Test #2

Due Jun 12 at 11:59pm **Points** 100 **Questions** 40
Available Jun 8 at 12:01am - Jun 12 at 11:59pm 5 days **Time Limit** 60 Minutes

Instructions

This is called a "quiz" on Canvas, but it is really 100-point Test #2.

There is a time limit of 60 minutes on this test. Once you start, you must finish. Canvas will not let you pause and come back later.

This test is Open Notes, Closed Friends, and Closed Internet.

This is your last graded-anything in this class, so do a good job!

Thanks for a *great* quarter!

Attempt History

	Attempt	Time	Score
EST	Attempt 1	41 minutes	100 out of 100

Correct answers will be available on Jun 13 at 12:01am.

Score for this quiz: **100** out of 100
Submitted Jun 9 at 12:57am
This attempt took 41 minutes.

Question 1 — 2.5 / 2.5 pts

What does the OpenCL call "gid = get_global_id(0)" return?

- It tells you how big the global dataset is
- It tells you how big the local dataset is
- It tells you where you are in the global dataset
- It tells you where you are in the local dataset

Question 2

2.5 / 2.5 pts

The OpenGL-created Vertex Buffer Object looks, to OpenCL, like:

- A table of XYZ coordinates
- A linked list of XYZ coordinates
- A collection of separate X[ ], Y[ ], and Z[ ] arrays
- A hash table of XYZ arrays

Question 3

2.5 / 2.5 pts

In the OpenCL call "gid = get_global_id(0)", what does the argument of 0 indicate?

- Since the time at which the program started
- Relative to the first element of the dataset
- That you want only one value returned

- [x] In the X dimension

Question 4

2.5 / 2.5 pts

What is special about using OpenCL/OpenGL interoperability?

- [ ] It saves electrical power
- [x] The data never leaves GPU memory
- [ ] It allows GPU graphics to be driven by CPU multicore computing
- [ ] The Khronos Group gives you a certificate for doing it

Question 5

2.5 / 2.5 pts

In running your GPU program on the DGX server, you needed to type, for example:

- [ ] ./montecarlo
- [ ] ./montecarlo.cu
- [ ] slurm montecarlo.csh
- [x] sbatch montecarlo.bash

Question 6

2.5 / 2.5 pts

Where is the OpenCL kernel compiler (as we used it this quarter) located?

- ○ On the Internet
- ○ As an external program
- ◉ In the OpenCL driver
- ○ In the GPU

Question 7

2.5 / 2.5 pts

When is it OK to use the less-precise "fast_normalize( )" call instead of the full-precision "normalize( )" call?

- ◉ When using OpenCL for computer graphics
- ○ When using OpenCL for scientific computing
- ○ Never
- ○ Always

Question 8

2.5 / 2.5 pts

Projects #1 and #5 ran roughly the same code on a CPU and a GPU, respectively. What can you say about their relative performance in Trials/Second?

- ◉ The GPU version was way faster

- The CPU version was way faster
- Within 10%, the two versions had about the same performance

Question 9

2.5 / 2.5 pts

Comparing CPUs and GPUs, it is correct to say:

- (selected) CPUs are better with linked-list data structures, GPUs are better with data parallel arrays
- GPUs are better with integers, CPUs are better with floating-point
- CPUs are better with integers, GPUs are better with floating-point
- GPUs are better with linked-list data structures, CPUs are better with data parallel arrays

Question 10

2.5 / 2.5 pts

In your CUDA program, how do you show that a function is the GPU Kernel function?

- By labeling it with __device__
- (selected) By labeling it with __global__
- By labeling it with __local__
- By labeling it with __kernel__

Question 11

2.5 / 2.5 pts

The primary purpose of MPI is to:

- ○ To allow multicore
- ◉ Allow parallel computing among separate computers
- ○ To get computing access to a GPU
- ○ To get SIMD performance

Question 12

2.5 / 2.5 pts

In Project #3, the Functional Decomposition project, each individual quantity's function needed to have three barriers. The *second* barrier was there to:

- ○ Indicate when the Watcher thread could print values
- ○ Indicate when it was time to increment the month
- ○ Indicate when that quantity's function was done computing that quantity's next value
- ◉ Indicate when that quantity's next value was done being copied to the global state

Question 13

2.5 / 2.5 pts

An MPI “Broadcast” operation involves:

- ○ Many functions: multiple broadcast senders and a single broadcast receiver
- ○ Many functions: a broadcast sender and one unique broadcast receiver function per CPU
- ○ Two functions: a broadcast sender and a broadcast receiver
- ◉ A single function regardless of if you are sending or receiving

