Федеральное государственное автономное образовательное учреждение высшего образования
"Национальный исследовательский университет "Высшая школа экономики"

Факультет компьютерных наук
Департамент программной инженерии

Рабочая программа дисциплины
Архитектура вычислительных систем

для образовательной программы «Программная инженерия»
направления подготовки 09.03.04 «Программная инженерия»
уровень - бакалавр

Разработчик программы:
Баканов В.М., д.т.н., профессор, vbakanov@hse.ru

Одобрена на заседании департамента программной инженерии «___»_____ 2018 г.
Руководитель департамента Авдошин С.М. _____

Утверждена Академическим советом образовательной программы
«___»_____ 2018 г., № протокола_____

Академический руководитель образовательной программы
Шилов В.В. _____

Москва, 2018

1 Область применения и нормативные ссылки

Настоящая программа учебной дисциплины "Архитектура вычислительных систем" устанавливает минимальные требования к знаниям и умениям студента и определяет содержание и виды учебных занятий и отчётности.

Программа предназначена для преподавателей, ведущих данную дисциплину, учебных ассистентов и студентов образовательной программы «Программная инженерия» направления подготовки 09.03.04 "Программная инженерия", изучающих дисциплину "Архитектура вычислительных систем".

Программа разработана в соответствии с:

- образовательным стандартом Федерального государственного автономного образовательного учреждения высшего образования "Национальный исследовательский университет "Высшая школа экономики" по направлению 09.03.04 "Программная инженерия";
- образовательной программой «Программная инженерия» направления подготовки 09.03.04 "Программная инженерия";
- рабочим учебным планом Университета по направлению 09.03.04 "Программная инженерия", утверждённым в 2018 г.

2 Цели освоения дисциплины

Цели освоения дисциплины "Архитектуры вычислительных систем":

- формирование у студентов профессиональных компетенций, связанных с использованием теоретических знаний в области архитектур вычислительных систем;
- ознакомление студентов с прогрессивными парадигмами развития архитектур вычислителей с целью заложить основы для последующих курсов, посвящённых созданию современных информационных систем
- получение практических навыков в области выбора архитектуры вычислительной системы, наилучшим образом раскрывающего потенциальные возможности заданного алгоритма с учётом заданных требований к программному обеспечению;
- развитие умений, основанных на полученных теоретических знаниях, позволяющих на творческом и репродуктивном уровне применять и создавать эффективные алгоритмы для решения задач обработки информации применительно к данной архитектуре вычислительной системы;
- получение студентами навыков самостоятельной исследовательской работы, предполагающей изучение специфических методов анализа архитектур вычислительных систем и функционирующего на них программного обеспечения, инструментов и средств, необходимых для решения актуальной, в аспекте программной инженерии, задачи выбора рациональных алгоритмов в зависимости от особенностей применения разрабатываемых программ.

3 Компетенции обучающегося, формируемые в результате освоения дисциплины

В результате освоения дисциплины студент должен:

- Знать:
  — методы представления числовой информации в вычислительных системах;

— методы обработки числовой информации в вычислительных системах;

— методы обмена информацией между компонентами вычислительных систем.

- Уметь:
  — оценивать компьютерные архитектуры вычислительных систем в точки зрения комплексных критериев качества;

  — планировать эксперимент, проводить экспериментальное исследование с помощью натурных или имитационных моделей вычислительных систем.
- Иметь навыки (приобрести опыт) и владеть:
  — методами анализа потоков данных в вычислителях различной архитектуры;

  — инструментами замера времени в программных реализациях алгоритмов;

  — основами технологий разработки программ для заданных архитектур вычислителей.

В результате изучения дисциплины студент осваивает следующие (в соответствие с образовательным стандартом НИУ ВШЭ программ бакалавриата по направлению подготовки 09.03.04 "Программная инженерия") компетенции:

Компетенция	***Код по ОС НИУ ВШЭ***
а) универсальные компененции (УК):	
Способен учиться, приобретать новые знания, умения, в том числе в области, отличной от профессиональной	УК-1
Способен выявлять научную сущность проблем в профессиональной области	УК-2
Способен работать с информацией: находить, оценивать и использовать информацию из различных источников, необходимую для решения научных и профессиональных задач (в том числе на основе системного подхода)	УК-5
Способен работать в команде	УК-7
Способен грамотно строить коммуникацию, исходя из целей и ситуации общения	УК-8
Способен критически оценивать и переосмыслять накопленный опыт (собственный и чужой), рефлексировать профессиональную и социальную деятельность	УК-9
б) профессиональные компетенции (ПК):	
Способен применять основные концепции, принципы, теории и факты, связанные с информатикой при решении научно-исследовательских задач	ПК-1
Способен к формализации в своей предметной области с учетом ограничений используемых методов исследования	ПК-2
Способен готовить презентации, оформлять научно-технические отчеты по результатам выполненной работы, публиковать результаты	ПК-5
Способен читать, понимать и выделять главную идею прочитанного исходного кода, документации	ПК-11
Способен оценивать временную и емкостную сложность программного обеспечения	ПК-13
Способен использовать различные технологии разработки программного обеспечения	ПК-16

Способен придерживаться правовых и этических норм в профессиональной деятельности	ПК-28
Способен к социальному взаимодействию, к сотрудничеству и разрешению конфликтов	ПК-31
Способен ориентироваться в системе общечеловеческих ценностей и ценностей мировой и российской культуры, понимает значение гуманистических ценностей для сохранения и развития современной цивилизации	ПК-36

4 Место дисциплины в структуре образовательной программы

Настоящая дисциплина относится к базовой части профессионального цикла дисциплин.

В соответствии с рабочим учебным планом по направлению "Программная инженерия" дисциплина "Архитектура вычислительных систем" читается студентам второго курса бакалавриата в 1-ом и 2-ом модулях.

Изучение данной дисциплины базируется на знаниях, полученных студентами при освоении учебных дисциплин "Дискретная математика", "Программирование", общих знаниях математики и основ программирования в части базовых алгоритмических конструкций.

Дисциплина является основой для последующего изучения дисциплин: "Проектирование и архитектура программных систем", научно-исследовательского семинара "Современные архитектуры вычислителей".

