Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Новосибирский государственный технический университет»
Кафедра вычислительной техники

Паспорт экзамена

по дисциплине «Многопоточное и распределенное программирование», 2 семестр

1. Методика оценки

Экзамен проводится в форме письменного тестирования с использованием электронной информационно-образовательной среда НГТУ (http:// www.nstu.ru/sveden/eos).

Тестовые задания охватывают все содержание дисциплины «Многопоточное и распределенное программирование».

Тест состоит из 82 вопросов различного вида и позволяет проверить результаты обучения по дисциплине, соотнесенные с установленными в программе индикаторами достижения компетенций.

2. Критерии оценки результатов обучения, соотнесенных с уровнями освоения индикаторов достижения компетенций

Выполнение теста засчитывается на **продвинутом** уровне, если студент правильно ответил на все вопросы теста, знает определения всех понятий, продемонстрировал способность безошибочно устанавливать последовательность и соответствие между процессами и явлениями, выявлять проблемы, предлагать механизмы их решения, представляет количественные и качественные характеристики определенных процессов. Совокупность результатов обучения по дисциплине и соотнесенных с ними индикаторов достижения компетенций не содержит пробелов. Установленные в программе компетенции сформированы в полном объеме. Оценка составляет *от 35 до 40 баллов.*

Выполнение теста засчитывается на **базовом** уровне, если студент правильно ответил на ⅔ вопросов теста, знает определения основных понятий, продемонстрировал способность устанавливать последовательность и соответствие между процессами и явлениями, правильно характеризует процессы, явления. Совокупность результатов обучения по дисциплине и соотнесенных с ними индикаторов достижения компетенций содержит несущественные пробелы. Установленные в программе компетенции сформированы на базовом уровне. Оценка составляет *от 34 до 27 баллов.*

Выполнение теста засчитывается на **пороговом** уровне, если студент правильно ответил от ½ до ⅔ вопросов теста, знает определения основных понятий, продемонстрировал способность устанавливать последовательность и соответствие между процессами и явлениями. Совокупность результатов обучения по дисциплине и соотнесенных с ними индикаторов достижения компетенций содержит пробелы. Установленные в программе компетенции сформированы на пороговом уровне. Оценка составляет *от 26 до 20 баллов.*

Выполнение теста считается **неудовлетворительным**, если студент правильно ответил менее чем на половину вопросов теста, не знает определений понятий, не продемонстрировал способность устанавливать последовательность и соответствие между процессами и явлениями. Совокупность результатов обучения по дисциплине и соотнесенных с ними индикаторов достижения компетенций содержит существенные

пробелы. Установленные в программе компетенции не сформированы. Оценка составляет *менее 20 баллов.*

3. Шкала оценки

Экзамен считается сданным, если сумма баллов по всем заданиям теста составляет от 20 до 40 баллов включительно. Сумма менее 20 баллов признается неудовлетворительным результатом промежуточной аттестации по дисциплине (модулю).

В общей оценке по дисциплине экзаменационные баллы учитываются в соответствии с правилами балльно-рейтинговой системы, установленными в НГТУ.

4. Примерный тест для экзамена

Утверждаю:
Зав. кафедрой ____ _____ ФИО
«____» _____ 20_ г.

Тест

по дисциплине «Многопоточное и распределенное программирование»[1]

Вопрос 1. В чем различие между массивно-параллельными процессорами и кластерами?

- Ответ 1: массивно-параллельные процессоры – это системы с общей памятью, а кластеры – с распределенной памятью
- **Ответ 2**: массивно-параллельные процессоры собираются из узлов/блоков, разработанных исключительно для них, а кластеры – из узлов/блоков, которые можно приобрести на рынке средств ВТ
- Ответ 3: массивно-параллельные процессоры предназначены для обработки больших массивов данных, а кластеры – для решения произвольных задач

Вопрос 2. Какие специфические проблемы возникают при разработке параллельных программ?

- **Ответ 1:** обеспечение масштабируемости
- **Ответ 2:** балансировка загрузки
- **Ответ 3:** обеспечение корректности межпроцессных взаимодействий
- **Ответ 4:** необходимость предотвращения взаимных блокировок
- **Ответ 5:** необходимость предотвращения гонок данных
- **Ответ 6:** недетерминированное поведение
- Ответ 7: резко увеличивающийся объем программы

Вопрос 3. Чем ограничена предельная производительность последовательного вычислителя?

- **Ответ 1:** физическими константами и законами
- Ответ 2: размером основной памяти
- Ответ 3: количеством транзисторов на процессорном чипе
- Ответ 4: количеством уровней кэша
- Ответ 5: операционной системой

Вопрос 4. Какие архитектуры вычислительных систем входят в классификацию Флинна?

- **Ответ 1:** ОКМД – один поток команд, много потоков данных
- **Ответ 2:** ОКОД – один поток команд, один поток данных
- **Ответ 3:** МКМД – много потоков команд, много потоков данных
- **Ответ 4:** МКОД – много потоков команд, один поток данных

[1] Правильные ответы выделены жирным шрифтом

Вопрос 5. Чем мультипроцессоры отличаются от мультикомпьютеров?

- **Ответ 1:** мультипроцессоры – это системы с общей памятью, а мультикомпьютеры – с распределенной памятью
- Ответ 2: мультипроцессоры – это системы с несколькими равноправными процессорами, а мультикомпьютеры – с одним главным и несколькими подчиненными процессорами
- Ответ 3: мультипроцессоры – это многоядерные системы, а мультикомпьютеры содержат только одноядерные процессоры

Вопрос 6. Параллельное программирование это:

- Ответ 1: одновременная разработка программы несколькими программистами
- **Ответ 2:** разработка программы для последующего исполнения ее на параллельной вычислительной системе
- Ответ 3: параллельная (одновременная) разработка и отладка программы

Вопрос 7. Информационная зависимость итераций цикла возникает, если:

- **Ответ 1:** в некоторой итерации для вычислений используется значение, вырабатываемое в другой итерации
- Ответ 2: в некоторой итерации для вычислений используется значение, вырабатываемое в этой же итерацией
- Ответ 3: в некоторой итерации для вычислений используется значение, вырабатываемое вне тела цикла

Вопрос 8. Какие существуют способы распределения итераций распараллеливаемого цикла?