Question 14

2.5 / 2.5 pts

MPI Reductions:

- ○ Must be implemented by your application
- ○ Are unnecessary because of the number of CPUs
- ○ Are unnecessary because of the SIMD units on the CPUs
- ◉ Are a built-in feature of the MPI API

Question 15

2.5 / 2.5 pts

GPU Reductions:

- Are unnecessary because of the GPU speed
- Are a built-in feature of the OpenCL API just like they are in OpenMP
- Are unnecessary because of the GPU hardware instruction set
- (selected) Must be implemented by the .cl function you write

Question 16

2.5 / 2.5 pts

The OpenCL function clCreateFromGLBuffer( ):

- Deletes an OpenCL buffer and replaces it with an OpenGL-compatible vertex buffer object
- Allocates an OpenCL device memory buffer
- (selected) Creates an OpenCL device memory pointer from an OpenGL graphics vertex buffer object
- Creates an OpenGL graphics vertex buffer object

Question 17

2.5 / 2.5 pts

A Sphere can be represented as four floats. What are they?

- The four hyperbolic radii
- The STP of the texture coordinates and the radius

- [x] XYZ of the center position and the radius
- [ ] XYZ of the surface normal and the radius

Question 18

2.5 / 2.5 pts

What is the advantage of a Fused-Multiply-Add?

- [ ] You only have to write one line of code instead of two
- [x] It can perform a multiply plus an add in about the same time as it could have done the multiply alone
- [ ] It implies that a SIMD operation should be performed
- [ ] It reduces the possibility of False Sharing

Question 19

2.5 / 2.5 pts

What does the function cudaMalloc( ) do?

- [ ] Pre-allocates space in the GPU cache
- [ ] Allocates space in both CPU and GPU memory
- [x] Allocates space in GPU memory
- [ ] Allocates space in CPU memory

Question 20

2.5 / 2.5 pts

There were several cases when OpenCL, in querying what sort of system it was running on, called the same function twice:

*status = **clGetDeviceIDs**(platform, CL_DEVICE_TYPE_ALL, 0, NULL, &numDevices);*

. . .

*status = **clGetDeviceIDs**(platform, CL_DEVICE_TYPE_ALL, numDevices, devices, NULL);*

Why?

- ○ So you could get the information from two separate platforms
- ○ Once to get the information from a CPU, and once to get it from a GPU
- ○ Once to get the information from a CPU/GPU, and once to get it from an FPGA (Field-Programmable Gate Array)
- ◉ Once to get the number of something, and once to retrieve that much information

Question 21

2.5 / 2.5 pts

MPI follows what parallel programming model?

- ○ Multiple Instructions, Multiple Data (MIMD)
- ○ Single Instruction, Multiple Data (SIMD)
- ◉ Single Program, Multiple Data (SPMD)
- ○ Single Instruction, Single Data (SISD)

Question 22

2.5 / 2.5 pts

In this class, the letters "MPI" stand for:

- ○ MegaCalculation Per Instruction
- ○ Many-Processor Interfaces
- ◉ Message Passing Interface
- ○ Millions of Processor Instructions

Question 23

2.5 / 2.5 pts

In Project #3, the Functional Decomposition project, each individual quantity's function needed to have three barriers. The *first* barrier was there to:

- ○

Indicate when that quantity's next value was done being copied to the global state

- ○ Indicate when it was time to increment the month
- ◉ Indicate when that quantity's function was done computing that quantity's next value
- ○ Indicate when the Watcher thread could print values

Question 24

2.5 / 2.5 pts

Jane Parallel uses this line of OpenCL code:

status = clEnqueueNDRangeKernel(cmdQueue, kernel, 1, NULL, A, B, C, D, E);

what is the E variable used for?

- ○ It specifies how many events to wait for
- ○ The context to use
- ○ The globalWorkSize
- ◉ It specifies what event to throw when this kernel is completed

Question 25

2.5 / 2.5 pts

When OpenCL and OpenGL work together:

- ○ It is just OpenCL that is able to access the vertex buffer

- They both access the vertex buffer at the same time
- It is just OpenGL that is able to access the vertex buffer
- They take turns accessing the vertex buffer

Question 26

2.5 / 2.5 pts

Joe Parallel wants to use OpenCL kernels to implement the graph execution structure shown here. How?

