Архитектура вычислительных систем

Программа учебной дисциплины

А. И. Легалов С. А. Виденин

25 сентября 2022 г.

Оглавление

Введение

Дисциплина «Архитектура вычислительных систем» направлена на изучение особенностей организации компьютеров и методов взаимодействия между их подсистемами. Рассматриваются особенности многоуровневого построения архитектур и взаимное влияние этих уровней, что необходимо учитывать для повышения эффективности разрабатываемого программного обеспечения. Основной акцент сделан на архитектуры уровня системы команд и используемый для программирования на данном уровне язык ассемблера. Показана взаимосвязь данного уровня с операционной системы и вышестоящим уровнем языков системного программирования. Представляются различные подходы к построению архитектур на уровне системы команд. Приводится поддержка параллелизма и рассмотрены ключевые архитектуры параллельных вычислительных систем.

Цели освоения курса

1. Формирование профессиональных компетенций, связанных с использованием теоретических знаний в области архитектур вычислительных систем.

2. Ознакомление с современными архитектурами вычислительных систем с целью заложить основы для последующих курсов, посвящённых созданию современных информационных систем и архитектур программного обеспечения.

3. Получение навыков в области выбора и сочетания архитектур вычислительной системы, наилучшим образом раскрывающего потенциальные возможности решаемой задачи с учётом заданных требований к программному обеспечению.

Планируемые результаты обучения

1. Понимание взаимосвязей между различными уровнями архитектурных решений и особенностей отображения вышестоящих уровней в нижестоящие.

2. Умение выбирать при разработке программного обеспечения архитектурные решения, повышающие эффективность современных компьютеров с учетом их аппаратных возможностей, операционных систем и языков программирования различного уровня организации.

3. Понимание тенденций развития архитектур современных вычислительных систем.

Технические требования

1. Проведение семинаров в компьютерных классах.

2. Компьютеры с архитектурой x86-64 (AMD64).

3. Операционная система Linux. Либо доступная отдельно, либо через виртуальную машину.

4. Поддержка любой графической оболочки и консольного режима.

5. Программное обеспечение:
   - компиляторы: C, ассемблер GAS из gcc, clang (опционально);
   - отладчики: gdb, ddd (опционально), edb (опционально);
   - утилиты, обеспечивающие анализ машинного кода: objdump и др.
   - текстовые редакторы, поставляемые с ОС Linux.

Примечание. *Используется только свободное программное обеспечение*

1 Формула оценки 2022

1.1 Общие параметры

Оценка прохождения дисциплины формируется по результатам участия в семинарах, выполнения заданий, сдачи экзамена.

1.1.1 Участие в семинарских занятиях

Каждое семинарское занятие оценивается следующим образом.

- отсутствие на семинарском занятии: 0 баллов;
- присутствие на занятии и выполнение заданий преподавателя: 7 баллов;
- демонстрация выполнения домашнего задания: +1 балл;
- демонстрация выполнения опционального домашнего задания: + 1 балл;
- решение тестового задания или контрольной задачи: +1 балл;
- пассивное времяпровождение на занятии: -1 балл;
- опоздание на занятие: -2 балла.

Общая оценка за все семинары формируется как среднее арифметическое значение по всем прошедшим занятием, округленное до целого по математическим правилам (математическое округление [11]):

```
seminars_estimation = math_round (
    sum(seminar_estimation[i]) / seminars_number
)
```

где `seminars_estimation` — усредненная оценка за семинары,
`seminar_estimation[i]` — оценка за `i`-е семинарское занятие,
`seminars_number` — количество проведенных семинарских занятий, `sum` — сумма всех оценок за семинарские занятия, `math_round` — математическое округление полученного значения.

1.1.2 Выполнение индивидуальных заданий

В ходе изучения дисциплины предлагается выполнение 4-х индивидуальных заданий:

1. Ассемблерная программа обработки одномерных целочисленных массивов.
2. Ассемблерная программа обработки строк символов.
3. Ассемблерная программа осуществляющая вычисления с плавающей точкой.
4. Параллельная программа на языке программирования C, использующая взаимодействие и синхронизацию потоков.

