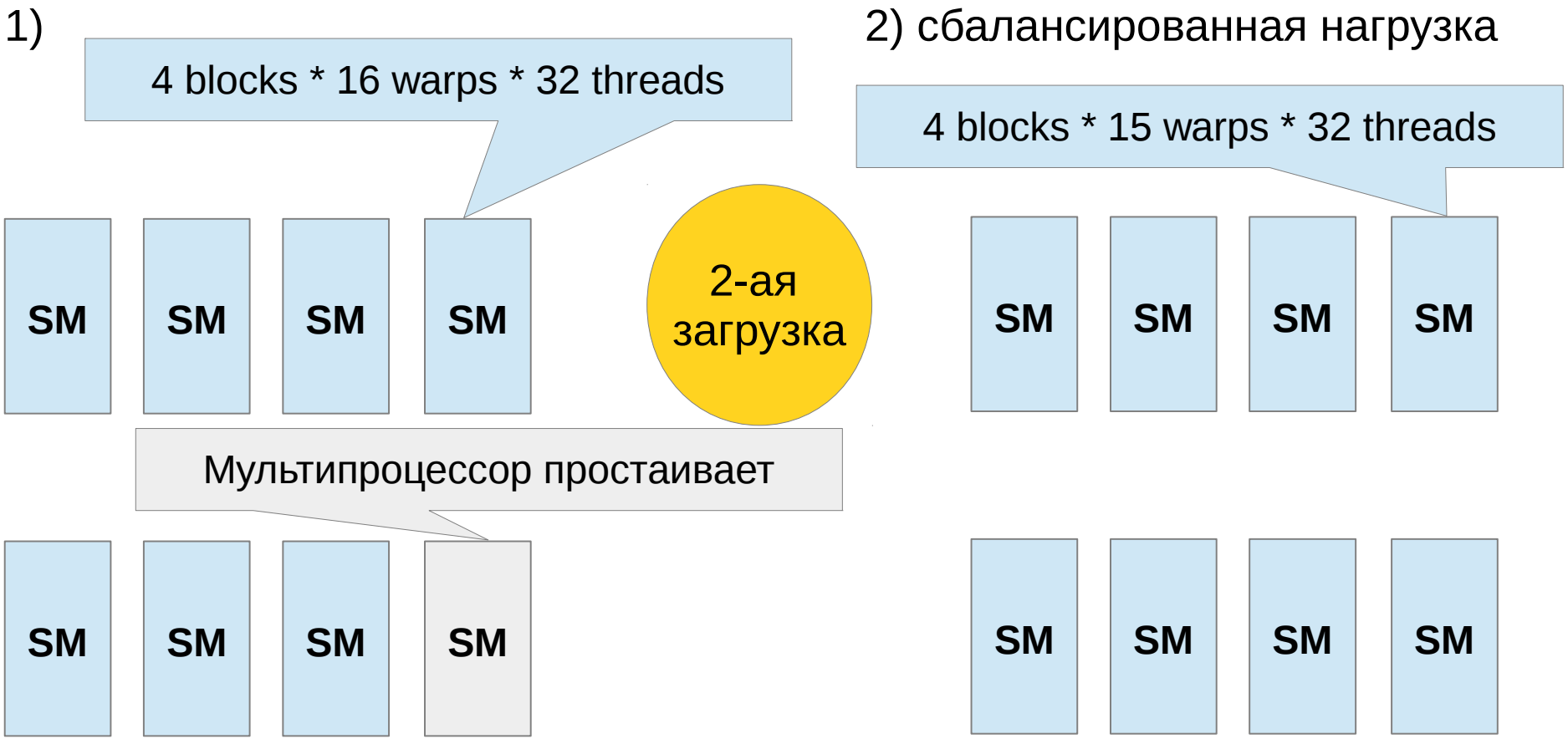
48+16 KB

Пример: как распределить 30720 нитей -
1) 512 threads * 60 blocks,
2) 480 threads * 64 blocks?

