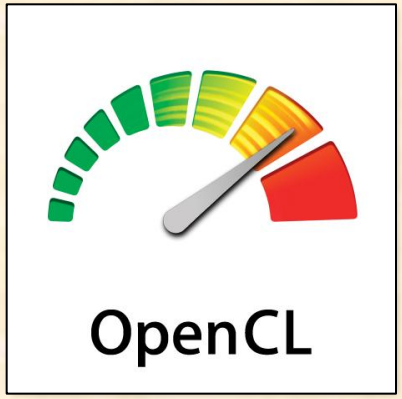


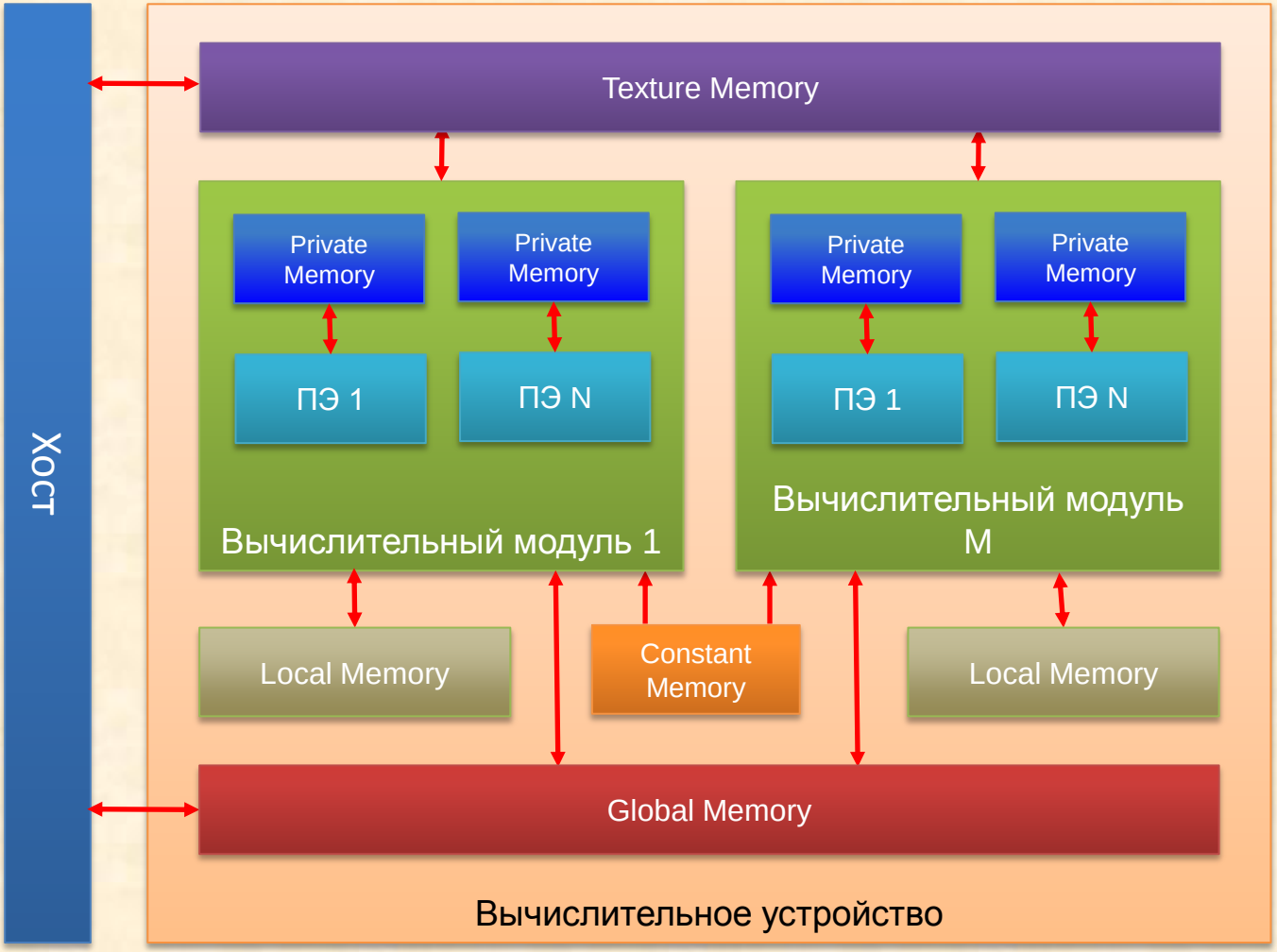
Основы OpenCL