- **Ответ 1:** статическое блочное
- **Ответ 2:** статическое циклическое
- **Ответ 3:** статическое блочно-циклическое
- **Ответ 4:** динамическое с фиксированным размером блока итераций
- **Ответ 5:** динамическое с переменным уменьшающимся размером блока итераций
- Ответ 6: динамическое с переменным увеличивающимся размером блока итераций
- Ответ 7: статическое с переменным размером блока итераций

Вопрос 9. Ситуация гонок данных возникает, если:

- **Ответ 1:** Два или более потока/процесса в близкие моменты времени считывают одно и то же значение общей переменной, затем каждый из них модифицирует его и каждый записывает полученный им результат в ту же переменную
- Ответ 2: Два или более потока/процесса в близкие моменты времени считывают одно и то же значение общей переменной, затем один из них модифицирует его и записывает полученный им результат в ту же переменную
- Ответ 3: Один поток/процесс постоянно модифицирует значение общей переменной, а другой (другие) считывает и использует значение этой переменной

Вопрос 10. Что такое блокировка (смертельное объятие - deadlock) двух процессов?

- **Ответ 1:** Это ситуация, когда, при наличии двух общих ресурсов (переменных) A и B, каждый процесс успешно захватывает в монопольное владение один ресурс (первый - ресурс A, второй - ресурс B) и пытается захватить второй (первому нужен ресурс B, а второму - ресурс A)
- Ответ 2: Это ситуация, когда, при наличии двух общих ресурсов (переменных) A и B, каждый процесс пытается захватить в монопольное владение один и тот же ресурс, например A; одному из них это удается, а второй переходит в состояние блокировки до освобождения нужного ему ресурса

Вопрос 11. Каким правилам нужно следовать, Каким правилам нужно следовать, чтобы избежать возможности попадания параллельной программы в deadlock

- Ответ 1: Не модифицировать общие ресурсы
- Ответ 2: Использовать для модификации любых общих ресурсов неименованную критическую секцию
- **Ответ 3:** Захватывать в любом потоке разные общие ресурсы строго в одном и том же порядке и освобождать их в обратном порядке
- Ответ 4: Захватывать в любом потоке разные общие ресурсы строго в одном и том же порядке и освобождать их в том же самом порядке

Вопрос 12. Что такое конкурентность?

- **Ответ 1:** Кажущийся параллелизм исполнения длительных последовательностей операций.
- Ответ 2: Способ реализации использования общих ресурсов параллельно выполняемыми вычислительными процессами.
- Ответ 3: Способ обеспечения возможности для одного из нескольких вычислительных процессов завершиться первым.

Вопрос 13. Какие способы организации вычислений позволяют реализовывать конкурентность?

- **Ответ 1:** Многозадачность
- **Ответ 2:** Многопоточность
- Ответ 3: Многоязычность
- Ответ 4: Многоплатформенность
- Ответ 5: Многоядерность

Вопрос 14. Многопоточность используется для:

- **Ответ 1:** Разделения ответственности
- **Ответ 2:** Разделения данных
- Ответ 3: Разделения пользователей
- Ответ 4: Разделения времени
- Ответ 5: Разделения свойств

Вопрос 15. Проблемы, возникающие при разделении данных в многопоточной программе состоят в:

- **Ответ 1:** Возможности возникновения дедлоков
- Ответ 2: Невозможности возникновения дедлоков
- **Ответ 3:** Несинхронизированном поведении: один поток читает данные из памяти, которые еще не занес в нее другой поток
- Ответ 4: Несинхронизированном поведении: потоки в произвольном порядке читают данные из памяти, ни один поток не меняет эти данные
- **Ответ 5:** Гонках данных: потоки в произвольном порядке изменяют значения одной и той же переменной путем выполнения возможно прерываемой последовательности действий "прочитать-модифицировать-записать".
- Ответ 6: Гонках данных: потоки в произвольном порядке изменяют значения одной и той же переменной путем выполнения атомарной последовательности действий "прочитать-модифицировать-записать".
- **Ответ 7:** Нарушения инвариантов: если в двусвязном списке элемент A ссылается на элемент B как на последующий, но элемент B ссылается на элемент C как на предыдущий.
- Ответ 8: Нарушения инвариантов: если в односвязном списке элемент A ссылается на элемент B как на последующий, и элемент B ссылается на элемент C как на последующий
- Ответ 9: Гонках данных: один поток вырабатывает результаты работы значительно быстрее, чем остальные

Вопрос 16. Можно ли заново присоединить к потоку ранее отсоединенный поток?

- Ответ 1: Можно
- **Ответ 2:** Нельзя

Вопрос 17. Что такое мьютекс?

- **Ответ 1:** Это примитив, реализуемый операционной системой и используемый для упорядочивания доступа нескольких потоков к общему ресурсу.
- Ответ 2: Это переменная, создаваемая родительским потоком и используемая для доступа к дочернему потоку.
- Ответ 3: Это переменная, которая может иметь одновременно много разных значений, присваиваемых ей разными потоками и используемая для реализации разделения данных

Вопрос 18. Какими бывают мьютексы?

- **Ответ 1:** Либо простыми, предоставляющими возможность однократного захвата, либо рекурсивными, предоставляющими возможность множественного захвата одним потоком.
- Ответ 2: Либо простыми, предоставляющими возможность однократного захвата, либо

рекурсивными, предоставляющими возможность множественного захвата несколькими потоками.
- Ответ 3: Либо целочисленными, либо булевскими.
- Ответ 4: Либо частными, принадлежащими одному потоку, либо разделяемыми между несколькими потоками.

Вопрос 19. Условные переменные используются для:
- **Ответ 1:** того, чтобы оповещать спящие потоки о возникновении условий для пробуждения.
- Ответ 2: того, чтобы хранить условие пробуждения спящего потока.
- Ответ 3: того, чтобы переключать поток в состояние ожидания по заданному условию.

Вопрос 20. Существуют следующие способы оповещения спящих потоков о возникновении условия для пробуждения:
- **Ответ 1:** Одиночное оповещение
- **Ответ 2:** Широковещательное оповещение
- Ответ 3:Селективное оповещение
- Ответ 4:Оповещение заданного количества N спящих потоков.
- Ответ 5: Многократное оповещение

Вопрос 21. Опишите возможности библиотеки MPI по выполнению сведения данных.
- **Ответ:** Для выполнения сведения данных в библиотеке MPI существует группа функций и, для некоторых функций, специальные типы данных. Простейшим способом получения сводного значения, вычисляемого путем применения некоторой операции (сложение, умножение, …) к данным, вырабатываемым всеми процессами коммуникатора в одном процессе является функция MPI_Reduce. Для таких операций, как нахождение минимального или максимального значения, библиотека предоставляет возможность получить кроме собственно минимума/максимума еще и идентификатор процесса, в котором он найден. Существуют различные усложнения операции сведения, такие как распространение сводного значения по все процессам или ступенчатое сведение, при котором каждый процесс получает сводное значение только для тех процессов, идентификаторы которых не превышают значение его идентификатора.