# *Оптимальное количество блоков и нитей в блоках (заполняемость мультипроцессоров)*



# *Оптимальное количество блоков и нитей в блоках (заполняемость мультипроцессоров)*



# *Оптимальное количество блоков и нитей в блоках (заполняемость мультипроцессоров)*

Для архитектуры Fermi:

- максимальное кол-во варпов, выполняемых одновременно на одном мультипроцессоре равно 48;
- максимальное кол-во блоков выполняемых одновременно на одном мультипроцессоре равно 8;

**Пример.**

32 нити в блоке, 512 блоков в гриде:
Кол-во одновременно выполняемых нитей=32*8,
Максимальное кол-во=48*32,
Заполняемость=32*8/48*32=1/6=16.(6)%.

512 нитей в блоке, 32 блока в гриде:
Кол-во одновременно выполняемых нитей=512*8,
Максимальное кол-во=48*32,
Заполняемость=512*8/48*32=266.(6)%.

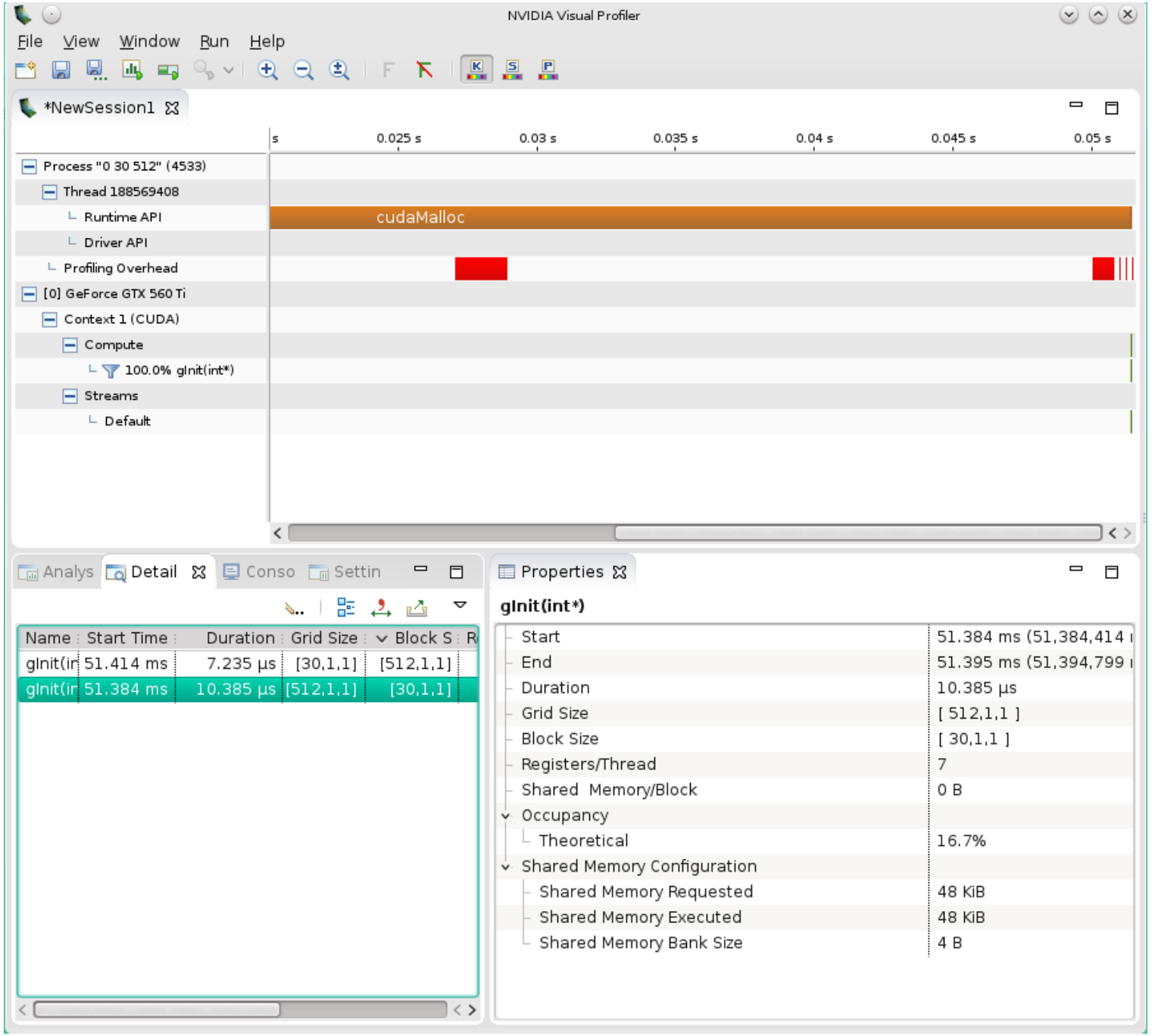
**Оптимальное кол-во нитей в блоке (для *Fermi*):** 48*32/8=192

