Государственное образовательное учреждение
«Приднестровский государственный университет им. Т.Г. Шевченко»

Физико-технический институт

Кафедра программного обеспечения вычислительной техники

УТВЕРЖДАЮ
Заведующий кафедрой ПОВТ

С.Г. Федорченко
«28» августа 2023 г.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

по дисциплине

ПРОГРАММИРОВАНИЕ СПЕЦИАЛИЗИРОВАННЫХ ВЫЧИСЛИТЕЛЬНЫХ УСТРОЙСТВ

Направление подготовки
09.04.04 Программная инженерия

Профиль подготовки
Разработка программно-информационных систем

Квалификация: **магистр**

Разработал:
к.т.н., доцент кафедры ПОВТ,

С.Г. Федорченко
«28» августа 2023 г.

Тирасполь, 2023

Государственное образовательное учреждение
«Приднестровский государственный университет им. Т.Г. Шевченко»

Физико-технический институт

Кафедра программного обеспечения вычислительной техники

Итоговый тест к экзамену

1. Архитектура GPU:
   1) содержит один арифметико-логический блок;
   2) работает на высокой тактовой частоте;
   3) работает на низкой тактовой частоте;
   4) похожа на структуру центрального процессора.

2. Программная модель CUDA:
   1) система компиляции и исполнения программ, работающих на CPU;
   2) система компиляции и исполнения программ, части которых работают на CPU и GPU;
   3) система компиляции и исполнения программ, работающих на GPU;
   4) система компиляции и исполнения программ, части которых работают на разных операционных системах.

3. Общий прием программирования для CUDA:
   1) группировка множества нитей в блоки;
   2) выделение главной нити;
   3) формирование блоков, реализующих ядро алгоритма вычислений;
   4) нити независимы в рамках одного блока взаимодействуют.

4. Оптимизация работы с глобальной памятью CUDA:
   1) адреса должны быть кратны 64;
   2) выполняется выравнивание адресов (кратные 256);
   3) запросы всех нитей варпа (полуварпа) выполняются независимо друг о друга;
   4) нет правильных ответов.

5. Асинхронные функции CUDA:
   1) хранение информации;
   2) запуск ядра;
   3) дублирование информации;
   4) функции, с произвольными именами.

6. Атомарные операции CUDA:
   1) AtomaticAdd;
   2) AtomaticS;
   3) AtomaticIn;
   4) AtomaticDe.

7. Разделяемая память CUDA:
   1) выделяется на уровне нитей;
   2) выделяется на уровне блоков;
   3) выделяется для каждого процесса;
   4) глобальная память.

8. Константная и текстурная память CUDA:
   1) используется для хранения кода программы;
   2) расположена в SRAM;
   3) обладает независимым кэшем;
   4) не обладает независимым кэшем.

9. Максимальная размерность блока памяти CUDA:
    1) <64К;
    2) 64К;
    3) 512К;
    4) 1024К.

10. Дивергентное ветвление, это:
    1) нити в составе варпа выполняют параллельно одну и ту же команду;
    2) если нити одного варпа должны идти по разным веткам кода, то выполнятся все проходимые ветки кода;
    3) пронумерованная группа нитей называется варпом;
    4) варп содержит 32 нити.

11. Для оптимизации CUDA – программ используют:
    1) оптимизацию доступа к памяти;
    2) таймеры компьютеров;
    3) вмешательство операторов;
    4) нет правильных ответов.

12. Максимальная производительность константной памяти, достигается при:
    1) обращении всеми потоками варпа по двум адресам;
    2) обращении всеми потоками варпа по одному адресу;
    3) увеличении числа потоков варпа;
    4) уменьшении числа потоков варпа.

13. Библиотека CUSPARSE позволяет:
    1) учитывать физическое время;
    2) реализует основные операции линейной алгебры для разреженных векторов и матриц;
    3) реализовать преобразование Фурье;
    4) реализует основные операции линейной алгебры для векторов и матриц.

14. Библиотека CUFFT реализует:
    1) преобразование Фурье;
    2) операции над матрицами;
    3) решение систем линейных уравнений;
    4) решение систем дифференциальных уравнений.

15. Библиотека CURAND содержит:
    1) функции для вычисления стандартных математических функций;
    2) набор констант;
    3) генераторы случайных чисел;
    4) генерацию стандартных сообщений.

16. Библиотека Thrust поддерживает:
    1) нечеткую логику;
    2) обработку сообщений с временными метками;
    3) параллельные алгоритмы обработки данных;
    4) оптимизацию потока сообщений.

17. При использовании нескольких GPU:
    1) используется уровень пользовательского интерфейса;
    2) нет механизма управления распределенным кластером;
    3) не создаются контексты устройств;
    4) работают на уровне протоколов.

18. Микропроцессор Cell:
    1) содержит 2 блок PPE и 16 блоков SPE;

2) содержит 1 блок PPE и 8 блоков SPE;
3) содержит 4 блок PPE и 32 блока SPE;
4) содержит 16 блоков SPE;

19. Микропроцессор Cell:
1) одно ядро имеет доступ к общей памяти;
2) все ядра имеют доступ к общей памяти;
3) каждое ядро SPE имеет свою локальную память;
4) каждое ядро SPE работает только с общей памятью.

20. Программирование Cell:
1) выполнено расширение языка Java;
2) выполнено расширение языка C++;
3) разработан собственный язык программирования;
4) использует удаленный вызов процедур (RPC).