5 Тематический план учебной дисциплины

№	*Название темы*	*Всего часов по дисциплине*	*Аудиторные часы*		**Самостоятельная работа**
			Лекции	*Практические занятия*	
1	История машинного счёта	8	2	2	6
2	Определения понятия "Архитектура" в применении к вычислительным системам	8	2	2	4
3	Уровни управления процессом вычислений. Архитектура процессоров x86 фирмы Intel	10	2	2	6
4	Общие требования к программному коду. Пото́ковые (DATA-FLOW) вычислители. Машинные команды, язык программирования Ассемблер	10	2	2	8
5	Недостаток процесса вычислений в позиционной системе счисления и альтернативные решения. Простейшие программы на Ассемблере -	10	2	2	6

	простые типы и описания данных				
6	Архитектура параллельных вычислительных систем. Ассемблер - целочисленная арифметика	10	2	2	8
7	Суперкомпьютеры. Ассемблер - программирование условных и безусловных переходов	10	2	2	8
8	Нейронные сети и нейрокомпьютеры	8	2	2	6
9	Ассемблер - практика разработки программ. Интерактивные среды разработки. Макроассемблер. Процессор x86 и программирование под DOC. Ассемблер для 32-х и 64-бит архитектур. Ассемблер для процессоров архитектуры ARM. Поддержка операций над числами с с плавающей запятой	12	2	2	8
10	Транспьютеры. Отладка программ на языке Ассемблер	8	2	2	6
11	Метакомпьютинг и концепция GRID. Использование ассемблерных программ при программировании на языках высокого уровня	10	2	2	6
12	Архитектура GPU фирмы NVIDIA и технология CUDA	12	2	2	4
13	Аналоговые вычислительные системы	8	2	2	4
14	Вычислители с программируемой архитектурой, пути усовершенствования архитектуры	10	2	2	4
15	Квантовые вычислители и системы передачи данных	8	1	1	4
16	Архитектура систем поиска информации в сети InterNet	10	1	1	4
	Итого:	**152**	**30**	**30**	**92**

6 Формы контроля знаний студентов

Тип контроля	*Форма контроля*	*Модули*		*Параметры*
		1	*2*	
Текущий (неделя)	Домашнее задание		5÷7-я недели	Разработка программы на языке Ассемблер (по индивидуальному заданию)
	Экзамен		■	Устный экзамен 120 мин.
Итоговый				

6.1 Критерии оценки знаний, навыков

Оценки по всем формам контроля выставляются по 10-ти балльной шкале.

Текущий контроль предусматривает домашнее задание, выполняемое во втором модуле.

Домашнее задание включает разработку, кодирование, тестирование и отладку программ реализации задачи на языке Ассемблера (по индивидуальному заданию), исследование и сравнительный анализ алгоритмов ее решения. По домашнему заданию оформляется отчет в электронном виде.

Домашнее задание размещается в LMS в разделе "Проекты". В установленный срок студент загружает в LMS архив, содержащий полностью оформленный отчёт и программу решения контрольного домашнего задания. Оценка за домашнее задание выставляется с учётом полноты выполнения задания и оформления результатов.

При определении итоговой оценки учитываются:

- работа студентов на практических занятиях, а именно: подготовка сообщений по одной из заданных тем, решение тестов по некоторым темам;
- выполнение домашнего задания.

В случае несвоевременной сдачи домашнего задания оценка снижается на один балл за каждый день задержки. При задержке по уважительной причине баллы не снимаются.

Итоговый контроль: экзамен в конце 2-го модуля (2 час). Экзамен проводится в устной форме.

6.2 Порядок формирования оценок по дисциплине

По всем видам работ выставляется 10-балльная оценка. Оценивается работа студентов на практических занятиях $О_{аудиторная}$, в которую входят:

- результаты опросов по текущей теме в устной форме.
- полнота освещения темы доклада. Доклад готовит каждый студент по выбранной теме.

Оценки за работу на практических занятиях выставляются в рабочую ведомость.

Оценивается самостоятельная работа студентов $О_{сам.работа}$: правильность и полнота выполнения домашних работ по темам практических занятий. Оценки за самостоятельную работу студента преподаватель выставляет в рабочую ведомость.

Накопленная оценка $О_{накопл.}$ за текущий контроль учитывает результаты студента по текущему контролю следующим образом:

$$О_{накопл.} = 0{,}4 \times О_{дом.задание} + 0{,}3 \times О_{ауд} + 0{,}3 \times О_{сам.работа},$$

Способ округления — арифметический.
Результирующая оценка во втором модуле определяется соотношением:

$$О_{результ.} = 0{,}5 \times О_{накопл.} + 0{,}5 \times О_{экз}.$$

Перевод в пятибалльную оценку осуществляется в соответствии со следующей таблицей.

Таблица соответствия оценок по десятибалльной и пятибалльной системам

По десятибалльной шкале	*По пятибалльной шкале*
1 – неудовлетворительно 2 – очень плохо 3 – плохо	неудовлетворительно – 2
4 – удовлетворительно 5 – весьма удовлетворительно	удовлетворительно – 3
6 – хорошо 7 – очень хорошо	хорошо – 4
8 – почти отлично 9 – отлично 10 – блестяще	отлично – 5

7 Содержание дисциплины

7.1 Содержание лекций

Тема 1. История машинного счёта

Предпосылки необходимости счёта. Мысли великих людей по поводу счёта. От пальцевого счёта к механизации вычислений. Примитивные орудия вычислений. Абак. Барон Джон Непер и его "счёт на палочках". Блез Паскаль и "Паскалина". Лейбниц и его арифмометр. Чарльз Беббидж, его механические вычислители и фактическое рождение понятия "Архитектура" в применении к вычислительным системам. Леди Байрон-Лавлейс как первая программистка. Гипотетическая машина Тьюринга. Фон-Неймановские принципы построения процессоров. Гарвардская и Принстонская архитектуры. Электромеханические и электронные вычислительные машины. "Из ряда вон выходящие" вычислительные системы (вычислители с трёхзначной логикой, вычислители на основе арифметики остаточных классов). Суперкомпьютеры и задачи класса GRAND CHALLENGES.

Литература по теме 1:

1. От абака до компьютера. [Электронный ресурс] — Режим доступа - URL: http://fromatoc.mosedu.ru/ (дата обращения: 10.IX.2017).
2. Макарский Д.Д, Никоноров А.В. История компьютерной эры. — М.: Эксмо, 2016. — 256 с.

Тема 2. Определения понятия "Архитектура" в применении к вычислительным системам

Общее и разница межу понятиями *структура* и *архитектура*. Современные определения понятия архитектуры вычислительных систем. Машина Тьюринга как классика архитектуры вычислителей. Пять принципов фон-Неймана построения вычислителей. Классическая фон-Неймановская (Принстонская) архитектура, Гарвардская архитектура. их сравнительные преимущества и недостатки. Основные архитектуры многопроцессорных и многокомпьютерных вычислительных систем. Сравнительные достоинства и недостатки SMP- и MPP-архитектур. Классификации архитектур вычислителей. Классификация Флинна. Методы управления процессом вычислений. Понятие регистра - счётчика команд (*Set Counter*, *Instruction Pointer*). Форматы машинных команд. Методы повышения производительности процессоров. CISC- и RISC-процессоры. Конвейерная и суперскалярная обработка данных. Истинный параллелизм.