# *Профилирование метрик*

.../Lab3> nvprof **-m achieved_occupancy** ./Lab3
==11929== NVPROF is profiling process 11929, command: ./Lab3
gTest1 took 3.56662
gTest2 took 2.41971
==11929== Profiling application: ./Lab3
==11929== Profiling result:
==11929== Metric result:

| Invocations | Metric Name | Metric Description | **Min** | **Max** | **Avg** |
|---|---|---|---|---|---|
| Device "GeForce GTX 560 Ti (0)" | | | | | |
| Kernel: gTest1(float*) | | | | | |
| 1 | achieved_occupancy | Achieved Occupancy | **0.702714** | **0.702714** | **0.702714** |
| Kernel: gTest2(float*) | | | | | |
| 1 | achieved_occupancy | Achieved Occupancy | **2.426397** | **2.426397** | **2.426397** |

>nvvp ./Lab3

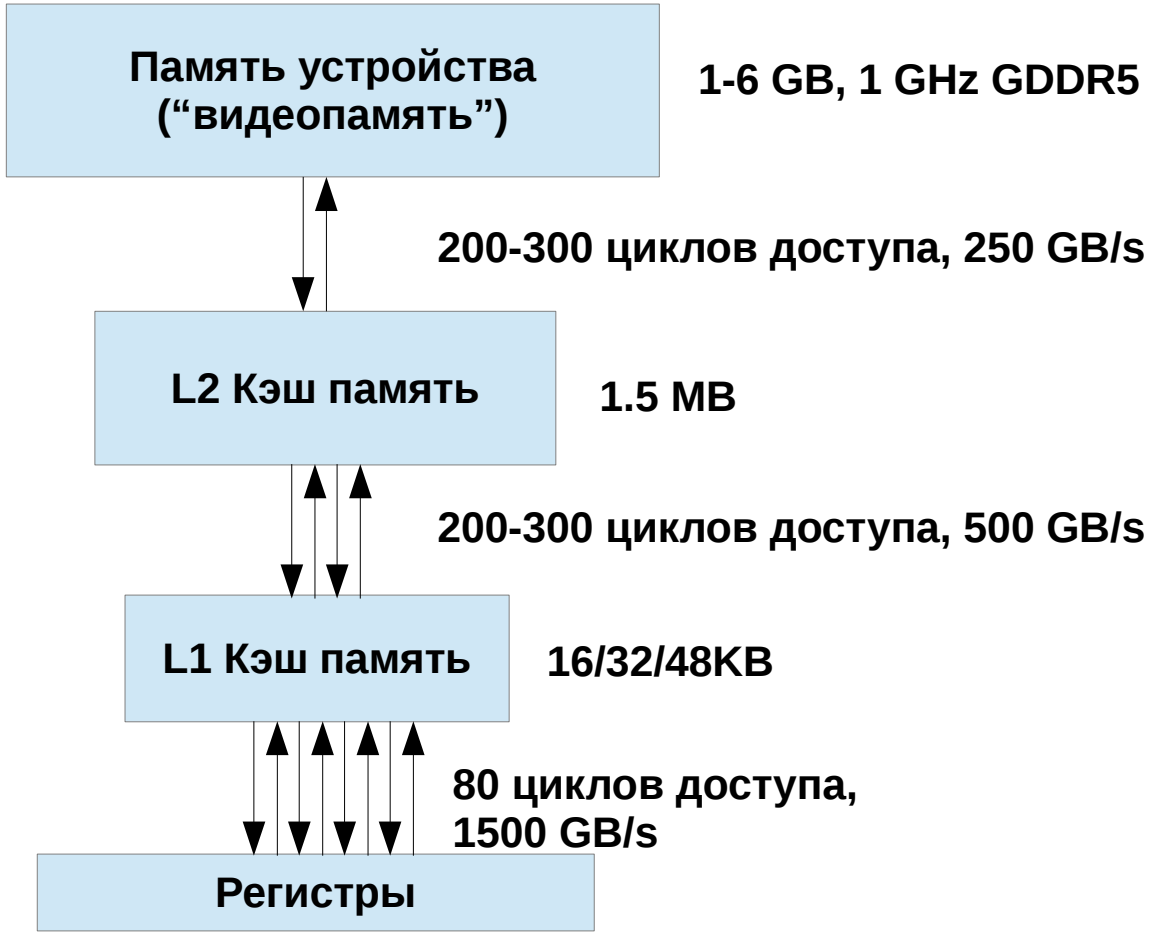

NVIDIA Visual Profiler
File View Window Run Help
*NewSession1
s 0.025 s 0.03 s 0.035 s 0.04 s 0.045 s 0.05 s
Process "0 30 512" (4533)
Thread 188569408
Runtime API
cudaMalloc
Driver API
Profiling Overhead
[0] GeForce GTX 560 Ti
Context 1 (CUDA)
Compute
100.0% gInit(int*)
Streams
Default
Analys Detail Conso Settin
Name Start Time Duration Grid Size Block S
gInit(in 51.414 ms 7.235 µs [30,1,1] [512,1,1]
gInit(in 51.384 ms 10.385 µs [512,1,1] [30,1,1]
Properties
gInit(int*)
Start 51.384 ms (51,384,414
End 51.395 ms (51,394,799
Duration 10.385 µs
Grid Size [ 512,1,1 ]
Block Size [ 30,1,1 ]
Registers/Thread 7
Shared Memory/Block 0 B
Occupancy
Theoretical 16.7%
Shared Memory Configuration
Shared Memory Requested 48 KiB
Shared Memory Executed 48 KiB
Shared Memory Bank Size 4 B

