NPTEL Online Certification Courses

Indian Institute of Technology Kharagpur

GPU Architectures

and Programming

Assignment- Week3

TYPE OF QUESTION: MCQ/MSQ

Number of questions: 10 **Total mark: 10 X 1 = 10**

MCQ Question

Question 1:

Memory management function for allocating memory in GPU device using CUDA is:

A. malloc
B. cudaMemcpy
C. cudaMalloc
D. calloc

Ans: B

Question 2:

Correct statement for allocation of memory chunk for 1024 floating point data element in GPU device is (Assume mem_chunk is device pointer):

A. cudaMalloc((void **)&mem_chunk, 1024*sizeof(float))
B. cudaMalloc((void **)&mem_chunk, 1024)
C. cudaMalloc((float **)&mem_chunk, 1024*sizeof(float))
D. cudaMalloc((float **)&mem_chunk, 1024)

Ans: A

Question 3:

Which of the following CUDA kernels correctly multiplies two 1D arrays element-wise:

A.
```
__global__ void multiplyArrays(float *A, float *B, float *C, int n) {
   int i = threadIdx.x + blockIdx.x * blockDim.x;
   if (i < n) {
      C[i] = A[i] * B[i];
   }
}
```

B.
```
__global__ void multiplyArrays(float *A, float *B, float *C, int n) {
   int i = threadIdx.x;
   C[i] = A[i] * B[i];
}
```

C.
```
__global__ void multiplyArrays(float *A, float *B, float *C, int n) {
   int i = blockIdx.x * blockDim.x;
   if (i < n) {
      C[i] = A[i] * B[i];
   }
}
```

D.
```
__global__ void multiplyArrays(float *A, float *B, float *C, int n) {
   int i = threadIdx.x + threadIdx.y * blockDim.x;
   if(i < n)
   C[i] = A[i] * B[i];
}
```

Ans: A

Question 4:

What is the output of the following code if a kernel fails during execution:

```
cudaError_t err = cudaGetLastError();
if (err != cudaSuccess) {
   printf("CUDA error: %s\n", cudaGetErrorString(err));
}
```

A. It prints the error code as an integer.
B. It prints the error message corresponding to the error code.
C. It resets the last error state and continues execution.
D. It terminates the program immediately.

Ans: B

Question 5:

Which among the following statements is True:

A. __device__ function can have return type other than void
B. __global__ function must always returns int
C. A function can not be declared as both __host__ and __device__ function
D. Every function is a default __global__ function

Ans: A

Question 6:

How is the total number of blocks to be launched determined in the code:

A. n / threadsPerBlock
B. threadsPerBlock / n
C. (n + threadsPerBlock - 1) / threadsPerBlock
D. (n - threadsPerBlock) / threadsPerBlock

Ans: C

Question 7:

Consider a vector addition kernel launch
vectorAdd<<<dim3(16, 8), dim3(32, 16)>>>(d_A, d_B, d_C, n);

What does the following CUDA kernel launch configuration imply:
A. 128 threads per block and 128 blocks in the grid.
B. 256 threads per block and 128 blocks in the grid.
C. 512 threads per block and 2048 blocks in the grid.
D. 512 threads per block and 128 blocks in the grid.

Ans: D

Solution:
Total number of blocks in the grid :16 * 8 = 128
Total number of threads in each block:32 * 16 =512

Question 8:

A typical CUDA program structure consists of five main steps:

1.Allocate GPU memories.
2.Copy data from CPU memory to GPU memory.
3.Invoke the CUDA kernel to perform program-specific computation.
4.Copy data back from GPU memory to CPU memory.
5.Destroy GPU memories.

CUDA API for the step no 1,2 and 5 respectively are

A. cudaMalloc(), cudaMemcpy(), cudaFree()
B. malloc(), copy(),free()
C. cudaAlloc(), cudaCopy(), cudaRelease()
D. cudaAlloc(), copy(), free()

Ans: A

Question 9:

Consider the following kernel specific system variables and the subsequent statements (a)-(d).

(i) gridDim.x
(ii) blockDim.y
(iii) blockidx.y
(iv) threadidx.x

(a) number of blocks in dimension x of multi-dim grid
(b) thread number inside a block in dimension x
(c) number of threads per block in dimension y of multi-dim block.
(d) block number for a thread in dimension y

Match and pair. Choose the correct option that represents the correct solution.

A. (i)-(a), (ii)-(c), (iii)-(d), (iv)-(b)
B. (i)-(a), (ii)-(b), (iii)-(c), (iv)-(d)
C. (i)-(d), (ii)-(a), (iii)-(b), (iv)-(c)
D. (i)-(a), (ii)-(c), (iii)-(b), (iv)-(d)

Answer A

Question 10:

A CUDA kernel is launched with the following configuration:
vectorMultiply<<<dim3(8, 2, 4), dim3(16, 32, 2)>>>(d_A, d_B, d_C, n);
How many total threads will be launched:

A. 4096
B. 16,384
C. 32,768
D. 65,536

Ans: D

Solution:

Total number of threads launched: 8*2*4*16*32*2= 65,536

***********END*************