Литература по теме 2:

1. Э.Таненбаум, Т.Остин. Архитектура компьютера (издание 6). — СПб.: Питер, 2017. — 816 с.

Тема 3. Уровни управления процессом вычислений

Управление последовательностью вычислений. Процессор как синтез операционного и управляющего автоматов. Системы счисления современных процессоров. Выбор рационального основания позиционной системы счисления и форматов представления чисел в ЭВМ. точность представления чисел. Стандарт IEEE 754 для представления вещественных чисел. Выполнение арифметических действий над числами с фиксированной запятой, проблемы "размножения ошибки" вследствие переноса значений битов и невозможности распараллеливания на битовом уровне. Использование конвейерной архитектуры для повышения производительности процессора. Особенности функционирования конвейера.

Классика - архитектура процессора Intel x86. История разработки, внутренние регистры процессора, их разрядность, обозначение и назначение. Связь с внешними устройствами. Принцип определения последовательности исполнения машинных команд.

Литература по теме 3:

1. Э.Таненбаум, Т.Остин. Архитектура компьютера (издание 6). — СПб.: Питер, 2017. — 816 с.
2. Барский А.Б. Параллельные информационные технологии. Учебное пособие. — М.: Бином, 2013. –503 с.

Тема 4. Общие требования к программному коду. Пото́ковые (DATA-FLOW) вычислители

Условия корректного выполнения программного кода. Принципы программного управления последовательностью выполнения операций (CONTROL-FLOW) и управления порядком выполнения операций самими данными (DATA-FLOW). Отрицательная роль регистра - счётчика команд на возможность распараллеливания вычислений. Исторические попытки модернизации классической фон-Неймановской архитектуры. Принципиальная возможность распараллеливания процесса вычислений по произвольному алгоритму без априорного указания последовательности действий. Использование ЯПФ (*Ярусно-Параллельной Формы*) информационного графа алгоритма с целью выявления параллельно исполняемых блоков (*гранул*) программы. Структурная схема вычислителя с управлением последовательностью вычислений

потоком данных. Реализации потоковых вычислителей. Проблема ассоциативной памяти. особенности программирования DATA-FLOW машин. Компьютерная модель (*симулятор*) потокового вычислителя и её использование для моделирования и оптимизации процесса вычислений. Понятие интенсивности вычислений и возможность целенаправленного управления ею. Кумулятивная кривая количества исполненных операций.

Язык Ассемблер и машинные команды. Препроцессор, компилятор языка Ассемблер, макроассемблер. Отличие уровня программирования на ассемблере от программирования непосредственно в командах процессора. Машинные команды - основные группы, синтаксис описания, а́дресность команд, время выполнения. Понятие микропрограммирования.

Литература по теме 4:

1. Барский А.Б. Параллельные информационные технологии. Учебное пособие. — М.: Бином, 2013. –503 с.
2. Стерлинг Томас. Многоточие Стерлинга. // Журнал “Суперкомпьютеры”, № 3(3), 2010, с. 17-20. [Электронный ресурс] — Режим доступа - URL: http://www.supercomputers.ru/images/stories/arhive/Supercomputers_03-2010.pdf (дата обращения: 09.IX.2017).
3. Баканов В.М. Управление динамикой вычислений в процессорах потоковой архитектуры для различных типов алгоритмов. // Журнал “Программная инженерия”. — М.: 2015, с.20-24.
4. Зубков С.В. Для программистов. Assembler для DOS, Windows и Unix. — М.: ДМК Пресс. 2017. — 638 с.

Тема 5. Недостаток процесса вычислений в позиционной системе счисления и альтернативные решения

Принцип поразрядного последовательного выполнения булевых операций при использовании ПСС (*Позиционной Системы Счисления*). Распространение ошибки булевых операций от младших к старшим разрядам вследствие "переноса в старший разряд", затруднение выявления ошибки выполнения арифметического действия вследствие этого. Невозможность распараллеливания собственно арифметической операции в ПСС. Методы ускорения вычислений в рамках ПСС. История непозиционных систем счисления. Китайская теорема об остатках. Теория вычетов. Модулярная алгебра. История применения СТО (*Системы Остаточных Классов*) при разработке арифметических устройств, персоналии. Архитектура арифметических устройств на основе СОК. Табличный метод определения результатов арифметической операции по заданном основанию vs вычислительный метод, условия выбора одного из этих методов. Реальные ЭВМ, использующие вычисления на основе СОК. Преимущества и недостатки арифметических устройств на основе СОК.

Программное обеспечения разработки программ на Ассемблере. Интегрированная среда emu8086, её возможности. Макроассемблеры. Секции программы на ассемблере - описание формата исполняемого файла, данных, собственно ассемблерный код. Знаковые и беззнаковые целые числа. Возможности структурирования программ на ассемблере - макросы, процедуры. Использование регистров процессора и стека при использовании процедур. Соглашения по вызову strcall, cdecl, fastcall. Простейшая программа на ассемблере, последовательность компиляции ассемблерной программы в машинный код.

Литература по теме 5:

1. Авдошин С.М., Набебин А.А. Дискретная математика. Модулярная алгебра, криптография, кодирование. — М.: ДМК Пресс, 2016. — 352 с.

2. Зубков С.В. Для программистов. Assembler для DOS, Windows и Unix. — М.: ДМК Пресс. 2017. — 638 с.

Тема 6. Архитектура параллельных вычислительных систем

Цель параллелизации обработки информации. Ускорение вычислений vs надёжность. Физические ограничения повышения производительности процессоров на едином кристалле. Зависимость тепловыделения процессора от его тактовой частоты, закон Рэлея. Доказательство возможности полного распараллеливания вычислений для конкретного алгоритма. Ярусно-Параллельная Форма (ЯПФ) информационного графа алгоритма. Параллелизация внешняя и истинная. Технологии параллелизации. Конвейерный принцип. Векторные процессоры. Контрфон-Неймановские архитектуры вычислителей. Типы параллелизаций - параллелизация вычислений и параллелизация по данным. Абстрактные модели параллельных вычислений. Концепция неограниченного параллелизма. Понятие *тонкой информационной структуры* алгоритма. Формальное определение гранулы (зерна, блока) параллелизации. Глубина распараллеливания.