# Определение реальной заполняемости GPU

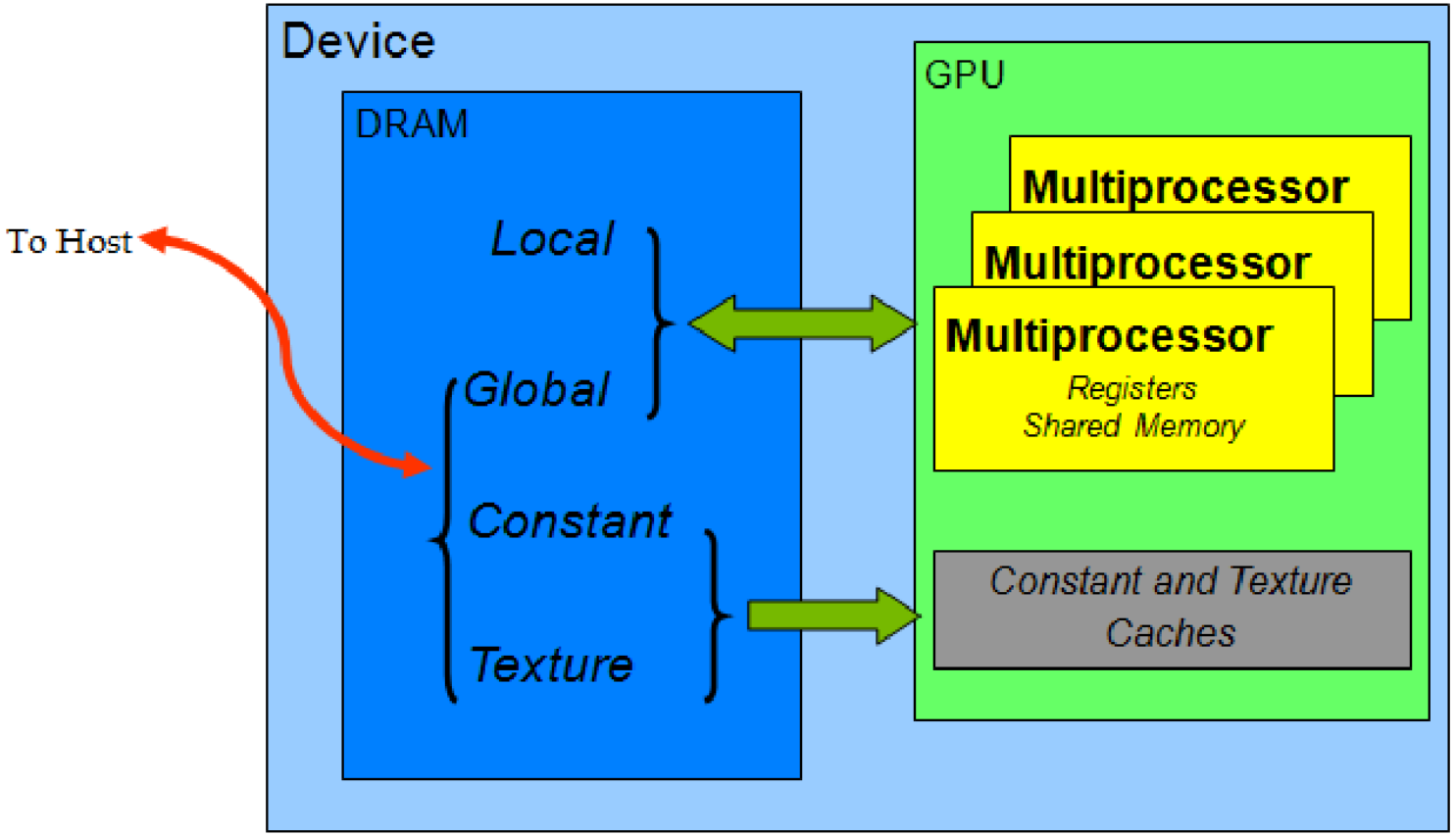
**Table 14. Technical Specifications per Compute Capability**

| | Compute Capability | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Technical Specifications** | **3.0** | **3.2** | **3.5** | **3.7** | **5.0** | **5.2** | **5.3** | **6.0** | **6.1** | **6.2** | **7.0** |
| Maximum number of resident blocks per multiprocessor | 16 | | | | 32 | | | | | | |
| Maximum number of resident warps per multiprocessor | 64 | | | | | | | | | | |
| Maximum number of resident threads per multiprocessor | 2048 | | | | | | | | | | |
| Number of 32-bit registers per multiprocessor | 64 K | | | 128 K | 64 K | | | | | | |
| Maximum number of 32-bit registers per thread block | 64 K | 32 K | 64 K | | | | 32 K | 64 K | | 32 K | 64 K |
| Maximum number of 32-bit registers per thread | 63 | 255 | | | | | | | | | |
| Maximum amount of shared memory per multiprocessor | 48 KB | | | 112 KB | 64 KB | 96 KB | 64 KB | | 96 KB | 64 KB | 96 KB |
| Maximum amount of shared memory per thread block | 48 KB | | | | | | | | | | 96 KB[20] |

