OpenCL C

OpenCL и CUDA

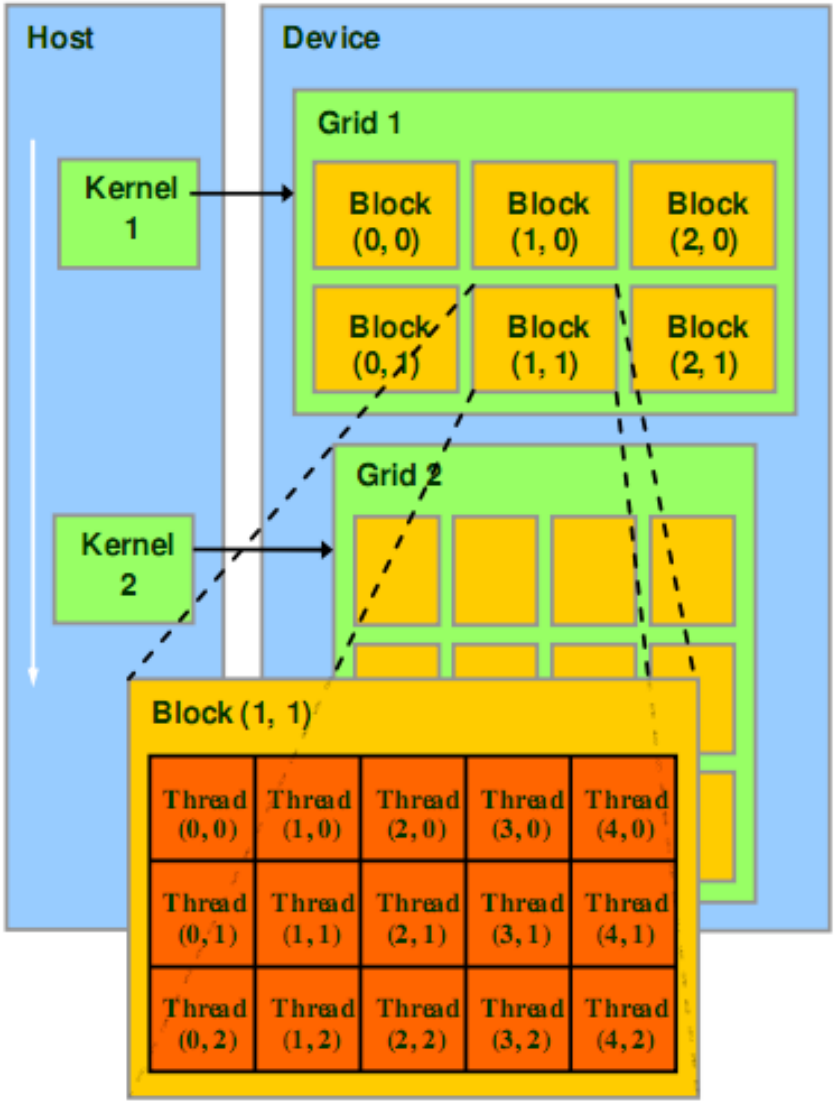
CUDA	OpenCL
Kernel	Kernel
Host program	Host program
Thread	Work item
Block	Work group
Grid	NDRange
Warp	Wavefront

OpenCL и CUDA

CUDA	OpenCL
`threadIdx.x`	`get_local_id(0)`
`blockIdx.x*blockDim.x + threadIdx.x`	`get_global_id(0)`
`gridDim.x`	`get_num_groups(0)`
`blockIdx.x`	`get_group_id(0)`
`blockDim.x`	`get_local_size(0)`
`gridDim.x * blockDim.x`	`get_global_size(0)`