Лекция 4: Работа с текстурной памятью

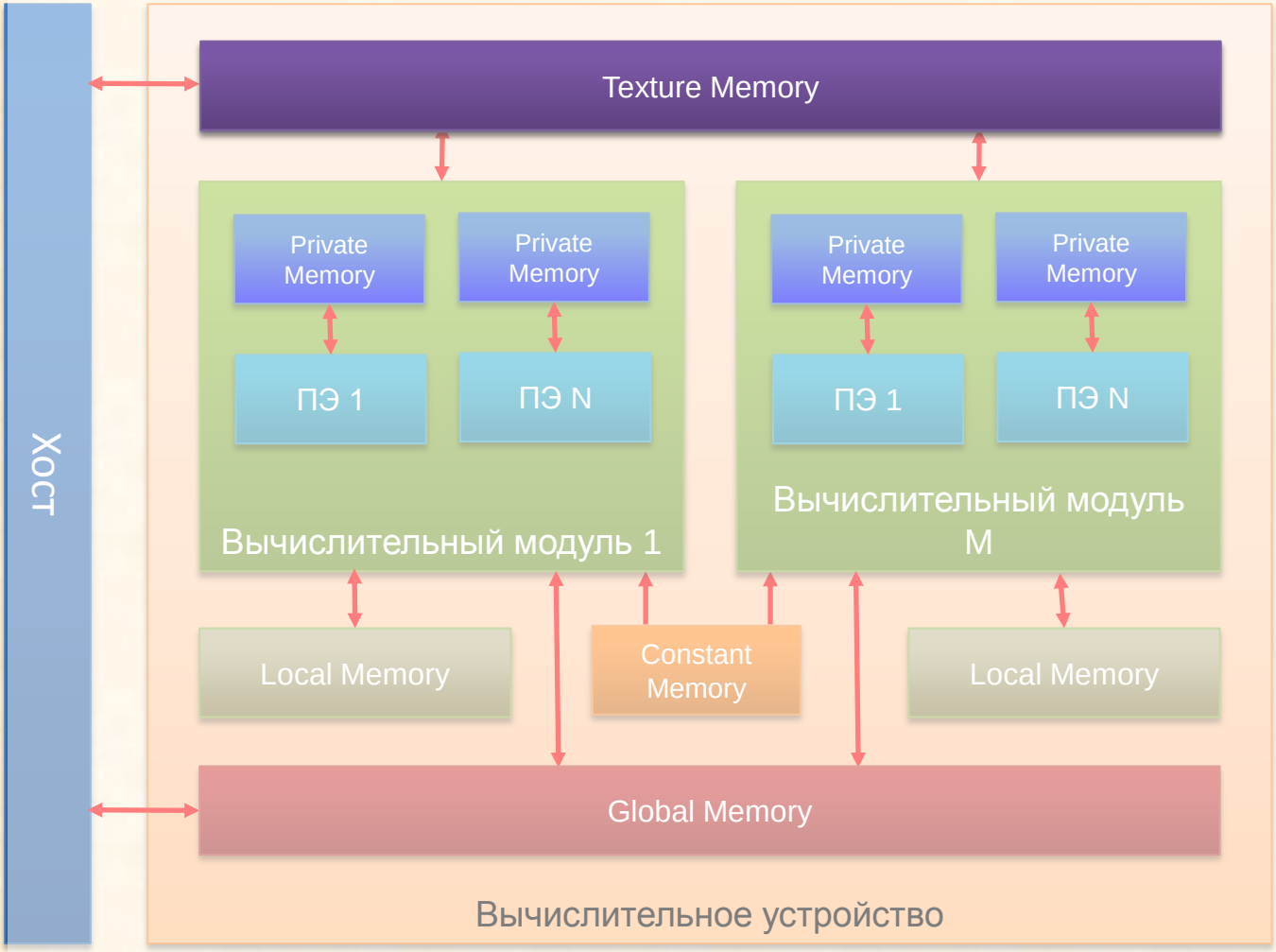
Авторы курса:

- Геллер Олег
- Казённов Андрей

Текстурная память



Текстурная память



Доступ к текстурной памяти в ядрах OpenCL

- Доступ к текстурной памяти осуществляется при помощи объектов специального типа **image2d_t** и **image3d_t**
- Переменная типа **image[2|3]d_t** может быть передана в качестве аргумента ядра.
- Каждая текстура может быть доступна только на чтение или только на запись в каждом ядре.
- Указать режим текстуры можно при помощи ключевых слов **read_only** и **write_only**
- По умолчанию, текстура считается текстурой для чтения.

Пример:

```
kernel void Nbody_kernel(write_only image2d_t OutImg, image2d_t InImg) {
…
};
```

Свойства объектов **image[2|3]d_t**

- Размеры: Ширина и высота. Для image3d_t так же глубина.

  int **get_image_width** (**image[2|3]d_t** *image)*
  int **get_image_height** (**image[2|3]d_t** *image)*
  int **get_image_depth** (**image3d_t** *image)*
  int2 **get_image_dim** (**image2d_t** *image)*
  int4 **get_image_dim** (**image3d_t** *image)*

- Количество каналов – 1, 2 или 4, в отдельное свойство не выносится, объединено с порядком каналов.
- Тип данных канала.

  int **get_image_channel_data_type** (image2d_t *image)*

  Доступные значения:

CLK_SNORM_INT8	CLK_UNORM_SHORT_101010	CLK_UNSIGNED_INT32
CLK_SNORM_INT16	CLK_SIGNED_INT8	CLK_HALF_FLOAT
CLK_UNORM_INT8	CLK_SIGNED_INT16	CLK_FLOAT
CLK_UNORM_INT16	CLK_SIGNED_INT32	
CLK_UNORM_SHORT_565	CLK_UNSIGNED_INT8	
CLK_UNORM_SHORT_555	CLK_UNSIGNED_INT16	

Свойства объектов **image[2|3]d_t**

- Порядок каналов

  int **get_image_channel_order** (**image[2|3]d_t** *image)*

  Доступные значения:

Значение	Компоненты
CL_R	(r, 0.0, 0.0, 1.0)
CL_A	(0.0, 0.0, 0.0, a)
CL_RG	(r, g, 0.0, 1.0)
CL_RA	(r, 0.0, 0.0, a)
CL_RGB	(r, g, b, 1.0)
CL_RGBA, CL_BGRA, CL_ARGB	(r, g, b, a)
CL_INTENSITY	(I, I, I, I)
CL_LUMINANCE	(L, L, L, 1.0)

Чтение из **image[2|3]d_t**. Сэмплеры

- Чтение данных осуществляется встроенными функциями OpenCL C с использованием специальных параметров – сэмплеров.
- Сэмплеры имеют тип **sampler_t**
- Сэмплер может быть передан в качестве аргумента ядра или определен как константа.
- **sampler_t** – 32-битный набор флагов.

sampler_t состоит из следующих полей:

- Нормализация координат. Значения:
  - **CLK_NORMALIZED_COORDS_TRUE**
  - **CLK_NORMALIZED_COORDS_FALSE**
- Обрезка адресов. Значения:
  - **CLK_ADDRESS_REPEAT**
  - **CLK_ADDRESS_CLAMP_TO_EDGE**
  - **CLK_ADDRESS_CLAMP**
  - **CLK_ADDRESS_NONE**
- Фильтрация. Значения:
  - **CLK_FILTER_NEAREST**
  - **CLK_FILTER_LINEAR**

Пример:

```
const sampler_t samplerA = CLK_NORMALIZED_COORDS_TRUE |
                           CLK_ADDRESS_REPEAT |
                           CLK_FILTER_NEAREST;
```

Чтение и запись в **image[2|3]d_t**.