Каждое из заданий имеет одинаковую весовую оценку. Поэтому общая оценка за все задания формируется как математическое округление [11] среднего арифметического значения оценок по всем заданиям.

```
tasks_estimation = math_round (
    (task_estimation1 +
     task_estimation2 +
     task_estimation3 +
     task_estimation4) / 4
)
```

где `tasks_estimation` — усредненная оценка за индивидуальные задания, `task_estimation1 ... task_estimation4` — оценка за каждое из заданий.

1.1.3 Экзамен

Допуск к экзамену осуществляется при положительной оценке, получаемой с учетом семинарских занятий и индивидуальных заданий, приведенных к единичному коэффициенту:

```
access_estimation = math_round (
  (
      0.15 * seminars_estimation
    + 0.6 * tasks_estimation
  ) / 0.75
)
```

где `access_estimation` — положительная оценка определяющая допуск к сдаче экзамена. В противном случае допуск к экзамену блокируется до пересдачи заданий выполненных на отрицательную оценку. Пересдача заданий осуществляется по завершению сессии до проведения переэкзаменовки в соответствии с регламентом. Пересдаются задания, имеющие отрицательную оценку или те задания, которые имеют минимальную положительную оценку.

Экзамен оценивается по 10 бальной шкале.

1.1.4 Итоговая оценка

Итоговая оценка определяется по следующей формуле:

```
result_estimation = math_round (
      0.15 * seminars_estimation
    + 0.6 * tasks_estimation
    + 0.25 * exam_estimation
)
```

где `result_estimation` - итоговая оценка, `exam_estimation` - оценка за экзамен.

1.1.5 Итоговая оценка автоматом

Получение итоговой оценки автоматом возможно при наличии усредненной положительной оценки (4 балла и выше) за семинарские занятия, а также при наличии положительных оценок за каждое из выполненных заданий. Итоговая оценка при этом рассчитывается с учетом приведения к единице:

```
result_estimation = math_round (
  (
      0.15 * seminars_estimation
    + 0.6 * tasks_estimation
  ) / 0.75
)
```

При несогласии с итоговой оценкой автоматом можно ее изменить путем сдачи экзамена. При этом оценка автоматом аннулируется.

1.2 Оценка заданий, разрабатываемых при изучении ассемблера

Ниже представлены требования которым должно удовлетворять задание по разработке программы на ассемблере для получения соответствующей оценки.

4 балла

- Приведено решение задачи на C.
- В полученную ассемблерную программу, откомпилированную без оптимизирующих и отладочных опций, добавлены комментарии, поясняющие эквивалентное представление переменных в программе на C.
- Из ассемблерной программы убраны лишние макросы за счет использования соответствующих аргументов командной строки и/или за счет ручного редактирования исходного текста ассемблерной программы.
- Модифицированная ассемблерная программа отдельно откомпилирована и скомпонована без использования опций отладки.
- Представлено полное тестовое покрытие, дающее одинаковый результат на обоих программах. Приведены результаты тестовых прогонов для обоих программ, демонстрирующие эквивалентность функционирования.
- Сформировать отчет, описывающий результаты тестовых прогонов и используемых опций компиляции и/или описания проведенных модификаций.

При невыполнении хотя бы одного из требований оценка снижается на балл.

5 баллов

В дополнение к требованиям на предыдущую оценку

- В реализованной программе использовать функции с передачей данных через параметры.
- Использовать локальные переменные.
- В ассемблерную программу при вызове функции добавить комментарии, описывающие передачу фактических параметров и перенос возвращаемого результата.
- В функциях для формальных параметров добавить комментарии, описывающие связь между параметрами языка Си и регистрами (стеком).
- Добавить информацию о проведенных изменениях в отчет.

При невыполнении хотя бы одного из требований оценка снижается на балл.

6 баллов

В дополнение к требованиям на предыдущую оценку

- Рефакторинг программы на ассемблере за счет максимального использования регистров процессора. Добавление этой программы к уже представленным.
- Добавление комментариев в разработанную программу, поясняющих эквивалентное использование регистров вместо переменных исходной программы на C.
- Представление результатов тестовых прогонов для разработанной программы. Оценка корректности ее выполнения на основе сравнения тестовых прогонов результатами тестирования предшествующих программ.
- Добавить новую информацию в отчет.

При невыполнении хотя бы одного из требований оценка снижается на балл.

7 баллов

В дополнение к требованиям на предыдущую оценку

- Реализация программы на ассемблере, полученной после рефакторинга, в виде двух или более единиц компиляции.
- Задание файлов с исходными данными и файла для вывода результатов с использованием аргументов командной строки.
- Добавить в отчет информацию о проделанных изменениях и результаты работы с тестовыми файлами.