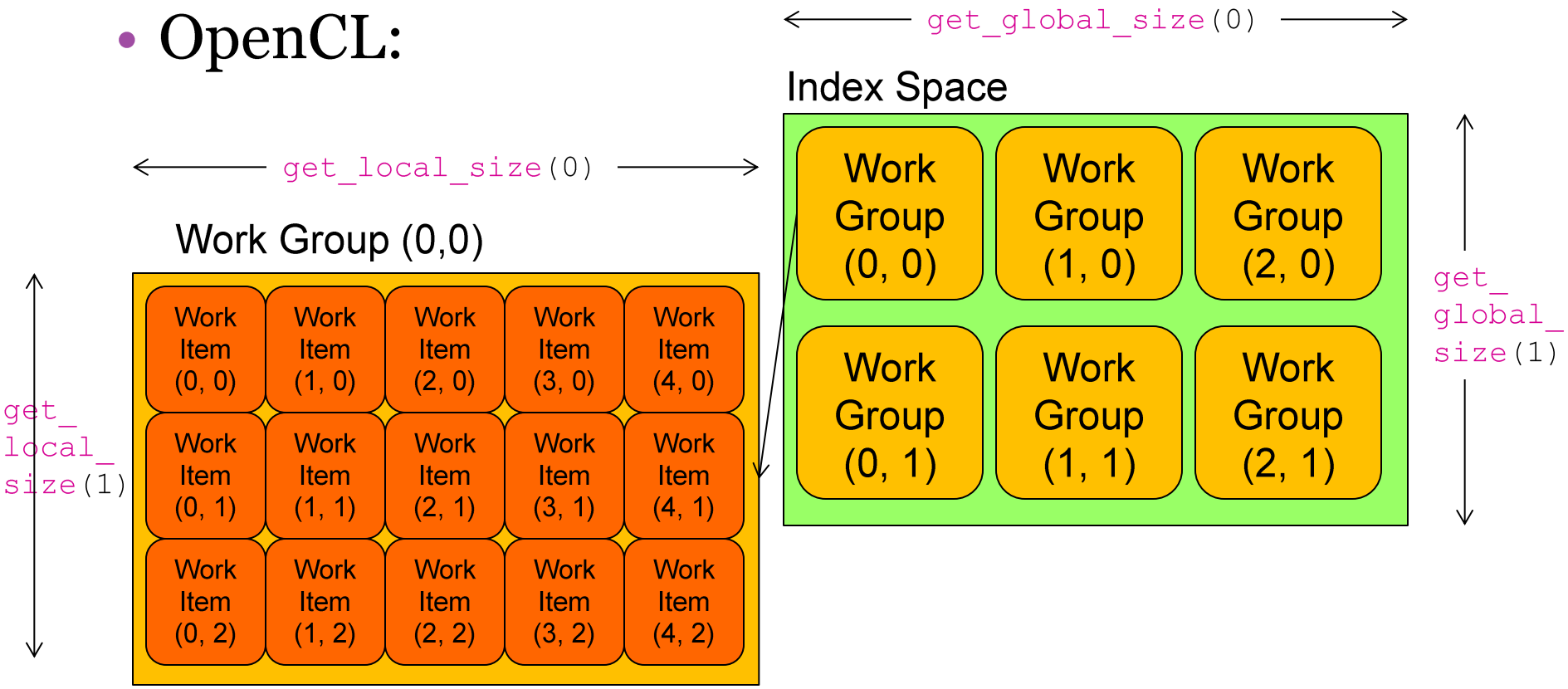
OpenCL и CUDA

- Модель исполнения CUDA:



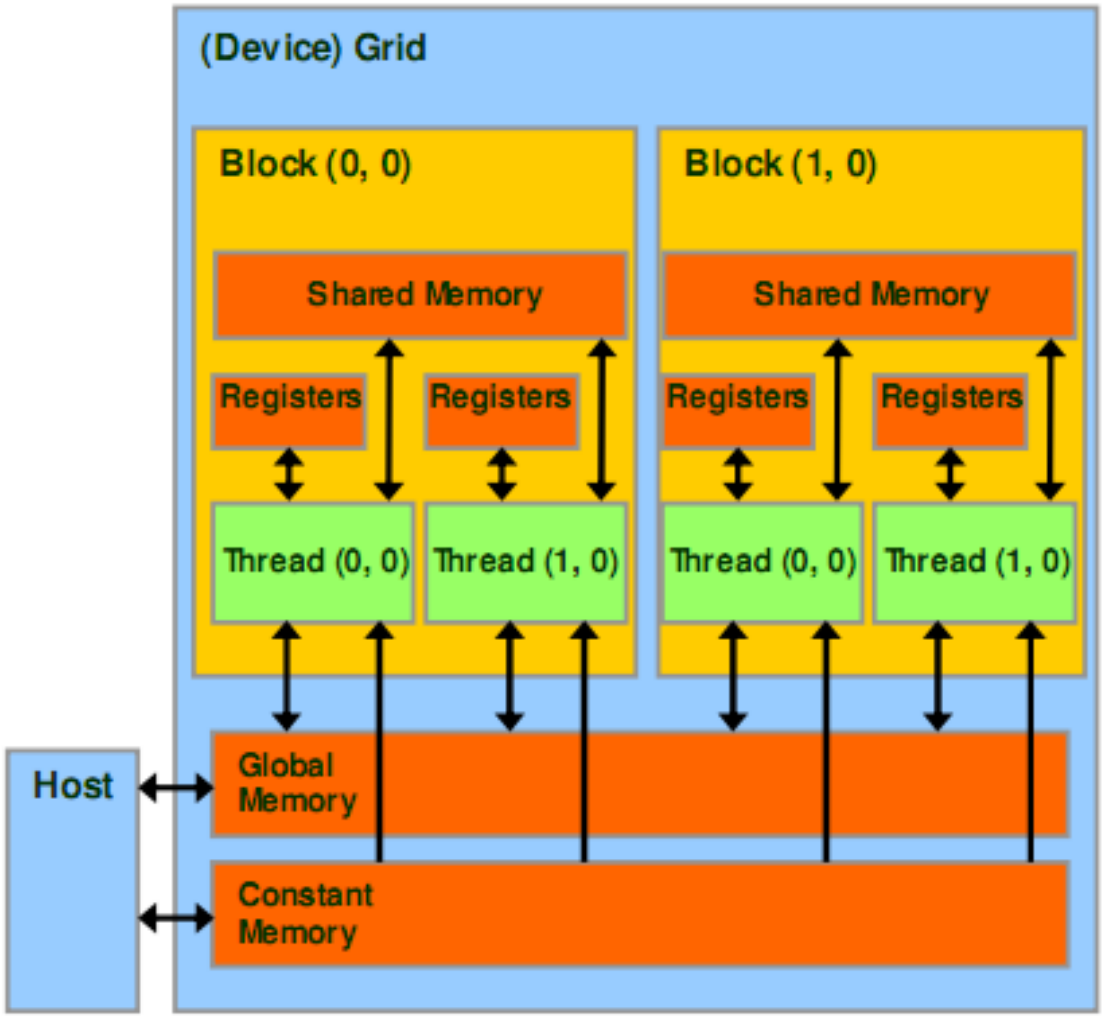
OpenCL и CUDA

- OpenCL:



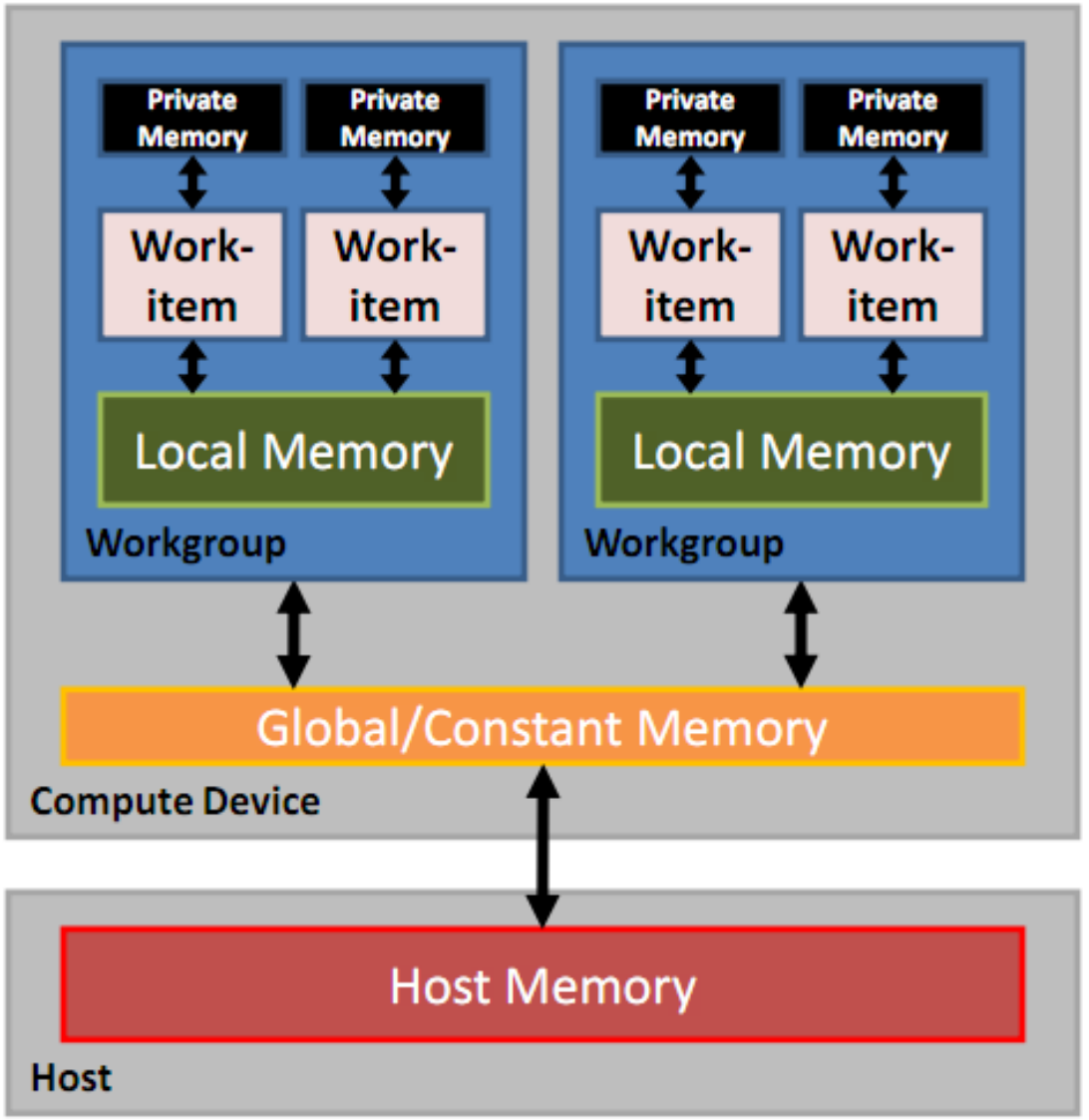
OpenCL и CUDA

- Модель памяти CUDA



OpenCL и CUDA

- OpenCL:



OpenCL и CUDA

CUDA	OpenCL
Global memory	Global memory
Constant memory	Constant memory
Shared memory	Local memory
Local memory	Private memory

OpenCL и CUDA

CUDA	OpenCL
`__syncthreads()`	`barrier()`

Programming kernels: OpenCL C Language

- **A subset of ISO C99**
  - But without some C99 features such as standard C99 headers, function pointers, recursion, variable length arrays, and bit fields
- **A superset of ISO C99 with additions for:**
  - Work-items and workgroups
  - Vector types
  - Synchronization
  - Address space qualifiers
- **Also includes a large set of built-in functions**
  - Image manipulation
  - Work-item manipulation,
  - Specialized math routines, etc.

Image from: http://www.khronos.org/developers/library/overview/opencl_overview.pdf

OpenCL и CUDA

- Kernel – функции в CUDA:

```
__global__ void sum(float *a,
  float *b, float *c)
{
   int i = blockIdx.x*blockDim.x
  + threadIdx.x;
   c[i] = a[i] + b[i];
}
```

OpenCL и CUDA

- OpenCL:

```
__kernel void vecAdd(__global
  float *a, __global float *b,
  __global float *c)
{
  int i = get_global_id(0);
  c[i] = a[i] + b[i];
}
```

Porting CUDA to OpenCL™

- Qualifiers

C for CUDA Terminology	OpenCL™ Terminology
__global__ function	__kernel function
__device__function	function (no qualifier required)
__constant__ variable declaration	__constant variable declaration
__device__ variable declaration	__global variable declaration
__shared__ variable declaration	__local variable declaration

Slide from: http://developer.amd.com/zones/OpenCLZone/courses/pages/Introductory-OpenCL-SAAHPC10.aspx

Data Types

Scalar Type	Vector Type (n = 2, 4, 8, 16)	API Type for host app
char, uchar	charn, ucharn	cl_char<n>, cl_uchar<n>
short, ushort	shortn, ushortn	cl_short<n>, cl_ushort<n>
int, uint	intn, uintn	cl_int<n>, cl_uint<n>
long, ulong	longn, ulongn	cl_long<n>, cl_ulong<n>
float	floatn	cl_float<n>

Slide from: http://developer.amd.com/zones/OpenCLZone/courses/pages/Introductory-OpenCL-SAAHPC10.aspx

Accessing Vector Components

- Accessing components for vector types with 2 or 4 components
  - <vector2>.xy, <vector4>.xyzw

```
float2 pos;
pos.x = 1.0f;
pos.y = 1.0f;
pos.z = 1.0f ; // illegal since vector only has 2 components

float4 c;
c.x = 1.0f;
c.y = 1.0f;
c.z = 1.0f;
c.w = 1.0f;
```


Slide from: http://developer.amd.com/zones/OpenCLZone/courses/pages/Introductory-OpenCL-SAAHPC10.aspx

