CUDA Chapter 1 - Strong MCQs for Active Recall

1. 1. Which part of the CUDA programming model handles the coordination between the host and the device?

- The driver API

- The runtime API

- The GPU kernel

- The PTX assembler

✔ Answer: The runtime API

1. 2. What distinguishes a thread block from a warp in CUDA?

- A warp contains all threads in the grid

- A thread block is the smallest executable unit

- A thread block is a group of threads scheduled together; a warp is the smallest unit of execution (32 threads)

- They are equivalent terms

✔ Answer: A thread block is a group of threads scheduled together; a warp is the smallest unit of execution (32 threads)

1. 3. Why is global memory access generally slower in CUDA?

- Because it's allocated by the OS

- Because it uses paging

- Because it resides in DRAM and is not cached per-thread

- Because it requires thread synchronization

✔ Answer: Because it resides in DRAM and is not cached per-thread

1. 4. In CUDA, what is the purpose of using shared memory?

- To share data between host and device

- To persist data after kernel execution

- To enable fast data exchange between threads in the same block

- To store constant variables only

✔ Answer: To enable fast data exchange between threads in the same block

1. 5. What CUDA construct defines how many threads execute a kernel in parallel?

- Host loop count

- Grid and block dimensions

- Device architecture

- Kernel parameters

✔ Answer: Grid and block dimensions

1. 6. Which of these is true about kernel launches in CUDA?

- They are called from device to host

- They are wrapped in if-else conditions

- The triple-chevron syntax <<<...>>> defines execution configuration

- They return output via return values

✔ Answer: The triple-chevron syntax <<<...>>> defines execution configuration

1. 7. Which CUDA memory type has the lowest access latency?

- Global memory

- Shared memory

- Constant memory

- Registers

✔ Answer: Registers

1. 8. What happens if threads in a warp diverge in control flow?

- The warp gets split and executed in parallel

- Only the first thread executes

- They serialize and execute one path at a time

- CUDA throws a runtime error

✔ Answer: They serialize and execute one path at a time

1. 9. What is a key design reason for massive parallelism on GPUs?

- To allow faster clock rates

- To use more power per core

- To hide memory access latency by context switching across warps

- To simplify thread scheduling

✔ Answer: To hide memory access latency by context switching across warps

1. 10. Which of the following best defines the host in CUDA?

- The GPU execution engine

- The memory allocator for threads

- The CPU system that controls and initiates GPU execution

- The device driver interface

✔ Answer: The CPU system that controls and initiates GPU execution

1. 11. Which part of the CUDA programming model handles the coordination between the host and the device? (Variant)

- The PTX assembler

- The driver API

- The runtime API

- The GPU kernel

✔ Answer: The runtime API

1. 12. Which CUDA memory type has the lowest access latency? (Variant)

- Global memory

- Registers

- Shared memory

- Constant memory

✔ Answer: Registers

1. 13. What is a key design reason for massive parallelism on GPUs? (Variant)

- To hide memory access latency by context switching across warps

- To simplify thread scheduling

- To use more power per core

- To allow faster clock rates

✔ Answer: To hide memory access latency by context switching across warps

1. 14. Which of the following best defines the host in CUDA? (Variant)

- The device driver interface

- The CPU system that controls and initiates GPU execution

- The GPU execution engine

- The memory allocator for threads

✔ Answer: The CPU system that controls and initiates GPU execution

1. 15. Which part of the CUDA programming model handles the coordination between the host and the device? (Variant)

- The PTX assembler

- The GPU kernel

- The runtime API

- The driver API

✔ Answer: The runtime API

1. 16. What is a key design reason for massive parallelism on GPUs? (Variant)

- To simplify thread scheduling

- To hide memory access latency by context switching across warps

- To use more power per core

- To allow faster clock rates

✔ Answer: To hide memory access latency by context switching across warps

1. 17. Which part of the CUDA programming model handles the coordination between the host and the device? (Variant)

- The driver API

- The GPU kernel

- The runtime API

- The PTX assembler

✔ Answer: The runtime API

1. 18. What is a key design reason for massive parallelism on GPUs? (Variant)

- To simplify thread scheduling

- To use more power per core

- To allow faster clock rates

- To hide memory access latency by context switching across warps

✔ Answer: To hide memory access latency by context switching across warps

1. 19. In CUDA, what is the purpose of using shared memory? (Variant)

