**CPSC/ECE 4780/6780**

**General-Purpose Computation on Graphical Processing Units (GPGPU)**

**Exam 2**

**On Canvas from 11:00am to 12:15pm Thursday 07/23/2020**

**Please submit your Answer Sheet back on Canvas before 12:15pm**

NAME: \_\_\_\_\_\_\_\_BIYANG FU\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ SCORE: \_89\_\_\_\_\_ / 100

1. **Multiple Choices (20 points): each question has more than one correct answer.**
2. (4 points) Which of the following are OpenGL Callback functions? ABCD

(A) glutDisplayFunc

(B) glutReshapeFunc

(C) glutIdleFunc

(D) glutMouseFunc

1. (4 points) Which of the following statements are true about OpenCL? ABCD
2. OpenCL is a platform for heterogeneous computing, which means it’s a portable language for GPUs, CPUs, and other processors.
3. OpenCL’s platform API allows applications to query for OpenCL devices and manage them through a context.
4. OpenCL’s runtime API uses contexts for managing objects such as command queues, memory objects, and kernel objects, etc.

-1

1. OpenCL kernels can only be queued in-order.
2. (4 points) The supported number of elements in an OpenCL vector data type can be? ABD
3. 2
4. 4
5. 6
6. 8
7. (4 points) The function barrier(mem\_fence\_flag) makes sure all work-items within a work-group has executed before any work-item can continue. What can the mem\_fence\_flag be? AC
8. CLK\_LOCAL\_MEM\_FENCE
9. CL\_RUNNING
10. CLK\_GLOBAL\_MEM\_FENCE
11. CL\_COMPLETE
12. (4 points) Which functions can probably be used to project a scene onto the screen in OpenGL? ACD
13. gluPerspective()
14. glViewPort()
15. glOrtho()
16. gluOrtho2D()
17. **Short Answer (20 points)**
18. (4 points) What is mipmapping in OpenGL? What advantages does it provide?

A series of prefiltered texture maps of decreasing resolutions, called mipmaps.

When using mipmapping, OpenGL automatically determines which texture map to use based on the size (in pixels) of the object being mapped. With this approach, the level of detail in the texture map is appropriate for the image that's drawn on the screen - as the image of the object gets smaller, the size of the texture map decreases. Mipmapping requires some extra computation and texture storage area; however, when it's not used, textures that are mapped onto smaller objects might shimmer and flash as the objects move.

Mipmapping allows for prefiltered texture maps of decreasing resolutions

• Lessens interpolation errors for smaller textured objects

• Requires extra computation and texture storage area

1. (4 points) Name at least two differences between OpenCL and CUDA (other than naming conventions).

*(1).OpenCL is implemented by many vendors(intel,nvidia,amd,xilinx…), CUDA is only implemented by Nvidia*

*(2). Using OpenCL, migrating to other platforms is easy. Using CUDA, it is harder to migrate because it has explicit optimization options to harness more GFLOPS percentage per architecture. Once optimized, it becomes more work to do. Without optimizations, OpenCL is very similar in performance.*

*(3). OpenCL abstracts much of hardware away from developer. So developer just writes parallelized code and fine tunes launch parameters later. CUDA doesn’t abstract that much and lets developers optimize specifically for the underlying Nvidia architecture.*

1. (4 points) Translate the following CUDA concepts to corresponding concepts in OpenCL: i) thread; ii) block; iii) shared memory; iv) \_\_syncthreads().

*(1). Work-item*

*(2). Work-group*

*(3). Local memory*

*(4). Barrier()*

1. (4 points) In OpenCL, what happens if I set global\_work\_size[0] = 100,000, local\_work\_size[0] = 36? How to resolve the problem?

*100,000/36 = 2777.78*

*(int) (100000+36-1)/36 = 2778*

*Set global\_work\_size[0] to 2778\*36 = 100,008*

*Specify more work-items than required*

*• int num\_work\_groups = (global\_work\_size[0] + local\_work\_size[0] - 1) /*

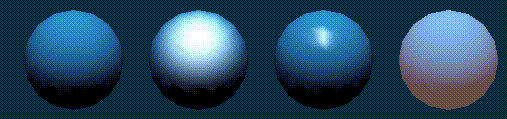
*local\_work\_size[0] = 2778;*

*• global\_work\_size[0] = num\_work\_groups \* local\_work\_size[0] = 100008;*

*• Set global\_work\_size[0] = 100008 (gives 2778 work-groups)*

*• Kernel must check for going out of bounds in your arrays*

1. (4 points) What kinds of materials properties (e.g., ambient, diffuse, specular, shininess, emission) can be observed from each of the following balls? (Just name the most significant material property for each ball.)



*(1). Diffuse reflection only; no ambient or specular*

*(2). Diffuse and specular reflection; low shininess; no ambient*

*(3). Diffuse and specular reflection; high shininess; no ambient*

*(4). Diffuse reflection; emission; no ambient or specular reflection*

1. **Problem Solving (60 points)**
2. (20 points) Suppose we wish to transform a primitive object as follows:
3. First, rotate by an angle of 270 degrees about the x axis;
4. Next, translate by 6 in x, 4 in y, and 8 in z;
5. Lastly, scale by ½ in y and 2 in z.

Write the corresponding function calls in OpenGL to perform the transformation.

*1). glRotatef(270.0, 1.0, 0.0, 0.0);*

*2). glTranslatef(6.0, 4.0, 8.0);*

*3). glScalef(1.0, 0.5, 2.0);*

1. (20 points) Suppose we initialized two vector type objects in OpenCL as following:

int4 v1 = (int4) -7;

int4 v2 = (int4) (0, 1, 2, 3);

After the following series of vector operations, what the final v3 will be?

v1 = abs(v1);

v2.s23 = v1.hi;

int8 v3 = (int8) (v1, v2.xw, v2.even);

v3 = v3 + 2;

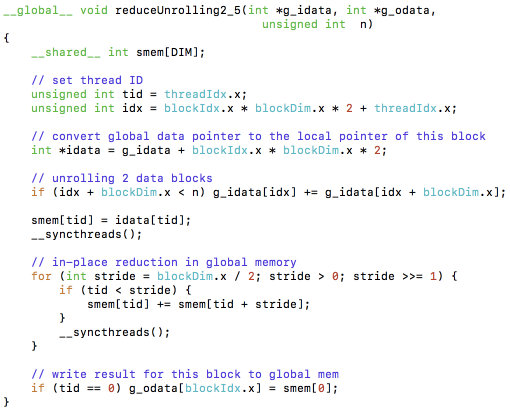
v1 = (7,7,7,7);

v2 = (0,1,7,7);

v3 = (7,7,7,7,0,7,0,7);

v3 = (9,9,9,9,2,9,2,9);

1. (20 points) In Parallel Reduction example in CUDA, we have one implementation as below:



Rewrite this kernel in OpenCL:

\_\_kernel void reduceUnrolling2\_5(\_\_global int \*g\_idata, \_\_global int \*g\_odata, uint n) {

\_local int

int loc\_id = get\_local\_id(0);

int group\_size = get\_local\_size(0);

-10

int gbl\_id = get\_global\_id(0);

g\_idata[loc\_id] = [gbl\_id];

for (uint stride = group\_size/2; stride > 0; stride /= 2) {

barrier(CLK\_LOCAL\_MEMFENCE);

if (loc\_id < stride)

g\_idata[loc\_id] += g\_idata[loc\_id + stride];

}

if (loc\_id == 0)

g\_odata[get\_group\_id(0)] = g\_idata[0];

}