Test #2

截止日期 6月12日 23:59 **分數** 100 **問題** 40
可用 6月8日 0:01 - 6月12日 23:59 5 天 **時間限制** 60 分鐘

說明

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!

嘗試記錄

	嘗試	時間	分數
最新的	嘗試 1	27 分鐘	得分：97.5；總分：100

正確答案將於 6月13日 0:01 可用。

此測驗的分數： 得分：**97.5**；總分：100
已提交6月8日 21:19
此嘗試持續 27 分鐘。

問題 1 — 2.5 / 2.5 分數

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

- (●) It makes it more obvious what her code is doing

The compiler requires it

Those were indeed the real OpenCL names for those types of variables

The OpenCL standard requires it

問題 2

2.5 / 2.5 分數

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

In the GPU

As an external program

On the Internet

In the OpenCL driver

問題 3

2.5 / 2.5 分數

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:

Totally online

In Washington, DC

In Dallas, TX

In Los Angeles, CA

問題 4

2.5 / 2.5 分數

What does the function cudaMalloc( ) do?

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

問題 5

2.5 / 2.5 分數

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

- With the cudaEnqueueNDRangeKernel( ) function
- With the cudaExecuteKernel( ) function
- With the >>> … <<< (chevron) syntax
- With the <<< … >>> (chevron) syntax

問題 6

2.5 / 2.5 分數

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 it was time to increment the month

Indicate when the Watcher thread could print values

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

Indicate when that quantity's function was done computing that quantity’s next value

問題 7

2.5 / 2.5 分數

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

./montecarlo.cu

slurm montecarlo.csh

sbatch montecarlo.bash

./montecarlo

問題 8

2.5 / 2.5 分數

In MPI, a computer’s “rank” is:

Its priority

Its processing power

The number of cores it has

Its integer identifier

問題 9

2.5 / 2.5 分數

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

- ○ XYZ of the surface normal and the radius
- ◉ XYZ of the center position and the radius
- ○ The four hyperbolic radii
- ○ The STP of the texture coordinates and the radius

問題 10

2.5 / 2.5 分數

In the CUDA call:

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

- ○ The GPU array A gets copied to the CPU array B
- ○ The GPU array B gets copied to the CPU array A
- ○ The CPU array A gets copied to the GPU array B
- ◉ The CPU array B gets copied to the GPU array A

問題 11

2.5 / 2.5 分數

MPI follows what parallel programming model?

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

問題 12

2.5 / 2.5 分數

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

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

問題 13

2.5 / 2.5 分數

Comparing CPUs and GPUs, it is correct to say:

GPUs are better with integers, CPUs are better with floating-point

CPUs are better with linked-list data structures, GPUs are better with data parallel arrays

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

問題 14

2.5 / 2.5 分數

What is one reason that OpenCL uses a Command Queue?

So that you don’t need to know what each command does

So OpenCL can gobble up commands as fast as it can

To be compatible with CPU-SIMD

This paradigm is forced by how the hardware works

問題 15

2.5 / 2.5 分數

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 CPU version was way faster

Within 10%, the two versions had about the same performance

- The GPU version was way faster

問題 16

2.5 / 2.5 分數

The primary purpose of MPI is to:

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

問題 17

2.5 / 2.5 分數

When OpenCL and OpenGL work together:

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

問題 18

2.5 / 2.5 分數

In CUDA, how many threads are in each Warp?

- 32
- 128
- 16
- 64

問題 19

2.5 / 2.5 分數

The OpenCL function clCreateFromGLBuffer( ):

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

問題 20

2.5 / 2.5 分數

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 it was time to increment the month

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

Indicate when that quantity's function was done computing that quantity's next value

Indicate when the Watcher thread could print values

問題 21

2.5 / 2.5 分數

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

Indicate when that quantity's function was done computing that quantity's next value

Indicate when the Watcher thread was done printing values

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

Indicate when it was time to increment the time of day

問題 22

2.5 / 2.5 分數

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

問題 23

2.5 / 2.5 分數

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 still there – it has just been replaced with the special CUDA “foreach” capability

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

問題 24

2.5 / 2.5 分數

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

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

問題 25

2.5 / 2.5 分數

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?

- The context to use
- 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

問題 26

2.5 / 2.5 分數

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?

- (●) Once to get the number of something, and once to retrieve that much information
- ( ) 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)

問題 27

2.5 / 2.5 分數

Function calls on GPU hardware:

- ( ) Happen through the special “GPU-stack”, which is different from a CPU-stack
- (●) End up being inlined because there is no stack to store arguments and return addresses
- ( ) Happen exactly the same way as CPU hardware implements them
- ( ) Happen through the special “GPU-stack”, which is the same as a CPU-stack

問題 28

2.5 / 2.5 分數

In this class, the letters “MPI” stand for:

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

不正確

問題 29

0 / 2.5 分數

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

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

問題 30

2.5 / 2.5 分數

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

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

A hash table of XYZ arrays

A linked list of XYZ coordinates

問題 31

2.5 / 2.5 分數

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

Work Group Size = (Global Data Set Size) * (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)

Number of Work Groups = (Work Group Size) * (Global Data Set Size)

問題 32

2.5 / 2.5 分數

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

By labeling it with __local__

By labeling it with __global__

By labeling it with __device__

By labeling it with __kernel__

問題 33

2.5 / 2.5 分數

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

- (●) In the X dimension
- ( ) Relative to the first element of the dataset
- ( ) That you want only one value returned
- ( ) Since the time at which the program started

問題 34

2.5 / 2.5 分數

MPI Reductions:

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

問題 35

2.5 / 2.5 分數

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

- (●) 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
- You only have to write one line of code instead of two

問題 36

2.5 / 2.5 分數

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
- (●) It specifies what event to throw when this kernel is completed
- The globalWorkSize
- The context to use

問題 37

2.5 / 2.5 分數

As of the writing of our class notes, the 2022 IEEE Visualization conference:

- Will be online-only
- (●) Will be held in Oklahoma City, OK
- Has been cancelled
- Will be held in Cambridge, MA

問題 38

2.5 / 2.5 分數

In MPI, a "derived type" is:

- 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.
- 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

問題 39

2.5 / 2.5 分數

One of the ways that CUDA differs from OpenCL is:

- In OpenCL, the C/C++ and GPU code are placed in the same file
- In CUDA, the C/C++ and GPU code are placed in the same file
- CUDA GPU code looks like Python and OpenCL GPU code looks like C
- CUDA GPU code looks like C and OpenCL GPU code looks like Python

問題 40

2.5 / 2.5 分數

GPU Reductions:

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

測驗分數： 得分：**97.5**；總分：100