- To enable fast data exchange between threads in the same block

- To persist data after kernel execution

- To share data between host and device

- To store constant variables only

✔ Answer: To enable fast data exchange between threads in the same block

1. 20. What happens if threads in a warp diverge in control flow? (Variant)

- The warp gets split and executed in parallel

- CUDA throws a runtime error

- Only the first thread executes

- They serialize and execute one path at a time

✔ Answer: They serialize and execute one path at a time

1. 21. Which CUDA memory type has the lowest access latency? (Variant)

- Global memory

- Registers

- Constant memory

- Shared memory

✔ Answer: Registers

1. 22. What CUDA construct defines how many threads execute a kernel in parallel? (Variant)

- Host loop count

- Grid and block dimensions

- Kernel parameters

- Device architecture

✔ Answer: Grid and block dimensions

1. 23. What happens if threads in a warp diverge in control flow? (Variant)

- They serialize and execute one path at a time

- The warp gets split and executed in parallel

- Only the first thread executes

- CUDA throws a runtime error

✔ Answer: They serialize and execute one path at a time

1. 24. What CUDA construct defines how many threads execute a kernel in parallel? (Variant)

- Host loop count

- Grid and block dimensions

- Device architecture

- Kernel parameters

✔ Answer: Grid and block dimensions

1. 25. What CUDA construct defines how many threads execute a kernel in parallel? (Variant)

- Device architecture

- Host loop count

- Grid and block dimensions

- Kernel parameters

✔ Answer: Grid and block dimensions

1. 26. Which of the following best defines the host in CUDA? (Variant)

- The device driver interface

- The CPU system that controls and initiates GPU execution

- The memory allocator for threads

- The GPU execution engine

✔ Answer: The CPU system that controls and initiates GPU execution

1. 27. What happens if threads in a warp diverge in control flow? (Variant)

- The warp gets split and executed in parallel

- CUDA throws a runtime error

- They serialize and execute one path at a time

- Only the first thread executes

✔ Answer: They serialize and execute one path at a time

1. 28. What is a key design reason for massive parallelism on GPUs? (Variant)

- To use more power per core

- To allow faster clock rates

- To hide memory access latency by context switching across warps

- To simplify thread scheduling

✔ Answer: To hide memory access latency by context switching across warps

1. 29. Which of these is true about kernel launches in CUDA? (Variant)

- They are wrapped in if-else conditions

- They return output via return values

- They are called from device to host

- The triple-chevron syntax <<<...>>> defines execution configuration

✔ Answer: The triple-chevron syntax <<<...>>> defines execution configuration

1. 30. Which part of the CUDA programming model handles the coordination between the host and the device? (Variant)

- The driver API

- The runtime API

- The GPU kernel

- The PTX assembler

✔ Answer: The runtime API

1. 31. What happens if threads in a warp diverge in control flow? (Variant)

- The warp gets split and executed in parallel

- They serialize and execute one path at a time

- CUDA throws a runtime error

- Only the first thread executes

✔ Answer: They serialize and execute one path at a time

1. 32. Which of these is true about kernel launches in CUDA? (Variant)

- They return output via return values

- They are wrapped in if-else conditions

- They are called from device to host

- The triple-chevron syntax <<<...>>> defines execution configuration

✔ Answer: The triple-chevron syntax <<<...>>> defines execution configuration

1. 33. Why is global memory access generally slower in CUDA? (Variant)

- Because it's allocated by the OS

- Because it uses paging

- Because it resides in DRAM and is not cached per-thread

- Because it requires thread synchronization

✔ Answer: Because it resides in DRAM and is not cached per-thread

1. 34. Which of the following best defines the host in CUDA? (Variant)

- The CPU system that controls and initiates GPU execution

- The device driver interface

- The GPU execution engine

- The memory allocator for threads

✔ Answer: The CPU system that controls and initiates GPU execution

1. 35. What distinguishes a thread block from a warp in CUDA? (Variant)

- They are equivalent terms

- A thread block is a group of threads scheduled together; a warp is the smallest unit of execution (32 threads)