float4 **read_imagef** (image2d_t *image*, sampler_t *sampler*, int2 *coord*)
float4 **read_imagef** (image2d_t *image*, sampler_t *sampler*, float2 *coord*)
float4 **read_imagef** (image3d_t *image*, sampler_t *sampler*, int4 *coord*)
float4 **read_imagef** (image3d_t *image*, sampler_t *sampler*, float4 *coord*)

int4 **read_imagei** (image2d_t *image*, sampler_t *sampler*, int2 *coord*)
int4 **read_imagei** (image2d_t *image*, sampler_t *sampler*, float2 *coord*)
int4 **read_imagei** (image3d_t *image*, sampler_t *sampler*, int4 *coord*)
int4 **read_imagei** (image3d_t *image*, sampler_t *sampler*, float4 *coord*)

unsigned int4 **read_imageui** (image2d_t *image*, sampler_t *sampler*, int2 *coord*)
unsigned int4 **read_imageui** (image2d_t *image*, sampler_t *sampler*, float2 *coord*)
unsigned int4 **read_imageui** (image3d_t *image*, sampler_t *sampler*, int4 *coord*)
unsigned int4 **read_imageui** (image3d_t *image*, sampler_t *sampler*, float4 *coord*)

void **write_imagef** (image2d_t *image*, int2 *coord*, float4 *color*)
void **write_imagei** (image2d_t *image*, int2 *coord*, int4 *color*)
void **write_imageui** (image2d_t *image*, int2 *coord*, unsigned int4 *color*)

Текстура OpenCL на Хосте

cl_mem **clCreateImage2D** (cl_context *context*, cl_mem_flags *flags*,
const cl_image_format **image_format*, size_t *image_width*,
size_t *image_height*, size_t *image_row_pitch*, void **host_ptr*,
cl_int **errcode_ret)*

cl_mem **clCreateImage3D** (cl_context *context*, cl_mem_flags *flags*,
const cl_image_format **image_format*, size_t *image_width*,
size_t *image_height*, size_t *image_depth*, size_t *image_row_pitch*,
size_t *image_slice_pitch*, void **host_ptr*, cl_int **errcode_ret)*

```
typedef struct _cl_image_format {
        cl_channel_order  image_channel_order;
        cl_channel_type  image_channel_data_type;
} cl_image_format;
```

Чтение, запись и копирование текстур OpenCL

cl_int **clEnqueueReadImage** (cl_command_queue *command_queue*, cl_mem *image*,
cl_bool *blocking_read*, const size_t *origin[3]*, const size_t *region[3]*,
size_t *row_pitch*, size_t *slice_pitch*, void *ptr, cl_uint *num_events_in_wait_list*,
const cl_event *event_wait_list, cl_event *event)

cl_int **clEnqueueWriteImage** (cl_command_queue *command_queue*, cl_mem *image*,
cl_bool *blocking_write*, const size_t *origin[3]*,const size_t *region[3]*,
size_t *input_row_pitch*, size_t *input_slice_pitch*, const void * *ptr*,
cl_uint *num_events_in_wait_list*, const cl_event *event_wait_list,
cl_event *event)

cl_int **clEnqueueCopyImageToBuffer** (cl_command_queue *command_queue*,
cl_mem *src_image*, cl_mem *dst_buffer*, const size_t *src_origin[3]*,
const size_t *region[3]*, size_t *dst_offset*, cl_uint *num_events_in_wait_list*,
const cl_event *event_wait_list, cl_event *event)

cl_int **clEnqueueCopyBufferToImage** (cl_command_queue *command_queue*,
cl_mem *src_buffer*, cl_mem *dst_image*, size_t *src_offset*, const size_t *dst_origin[3]*,
const size_t *region[3]*, cl_uint n*um_events_in_wait_list*,
const cl_event *event_wait_list, cl_event *event)

Сэмплеры на Хосте

cl_sampler – дескриптор сэмплера

cl_sampler **clCreateSampler** (cl_context *context,* cl_bool *normalized_coords,*
cl_addressing_mode *addressing_mode,* cl_filter_mode *filter_mode,*
cl_int **errcode_ret)*

cl_int **clGetSamplerInfo** (cl_sampler *sampler,* cl_sampler_info *param_name,*
size_t *param_value_size,* void **param_value,*
size_t **param_value_size_ret)*

cl_sampler_info:

- CL_SAMPLER_ADDRESSING_MODE
- CL_SAMPLER_FILTER_MODE
- CL_SAMPLER_NORMALIZED_COORDS

cl_int **clReleaseSampler** (cl_sampler *sampler)*