Вопрос 22. Что такое коммуникаторы в технологии MPI, какие есть виды коммуникаторов?
- **Ответ:** Коммуникатором в технологии MPI называется дескриптор (описатель) коллектива ветвей. Коммуникатор создается и хранится во внутренних структурах данных библиотеки, в MPI-программах он представлен указателем (handle). Коммуникаторы являются обязательными аргументами любых функций библиотеки, имеющих отношение к процессам взаимодействия ветвей параллельной программы. Коммуникаторы бываю двух видов интра- и интер-. Интра-коммуникаторы создаются и используются в любой MPI-программе и определяют подколлективы того коллектива ветвей, который был создан в момент запуска программы на выполнение. Интер-коммуникаторы создаются только тогда, когда MPI-программа запрашивает расширение коллектива ветвей и используются для организации взаимодействий между ветвями исходного и расширенных коллективов.

Вопрос 23. Перечислите и охарактеризуйте коммуникаторы, которые автоматически создаются при запуске параллельной MPI-программы?
- **Ответ:** При запуске на выполнение любой параллельной MPI-программы в каждом процессе автоматически создаются три коммуникатора:1. MPI_COMM_WORLD – коммуникатор, который содержит все процессы, созданные при запуске программы. 2. MPI_COMM_SELF – коммуникатор, содержащий единственный процесс; с использованием этого коммуникатора каждый процесс может передавать данные сам себе. 3. MPI_COMM_NULL – фиктивный коммуникатор, не содержащий ни одного процесса; он может быть использован для немедленного аварийного завершения программы

Вопрос 24. Опишите последовательность действий, которую необходимо выполнить для того, чтобы решить некоторую задачу с использованием технологии CUDA?
- **Ответ:** Эта последовательность должна содержать как минимум следующие действия:1. Получение от CUDA-драйвера сведений о количестве и характеристиках графических процессоров, подключенных к хост-компьютеру. 2. Выбрать наиболее подходящее устройство, если их несколько. 3. Сформировать данные для обработки. 4. Захватить

нужные блоки памяти в глобальной памяти GPU, загрузить в них подготовленные данные. 5. Оформить функцию-ядро для параллельной обработки данных, подготовить значения аргументов 6. Запустить ядро на выполнение. 7. Скопировать результаты выполнения ядра из памяти графического процессора в основную память компьютера, выполнить их окончательную обработку или вывести на внешний носитель.

Вопрос 25. Опишите модель памяти технологии CUDA.

- **Ответ:** Каждое CUDA-устройство содержит:1. Глобальную память большого объема (десятки гигабайт), но невысокого быстродействия, доступную как по чтению, так и по записи и из хост-компьютера, и из любого потока параллельной программы, выполняемой устройством. 2. Память констант небольшого объема (десятки или сотни килобайт) и невысокого быстродействия, доступную по чтению и записи из хост-компьютера и только по чтению из потоков выполняемой программы. 3. Память текстур среднего объема (сотни килобайт) и невысокого быстродействия, доступную по чтению и записи из хост-компьютера и только по чтению из потоков выполняемой программы. 3. В каждом мультипроцессоре (группа вычислителей, работающая по принципу ОКМД) есть разделяемая память небольшого объема (до сотни килобайт), но высокого быстродействия (на 2-3 порядка быстрее, чем глобальная память). Разделяемая память доступна по чтению и записи из любого потока блока потоков, выполняемых в данный момент этим мультипроцессором, и недоступна ни из хост-компьютера, ни из любого другого мультипроцессора. 4. Каждый вычислитель каждого мультипроцессора имеет регистровую память очень небольшого объема (единицы килобайт), но высокого быстродействия. К регистровой памяти имеет доступ и по чтению и по записи только единственный поток, выполняемый в данный момент данным вычислителем. В том случае, если в коде функции-ядра объявлено слишком много переменных, так что объема регистровой памяти не хватает для хранения их всех, потоку выделяется фрагмент глобальной памяти для хранения переменных, не помещающихся в регистровой памяти. Такой фрагмент называется локальной памятью.

Вопрос 26. Как в CUDA-программе определить переменные или массивы, которые должны размещаться в разделяемой памяти блока потоков?

- **Ответ 1:** Объявлять такие переменные или массивы с использованием ключевых слов __device__ __shared__ (если используется слово __shared__, то слово __device__ не обязательно).
- Ответ 2: Не нужно никак специально объявлять такие переменные или массивы, компилятор nvcc оптимизирует размещение данных по уровням памяти графического процессора.
- Ответ 3: Объявлять такие переменные или массивы с использованием ключевых слов __device__ __shared__ (если используется слово __device__, то слово __shared__ не обязательно).

Вопрос 27. Как гнездо из двух Как гнездо из двух (или трех) цикловfor(... ; ... ; ...) for(... ; ... ; ...) { cycle_body } } без информационной зависимости их итераций друг от друга формально можно преобразовать в функцию-ядро CUDA?

- **Ответ:** Для этого нужно:1. Заменить оба (или три) заголовка цикла на заголовок функции-ядра (__global__ void kernelName). 2. Добавить список аргументов этой функции (с указанием их типов), включающий все те переменные и массивы, которые объявлены и инициализируются вне цикла. 3. Вставить вычисление идентификатора потока по двум координатам и присвоение их значений идентификаторам потоков по соответствующим измерениям (например i = blockDim.x * blockIdx.x + threadIdx.x; и j = blockDim.y * blockIdx.y + threadIdx.y; для третьего идентификатора - аналогично, но с использованием поля .z) перед телом цикла.

Вопрос 28. Опишите добавочные виды функций для языков C/C++ в синтаксическом расширении CUDA.

- **Ответ:** К добавочным видам функций синтаксического расширения языков C/C++ в CUDA относятся:1. __global__ void - функция исполняется на графическом процессоре, вызывается из программы центрального процессора 2. __device__ TYPE - функция исполняется на графическом процессоре, вызывается из программы графического процессора 3. __host__ TYPE - функция исполняется на центральном процессоре, вызывается из программы центрального процессора

Вопрос 29. Как в CUDA-программе определить переменные или массивы, которые должны размещаться в регистровой памяти потока?

- **Ответ 1:** Объявить их внутри функции-ядра.
- Ответ 2: Объявить их с использованием ключевого слова __register__
- Ответ 3: Не нужно никак специально объявлять такие переменные или массивы, компилятор nvcc оптимизирует размещение данных по уровням памяти графического процессора.

Вопрос 30. В чем сходство и в чем различие между массивно-параллельными процессорами и высокопроизводительными вычислительными кластерами?

- **Ответ:** Различие состоит в том, что для массивно-параллельных процессоров (МПП) фирма-производитель производит узлы и блоки, разработанных исключительно для них, а кластеры собираются из узлов/блоков, которые можно приобрести на рынке средств вычислительной техники. Сходства состоят в том, что:1. И МПП и кластеры представляют собой совокупность большого количества вычислительных узлов. 2. Узла МПП и кластеров содержать только процессоры (возможно, многоядерные), память с несколькими уровнями кэша, сетевое оборудование и, возможно, вычислительные акселераторы (графические процессоры). 3. Все узлы объединены несколькими сетями передачи данных: основной (высокопроизводительной), сетью синхронизации, сетью управления, сетью тестирования.