Целочисленная арифметика при программировании на ассемблере. Синтаксис команд целочисленной арифметики. Анализ данных и трассировка ассемблерной программы в эмуляторе emu8086. Анализ выполнения машинной команды с помощью регистра состояния процессора, битовые флаги состояния.

Литература по теме 6:

1. Э.Таненбаум, Т.Остин. Архитектура компьютера (издание 6). — СПб.: Питер, 2017. — 816 с.
2. Воеводин В.В., Воеводин Вл.В. Параллельные вычисления. — СПб.: БХВ-Петербург, 2004. — 609 с.
3. Т.Кормен, Ч.Лейзерсон, Р.Ривест, К.Штайн. Алгоритмы. Построение и анализ (издание 3). — М.: Вильямс, 2013. — 1324 с.
4. Зубков С.В. Для программистов. Assembler для DOS, Windows и Unix. — М.: ДМК Пресс. 2017. — 638 с.

Тема 7. Суперкомпьютеры

Определение суперкомпьютера. Психоэмоциональное состояние "механетикс" (*шутл.*). Задачи и области применения суперкомпьютеров. Проблемы класса GRAND CHALLENGES (*ЗАДАЧИ БОЛЬШОГО ВЫЗОВА*). Методы определения производительности суперкомпьютеров, требования к методам. История - вычислительные смеси Гибсона. Тесты LinPACK, HPL (*High-Performance Linpack benchmark*). Диапазон производительности современных суперкомпьютеров. Вычислительные кластеры. Реальное и пиковое быстродействие. Проблемы пета- и экзафлопса. Энергетическая стоимость одной арифметической операции. Оценка погрешности вычислений в зависимости от точности представления данных и числа выполненных операций. Обоснование использования вещественной арифметики двойной точности (IEEE 754) при суперкомпьютерных вычислениях. Топологии коммуникационных сред суперкомпьютеров. Концепция неограниченного параллелизма. Закон Амдаля и сетевой закон Амдаля.

Принципы организации ветвлений при программировании на Ассемблере. анализ конкретных битовых флагов регистра состояния и команды условного перехода. Реализации "ближних" и "дальних переходов", дополнительные возможности реализации циклов. 32-битовая архитектура и "плоская" память.

Литература по теме 7:

1. Воеводин В.В., Воеводин Вл.В. Параллельные вычисления. — СПб.: БХВ-Петербург, 2004. — 609 с.
2. Зубков С.В. Для программистов. Assembler для DOS, Windows и Unix. — М.: ДМК Пресс. 2017. — 638 с.

Тема 8. Нейронные сети и нейрокомпьютеры.

Биологический нейрон. Дерево входов (*дендриты*), выход (*аксон*), понятия *возбуждения* и *торможения* нейрона. Связь нейронов в коре головного мозга. Искусственные нейроноподобные структуры. Нейронная сеть (НС). Функция активации нейрона. Множества входов и выходов НС. Перцептрон. Понятие "решить задачу" в применении к НС. Обучение нейронной сети. Нейронные сети Кохонена, Хопфилда. Когнитрон. НС с обратным распространением информации. Обучающая, тестовая, рабочая последовательности. Признак окончания обучения НС. Одно- и многослойные НС. Процедуры обучения НС. Метод обратного распространения ошибки (*back propagation*), условие применимости этого метода. Самообучающиеся НС. Программные реализации НС. Аппаратная реализация НС. Применения НС. Нейронная сеть как решатель, функционирующий на основе теории нечёткой логики.

Литература по теме 8:

1. Рашид Тарик. Создаём нейронную сеть. — М.: Вильямс, 2017. — 272 с.
2. Боресков А.В., Харламов А.А. Основы работы с технологией CUDA. — М.: ДМК Пресс, 2016. — 232 с.

Тема 9. Зависимость производительности вычислительного кластера МРР-архитектуры от параметров оборудования и решаемой задачи

Вычислительный кластер как представитель многопроцессорных вычислительных систем (МВС) архитектуры МРР. Вычислительные кластеры BEOWULF. МВС как симбиоз вычислительных узлов (ВУ) и коммуникационной сети (КС). Потоки данных в МВС. Необходимость синхронизации времени отдельных ВУ кластера. Задержка данных при обмене информацией через КС как источник простóя ВУ. Латентность (*инерционность*) сети передачи данных. Понятие *тонкой информационной структуры* программы и *гра́нулы* (зёрна, блока) параллелизма. Коэффициент гранулярности. технологии разработки параллельных программ. Параллельное программирование с использованием передачи сообщений MPI (*Messages Passing Interface*). Многозначность параллельных реализаций единого математического алгоритма. Пример - ленточный алгоритм параллельного умножения матриц классическим способом. Системные скрипты компиляции и запуска на исполнение MPI-программ. Определение зависимости времени обмена по компьютерной сети и производительности сети от размера сообщения (при обменах типа “точка - точка”). Исследование зависимости производительности МВС от размера обрабатываемых данных и количества вычислительных узлов. Верификация кубичной зависимости времени выполнения параллельной программы от размерности данных. Интерпретация результатов.

Разница между форматами исполняемых файлов для 16, 32 и 64-х битовых архитектур. программные сегменты и их реализация для 16-ти битовых процессоров Intel и для "плоской" (flat) организации памяти. Реализация многозадачности, разграничение доступа к памяти. Двоично-десятичная арифметика как способ представления чисел с фиксированной точкой. Арифметический сопроцессор действий с плавающей запятой - его организация, машинные команды.

Литература по теме 9:

1. Гергель В Э.Таненбаум, Т.Остин. Архитектура компьютера (издание 6). — СПб.: Питер, 2017. — 816 с.
2. Воеводин В.В., Воеводин Вл.В. Параллельные вычисления. — СПб.: БХВ-Петербург, 2004. — 609 с.
3. Зубков С.В. Для программистов. Assembler для DOS, Windows и Unix. — М.: ДМК Пресс. 2017. — 638 с.

Тема 10. Транспьютеры

Предпосылки перехода к распределённой архитектуре вычислений. Энтони Хоар как инициатор разработки транспьютерной архитектуры. Транспьютер как специализированный процессор с большим количеством каналов связи с подобными. Роль компании Inmos в разработке первых транспьютеров. Архитектура транспьютера T805 фирмы Inmos. Топологии объединения транспьютеров. Языки программирования для транспьютеров. Современное состояние транспьютерной индустрии. Транспьютеры как элементы реализации архитектур систолических массивов, вычислителей с реконфигурируемой структурой.