При невыполнении хотя бы одного из требований оценка снижается на балл.

8 баллов

В дополнение к требованиям на предыдущую оценку

- Добавление в программу генератора случайных наборов данных, расширяющих возможности тестирования. Подключение генератора к программе с выбором в командной строке варианта ввода данных.
- Расширение анализа командной строки для выбора способа порождения исходных данных. Добавление данных, порождаемых генератором.

- Модификация программы на C и программы на ассемблере, полученной после рефакторинга, для проведения сравнения на производительность. Необходимо добавить замеры во времени, которые не учитывают время ввода и вывода данных. Для увеличения времени работы минимум до 1 секунды, в зависимости от особенностей программы, можно либо выбирать соответствующие размеры исходных данных, либо зацикливать для многократного выполнения ту часть программы, которая выполняет вычисления.

- Представить полученные данные в отчете для разных вариантов тестовых прогонов

При невыполнении хотя бы одного из требований оценка снижается на балл.

9 баллов

В дополнение к требованиям на предыдущую оценку

- Используя опции оптимизации по скорости, сформировать из модифицированной программы на C исходный код ассемблере. Провести сравнительный анализ с предыдущими ассемблерными программами по размеру ассемблерного кода, размеру исполняемого файла и производительности.

- Аналогично, используя опции оптимизации по размеру, сформировать код на ассемблере. Провести сравнительный анализ с предыдущими ассемблерными программами по размеру ассемблерного кода, размеру исполняемого файла и производительности.

- Представить в отчете полученные результаты, дополнив данные представленные в предыдущем отчете.

При невыполнении хотя бы одного из требований оценка снижается на балл.

10 баллов

В дополнение к требованиям на предыдущую оценку

- Использование вместо `libc` библиотеки, опирающейся на системные вызовы операционной системы, разработанной самостоятельно или адаптированной из найденных информационных источников. По сути в данном случае вместо компиляции и рефакторинга программы на C, будет получена программа, написанная «вручную».

- Провести тестовые прогоны данной программы. Оценить производительность.
- Расширить отчет, дополнив его новыми данными.

При невыполнении хотя бы одного из требований оценка снижается на балл.

1.3 Оценка многопоточного приложения

Ниже представлены требования которым должно удовлетворять задание по многопоточному программирования для получения соответствующей оценки.

При отсутствии корректной реализации взаимодействия потоков задача считается нерешенной (0 баллов).

4 балла

- Приведено условие задачи.
- Представлена модель параллельных вычислений используемая при разработке многопоточной программы.
- Описаны входные данные программы, включающие вариативные диапазоны, возможные при многократных запусках.
- Реализовано консольное приложение, решающее поставленную задачу с использованием одного варианта синхропримитивов.
- Ввод данных в приложение реализован с консоли.
- Результаты работы приведены в отчете.

При невыполнении хотя бы одного из требований оценка снижается на балл.

5 баллов

В дополнение к требованиям на предыдущую оценку

- В программу добавлены комментарии, поясняющие выполняемые действия и описание используемых переменных.
- В отчете должен быть приведен сценарий, описывающий одновременное поведение представленных в условии задания сущностей в терминах предметной области. То есть, описано поведение объектов разрабатываемой программы как взаимодействующих субъектов, а не то, как это будет реализовано в программе.

При невыполнении хотя бы одного из требований оценка снижается на балл.

6 баллов

В дополнение к требованиям на предыдущую оценку

- В отчете подробно описан обобщенный алгоритм, используемый при реализации программы исходного словесного сценария. В котором показано, как на программу отображается каждый из субъектов предметной области.
- Реализован ввод данных из командной строки.
- Результаты изменений отражены в отчете.

При невыполнении хотя бы одного из требований оценка снижается на балл.

7 баллов

В дополнение к требованиям на предыдущую оценку

- В программу добавлены ввод данных из файла и вывод результатов в файл.
- Приведены входные и выходные файлы с различными результатами выполнения программы.
- Результаты работы программы должны выводиться на экран и записываться в файл.
- Ввод данных из командной строки расширен с учетом введенных изменений.
- Результаты изменений отражены в отчете.

При невыполнении хотя бы одного из требований оценка снижается на балл.