- A thread block is the smallest executable unit

- A warp contains all threads in the grid

✔ Answer: A thread block is a group of threads scheduled together; a warp is the smallest unit of execution (32 threads)

1. 36. What is a key design reason for massive parallelism on GPUs? (Variant)

- To allow faster clock rates

- To hide memory access latency by context switching across warps

- To use more power per core

- To simplify thread scheduling

✔ Answer: To hide memory access latency by context switching across warps

1. 37. Why is global memory access generally slower in CUDA? (Variant)

- Because it uses paging

- Because it's allocated by the OS

- Because it resides in DRAM and is not cached per-thread

- Because it requires thread synchronization

✔ Answer: Because it resides in DRAM and is not cached per-thread

1. 38. Which of these is true about kernel launches in CUDA? (Variant)

- They are wrapped in if-else conditions

- They return output via return values

- The triple-chevron syntax <<<...>>> defines execution configuration

- They are called from device to host

✔ Answer: The triple-chevron syntax <<<...>>> defines execution configuration

1. 39. What is a key design reason for massive parallelism on GPUs? (Variant)

- To use more power per core

- To simplify thread scheduling

- To hide memory access latency by context switching across warps

- To allow faster clock rates

✔ Answer: To hide memory access latency by context switching across warps

1. 40. Why is global memory access generally slower in CUDA? (Variant)

- Because it resides in DRAM and is not cached per-thread

- Because it's allocated by the OS

- Because it uses paging

- Because it requires thread synchronization

✔ Answer: Because it resides in DRAM and is not cached per-thread

1. 41. What CUDA construct defines how many threads execute a kernel in parallel? (Variant)

- Kernel parameters

- Device architecture

- Grid and block dimensions

- Host loop count

✔ Answer: Grid and block dimensions

1. 42. Which of these is true about kernel launches in CUDA? (Variant)

- They are called from device to host

- They are wrapped in if-else conditions

- The triple-chevron syntax <<<...>>> defines execution configuration

- They return output via return values

✔ Answer: The triple-chevron syntax <<<...>>> defines execution configuration

1. 43. What distinguishes a thread block from a warp in CUDA? (Variant)

- They are equivalent terms

- A thread block is a group of threads scheduled together; a warp is the smallest unit of execution (32 threads)

- A warp contains all threads in the grid

- A thread block is the smallest executable unit

✔ Answer: A thread block is a group of threads scheduled together; a warp is the smallest unit of execution (32 threads)

1. 44. What is a key design reason for massive parallelism on GPUs? (Variant)

- To allow faster clock rates

- To hide memory access latency by context switching across warps

- To simplify thread scheduling

- To use more power per core

✔ Answer: To hide memory access latency by context switching across warps

1. 45. Why is global memory access generally slower in CUDA? (Variant)

- Because it requires thread synchronization

- Because it uses paging

- Because it resides in DRAM and is not cached per-thread

- Because it's allocated by the OS

✔ Answer: Because it resides in DRAM and is not cached per-thread

1. 46. What distinguishes a thread block from a warp in CUDA? (Variant)

- A warp contains all threads in the grid

- They are equivalent terms

- A thread block is the smallest executable unit

- A thread block is a group of threads scheduled together; a warp is the smallest unit of execution (32 threads)

✔ Answer: A thread block is a group of threads scheduled together; a warp is the smallest unit of execution (32 threads)

1. 47. What happens if threads in a warp diverge in control flow? (Variant)

- CUDA throws a runtime error

- Only the first thread executes

- The warp gets split and executed in parallel

- They serialize and execute one path at a time

✔ Answer: They serialize and execute one path at a time

1. 48. What CUDA construct defines how many threads execute a kernel in parallel? (Variant)

- Kernel parameters

- Host loop count

- Device architecture

- Grid and block dimensions

✔ Answer: Grid and block dimensions

1. 49. Which of these is true about kernel launches in CUDA? (Variant)

- The triple-chevron syntax <<<...>>> defines execution configuration

- They return output via return values

- They are wrapped in if-else conditions

- They are called from device to host

✔ Answer: The triple-chevron syntax <<<...>>> defines execution configuration

1. 50. Which part of the CUDA programming model handles the coordination between the host and the device? (Variant)

- The driver API

- The GPU kernel

- The PTX assembler

- The runtime API

✔ Answer: The runtime API