Отладка ассемблерных программ. Режим эмуляции и пошагового исполнения в интегрированных средах разработки программ на Ассемблере. Повышение надёжности ассемблер-программирования путём использования готовых библиотек. Частые ошибки при программировании на ассемблере - некорректность работы со стеком, неверные манипуляции со счётчиком команд, "затира́ние" регистров процессора. Полезность применения известных схем (шаблонов) работы с подпрограммами и тщательного документирования (в т.ч. комментирования) ассемблерных программ. Понятие и применение пролога и эпилога при использовании процедур.

Литература по теме 10:

1. Транспьютерные системы – становление в России ЭВМ с массовым параллелизмом. [Электронный ресурс] — Режим доступа - URL: http://www.computer-museum.ru/histussr/transputer.htm (дата обращения: 10.IX.2017).
2. Сэр Чарльз Энтони Ричард Хоар. [Электронный ресурс] — Режим доступа - URL: https://habrahabr.ru/post/274865/ (дата обращения: 10.IX.2017).
3. Зубков С.В. Для программистов. Assembler для DOS, Windows и Unix. — М.: ДМК Пресс. 2017. — 638 с.

Тема 11. Метакомпьютинг и концепция GRID

Понятия метакомпьютера и метакомпьютинга. Вычислительная сеть (GRID). Архитектура метакомпьютера. Отличия метакомпьютера от традиционного компьютера. Понятие *добровольных вычислений*. Инструментальные системы организации и управления метакомпьютинга. Реализации метакомпьютерных вычислительных систем. Облачные вычисления как частный случай метакомпьютинга. Концепция доверительных отношений между заказчиком и фирмой, предоставляющей услугу облачного сервиса. Облачный сервис с точки зрения конечного пользователя. Доводы *за* и *против* облачных технологий. Ричард Столлман против использования проприентарных программ и ресурсов.

Принцип "90/10 - 90% времени тратится на выполнение всего 10% кода, причём обычно даже на меньшую часть кода". Определение критических по времени выполнения участков программы методом анализа исходного кода на языках программирования высокого уровня и вставка ассемблер-фрагментов с помощью ключевого слова asm. Дизассемблирование исполня-

емого файла программы с дальнейшим выявлением неоптимально скопилированных участков кода и заменой его ассемблерного представления.

Литература по теме 11:

1. Воеводин В.В., Воеводин Вл.В. Параллельные вычисления. — СПб.: БХВ-Петербург, 2004. — 609 с. [стр. 154–162].
2. Э.Таненбаум, Т.Остин. Архитектура компьютера (издание 6). — СПб.: Питер, 2017. — 816 с.

3.

Тема 12. Архитектура GPU фирмы NVIDIA и технология CUDA

История совершенствования графических карт. Шейдерный механизм вычислений как основа использования карт в качестве графических процессоров (GPU - *Graphics Processing Unit*). GPU как процессор массового параллелизма архитектуры SIMD. Графические процессоры фирм NVIDIA и AMD. направление GPGPU (*General-Purpose computing on Graphics Processing Units*). Потоковый процессор, вычислительные ядра потокового процессора. Программирование графических процессоров. Модель программирования в CUDA (*Compute Unified Device Architecture*). Понятие устройства (*device*), центрального процессора (*host*), потока (*нити, thtread*). Компилятор NVCC. Значение Compute Capability как показатель вычислительных возможностей GPU. Принципиальные различия между нитями GPU и нитями CPU. Бесплатность SDK CUDA и уровень поддержки пользователей и разработчиков фирмой NVIDIA. Типовой шаблон работы с GPU на языках высокого уровня. Поддерживаемые языки высокого уровня с CUDA. Программирование с использованием низкоуровневого CUDA driver API и высокоуровневого CUDA runtime API. Идентификация вычислительных ядер и распределение блоков данных для обработки на конкретных ядрах. Типовые примеры (язык C++ интегрированной среды Microsoft Visual Studio) программ для CUDA. Экспериментальная оценка производительности GPU vs CPU.

Литература по теме 12:

1. Боресков А.В., Харламов А.А. Основы работы с технологией CUDA. — М.: ДМК Пресс, 2016. — 232 с.

Тема 13. Ана́логовые вычислительные системы

Ана́логовый и цифровой способы представления и хранения информации. История механических аналоговых вычислителей. Аналоговая вычислительная машина (АВМ) - *механический дифференциальный анализатор* (Уильям Томсон, лорд Кельвин). Классификация АВМ по используемому рабочему телу. Интегрирование (суммирование) на гидроинтеграторах. Нейронные сети как аналоговые вычислителя. Архитектура аналоговых вычислителей. Базовые элементы электронных АВМ. Операционный усилитель как основа электронной АВМ. Устройства ввода и вывода АВМ. Модели аналоговых вычислителей и их функциональные возможности. Метод электрогидродинамических аналогий (ЭГДА) и его применение к решению уравнений Лапласа. Области применения АВМ. Достоинства и недостатки АВМ. Гибридные вычислительные системы, преимущества перед цифровыми вычислительными системами. Аналого-цифровые (АЦП) и цифро-аналоговые (ЦАП) преобразователи.

Литература по теме 13:

1. Аналоговые вычислители: жизнь до и во время эпохи Цифры. . [Электронный ресурс] — Режим доступа - URL: http://old.computerra.ru/vision/668463/ (дата обращения: 07.IX.2017).

2. Обухов Игорь. Сны об аналоговых вычислениях. // Журнал "Суперкомпьютеры", № 3(11), 2012, с. 59-61. [Электронный ресурс] — Режим доступа - URL: http://www.supercomputers.ru/images/stories/arhive/Supercomputers_11-2012.pdf (дата обращения: 05.IX.2017).
3. Аналоговая вычислительная машина. Русская энциклопедия ТРАДИЦИЯ. [Электронный ресурс] — Режим доступа - URL: http://traditio-ru.org/wiki/%D0%90%D0%BD%D0%B0%D0%BB%D0%BE%D0%B3%D0%BE%D0%B2%D0%B0%D1%8F_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D0%BC%D0%B0%D1%88%D0%B8%D0%BD%D0%B0 (дата обращения: 05.IX.2017).

Тема 14. Вычислители с программируемой архитектурой. Пути усовершенствования архитектур вычислителей

Предпосылки разработки вычислительных систем с программируемой архитектурой. Реконфигурация как средство создания вычислительной структуры, максимально эффективной для заданного алгоритма. Связь реконфигурируемых вычислительных систем с идеей *клеточных автоматов* фон-Неймана (1948). Понятие процессорного элемента (ПЭ), универсальной коммутационной среды (УКС). Два этапа настройки реконфигурируемых вычислительных систем на выполнение конкретной задачи. Программируемые логические интегральные схемы (ПЛИС) как основа создания реконфигурируемых систем. Эффективность систем с программируемой архитектурой. Перспективные пути совершенствования архитектур вычислителей - переход к принципу перемещения кода к данным (а не наоборот, как принято сейчас), обработка данных по мере их готовности (пот*о*ковый принцип), т*е*говая память, домены санкционированного доступа в память, реализация одноуровневой памяти.

