

CUDA. Texturing

F.CS306 ПАРАЛЛЕЛ ПРОГРАММЧЛАЛ – Лекц 13

Г.ГАНБАТ
ganbatg@must.edu.mn

Хичээлийн агуулга

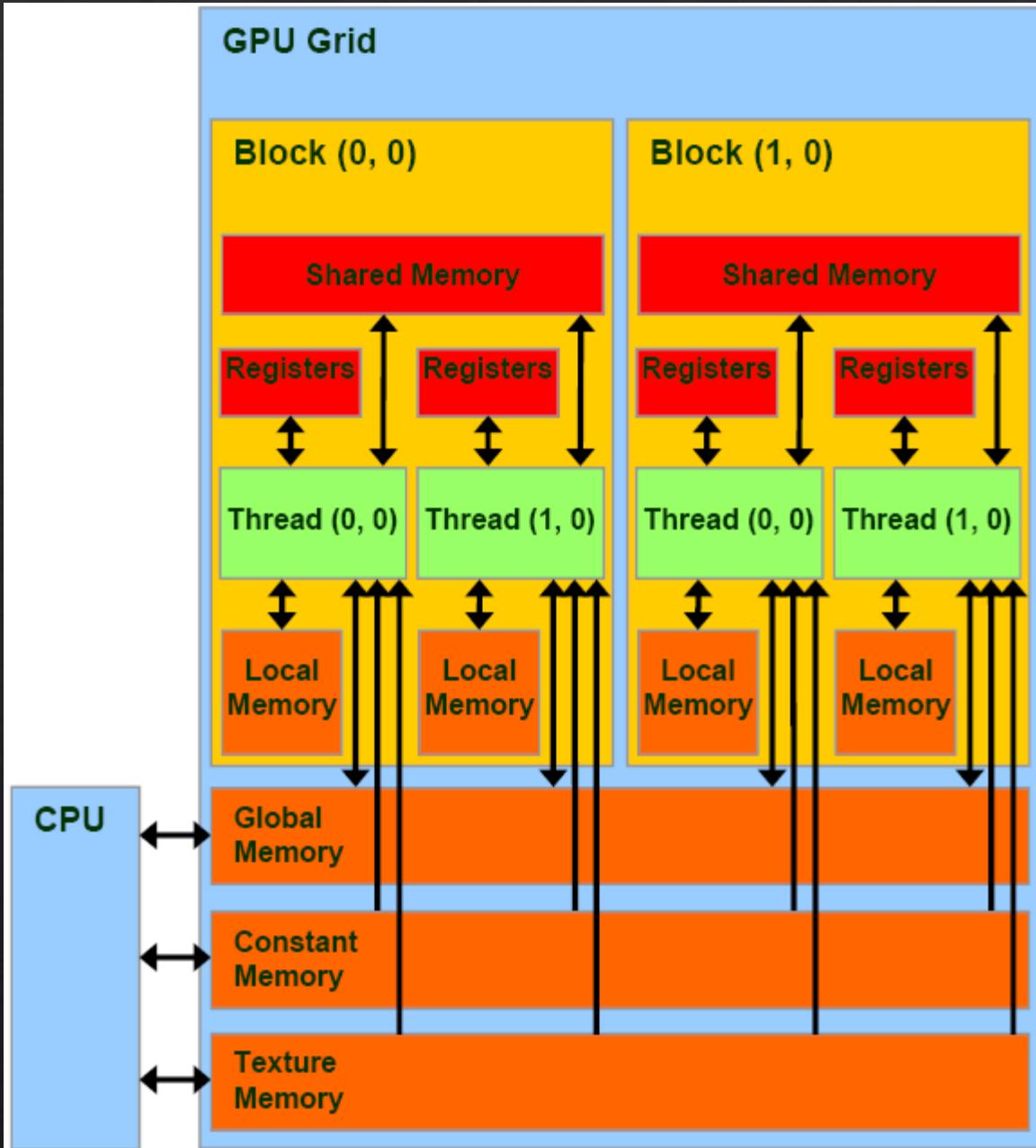
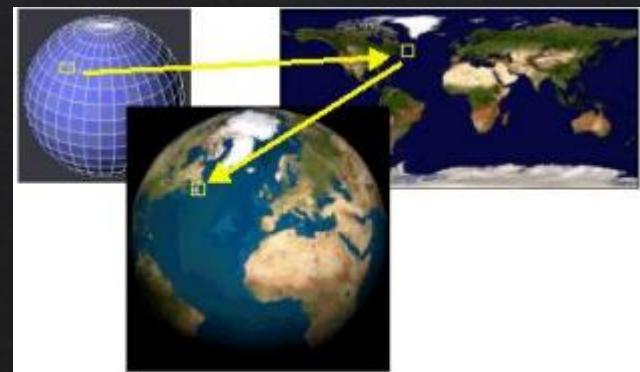
- ❖ Texture санах ой
- ❖ CUDA массив
- ❖ Texture fetch
- ❖ Texture санах ой хэрэглэх
- ❖ Зарим CUDA сангүүд