# *Иерархия памяти графического процессора (Fermi, Kepler)*



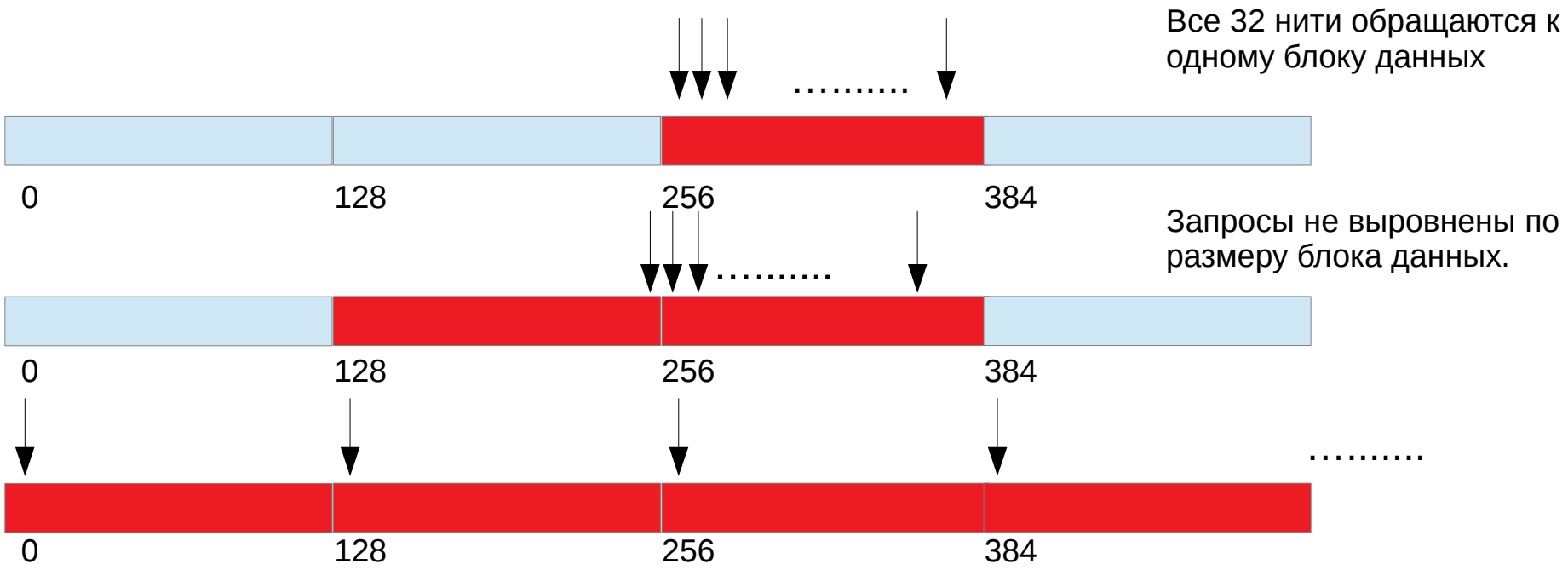
# Иерархия памяти устройства CUDA




*CUDA C Best Practices Guid, Nvidia, october 2012*

# *Объединение запросов к глобальной памяти (coalescing)*

Запросы на чтение и запись к глобальной памяти нитями одного варпа (*a warp*) объединяются в транзакции, количество которых равно количеству необходимых для выполнения запросов блоков данных (*cache lines*) L1 кэша размером в 128 байт. **COMPUTE CAPABILITIES 2.* !**



# Нарушение coalescing'а (пример)

```
__global__ void gTest1(float* a){
  int i=threadIdx.x+blockIdx.x*blockDim.x;
  int j=threadIdx.y+blockIdx.y*blockDim.y;
  int I=gridDim.x*blockDim.x;
  //int J=gridDim.y*blockDim.y;
  a[i+j*I]=(float)(threadIdx.x+blockDim.y*blockIdx.x);
  }

__global__ void gTest2(float* a){
  int i=threadIdx.x+blockIdx.x*blockDim.x;
  int j=threadIdx.y+blockIdx.y*blockDim.y;
  //int I=gridDim.x*blockDim.x;
  int J=gridDim.y*blockDim.y;
  a[j+i*J]=(float)(threadIdx.y+threadIdx.x*blockDim.y);
  }
```

Спасибо за внимание