Литература по теме 14:

1. Дордопуло А.И., Каляев И.А., Левин И.И. и др. Высокопроизводительные реконфигурируемые вычислительные системы. // Журнал "Суперкомпьютеры", № 3(3), 2010, с. 44-47. [Электронный ресурс] — Режим доступа - URL: http://www.supercomputers.ru/images/stories/arhive/Supercomputers_03-2010.pdf (дата обращения: 06.IX.2017).
2. Стерлинг Томас. Многоточие Стерлинга. // Журнал “Суперкомпьютеры”, № 3(3), 2010, с. 17-20. [Электронный ресурс] — Режим доступа - URL: http://www.supercomputers.ru/images/stories/arhive/Supercomputers_03-2010.pdf (дата обращения: 05.IX.2017).

Тема 15. Квантовые вычислители и системы передачи данных

Определение квантового компьютера, история идеи. Принципиальная возможность сверхбыстрого выполнения вычислений. Понятие кубита, квантовой связанности (*спутанности*). Число линейно независимых состояний системы N кубитов. Физическая реализация кубитов. Квантовые проблемы задания исходных данных и считывания результата. Фундаментально *вероятностный характер* квантовых вычислений. Известные алгоритмы для квантовых компьютеров (алгоритмы Гровера, Залки-Визнера, Шора). Современные реализации квантовых вычислителей. Неэффективность известных методов шифрования при использовании квантового компьютера для дешифрации. Недоставки современных линий передачи информации. Использование квантовых эффектов для детектирования перехвата данных копированием.

Литература по теме 15:

1. Альбов А.С. Квантовая криптография. — СПб.: Страта, 2015. — 248 с.
2. Сысоев Сергей. Квантовые вычисления: от бита к кубиту. // Журнал "Суперкомпьютеры", № 2(10), 2012, с. 30-33. [Электронный ресурс] — Режим доступа - URL: http://www.supercomputers.ru/images/stories/arhive/Supercomputers_10-2012.pdf (дата обращения: 05.IX.2017).

Тема 16. Архитектура систем поиска информации в сети InterNet

Краткая история и современные параметры сети InterNet. Проблемы и технологии поиска информации в сети InterNet. Технологии Data Mining. Классификация поисковых систем в сети InterNet. Каталоги и специализированные базы данных сетевых ресурсов. Активные поисковые системы. Закон Зипфа (*George Kingsley Zipf*) в применении к системам поиска информации. Архитектура и принципы функционирования машин поиска информации в сети InterNet. Принципы ранжирования информации в поисковых системах. Понятие релевантности. Поисковая машина Google. История развития компании Google. Отличия принципов ранжирования и поиска информации Google существующих. Архитектура аппаратной части поисковых машин Google. Принцип построения серверов Google – простота и дублирование.

Литература по теме 16:

1. Попов Артём. Эффективная методика поиска информации в сети Интернет. [Электронный ресурс] — Режим доступа - URL: http://citforum.ru/pp/search_03.shtml (дата обращения: 05.IX.2017).

7.2 Содержание практических занятий

Примерный перечень тем практических занятий:

1. Углубленное изучение специальных вопросов по темам лекций (с использованием информационных материалов преподавателя и/или ресурсов сети InterNet).
2. Практика подготовки, отладки и выполнения программ на языке Ассемблер для процессоров семейства x86 фирмы Intel.
3. Практика подготовки, отладки и выполнения CUDA-программ (Microsoft Visual Studio 2010, CUDA SDK).
4. Подготовка, отладка и выполнение параллельных программ в технологии MPI на вычислительном кластере вычислительного центра ВШЭ (Microsoft Visual Studio 2015, MPICH) в режиме удалённого доступа.
5. Ознакомление с практикой работы на программном симуляторе потокового (DATA-FLOW) вычислителя, подготовка простейших программ, их отладка и выполнение в режиме интерпретации; импорт данных с целью графической интерпретации результатов.

На практических занятиях языком программной реализации алгоритмов является язык программирования C++, в связи с этим на занятиях повторяются основы языка C++ в объёме, необходимом для выполнения практических заданий. Дополнительно изучается язык программирования Ассемблер в версии процессоров семейства x86 фирмы Intel.

8 Образовательные технологии

Работа на практических занятиях предполагает подробное изучение материала по текущей теме лекции (с использованием сети InterNet и материалов преподавателя); освоение подготовки программ на языке Ассемблера и их отладки, подготовку, отладку и выполнение программ

для определения эффективности арифметических ускорителей на основе графических карт фирмы NVIDIA.

Домашнее задание предполагает самостоятельный анализ задания, выбора инструментов для создания программы на языке ассемблер, отладку программ на этом языке. Задания индивидуальные.

9 Оценочные средства для текущего контроля и аттестации студента

9.1 Тематика заданий текущего контроля

9.1.1 Домашнее задание

1. Разработка программы на языке Ассемблер для процессоров архитектуры x86 фирмы Intel.

Примерное задание - по заданному алгоритму требуется**:**

1) Разработать программу на языке ассемблера для нахождения наибольшего общего делителя (НОД, GCD - *Greatest Common Divisor*) двух положительных чисел по методу Эвклида вычитанием. Программу разработать в среде Ассемблер-компилятора fasm с использованием 32-ти битовых регистров и оформить в виде ассемблер-процедуры (исходные данные в регистрах EAX,EBX, результат вернуть через EAX).

Формат входных и выходных данных:

1. Для набора текста программы применяется интегрированная среда разработки Ассемблерных программ emu8086, компиляция производится компилятором fasm.
2. Команды программы находятся в текстовом файле GCD_Euclid_32.asm, скомпилированный бинарный файл (версия 32-бит) - GCD_Euclid_32.exe. Бинарный файл исполняется аппаратно 32-битовым процессором Intel.

Формат выходных данных

Выходные данные - исполняемый бинарный 32-битный файл GCD_Euclid_32.exe.