8 баллов

В дополнение к требованиям на предыдущую оценку

- В программу добавлена генерация случайных данных в допустимых диапазонах.

- Приведены входные и выходные файлы с различными результатами выполнения программы.
- Ввод данных из командной строки расширен с учетом введенных изменений.
- Результаты изменений отражены в отчете.

При невыполнении хотя бы одного из требований оценка снижается на балл.

9 баллов

В дополнение к требованиям на предыдущую оценку

- Проведено исследование поведения программы в случаях, когда отключены различные синхропримитивы, обеспечивающие взаимодействие потоков. Приведены варианты данных, демонстрирующие некорректное поведение.
- Описаны ситуации, которые ведут к конфликтам при различных вариантах отключения.
- Результаты изменений отражены в отчете.

При невыполнении хотя бы одного из требований оценка снижается на балл.

10 баллов

В дополнение к требованиям на предыдущую оценку

- Разработано альтернативное решение, построенное на других синхропримитивах.
- Приведен сравнительный анализ поведения с ранее разработанной программой.
- Результаты изменений отражены в отчете.

При невыполнении хотя бы одного из требований оценка снижается на балл.

2 Содержание лекционного курса

1. **Архитектуры вычислительных систем (ВС). Основные понятия и определения.**
   Разновидности ВС. Разновидности задач, решаемых на ВС. Архитектура ВС, как многоуровневое понятие. Уровни архитектур ВС. Связь между процессом разработки ПО и многоуровневостью архитектур ВС. Критерии качества ПО и их поддержка на различных архитектурных уровнях. Многоуровневость архитектур как средство преодоления семантического разрыва в процессе разработки ПО. Методические и технические приемы преодоления семантического разрыва. Необходимость понимания многоуровневости архитектур ВС и использования ее в процессе разработки ПО.

2. **Поддержка однозначного выполнения операций. Системы типов. Отображение на архитектурные решения.**
   Структура абстрактного алгоритмического вычислителя и ее неоднозначность. Понятие системы типов и однозначности операций. Задание однозначности операций и функций. Бестиповой подход и операционная однозначность. Статическая типизация и статический полиморфизм. Динамическая типизация и динамический полиморфизм. Использование разных видов типизации и однозначности. Архитектуры ВС с бестиповой организацией. Архитектуры ВС, использующие статическую типизацию. Архитектуры ВС с динамической типизацией.

3. **Представление данных в вычислительных системах.**
   Типы данных, представляемые в архитектурах ВС. Позиционная система счисления. Представление целочисленных данных. Представление данных с плавающей точкой. Представление символьных данных. Непозиционные системы счисления.

4. **Архитектура уровня системы (набора) команд. Классификация архитектур.**
   Особенности уровня как физического, а не виртуального компьютера. Обобщенная структура ВС. Система типов уровня и организация однозначности операций. Основные блоки ВС, их функциональное назна-

чение. Организация системы команд. Классификация системы команд. Влияние состава системы команд на тип архитектуры ВС. Организация памяти на уровне системы команд.

5. **Особенности CISC архитектур на примере процессоров компании Intel . Архитектура процессоров семейства x86.**
   История становления и развития архитектур семейства x86. Ключевые процессоры семейства. Повышение разрядности в ходе эволюции. Сохранение преемственности. Особенности 8-разрядного процессора Intel 8080. 16-разрядный процессор Intel 8086. Процессоры с 32-разрядной архитектурой. 64-разрядные процессоры.

6. **Особенности RISC архитектур. Архитектуры и системы команд процессоров ARM и RISC-V.**
   Причины появления конкурирующих альтернативных архитектур. Специфические решения, повышающие эффективность альтернативных архитектур. Архитектура процессоров ARM. Особенности ассемблера процессоров ARM. Архитектура процессоров RISC-V. Особенности ассемблера процессоров RISC-V. Другие направления развития. VLIW архитектуры.

7. **Уровень микроархитектуры. Структурные решения, обеспечивающие повышение эффективности на уровне системы команд.**
   Микроархитектура как средство повышенияе эффективности архитектуры уровня системы команд за счет аппаратных решений. Реализации цикла выполнения команд. Микрокод. Конвейеризация команд. Кэширование. Прогнозирование ветвлений. Суперскалярность. Внеочередное исполнение. Переименование регистров. Многопроцессорность и многопоточность.