Accessing Vector with Numeric Index

Vector components	Numeric indices
2 components	0, 1
4 components	0, 1, 2, 3
8 components	0, 1, 2, 3, 4, 5, 6, 7
16 components	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, A, b, B, c, C, e, E, f, F

```
float8 f;
f.s0 = 1.0f; // the 1st component in the vector
f.s7 = 1.0f; // the 8th component in the vector

float16 x;
f.sa = 1.0f; // or f.sA is the 10th component in the vector
f.sF = 1.0f; // or f.sF is the 16th component in the vector
```


Slide from: http://developer.amd.com/zones/OpenCLZone/courses/pages/Introductory-OpenCL-SAAHPC10.aspx

Handy addressing of Vector Components

Vector access suffix	Returns
.lo	Returns the lower half of a vector
.hi	Returns the upper half of a vector
.odd	Returns the odd components of a vector
.even	Returns the even components of a vector

```
float4 f = (float4) (1.0f, 2.0f, 3.0f, 4.0f);
float2 low, high;
float2 o, e;

low = f.lo;      // returns f.xy (1.0f, 2.0f)
high = f.hi;     // returns f.zw (3.0f, 4.0f)
o = f.odd;       // returns f.yw (2.0f, 4.0f)
e = f.even;      // returns f.xz (1.0f, 3.0f)
```


Slide from: http://developer.amd.com/zones/OpenCLZone/courses/pages/Introductory-OpenCL-SAAHPC10.aspx

Еще несколько примеров

- float4 c;
- c.xyzw = (float4)(1.0f, 2.0f, 3.0f, 4.0f);

- float4 pos = (float4)(1.0f, 2.0f, 3.0f, 4.0f);
- float4 swiz= pos.wzyx;
- float4 dup = pos.xxyy;
- pos.xw = (float2)(5.0f, 6.0f);

- float4 a, b, c, d;
- float16 x;
- x = (float16)(a.xxxx, b.xyz, c.xyz, d.xyz, a.yzw);

Математические функции

- sin, cos, tan, sinh, cosh, tanh,
- обратные к перечисленным (вначале добавить букву a)
- gentype sincos (gentype x, gentype *cosval) – вычисляет сразу оба
- atan2(y,x) = arctg(y/x), atanpi = arctg/pi и др arc
- sqrt – квадратный корень, cbrt – кубический корень,
- rootn(x,y) = x^(1/y)
- erf(z) – интеграл ошибок
- exp, exp2,exp10 – e^x, 2^x, 10^x, expm1 = (e^x)-1, log, log2, log10
- ceil, floor, round – округление
- fabs, fmax(x,y), fmin(x,y) – поэлементно!
- hypot(x,y) – sqrt(x^2+y^2)
- mad(a,b,c) = a*b+c, pow(x,y), pown(x,n)
- remainder(x,y) – остаток от деления x на y
- clamp (x, minval, maxval) = min(max(x, minval), maxval)
- degrees (radians) radians (degrees)
- gentype step (gentype edge, gentype x) – функция Хевисайда
- smoothstep (edge0, edge1, x) – сглаженный Хевисайд
- gentype sign (gentype x)

Константы

- M_E_F = e
- M_LOG2E_F = log2e
- M_LOG10E_F = log10e
- M_LN2_F = loge2
- M_LN10_F = loge10
- M_PI_F = π
- M_PI_2_F = π / 2
- M_PI_4_F = π / 4
- M_1_PI_F = 1 / π
- M_2_PI_F = 2 / π
- M_2_SQRTPI_F = 2 / √π
- M_SQRT2_F = √2
- M_SQRT1_2_F = 1 / √2

Геометрические функции

- float3 cross (float3 p0, float3 p1) - векторное произведение
- float dot (floatn p0, floatn p1) – скалярное произведение
- float distance (floatn p0, floatn p1) – расстояние
- float length (floatn p) – норма (евклидова)
- floatn normalize (floatn p) - нормирование вектора

Функции сравнения

- intn isequal (floatn x, floatn y) – возвращает вектор 0 – false и -1 – true
- intn isnotequal (floatn x, floatn y)
- intn isgreater (floatn x, floatn y)
- intn isgreaterequal (floatn x, floatn y)
- intn isless (floatn x, floatn y)
- intn islessequal (floatn x, floatn y)
- int any (igentypen x) – проверяет самый старший бит у каждого элемента вектора (если элемент = 1 - > это false)
- int all (igentypen x)