Пример входных и выходных данных

Входные данные – файл GCD_EUCLID_32.ASM

```
format  PE GUI 4.0 ; GUI-приложение для Windows (EXE-файл)
;
include "c:/emu8086/fasm/include/win32ax.inc" ; путь к подкаталогу /include/
;
.data ; секция данных
formats db "НОД чисел %d и %d суть %d (десятичные)",0
result db 256 dup(?) ; строка для форматного вывода
;
A dd 81d ; исходные данные (1-е число для нахождения НОД - Наиб. Общего Делителя)
B dd 54d ; 2-е число для того же
;
.code ; начало секция кода
start: ; точка старта программы
;
```

```
mov eax,dword[A] ; исходные данные присваиваем регистрам
mov ebx,dword[B]
;
call GCD_2S_Euclid_32 ; вызов процедуры GCD_2S_Euclid_32 (поиск НОД двух чисел)
;
invoke wsprintf,result,formats, [A],[B],eax ; форматирование строки для вывода
add esp,20 ; восстановим стек (соглашение о вызовах cdecl)
;
invoke MessageBox,0,result,"Данные расчёта",MB_OK ; EAX ← код нажатой кнопки
invoke ExitProcess,0 ; возврат в Windows с кодом возврата 0
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
proc GCD_2S_Euclid_32 ; USES EAX,EBX; RESULT IN EAX
; фактически используется fastcall-соглашение (передача формальных
; параметров не посредством стека, а через регистры процессора
; ищет GCD от EAX,EBX по Эвклиду (при EAX==0 || EBX==0 возвращается 0)
; GCD is abbreviation of 'Greatest Common Divisor'
; эта процедура изменяет только регистры EAX и EBX
;
; while(EAX!=0 && EBX!=0) { if (EAX>EBX) EAX-=EBX else  EBX-=EAX; } return EAX+EBX;
;
cmp eax,0 ; сравниваем EAX с 0
jne ax_jne ; переход, если EAX!=0
ret ; возвращаем EAX=0 как результат
ax_jne: ; вариант EAX!=0
; cmp eax,0 ; сравниваем EAX с 0 (повторять сравнение не надо)
jge ax_jge ; переход, если EAX>=0
neg eax ; изменили знак EAX (получили модуль EAX)
ax_jge: ; вариант EAX>=0
;
cmp ebx,0 ; сравниваем EBX с 0
jne bx_jne ; переход, если EAX!=0
mov eax,ebx ; готовим возврат EAX ← 0
ret ; возвращаем EAX=0 (result)
bx_jne: ; вариант EBX!=0
; cmp ebx,0 ; сравниваем EBX с 0 (повторять сравнение не надо)
jge bx_jge ; переход, если EBX>=0
neg ebx ; изменили знак EBX (получили модуль EBX)
bx_jge: ; вариант EAX>=0
;
cycle: ; начинаем основной цикл метода Эвклида ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
cmp eax,ebx ; сравним EAX и EBX
je ended ; при EAX==EBX (ZF==1) выход с возвратом EAX
;
jl metka ; при EAX<EBX (SF#OF) переход на metka
sub eax,ebx ; EAX ← EAX-EBX
jmp cycle ; безусловный переход на cycle
metka:
sub ebx,eax ; EBX ← EBX-EAX
jmp cycle ; безусловный переход на cycle
ended: ; EAX ← Result ( GCD )
;
ret ; возврат из процедуры GCD_2S_Euclid_32
;
```

```
endp ; конец процедуры GCD_2S_Euclid_32

.end start ; конец программы
```

Выходные данные

При входных данных $EAX=81_{10}$ и $EBX=54_{10}$ после выполнения процедуры GCD_Euclid_32 правильный ответ $EAX=1B_{16}=27_{10}$

Оценивание домашней работы

1) Верно выполняется тестовый пример, приведённый выше - 3 балла.
2) Отсутствуют синтаксические ошибки в заданном примере - 4 балла.
3) Заданный пример выполняется не на всех входных данных - (+ до 2 баллов) (6-7 баллов).
4) Заданный пример выполняется на всех входных данных (+ до 2 баллов) (8-9 баллов).

9.2 Вопросы для оценки качества освоения дисциплины

1. История автоматизированных вычислений. Первые появления понятий “структура” и “архитектура” вычислительных систем.
2. Архитектура вычислительной системы Ч.Беббеджа. Роль Ады Лавлейс в её проектировании и программировании.
3. Понятие “архитектура” в применении к вычислительным системам.
4. Машина Тьюринга. Её архитектура, связь с архитектурой современных вычислительных системам.
5. Язык программирования Ассемблер - история, область применения.
6. Выбор основания системы счисления. “Двои́чные” и “трои́чные” вычислители.
7. Архитектура вычислительных систем фон Неймана. Пять принципов построения вычислителей, достоинства и недостатки “принстонской” и “гарвардской” архитектур.
8. Архитектуры SMP и MPP. Преимущества и недостатки этих архитектур, области применения.
9. Особенности разработки программ в архитектуре SMP – достоинства и недостатки.
10. Особенности разработки программ в архитектуре MPP – достоинства и недостатки.
11. Классификация архитектур по М.Флинну. Параметры классификации, примеры реализации архитектур.
12. Повышение быстродействия вычислительных систем с помощью совершенствования их архитектуры. Конвейерные вычислители.
13. Повышение быстродействия вычислительных систем с помощью совершенствования их архитектуры. Векторные вычислители.
14. Повышение быстродействия вычислительных систем с помощью совершенствования их архитектуры. Вычислители со сверхдлинным командным словом (VLIW).
15. Архитектура систем команд вычислителей. А́дресность команд, CISC и RISC – системы команд.
16. Арифметические операции с числами в позиционной системе счисления. Недостаток этого метода, альтернативы.
17. Упрежд**а**ющая загрузка и спекулятивное выполнение команд в современных процессорах. Проблемы и решения этого подхода.
18. Основные требования к программному коду. Особенности выполнение его в вычислителях классической фон Неймановской архитектуры и потоковых (DATA-FLOW) вычислителях.
19. Понятие ярусно-параллельной формы (ЯПФ) информационного графа алгоритма и услов**и**я гот**о**вности операций к выполнению.