8. **Архитектура уровня операционной системы. Основные функции, обеспечивающие поддержку уровня системы команд. Стандарт POSIX.**
   Назначение данного уровня. Основные функции, выполняемые на уровне ОС. Использование уровня ОС на других уровнях. Уровень ОС как предметно–ориентированный язык.

9. **Архитектуры параллельных вычислительных систем. Основные структурные решения и концепции.**
   Области применения. Влияние на все архитектурные уровни. Основные понятия и определения. Классификация Флинна. Примеры структурных решений. SMP и MPP архитектуры, их достоинства и недостатки. Причина повышенной масштабируемости MPP. Разница в средствах программирования для SMP и MPP. Суперкомпьютеры.

10. **Многопоточные архитектуры и многопоточное программирование.**
Основные понятия и определения. POSIX Threads — стандарт POSIX-реализации потоков. Примеры построения программ для многопоточных архитектур. Дуализм понятия параллелизма: parallelism vs. concurrency. Модели многопоточных приложений. Механизмы синхронизации. Выполнение многопоточных программ в многоядерных архитектурах. Инструментальные и языковые средства, поддерживающие многопоточность. Дополнительная инструментальная поддержка. OpenMP.

11. **Многопроцессные архитектуры и распределенные вычисления.**
Отличие распределенных ВС от многопоточных. Особенности поддержки распределенных вычислений в ОС Linux. Процессы. Дополнительная инструментальная поддержка. Message Passing Interface (MPI). Организация обмена сообщениями между процессами. Основные функции MPI. Синхронизация в MPI. Использование симбиоза распределенных и многопоточных ВС.

12. **Специализированные параллельные архитектуры**
Графические ускорители. Отличие в архитектуре от многопоточных ВС. CUDA (Compute Unified Device Architecture). Организация потоков в CUDA. Программирования GPU. Гибридные системы на основе CPU и GPU. Вычислители с программируемой архитектурой. Принцип реконфигурации вычислительной системы. Изменчивость структуры вычислительной системы против статической системы. Реконфигурируемые вычислительные системы, программируемые логические интегральные схемы (ПЛИС).

13. **Альтернативные пути развития архитектур параллельных вычислительных систем (обзор).**
Нетрадиционные архитектуры ВС. Процессоры потока данных (dataflow архитектуры). Функционально-потоковое параллельное программирование и событийные машины. Нейронные сети и нейрокомпьютеры. Аналоговые компьютеры. Квантовые компьютеры.

3 Содержание семинарских занятий

1. Применение ОС Linux для выполнения заданий. Установка ОС. Использование виртуальной машины. Основные инструментальные средства. Компиляция и выполнение программ в ОС Linux. Режимы компиляции. Отображение ассемблерных программ в нотациях Intel и AT&T.

2. Использование отладочных средств и простейших средств разработки применительно к программе на языке программирования С. Отладчики GDB, DDD, EDB.

3. Инструментальная поддержка непосредственной разработки программ на ассемблере. Отладчики GDB, DDD, EDB. Отображение объектных файлов с использованием OBJDUMP. IDE SASM. Основные ассемблерные команды, обеспечивающие поддержку целочисленной арифметики. Линейные программы. Использование библиотеки libc и реализация ассемблерных программ с использованием только системных вызовов.

4. Регистр флагов. Организация ветвлений и циклов. Отображение в архитектуре уровня системы команд высокоуровневых операторов языка программирования С. Условные операторы. Операторы цикла. Операторы переключения.

5. Использование подпрограмм. Рекурсия. Передача параметров. Использование локальных данных функции.

6. Представление составных и данных. Массивы. Структуры. Объединения.

7. Обработка массивных данных с использованием массовых операций. Работа со строками символов.

8. Арифметика с плавающей точкой. Особенности представления чисел с плавающей точкой и работа с ними.

9. Основы многопоточного программирования. Простейшие многопоточные программы. Основные функции библиотеки POSIX Treads.

10. Использование многопоточности для конкурентного программирования. Синхропримитивы.

11. Использование многопоточности для параллельного программирования. Обработка больших данных. Сравнение времени выполнения последовательной и многопоточной программ.

12. Распределенные параллельные вычисления. Поддержка на уровне операционной системы Linux.