Вопрос 31. Перечислите основные специфические проблемы, возникающие при разработке параллельных программ.

- **Ответ:** К таким проблемам относятся:1. Необходимость обеспечения масштабируемости программы (возможности ее запуска на различном количестве параллельно работающих процессоров/ядер). 2. Необходимость обеспечения балансировки загрузки оборудования. 3. Обеспечение корректности межпроцессных взаимодействий. 4. Необходимость предотвращения взаимных блокировок ветвей программы. 5. Необходимость предотвращения гонок данных. 6. Возможное недетерминированное поведение.

Вопрос 32. Какими факторами ограничена предельная производительность одиночного последовательного вычислителя?

- **Ответ:** Предельная производительность одного процессора ограничена рядом физических законов и констант. Главным из этих факторов является максимальное расстояние между логическими элементами в кристалле процессора, ограничивающее сверху частоту переключений этих элементов в силу конечности скорости распространения электромагнитной волны по соединительным проводникам. Следующим фактором является выделяемая при переключении одного элемента мощность, зависящая от скорости переключения и величины разности переключаемых потенциалов. Количество тепла, выделяемого в кристалле при множественных одновременных переключениях, не может превышать возможностей системы охлаждения, зависящих от теплопроводности материала чипа, иначе кристалл выйдет из строя вследствие перегрева.

Вопрос 33. Какова классификация Флинна для средств вычислительной техники?

- **Ответ:** По Флинну все средства вычислительной техники можно отнести к одному из следующих классов:ОКМД – один поток команд, много потоков данных ОКОД – один поток команд, один поток данных МКМД – много потоков команд, много потоков данных МКОД – много потоков команд, один поток данных

Вопрос 34. Опишите сходства и отличия Опишите сходства и отличия мультипроцессоров от мультикомпьютеров.

- **Ответ:** И мультипроцессоры и мультикомпьютеры являются параллельными вычислительными системами класса «Много потоков команд много потоков данных». И те и другие представляют собой системы с равноправными элементами. Основное отличие состоит в том, что мультикомпьютеры – это системы с распределенной памятью, а мультипроцессоры - с общей памятью. В мультипроцессорных системах могут реализовываться как равноправность всех процессоров по доступу ко всем блокам памяти (UMA-архитектуры), так и неравноправность (NUMA-архитектуры)

Вопрос 35. Какие цели преследуются при разработке параллельных программ и какие существуют виды параллелизма?

- **Ответ:** Основной целью разработки параллельных программ является ускорение решения задач за счет одновременного выполнения разных частей вычислений на разных

процессорах. Параллелизм бывает: скрытый, многозадачный и явный. Скрытый параллелизм реализуется в ядрах процессора в виде конвейеризации обработки потока команд, суперскалярности, векторизации и других технологических способов. Многозадачность состоит в разбиении общей задачи на набор подзадач, для решения которых разрабатываются отдельные программы, возможно, выполняющиеся одновременно на разных компьютерах. Явный параллелизм состоит в таком преобразовании последовательного алгоритма решения задачи, при котором одна и та же программа, выполняясь на разных ядрах/процессорах, реализует на каждом из них обработку разных частей данных.

Вопрос 36. Какие циклы могут, а какие циклы не могут распараллеливаться?

- **Ответ:** Могут быть распараллелены только информационно-независимые циклы. Цикл называется информационно-независимым, если никакая его итерация не использует результат, вырабатываемый при выполнении некоторой другой итерации. Поэтому выполнение всех (в пределе) итераций такого цикла может быть организовано одновременно на разных вычислительных устройствах. Если же все или некоторые итерации используют при вычислениях результаты, вырабатываемые при выполнении других итераций, то их выполнение не может быть одновременным.

Вопрос 37. Перечислите возможные способы распределения итераций распараллеливаемого цикла

- **Ответ:** Для того, чтобы цикл можно было распараллелить, он должен быть информационно независимым, т.е. никакая итерация не должна использовать данные, вырабатываемые в другой итерации. Существуют следующие способы распределения итераций:- статическое блочное; - статическое циклическое; - статическое блочно-циклическое; - динамическое с фиксированным размером блока итераций; - динамическое с переменным уменьшающимся размером блока итераций.

Вопрос 38. Что такое блокировка (смертельное объятие - deadlock) двух процессов?

- **Ответ:** Это ситуация, когда выполняется более одного вычислительного процесса (для простоты два: P1 и P2). Каждый них время от времени нуждается в том, чтобы модифицировать два общих ресурса (переменные) A и B. Если первый процесс (P1) успешно захватывает в монопольное владение ресурс A, а второй (P2) - ресурс B, а затем первый пытается захватить ресурс B, а второй - ресурс A, то возникает их взаимная блокировка. Ни один процесс не может продолжить выполнение, поскольку каждый из них ожидает освобождения ресурса, захваченного другим процессом.

Вопрос 39. Что такое гонки данных?

- **Ответ:** Это ситуация, когда два или более потока/процесса, работая с общей памятью, в близкие моменты времени считывают одно и то же значение общей переменной для последующей модификации. Затем каждый из них модифицирует это значение и формирует новое собственное значение. Далее каждый из этих потоков/процессов записывает полученный им результат в ту же переменную. В итоге переменная получает значение, вычисленное последним (по времени) из потоков/процессов, а не суммарный результат модификации всеми участвующими в гонке потоками/процессами. Как правило программа, в которой возникают гонки данных, вырабатывает результаты, отличные от тех, которые выдает нераспараллеленная программа.

Вопрос 40. Каким правилам нужно следовать, Каким правилам нужно следовать, чтобы избежать возможности попадания параллельной программы в deadlock?

- **Ответ:** Если разные потоки/процессы параллельной программы для обеспечения согласованности изменений данных должны захватывать более одного общего ресурса, то в каждом из них необходимо строго соблюдать определенную дисциплину. Общие ресурсы должны быть упорядочены (например – пронумерованы). Захват ресурсов должен осуществляться строго по увеличению (или наоборот по уменьшению) их номеров, например, ресурс2, потом ресурс5, затем ресурс7. Освобождение ресурсов должно производиться строго в обратной последовательности.

Вопрос 41. Опишите модель памяти технологии OpenCL.