Texture санах ой

- ❖ Read-Only
- ❖ Хандалтын тодорхой pattern хэрэглэн уншихад бүтээмжийг нэмэгдүүлах ба санах ойн ачааллыг бууруулах боломжтой
- ❖ Анх сонгодог OpenGL ба DirectX гадаргууг зурахад зориулан зохиогдсон.
- ❖ Гэхдээ тооцоололд, ялангуяа компьютерийн харааны программд ашигтай зарим чанаруудтай
- ❖ Texture санах ой чипэн дээр хадгалагддаг
 - ❖ Зарим тохиолдолд санах ойн хүсэлтийг багасгаснаар илүү ашигтай bandwith-ийг бий болгодог

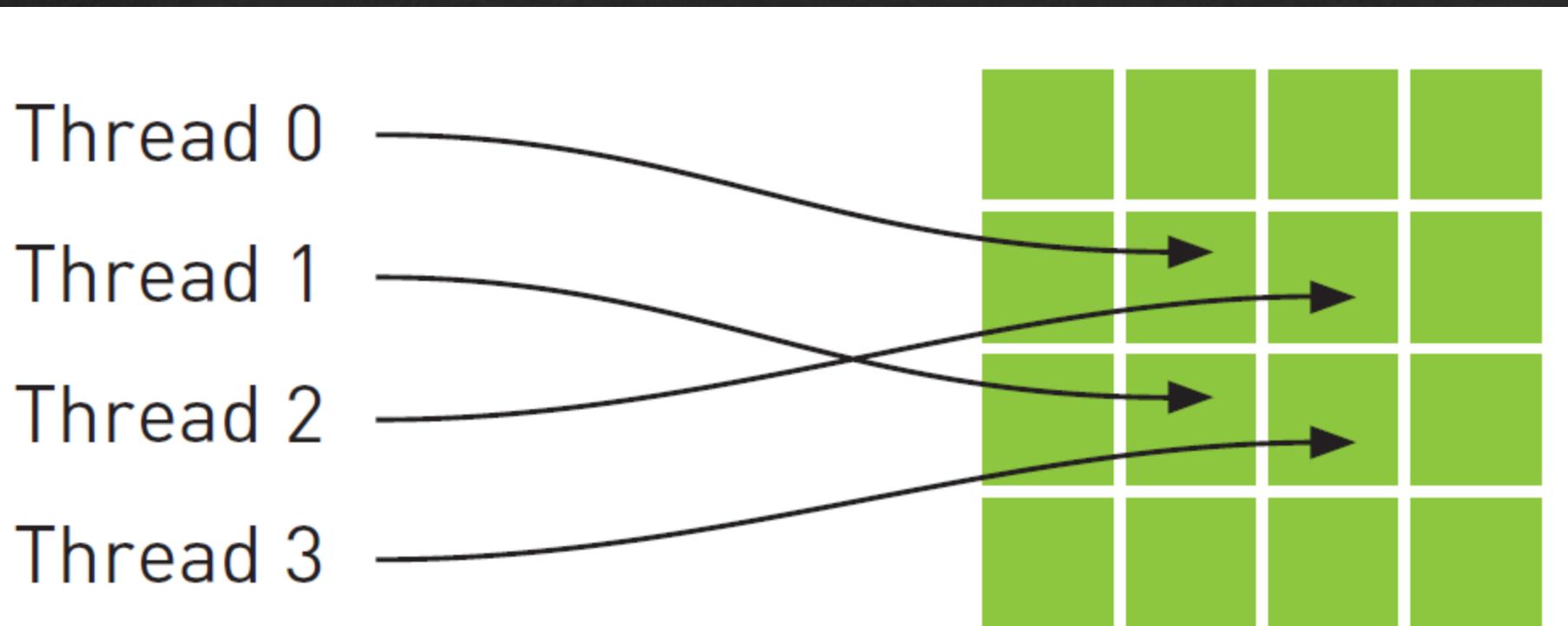
Texture саnahах ой

- ◆ 3D олон өнцөгт
модел дээр 2D
“skin”
харгалзуулах
зэрэг
үйлдлүүдийг
гүйцэтгэдэг



Texture санах ой

- ❖ Texture кэшүүд нь spatial locality-д суурилсан санах ойн хандалтын pattern-тэй график программуудад зориулагдан зохиогдсон
 - ❖ Тооцооллын хувьд, thread нь ойролцоох thread-үүдийн уншсан хаягны ойролцоо хаягнаас унших магадлалтай гэсэн үг юм.