- He can't -- it is not possible in the current version of OpenCL, but might be in the future
- He has A, B, and C each throw events, and has C, C, and D (respectively) wait for those events.
- He turns C and D into special OpenCL reduction functions
- He sets up barriers at C and D

Question 27

2.5 / 2.5 pts

The advantage of using SSE SIMD is:

- It uses a stack variable to hold each multiplication product
- You should be able to get a 4x performance improvement in array multiplication
- It uses the stack to hold its pointers
- You can perform a multiply and an add in one instruction

Question 28

2.5 / 2.5 pts

As of the writing of our class notes, the 2022 SC conference (International Conference for High Performance Computing, Networking, Storage, and Analysis) will be held:

- In Los Angeles, CA
- In Washington, DC
- In Dallas, TX
- Totally online

Question 29

2.5 / 2.5 pts

Let's beat the Yellow Robot metaphor to death. The yellow robot's grippers represent:

- Compute Units

- [x] Processing Elements
- [ ] Separate CPU cores
- [ ] SIMD banks

Question 30

2.5 / 2.5 pts

In the CUDA call:

cudaMemcpy(**A, B,** NUM_ELEMENTS*sizeof(float), cudaMemcpyHostToDevice); **:**

- [ ] The CPU array A gets copied to the GPU array B
- [ ] The GPU array A gets copied to the CPU array B
- [ ] The GPU array B gets copied to the CPU array A
- [x] The CPU array B gets copied to the GPU array A

Question 31

2.5 / 2.5 pts

In Project #1, you performed a multicore Monte Carlo simulation by using the NUMTRIALS for-loop. In Project #5, you re-created that same simulation using CUDA without any for-loop. Where did that NUMTRIALS for-loop go?

- [ ] You don't need to include it– CUDA is smart enough to figure out what you are trying to do and adds it for you

It is still there – it has just been written in CUDA-code instead of C/C++

It is not needed – it has been replaced by duplicating the simulation onto thousands of threads

It is still there – it has just been replaced with the special CUDA "foreach" capability

Question 32

2.5 / 2.5 pts

Jane Parallel uses this line of OpenCL code:

status = clEnqueueNDRangeKernel(cmdQueue, kernel, 1, NULL, A, B, C, D, E);

what are the C and D variables used for?

They specify what event to throw when this kernel is completed

The globalWorkSize and the localWorkSize

They specify how many events to wait for and which ones they are

The context to use

Question 33

2.5 / 2.5 pts

What is the relationship between Global Data Set Size, Work Group Size, and the Number of Work Groups?

- Global Data Set Size = (Number of Work Groups)^2 [i.e., squared]
- Global Data Set Size = (Number of Work Groups) * (Work Group Size)
- Work Group Size = (Global Data Set Size) * (Number of Work Groups)
- Number of Work Groups = (Work Group Size) * (Global Data Set Size)

Question 34

2.5 / 2.5 pts

Why did Jane Parallel use those typedefs (point, vector, color, sphere) in her OpenCL code?

- The compiler requires it
- Those were indeed the real OpenCL names for those types of variables
- It makes it more obvious what her code is doing
- The OpenCL standard requires it

Question 35

2.5 / 2.5 pts

In MPI, the phrase “scatter/gather” means:

To break a problem up into pieces, give each piece to a separate computers, and then gather up the results

To gather input data from the disk and then scatter it out to different CPUs for computation

To setup Barriers across all CPUs

To use the MPI_Bcast( ) function to get information to all other CPUs

Question 36

2.5 / 2.5 pts

In your C/C++ CUDA program, how do you show that you are making a call to the GPU Kernel function?

With the cudaExecuteKernel( ) function

With the cudaEnqueueNDRangeKernel( ) function

With the >>> … <<< (chevron) syntax

With the <<< … >>> (chevron) syntax

Question 37

2.5 / 2.5 pts

In MPI, a "derived type" is:

Linking multiple MPI calls together to send both MPI_FLOATs and MPI_INTs

- Being able to pack multiple MPI_INTs into a single MPI_LONG
- Being able to pack multiple MPI_CHARs into a single MPI_LONG
- You creating a struct that can act just like MPI_FLOAT, MPI_INT, etc.

Question 38

2.5 / 2.5 pts

What is one reason that OpenCL uses a Command Queue?

- To be compatible with CPU-SIMD
- So OpenCL can gobble up commands as fast as it can
- So that you don't need to know what each command does
- This paradigm is forced by how the hardware works

Question 39

2.5 / 2.5 pts

In MPI, a computer's "rank" is:

- Its integer identifier
- Its processing power
- The number of cores it has
- Its priority

Question 40

2.5 / 2.5 pts

Let's beat the Yellow Robot metaphor to death. The yellow robot represents:

- ○ A separate CPU core
- ○ A Processing Element
- ◉ A Compute Unit
- ○ A SIMD bank

Quiz Score: **100** out of 100