- **Ответ:** Каждое OpenCL устройство содержит:1. Глобальную память большого объема (десятки гигабайт), но невысокого быстродействия, доступную как по чтению, так и по записи и из хост-компьютера, и из любого потока параллельной программы, выполняемой устройством. 2. Память констант небольшого объема (десятки или сотни килобайт) и невысокого быстродействия, доступную по чтению и записи из хост-компьютера и только

по чтению из потоков выполняемой программы. 3. В каждом compute_device (группа вычислителей, работающая по принципу ОКМД) есть локальная память небольшого объема (до сотни килобайт), но высокого быстродействия (на 2-3 порядка быстрее, чем глобальная память). Локальная память доступна по чтению и записи из любого потока блока потоков, выполняемых в данный момент этим compute_device'ом, и недоступна ни их хост-компьютера, ни из любого другого compute_device'a. 4. Каждый вычислитель (processing_element) каждого compute_device'a имеет частную память очень небольшого объема (единицы килобайт), но высокого быстродействия. К частной памяти имеет доступ и по чтению и по записи единственный поток, выполняемый в данный момент данным processing_element'ом.

Вопрос 42. Что такое OpenCL-ядро с точки зрения компилятора программы для центрального процессора?

- **Ответ 1:** Строка символов
- Ответ 2: Функция, оформленная специальным образом
- Ответ 3: Структура данных, заполняемая аргументами командной строки
- Ответ 4: Функция, содержащая операторы специального вида

Вопрос 43. Сколько очередей работ может быть создано для одного OpenCL-устройства?

- **Ответ 1:** Несколько, но не менее одной (1, 2, 3, ...)
- Ответ 2: В точности одна
- Ответ 3: Создавать очереди не надо, это делается автоматически
- Ответ 4: Две: входная и выходная

Вопрос 44. Опишите перечень групп директив и их назначение в технология OpenMP?

- **Ответ:** В технологии OpenMP все директивы делятся на три группы:1. Создания параллельных регионов 2. Распределения вычислительных работ между потоками региона 3. Синхронизации потоков Директивы первой группы включают #pragma omp parallel и #pragma omp sections, обеспечивающие создание нескольких (зависит от опций директив или значений переменных окружения) потоков, которые выполняют один и тот же (в случае директивы parallel) или разные (для директивы sections) фрагменты программы. Директивы второй группы обеспечивают распределение вычислительных операций для итеративного (#pragma omp for) или рекурсивного (#pragma omp task) видов параллелизма. Третья группа директив используется для предотвращения гонок данных и дедлоков.

Вопрос 45. Перечислите и охарактеризуйте по быстродействию способы синхронизации доступа потоков OpenMP для модификации общих ресурсов.

- **Ответ:** 1. Опция reduction директивы #pragma omp parallel – наиболее быстродействующий способ, существенным недостатком которого является недоступность сводного значения общего ресурса внутри параллельного региона; до момента его окончания каждому потоку доступно только его собственное частное значение. 2. Директива #pragma omp atomic – следующий по быстродействию способ, основным недостатком которого является невозможность согласованного изменения двух или более общих ресурсов. 3. Критические секции, т.е. директива #pragma omp critical – еще менее быстродействующий способ, обладающий в то же время достаточным функционалом и гибкостью. 4. Замки (locks) и функции захвата/освобождения замков – наименее быстродействующий механизм, обладающий максимальной гибкостью, но его использование для управления доступом к последовательно захватываемым ресурсам потенциально может приводить к дедлокам.

Вопрос 46. Перечислите и охарактеризуйте возможные способы распределения итераций цикла между потоками параллельного региона в OpenMP.

- **Ответ:** Способ распределения итераций цикла задается опцией schedule(type [,chunk]) директивы #pragma omp for. Первым аргументом этой опции является одно из ключевых слов static, dynamic, guided, auto или runtime, второй, необязательный, аргумент определяет размер блока итераций. Ключевое слово static указывает на то, что итерации цикла должны распределяться блочно-циклическим образом с постоянным указанным размером блока. Блоки итераций назначаются потокам для выполнения в момент входа в цикл и не могут быть перераспределены, что может привести к неравномерности загрузки ядер процессора. Словом dynamic задается такой-же в начальный момент способ распределения блоков между потоками, но затем поток, завершивший выполнение некоторого блока итераций, может получить другой блок, предназначавшийся другому потоку. При небольших размерах блоков это позволят сгладить неравномерность загрузки ядер. Тем не менее,

фиксированный размер блока может оставить неравномерность довольно значительной. Способ guided реализует динамическое распределение с постепенно уменьшающимся размером блока, что позволяет надеяться на еще большее сглаживание неравномерности загрузки. Слово auto передает выбор способа распределения итераций компилятору и исполняющей системе OpenMP, а слово runtime означает, что для распределения будет использоваться способ, выбранный администратором системы и зафиксированный им в качестве значения переменной окружения OMP_SCHEDULE.

Вопрос 47. Опишите назначение, сходство и различия опций private, firstprivate и lastprivate директив OpenMP.

- **Ответ:** Все три опции *private предназначены для «преобразования» общих переменных в частные. При их использовании перечисленные в качестве параметров опций общие переменные «размножаются», превращаясь в частные переменные потоков параллельного региона. Если это делается с помощью опций private или lastprivate, то в момент начала исполнения параллельного региона начальные значения частных переменных не определены, а если с помощью опции firstprivate, то во всех потоках в качестве начальных будут установлены значения общих переменных. С помощью опции lastprivate обеспечивается сохранение значений частных переменных из потока, выполнявшего последнюю итерацию распараллеливаемого цикла в общие переменные.

Вопрос 48. Что такое замки (locks) в технологии OpenMP и чем множественный замок отличается от простого?

- **Ответ:** Замками зазываются переменные специального типа, используемые для обеспечения синхронизации потоков при доступе к общим ресурсам. Считается что каждый замок связан с одним общим ресурсом. Простой замок может быть однократно захвачен одним потоком с помощью вызова функции omp_set_lock(&lockar). После вызова этой функции поток считается монопольным владельцем соответствующего ресурса и может модифицировать его произвольным образом. Любой другой поток, нуждающийся в доступе к этому ресурсу и вызывающий для этого функцию omp_set_lock, будет "усыплен" по меньшей мер до тех пор, пока поток, владеющий ресурсом, не вызовет функцию omp_unset_lock(&lockar). Множественный замок может быть многократно захвачен владеющим этим ресурсом потоком, но для полного освобождения ресурса функция unset должна быть вызвана ровно столько же раз.

Вопрос 49. Что произойдет, если два потока одновременно вызовут функцию omp_set_nest_lock(&mLock)? где mLock - проинициализированный множественный замок?

- **Ответ:** Один из потоков захватит этот замок (в данный момент – однократно) и продолжит свое выполнение. Далее, при необходимости, этот поток может еще один или несколько раз захватить этот замок. Для его полно освобождения этот поток должен будет вызвать функцию omp_unset_nest_lock(&mLock) ровно столько раз, сколько раз вызывалась функция omp_set_nest_lock(&mLock). Второй поток будет заблокирован, ему не будет возвращено управление, до тех пор, пока первый не освободит захваченный замок полностью.