20. Архитектура пот**о**кового (DATA-FLOW) вычислителя. Проблемы реализации такой вычислительной системы.
21. Понятие гра́нулы (*зерна, блока*) параллелизма. Размер гранул параллелизма в вычислителях различной архитектуры.
22. Архитектура суперкомпьютеров. Задачи, требующие использования супер-ЭВМ; основные параметры супер-ЭВМ.
23. Группы инструкций процессора. Использование ими регистров процессора.
24. Недостатки выполнения арифметических действий на арифметико-логических современной архитектуры, потенциал использования непозиционных систем счисления. Вычислители на основе СОК (системы оста́точных классов), их достоинства и недостатки, реализация, перспективы.
25. Метод проведения математических вычислений на графических процессорах (GPU). Архитектура GPU, конкретные модели, основные параметры, область эффективного применения.
26. Принципы разработки программ для использования в технологии CUDA. Понятия “хоста” (*host*) и “девайса” (*device*), приёмы программирования, используемые ср*е*ды создания приложений для архитектуры CUDA.
27. Использование устройств архитектуры CUDA в современных суперкластерах. Разделение задач по эффективности решения на GPU.
28. особенности архитектуры и системы инструкций процессоров ARM.
29. Архитектура вычислительных систем на основе нейронных сетей. Класс реш**ае**мых задач, процесс обучение нейронной сети, метод обратного распространения ошибки.
30. Ана́логовые вычислители – архитектура, составные элементы аналоговых вычислителей, область применения, достоинства и недостатки. Гибридные вычислительные системы.
31. Архитектура вычислителей на основе транспьютеров. Понятие транспьютера, история разработки, потенциал агрегации транспьютеров, современное состояние.
32. Архитектура вычислителей на основе систол*и*ческих матриц. Принцип обработки данных, реализации, области применения.
33. Стековая машина, связь с обратной польской нотацией записи выражений. Достоинства и недостатки стековой машины, примеры реализации.
34. Сравнение архитектур SIMD и SMP (на примере GPU и CPU). Достигнутые вычислительные мощности, стоимость, области эффективного применения, перспективы.
35. Архитектура поиск**о**вых систем в сети InterNet. Принципы поиска информации, формирование ключевых слов, аппаратная реализация поисковых машин.
36. Недостатки архитектур современных вычислителей, направления их совершенствования.
37. Архитектура математического сопроцессора плавающей точки i87 - кольцевой стек, точность представления чисел, принцип разделения команд между центральным процессором и сопроцессором.
38. Квантовые вычислители и сети передачи данных. Основные принципы действия, потенциальные возможности, известные реализации отдельных компонентов.
39. Архитектуры вычислителей с наличием регистра-счётчика команд и без **о**ного. Примеры, достоинства и недостатки.
40. Архитектура метакомпьютера. Отличия метакомпьютера от обычного компьютера, гетерогенность метакомпьютера, реальные системы метакомпьютинга.
41. Архитектура “облачных вычислений” (*cloud computing*). Понятие облачного сервиса, прозрачности и гибкости изменения потребляемых клиентом ресурсов, критика полезности и безопасности технологии облачных вычислений.
42. Взаимодействие параллельных процессов, известные методы программной синхронизации. Аппаратная синхронизация.

43. Принципы управление последовательностью выполнения процессорных инструкций в вычислителях традиционной архитектуры и потоковой (DATA-FLOW) архитектуры.
44. Ограничения производительности вычислителей многопроцессорных архитектур – закон Амдаля, сетевой закон Амдаля. Формулировка, исходные предпосылки, применимость.
45. Особенности использования внутренних регистров процессора командами систем CISC и RISC.
46. Организация и использование стека в процессорах x86. Пролог и эпилог в процедурах на ассемблере.
47. CISC и RISC системы команд. Основные отличия, особенности декодирования и исполнения, преимущества.
48. Абстрактные модели параллельных вычислений. Теорема Брента.
49. Конвейерная архитектура как средство повышения производительности вычислителя. Условия эффективной работы конвейера.
50. Причины необходимости следования концепции однократного присваивания в вычислителях потоковой (DATA-FLOW) архитектуры.
51. Микропрограммная архитектура процессоров x86. Достоинства и недостатки микропрограммной архитектуры.

10 Учебно-методическое и информационное обеспечение дисциплины

10.1 Основная литература:

1. Э.Таненбаум, Т.Остин. Архитектура компьютера (издание 6). — СПб.: Питер, 2017. — 816 с.

10.2 Дополнительная литература и источники

1. Randal E.Bryant, David R.O'Hallaron. Computer systems: a programmer's perspective. 2-nd ed. — Prentice Hall, 2010. — 1043 p. (есть русск. перевод: Рэндал Э.Брайант, Дэвид Р.О'Халларон. Компьютерные системы. — СПб.: БХВ-Петербург, 2005. — 1104 с.)
2. Зубков С.В. Для программистов. Assembler для DOS, Windows и Unix. — М.: ДМК Пресс. 2017. — 638 с.
3. Воеводин В.В., Воеводин Вл.В. Параллельные вычисления. — СПб.: БХВ-Петербург, 2004. — 609 с.
4. Барский А.Б. Параллельные информационные технологии. Учебное пособие. — М.: Бином, 2013. –503 с.
5. Рашид Тарик. Создаём нейронную сеть. — М.: Вильямс, 2017. — 272 с.
6. Боресков А.В., Харламов А.А. Основы работы с технологией CUDA. — М.: ДМК Пресс, 2016. — 232 с.

10.3 Справочники, словари, энциклопедии

1. NVIDIA: архитектура графических карт и технология CUDA. [Электронный ресурс] — Режим доступа - URL: http://www.nvidia.ru (дата обращения: 10.IX.2017).
2. Программирование на языке Ассемблер. [Электронный ресурс] — Режим доступа - URL: https://nova.rambler.ru/search?query=%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5%20%D0%BD%D0%B0%20%D1%8F%D0%B7%D1%8B%D0%BA%D0%B5%20%D0%90%D1%81%D1%81%D0%B5%D0%BC%D0%B1%D0%BB%D0%B5%D1%80 (дата обращения: 25.V.2017).

3. Журнал "Суперкомпьютеры". [Электронный ресурс] — Режим доступа - URL: http://www.supercomputers.ru (дата обращения: 12.VI.2017).
4. AlgoWiki. Открытая энциклопедия свойств алгоритмов. [Электронный ресурс] — Режим доступа - URL: https://algowiki-project.org/ru/ (дата обращения: 13.IX.2017).

10.4 Программные средства

Практические занятия проводятся в компьютерном классе с выходом в Интернет и доступом к ресурсам электронной библиотеки НИУ ВШЭ. Каждый студент должен иметь рабочее место. Необходимое программное обеспечение:

1. Microsoft Office Professional 2007-2015
2. Microsoft Visual Studio 2008-2015.
3. Интегрированная среда программирования на языке Ассемблер emu8086, компилятор с языка Ассемблер fasm.

10.5 Дистанционная поддержка дисциплины

Дистанционная поддержка дисциплины обеспечивается использованием LMS. В разделе дисциплины "Архитектура вычислительных систем" размещаются материалы лекций и практических занятий, тесты для самоподготовки, проекты, оценки текущего и итогового контроля.

11 Материально-техническое обеспечение дисциплины

Проектор для лекций и семинаров, классы для семинаров с компьютерами, на которых установлена инструментальная среда Microsoft Visual Studio 2015, эмулятор emu8086, компилятор fasm.