13. Прикладные системы параллельного программирования

4 Список вопросов по содержанию дисциплины

Архитектура ВС. Основные понятия
Архитектуры параллельных ВС (структурные аспекты)

1. Определения архитектуры вычислительной системы. Многообразие определений.
2. Многоуровневость архитектур ВС. Основные уровни.
3. Особенности различных уровней архитектур ВС. Цифровой логический уровень.
4. Особенности различных уровней архитектур ВС. Уровень микроархитектуры.
5. Особенности различных уровней архитектур ВС. Уровень набора команд.
6. Особенности различных уровней архитектур ВС. Уровень операционной системы.
7. Особенности различных уровней архитектур ВС. Уровень ассемблера.
8. Особенности различных уровней архитектур ВС. Уровень промежуточных языков (промежуточного представления).
9. Особенности различных уровней архитектур ВС. Уровень языков системного программирования.
10. Особенности различных уровней архитектур ВС. Уровень языков прикладного программирования.
11. Цели и задачи процесса разработки ПО. Связь с архитектурами ВС.
12. Модели задачи, исполнителя, промежуточные модели, семантический разрыв.

13. Трудоемкость процесса разработки ПО. Основные проявления, связанные с трудоемкостью процесса разработки ПО.

14. Основные методические приемы, обеспечивающие переход к целевой архитектуре ВС.

15. Особенности формализации предметной области. Отличие процесса разработки с использование формализации. Общий механизм перехода.

16. Предметно-ориентированные и специализированные архитектуры. Особенности преодоления в них семантического разрыва. Достоинства и недостатки предметно-ориентированных архитектур.

17. За счет чего повышается эффективность разработки ПО при формализации предметной области.

18. Методики разработки ПО и их влияние на целевую архитектуру ВС. В чем проявляется эффективность использования методик разработки ПО?

19. Технические приемы и их влияние на разработку ПО для разных целевых архитектур. Разновидности технических приемов.

20. Параллельные ВС (ПВС). Определения. Цели и задачи создания.

21. Классификация Флинна.

22. ПВС. Разновидности структурных решений. SMP. Примеры современных решений.

23. ПВС. Разновидности структурных решений. MPP. Примеры современных решений

24. Кластеры. Особенности и варианты реализации.

25. Суперкомпьютеры. Определение. Особенности построения. Оценка суперкомпьютеров.

26. Понятие однозначности. Методы задания однозначности. Связь с типизацией.

27. Операционная однозначность. Основные особенности. Отражение операционной однозначности на архитектуру ВС и ее программирование.

28. Статическая однозначность. Основные особенности. Отражение статической однозначности на архитектуру ВС и ее программирование.

29. Динамическая однозначность. Основные особенности. Отражение динамической однозначности на архитектуру ВС и ее программирование.

30. Архитектура уровня статически типизированного процедурного языка. Особенности использования и реализации. Отображение на архитектуру машинного уровня.

31. Архитектура уровня динамически типизированного языка. Особенности использования и реализации. Отображение на архитектуру машинного уровня.

32. Архитектура уровня набора команд. Основные составляющие архитектуры данного уровня.

33. Структура компьютера, основные функциональные узлы и их объединение.

34. Особенности организации центрального процессора.

35. Понятие системы команд. Формат команды.

36. Классификация системы команд по функциональному назначению. Примеры команд различного назначения.

37. Классификация системы команд по способу передачи управления и количеству адресов. Отношения между командами. Примеры.

38. Классификация системы команд по длине команды и способу кодирования операций. Примеры.

39. Классификация системы команд по методам адресации. Примеры.

40. Типы архитектур процессора. CISC, RISK, VLIW архитектуры. Их особенности и отличия.

41. Организациия памяти. Особенности регистровой памяти. Классификация регистровой памяти.

42. Организациия памяти. Особенности общей памяти. Классификация общей памяти.

43. Основные характеристики фон Неймановской архитектуры. Достоинства и недостатки.

44. Основные характеристики фон Гарвардской архитектуры. Достоинства и недостатки.