Вопрос 50. Опишите модель исполнения программ в технологии OpenMP

- **Ответ:** Эта модель основывается на многопоточном программировании. Выполнение любой программы начинается с одного потока, называемого мастер-потоком. Мастер-поток готовит необходимые значения общих данных в памяти и создает путем вызова функций операционной системы параллельный регион как несколько одновременно работающих потоков. Все потоки выполняют один и тот же фрагмент программы, но для ускорения решения задачи каждый из них обрабатывает свои собственные данные.

Вопрос 51. Что такое итеративный и рекурсивный способы задания параллелизма в OpenMP?

- **Ответ:** Итеративный параллелизм в технологии OpenMP задается с использованием директивы #pragma omp for внутри параллельного региона. Эта директива распределяет итерации следующего за ней по тексту программы цикла между потоками региона. Используя опцию shedule этой директивы, можно управлять способом распределения итераций. Рекурсивный параллелизм задается с использованием директивы #pragma omp task. Этот способ применяется в случаях, когда количество повторений циклического участка программы заранее неизвестно и может определяться значениями обрабатываемых переменных, а также для обработки сложных структур данных, например - для обхода деревьев.

Вопрос 52. Каким ограничениям должен удовлетворять заголовок цикла для того, чтобы к нему можно было применить директиву OpenMP #pragma omp for ?

- **Ответ:** 1. В заголовке должна использоваться единственная целочисленная индексная переменная.2. Начальное значение индексной переменной должно явно инициализироваться константным выражением. 3. Условие продолжения выполнения цикла должно выглядеть как сравнение индексной переменной с константным выражением. 4. Модификация индексной переменной цикла при переходе к следующей итерации должна выполняться с использованием константного выражения

Вопрос 53. Опишите способы создания параллельных регионов в OpenMP

- **Ответ:** Создание параллельных регионов осуществляется двумя способами, выбор между которыми зависит от того, одинаковые или разные фрагменты программы должны выполнять потоки региона. В первом случае, когда потоки, выполняя одну и ту же последовательность инструкций, должны обрабатывать различные данные, нужно использовать директиву #pragma omp parallel. Если же имеется несколько слабо связанных, но времяемких фрагментов программы, то перед каждым из них (при необходимости обрамляемым фигурными скобками) пишется директива #pragma omp section, вся совокупность таких фрагментов тоже заключается в фигурные скобки и перед этим структурным блоком указывается директива #pragma omp sections.

Вопрос 54. Что такое сведение данных в технологии OpenMP?

- **Ответ:** Это средство, позволяющее "размножить" общую переменную, которая должна модифицироваться несколькими потоками, по всем потокам параллельного региона. В зависимости от того, какую операцию модификации (аддитивную или мультипликативную) будут использовать потоки, начальное значение каждой копии устанавливается в 0 или в 1. Далее каждый поток безопасным образом оперирует со своей частной "копией" этой переменной. В момент завершения параллельного региона исполняющая систем OpenMP собирает значения копий из всех потоков и выполняет с общей переменной указанную операцию модификации, "сводя" тем самым наработанные потоками частные суммы или произведения в единое значение.

Вопрос 55. В какой момент (какие моменты) времени выполняется неявная барьерная синхронизация потоков в OpenMP?

- **Ответ 1:** В момент завершения параллельного региона, цикла или совокупности секций, если в создающих их директивах не указана опция nowait
- Ответ 2: В момент завершения параллельного региона, цикла или совокупности секций, если в создающих их директивах указана опция nowait
- Ответ 3: В момент выполнения директивы #pragma omp barrier
- Ответ 4: В момент входа в критическую секцию
- Ответ 5: В момент выхода из критической секции

Вопрос 56. Перечислите средства технологии OpenMP, которые можно использовать для организации корректной модификации общей переменной несколькими потоками

- **Ответ:** К этим средствам относятся:1. Именованные и неименованные критические секции 2. Простые и множественные замки (lock's) и функции захвата/освобождения замков 3. Директива @pragma omp atomic 4. Опция reduction в директивах parallel, for и sections

Вопрос 57. Что такое критические секции в OpenMP и для чего они используются?

- **Ответ:** Критической секцией называется фрагмент программы, охваченный директивой #pragma omp critical. Выполнением операций всего этого фрагмента в любой момент времени может заниматься не более одного потока. Если какой-либо поток вошел в критическую секцию, то все остальные, подошедшие к очке входа в нее, будут "усыплены" исполняющей системой OpenMP до тех пор, пока занимающий секцию поток не выйдет из нее. В этот момент какой-либо из ожидающих потоков (в точности один, если такие потоки есть) будет "разбужен" и войдет в секцию. Критические секции предназначены для предотвращения гонок данных.

Вопрос 58. Опишите типы замков (locks) и способы их захвата.

- **Ответ:** Существует два типа замков: с единичным захватом и с множественным захватом. Каждый замок логически связан с некоторым защищаемым общим ресурсом. Замки объявляются как переменные специальных типов (omp_lock_t и omp_nest_lock_t) и до вызова функции инициализации (omp_init*lock, где * – это «_» для единичного замка и «_nest_» для множественного) не могут быть использованы. После инициализации замок

находится в разблокированном (незахваченном) состоянии, что означает недоступность ресурса для модификации. Поток, нуждающийся в доступе к ресурсу для модификации, может попытаться захватить защищающий его замок либо блокирующим (omp_set*lock), либо неблокирующим (omp_test*lock) способом. Если защищаемый ресурс не захвачен, то при блокирующем способе замок переводится в заблокированное состояние и поток продолжает работу, при неблокирующем выполняется то же самое и функция omp_test*lock возвращает истину. Если же защищаемый ресурс захвачен другим потоком, то при блокирующем способе замок не меняет своего состояния, но поток «засыпает», при неблокирующем функция omp_test*lock возвращает ложь и получивший это значение поток не имеет права модифицировать значение ресурса; он может выполнять другую работу, а позже заново проверить состояние замка.

Вопрос 59. Перечислите и опишите способы установки нужного количества потоков для создания очередного параллельного региона в OpenMP.

- **Ответ:** Требуемое количество потоков при создании параллельного региона в OpenMP можно установить тремя способами. Первый способ заключается в использовании опции num_threads директивы #pragma omp parallel, т.е. непосредственно в момент создания региона. Второй способ предполагает вызов функции omp_set_num_threads до того, как регион будет создаваться. Значение, переданное в эту функцию, будет использоваться в процессе исполнения программы при каждом создании нового региона, но только в том случае, если в директиве его создания не используется опция num_threads. Третий способ заключается в задании администратором системы значения переменной окружения OMP_NUM_THREADS. Это значение будет использоваться в каждой запускаемой параллельной программе, если в ней не вызывалась функция omp_set_num_threads и не используется опция num_threads.