Texture санах ой

- ❖ Texture кэш нь 2D spatial locality-д илүү тохиромжтой
- ❖ DRAM-ын нэг хэсэг
- ❖ Texture унших процессийг “*texture fetch*” гэнэ.
- ❖ 1D, 2D болон 3D хэмжээст массив илэрхийлэгдэх боломжтой
- ❖ Массивын элементүүдийг “texels” гэнэ

CUDA массив

- ❖ Texturing-ийг дэмжих зориулалттай
- ❖ Device санах ойд байрладаг
- ❖ Дэлгэмэл, шугаман бүтэцтэй
- ❖ Заагчаар хаяглагдах боломжгүй
- ❖ Санах ойн үүрүүдийн хаяглалт дараах 2 агуулгыг багтаана
 - ❖ Массивын handle
 - ❖ 1D, 2D болон 3D координатуудын олонлог

Texture fetch

- ❖ Texture заавар
 - ❖ texture санах ойн аль хэсэг fetch хийгдэхийг тодорхойлно
 - ❖ texture санах ойрүү Runtime функцээр холбогдсон байх ёстой
 - ❖ Атрибут:
 - ❖ Texture нь 1D, 2D эсвэл 3D-ээр хаяглагдсан байна
 - ❖ Оролтын болон гаралтын өгөгдлийн төрөлтэй
 - ❖ Оролтын координатууд нь кодчилогдсон байна
 - ❖ Гүйцэтгэх ёстой боловсруулалт
 - ❖ Texels нь энгийн төрөлтэй: integer, single/double precision floating point гэх мэт

Texture санах ой хэрэглэх

1. **Declare:** CUDA дотор texture санах ойг тодорхойлох
2. **Bind:** CUDA доторх texture санах ойг texture заавартаа харгалзуулах
3. **Read:** texture санах ойг CUDA кернел дотор texture заавартаа унших
4. **Unbind:** texture санах ойг CUDA доторх texture заавараас чөлөөлөх

Алхам 1: Declare

texture <type, dim, readmore> texture_reference;

- ❖ **texture_reference:** хувьсагч
- ❖ **type:** texels-ын төрөл –int, float
- ❖ **dim:** 1 (default),2 ,3
- ❖ **readmore:** хандалтаар буцаах texel-ын хувилбарын флаг
 - ❖ `cudaReadModeElementType` (default) : өөрчлөлтгүй
 - ❖ `cudaReadModeNormalizedFloat`
 - ❖ Int төрөлтэй байхад буцаах тэмдэгтэй утга [-1.0,1.0], тэмдэггүй утга [0.0, 1.0] байна.
- ❖ **Жнь:** `texture <float, 2, cudaReadMoreElementType> texture_reference;`

Алхам 2: Bind

`cudaBindtexture (size *t offset, & texture_reference , const void * devptr, size_t size) ;`

- ❖ DevPtr-р зааж өгсөн санах ойн талбайн хэмжээтэй байтыг текстэн лавлагаа texture_reference руу холбож өгдөг.
- ❖ devPtr: Төхөөрөмж дээрх санах ойн талбар
- ❖ хэмжээ: devPtr-ээр зааж өгсөн санах ойн талбарын хэмжээ

Алхам 3: Read

tex1Dfetch()

```
texture <int,1,cudaReadModeElementType> texref;  
  
__global__ void textureTest(int *out){  
    int tid = blockIdx.x * blockDim.x + threadIdx.x;  
    float x;  
    int i;  
    for(i=0; i<30; i++)  
        x = tex1Dfetch(texref, i);  
}
```

Алхам 4: Unbind

cudaBindtexture (testure_reference) ;

CUDA Libraries: CUBLAS

- ❖ • Cuda Based Linear Algebra Subroutines
 - ❖ – conjugate gradient, linear solvers, ...
- ❖ • Single GPU or Multiple GPUs
- ❖ • Support CUDA Stream
- ❖ • Basic preparation
 - ❖ – Install CUDA Toolkit
 - ❖ – Include cublas_v2.h
 - ❖ – Link cublas.lib

CUDA Libraries: CUBLAS

- ❖ Some basic tips
 - ❖ – Every CUBLAS function needs a handle
 - ❖ – The CUBLAS function must be written between `cublasCreate()` and `cublasDestory()`
 - ❖ – Every CUBLAS function returns a `cublasStatus_t` to report the state of execution.
 - ❖ – Column-major storage

CUDA Libraries: CUFFT

- ❖ Cuda Based Fast Fourier Transform Library.
 - ❖ • The FFT is a divide-and-conquer algorithm
 - ❖ • Computes FFT on Nvidia CUDA
 - ❖ • 1D, 2D, and 3D
 - ❖ • The CUFFT library is freely available as part of the CUDA Toolkit
 - ❖ • `#include<cufft.h>`

Баярлалаа.