45. Выравнивание данных в памяти. Цели, достоинства, недостатки.
46. Порядок размещения данных в общей памяти. Примеры размещения в различных процессорах.
47. Архитектура процессора Intel x86. Регистры. 16-разрядная версия.
48. Архитектура процессора Intel x86. Особенности формата команд. 16-разрядная версия.
49. Архитектура процессора Intel x86. Регистры. 32-разрядная версия.
50. Архитектура процессора Intel x86. Особенности формата команд. 32-разрядная версия.
51. Архитектура процессора Intel x86. Регистры. 64-разрядная версия.
52. Архитектура процессора Intel x86. Особенности формата команд. 64-разрядная версия.
53. Многопоточность. Основные определения.
54. Отличие потоков от процессов.
55. Библиотека pthread. Функции создания и завершения потоков.
56. Два основных понимания параллелизма (parallelism, concurrency). Отличие в восприятии.
57. Модели многопоточных приложений.
58. Понятие критической секции.
59. Синхронизация потоков. Использование мьютексов. Ключевые функции.
60. Синхронизация потоков. Использование семафоров. Ключевые функции.
61. Синхронизация потоков. Использование условных переменных. Ключевые функции.
62. Синхронизация потоков. Блокировки чтения-записи. Ключевые функции.
63. Синхронизация потоков. Барьеры. Ключевые функции.
64. Многопоточное программирование с применением OpenMP. Основнная специфика. Назначение.

65. Основные прагмы OpenMP. Примеры.
66. Процессы. Порождение процессов.
67. Организация взаимодействия процессов.
68. Интерфейс передачи сообщений. Назначение. Определения. Сравнение с многопоточными системами.
69. MPI. Основые функции по работе с процессами. Общие функции.
70. MPI. Основые функции по работе с процессами. Передача данных между процессами.
71. GPU. Основные определения. Изначальное назначение. Использование для параллельных вычислений.
72. GPU. Особенности функциональнного распределения вычислительных ресурсов на кристалле. Отличие от CPU.
73. Технология CUDA. Организация потоков.
74. Общие принципы программирования гибридных систем на основе CPU+GPU.
75. Нетрадиционные архитектуры ВС. Процессоры потока данных (dataflow архитектуры).
76. Нетрадиционные архитектуры ВС. Функционально-потоковое параллельное программирование. Событийные машины.
77. Нетрадиционные архитектуры ВС. Нейронные сети и нейрокомпьютеры.
78. Нетрадиционные архитектуры ВС. Аналоговые компьютеры.
79. Нетрадиционные архитектуры ВС. Квантовые компьютеры.

Литература

[1] Таненбаум Э. Архитектура компьютера. 6-е изд. — СПб.: Изд. Питер, 2017. — 816 с.

[2] Гагарина Л.Г., Кононова А.И. Архитектура вычислительных систем и Ассемблер с приложением методических указаний к лабораторным работам. Учебное пособие. — М.: СОЛОН-Пресс, 2019. — 368 с.

[3] Харрис Сара Л., Харрис Дэвид. Цифровая схемотехника и архитектура компьютера: RISC-V. — М.: ДМК Пресс, 2021. — 810 с.

[4] Харрис Дэвид, Харрис Сара Л. Цифровая схемотехника и архитектура компьютера. Дополнение по архитектуре ARM. — М.: ДМК Пресс, 2019. — 356 с.

[5] Рэндал Э. Брайант, Дэвид Р. О'Халларон. Компьютерные системы: архитектура и программирование. 3-е изд. — М.: ДМК Пресс, 2022. — 994 с.

[6] Прохоренок Н.А. Язык С. Самое необходимое. — СПб.: БХВ-Петербург, 2020. — 480 с.

[7] Йо Ван Гуй. Программирование на ассемблере x64: от начального уровня до профессионального использования AVX. — М.: ДМК Пресс, 2021. — 332 с.

[8] Смит Б. Ассемблер для Raspberry Pi. Практическое руководство. 4-е изд. — СПб.: БХВ-Петербург, 2022. — 320 с.

[9] Plantz Robert G. Introduction to Computer Organization. — 2022

[10] Suzanne J. Matthews, Tia Newhall, Kevin C. Webb. Dive into Systems. — 2022.

[11] Округление. Статья в Википедии. — `https://ru.wikipedia.org/wiki/%D0%9E%D0%BA%D1%80%D1%83%D0%B3%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5`

[12] Формат файла ELF64. — `https://uclibc.org/docs/elf-64-gen.pdf`

[13] Ричард Столмен, Роланд Пеш, Стан Шебс и др. Отладка с помощью GDB. — 2000. `https://www.opennet.ru/docs/RUS/gdb/`

[14] Андреас Целлер Почему не работают программы. — М.: Эксмо, 2011. — 560 с.