Вопрос 60. Что такое барьерная синхронизация в технологии OpenMP?

- **Ответ:** Барьерной называется такой вид синхронизации, в котором обязательно участвуют все потоки параллельного региона. Для осуществления барьерной синхронизации в нужной точке параллельного региона пишется директива #pragma omp barrier. Каждый из потоков, достигая при своем выполнении этой точки, "засыпает", дожидаясь последнего по времени. В тот момент, когда последний поток выполнит эту директиву, все потоки "просыпаются" и продолжают свою работу.

Вопрос 61. Каково назначение тегов и в каких операциях они используются?

- **Ответ:** Теги - это положительные целые числа, которые могут быть указаны в качестве аргументов некоторых функций, выполняющих индивидуальные (точка-точка) взаимодействия ветвей MPI-программы. С использованием тегов программист может упорядочивать потоки сообщений, движущихся между любой парой взаимодействующих ветвей. Библиотека MPI для индивидуальных взаимодействий реализует протокол, похожий на UDP, поэтому последовательность поступления сообщений в приемную ветвь может не совпадать с той, в которой они отправлялись источником. Если это может повлиять на правильность вычислений, то с помощью тегов можно восстанавливать правильную последовательность сообщений для обработки.

Вопрос 62. Какими параметрами описываются сообщения в индивидуальных MPI-взаимодействиях?

- **Ответ:** Любое взаимодействие определяет либо передачу, либо прием, либо совмещенные прием и передачу некоторых данных. В каждом индивидуальном взаимодействии принимает участие либо две, либо три ветви некоторого коммуникатора. Для всех остальных его ветвей этого взаимодействия просто не существует. Индивидуальное взаимодействие может выполняться либо с блокировкой участвующей ветви до его завершения, либо с продолжением ее выполнения параллельно процессам доставки данных. В коллективных взаимодействиях обязательно участвуют все ветви данного коммуникатора. Все коллективные взаимодействия всегда выполняются с блокировкой каждой ветви до момента окончания ее участия во взаимодействии.

Вопрос 63. Какими параметрами задается (определяется) буфер данных сообщения в MPI?

- **Ответ 1:** Адрес первого байта данных
- **Ответ 2:** Тип данных
- **Ответ 3:** Размер в единицах типа
- Ответ 4: Размер в байтах

- Ответ 5: Тэг сообщения
- Ответ 6: Коммуникатор
- Ответ 7: Статус сообщения

Вопрос 64. Опишите модель выполнения параллельной программы графическим процессором CUDA.

- **Ответ:** Центральный процессор запускает на выполнение параллельную программу в виде сетки блоков потоков. Сетка в качестве совокупности блоков может иметь одно, два или три измерения. Аналогично, блоки потоков данной сетки могут представлять собой одно-, дву- или трехмерную совокупность потоков (для всех блоков количество измерений и размерности в потоках по измерениям одинаковы). Каждый поток представляет собой процесс выполнения функции-ядра. Графический процессор на аппаратном уровне реализует распределение блоков потоков по мультипроцессорам, управление выполнением каждого блока, переключениями мультипроцессора с одного блока потоков на другой. Блоки потоков не могут мигрировать с одного мультипроцессора на другой.

Вопрос 65. Что такое ядро в терминологии CUDA?

- Ответ 1: Это вычислитель без устройства выборки и дешифрации команд, работающий в совокупности с другими ядрами по принципу ОКМД
- **Ответ 2:** Это специальным образом оформленная функция в коде программы
- Ответ 3: Это центральная структура данных, содержащая все параметры для запуска параллельной программы
- Ответ 4: Это аппаратное ядро центрального процессора

Вопрос 66. Как одиночный цикл for(... ; ... ; ...) { cycle_body } без информационной зависимости итераций друг от друга формально можно преобразовать в функцию-ядро CUDA?

- **Ответ:** Для этого нужно:1. Заменить заголовок цикла на заголовок функции-ядра (__global__ void kernelOne) 2. Добавить список аргументов этой функции (с указанием их типов), включающий все те переменные и массивы, которые объявлены и инициализируются вне цикла 3. Вставить перед телом цикла вычисление идентификатора потока и присвоение его значения идентификатору потока (например i = blockDim.x * blockIdx.x + threadIdx.x).

Вопрос 67. Опишите назначение и способ использования функции cudaGetDeviceProperties

- **Ответ:** Вызов этой функции должен предшествовать любым операциям по использованию графического процессора. С ее помощью программист получает объемы глобальной, разделяемой и регистровой памяти (две последние – на один потоковый мультипроцессор) устройства, предельные размеры сетки в блоках и блоков в потоках по каждому измерению и другие важные для исполнения CUDA-программы параметры устройства. Выбранные значения следует использовать для формирования параметров оператора запуска ядра на графическом процессоре.

Вопрос 68. Перечислите и охарактеризуйте встроенные векторные переменные, которые автоматически создаются и инициализируются при запуске любого ядра в технологии CUDA.

- **Ответ:** При запуске ядра на графическом процессоре создаются и инициализируются на основе параметров операции запуска следующие векторные переменные:1. dim3 gridDim – размеры сетки в блоках 2. dim3 blockDim – размеры одного блока в потоках 3. dim3 blockIdx – индекс блока внутри сетки 4. dim3 threadIdx – индекс потока внутри блока

Вопрос 69. Опишите модель выполнения параллельной программы в технологии OpenCL.

- **Ответ:** Хост-компьютер запускает на выполнение параллельную программу в виде сетки блоков потоков. Сетка в качестве совокупности блоков может иметь одно, два или три измерения. Аналогично, блоки потоков данной сетки могут представлять собой одно-, дву- или трехмерную совокупность потоков (для всех блоков количество измерений и размерности в потоках по измерениям одинаковы). Каждый поток представляет собой процесс выполнения функции-ядра. Compute_device на аппаратном уровне реализует распределение блоков потоков по compute_unit'ам, управление выполнением каждого блока, переключениями compute_unit'ов с одного блока потоков на другой. Блоки потоков не могут мигрировать с одного compute_unit'а на другой.

Вопрос 70. Опишите процесс компиляции и исполнения OpenCL-программы

- **Ответ:** OpenCL-программа всегда состоит не менее, чем из двух частей – последовательной для хост-компьютера и параллельной (или нескольких параллельных) для ОКМД-устройства. Параллельная часть представляет сосбой набор строковых данных

последовательной части или набор внешних файлов. Последовательная часть компилируется и собирается до запуска программы на хост-компьютере. Параллельная часть компилируется во время исполнения последовательной части после выбора нужного устройства для ее выполнения. Для этого последовательная часть вызывает/запускает компилятор, имеющийся в SDK для выбранного устройства.

Вопрос 71. Опишите особенности локальной памяти потока в архитектуре CUDA

- **Ответ:** Локальная память потока выделяется компилятором и драйвером как фрагмент глобальной памяти устройства для хранения тех переменных/массивов, которые объявлены в функции-ядре, но для которых при заданных параметрах запуска ядра не хватает регистровой памяти. Обращение потока к любой переменной, размещенной в локальной памяти, выполняется значительно медленнее (примерно в 500 раз дольше), чем обращение к любой регистровой переменной. Поэтому при программировании следует всегда оценивать суммарный размер переменных, объявляемых внутри ядра и делить объем регистровой памяти (обычно он составляет 64 – 96 Кбайт) на этот размер для получения верхней оценки количества потоков в блоке. Следует помнить, что максимальное количество потоков в блоке не должно превышать 1024.

Вопрос 72. В каком диапазоне можно установить размер блоков потоков при запуске ядра в технологии CUDA?

- **Ответ 1:** 1 - 1024
- Ответ 2: 32 - 1024
- Ответ 3: 0 - 1024
- Ответ 4: В любом
- Ответ 5: -32 - 32

Вопрос 73. Выберите правильное отношение вложенности понятий технологии CUDA

- **Ответ 1:** Grid <- Block <- Warp <- Thread
- Ответ 2: Grid <- Warp <- Block <- Thread
- Ответ 3: Grid <- Block <- Thread <- Warp

Вопрос 74. Должны ли в CUDA-программе совпадать количества измерений сетки блоков и блока потоков?

- Ответ 1: Да
- **Ответ 2:** Нет
- Ответ 3: Размерность сетки должна быть всегда больше размерности блока
- Ответ 4: Размерность сетки должна быть всегда меньше размерности блока

Вопрос 75. Какие уровни памяти графического процессора доступны центральному процессору в технологии CUDA?

- **Ответ 1:** Глобальная память
- **Ответ 2:** Память констант
- **Ответ 3:** Память текстур
- Ответ 4: Разделяемая память
- Ответ 5: Регистровая память
- Ответ 6: Локальная память
- Ответ 7: Стековая память
- Ответ 8: Кэш

Вопрос 76. Какой тип расширения базового языка реализован в технологии OpenCL?

- **Ответ 1:** Библиотечный
- Ответ 2: Синтаксический
- Ответ 3: Семантический
- Ответ 4: Крупноблочный
- Ответ 5: Декларативный

Вопрос 77. Перечислите уровни памяти гетерогенного вычислительного устройства в технологии OpenCL и охарактеризуйте их

- **Ответ:** В любом OpenCL-устройстве существуют следующие уровни памяти: 1. Глобальная память (global memory) – память довольно большого объема, сопоставимого с объемом памяти хост-компьютера, но относительно невысокого быстродействия. Эта память доступна как по чтению , так и по записи и хост-компьютеру и любому потоку устройства. 2. Память констант (constant memory) – несколько килобайт с невысоким быстродействием, доступных из хост-компьютера по чтению и по записи и только по

чтению из потоков устройства. 3. Локальная память (local memory) – память небольшого (десятки килобайт) объема, но высокого быстродействия, доступная и по чтению и по записи только потокам одного блока (каждый блок потоков имеет собственную локальную память) 4. Частная память (private memory) – небольшого объема (до килобайта) и высокого быстродействия фрагмент, доступный как по чтению, так и по записи только одному потоку.

Вопрос 78. Чем технология OpenCL похожа на технологию CUDA и чем они различаются?

- **Ответ :** Обе технологии ориентированы на разработку параллельных программ для вычислителей, построенных по архитектуре «Один поток команд много потоков данных». И та, и другая технологии предполагают наличие в составе вычислительной системы одного хост-компьютера (центрального процессора) и одного или нескольких ОКМД устройств. Технология CUDA может использоваться для программирования только графических процессоров компании NVIDIA. Технология OpenCL пригодна для программирования любых гетерогенных вычислительных систем при условии, что на этой системе установлен реализующий эту технологию Software Development Kit.

Вопрос 79. Что такое виртуальные топологии и какие виды топологий предоставляет библиотека MPI?

- **Ответ:** Виртуальной топологией называется логическая (условная) структура связей между узлами вычислительной системы, соответствующая характеру связей данных, обрабатываемых MPI-программой. Эта условная структура, передаваемая из MPI-программы в библиотеку MPI, затем используется для эффективного преобразования идентификаторов ветвей программы в физические адреса узлов вычислительной системы при выполнении взаимодействий. Существует два основных вида виртуальных топологий - декартова решетка заданной размерности и произвольный граф.

Вопрос 80. Чем различаются блокирующие и неблокирующие варианты передач/приемов сообщений в MPI?

- **Ответ:** При вызове блокирующего варианта функции приема/передачи управление возвращается в вызывающую программу только после полного завершения всех связанных с этой операцией действий в данном узле системы. В зависимости от разновидности операции момент такого завершения может быть различным и определяться состоянием партнера по взаимодействию. При вызове неблокирующего варианта управление возвращается в вызывающую программу сразу после того, как станет доступным для модификации указанный в функции буфер.

Вопрос 81. Перечислите и охарактеризуйте коллективные функции сбора данных из нескольких процессов.

- **Ответ:** Сбор данных в один или несколько процессов из всех процессов некоторого коммуникатора выполняется путем вызова одного из вариантов функции MPI_Gather. Самым простым вариантом является сбор порций данных одного типа и размера в один указанный процесс из всех процессоров коммуникатора. Более сложный вариант MPI_Gatherv позволяет собирать в один процесс блоки данных различных размеров из всех процессов и управлять размещением этих блоков в приемном буфере. Варианты MPI_Allgather и MPI_Allgatherv позволяют собирать блоки одинакового или различных размеров во все процессы коммуникатора и формировать одинаковые результаты во всех приемных буферах. Еще более сложный механизм рассылки разных порций данных из каждого процесса и сбора этих порций во все процессы коммуникатора предоставляет функция MPI_Alltoall.

Вопрос 82. Что такое взаимодействия в MPI, какие существуют виды взаимодействий по передаче данных?

- **Ответ:** Под взаимодействием понимается согласованный вызов некоторой функции (или разных функций) двумя или большим количеством процессов. В результате выполнения этих функций библиотека MPI обеспечивает перемещение указанных данных между процессами. Взаимодействия делятся на индивидуальные и коллективные. В индивидуальных принимают участие два процесса: приемники передатчик. Существуют варианты, в которых каждый процесс выполняет одновременно роль и приемника, и передатчика. Индивидуальные взаимодействия могут выполняться в блокирующем (процесс ожидает завершения обмена данными) и неблокирующем вариантах. В коллективных взаимодействиях принимают участие все процессы какого-либо

коммуникатора (коллектива процессов). Эти взаимодействия выполняются всегда в блокирующем режиме.