МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра «Конструирования и технологии радиоэлектронных средств»

Информатика, часть 1

(Лекции)

Преподаватель: Алексей Анатольевич, Бизяев
Место работы: НГТУ, РЭФ, КТРС, IV-530а
Контактный телефон: +7-905-958-6134
Web: http://vk.com/computer_science_1
https://ciu.nstu.ru/WebInput/persons/20761/edu_actions/pcources/informatika_chast_1
Консультации: чётный четверг, 15:45-17:15, IV-530а (предварительно договориться)

Новосибирск, 2018

Зачётные единицы

- **Информационные технологии, часть 1**

- **Основные понятия информатики**.
- **Основные этапы развития вычислительной техники**
- **Меры и единицы измерения информации.**
- **Системы счисления. Кодирование данных в ЭВМ.**
- **Основные понятия алгебры логики. Логические основы ЭВМ.**
- **Состав и назначение основных элементов персонального компьютера.**
- **Запоминающие устройства: классификация, принцип работы, основные характеристики. Устройства ввода-вывода данных, их разновидности и основные характеристики**.
- **Классификация программного обеспечения. Виды программного обеспечения и их характеристики.**
- **Понятие и назначение операционной системы. Разновидности операционных систем. Служебное (сервисное) обеспечение.**
- **Файловая структура операционных систем.**
- **Основы машинной графики**.
- **Программное обеспечение обработки текстовых данных**.
- **Принципы построения сетей. Сетевое оборудование**.
- **Сервисы Интернета**.
- **Защита информации в локальных и глобальных компьютерных сетях**.
- **Шифрование данных. Электронная подпись.**

Зачётные единицы

- **Информационные технологии, часть 2**

- **Моделирование как метод познания.**
- **Классификация и формы представления моделей.**
- **Методы и технологии моделирования.**
- **Информационная модель объекта**.
- **Алгоритм и его свойства**. **Способы записи алгоритмов**.
- **Линейная алгоритмическая структура**.
- **Разветвляющаяся алгоритмическая структура**.
- **Циклические алгоритмические структуры**.
- **Основные операторы циклов и ветвления**.
- **Типовые алгоритмы**.
- **Трансляция, компиляция и интерпретация.**
- **Эволюция и классификация языков программирования.**
- **Интегрированные среды программирования.**
- **Этапы решения задач на компьютерах.**
- **Структурное программирование.**
- **Объектно-ориентированное программирование**.
- **Общее понятие о базах данных (БД)**.
- **Модели данных в информационных системах**. **Реляционная модель базы данных**.
- **Основные операции с данными**.
- **Назначение и основы использования систем искусственного интеллекта.**

Список литературы

- **Литература**

- Примерная программ дисциплины «Информатика». Министерство образования РФ, 2000.
- Информатика. Базовый курс. Учебник для ВУЗов. Под ред. С.В. Симановича. Питер, 2006.
- Могилев А.В., Пак Н.И., Хеннер Е.К. Информатика. Учебное пособие для студентов пед. ВУЗов. М.: Академия, 2004.
- Соболь Б.В., Галин А.Б., Панов Ю.В., Рашидова Е.В., Садовой Н.Н. Информатика. Учебник. – Ростов н/Д, Феникс, 2005.
- Сырецкий Г.А. Информатика. Базовый курс.
- Материалы тестов ФЭПО для технических направлений и специальностей. http://www.fepo.ru/index.php?menu=structs_demo
- Таненбаум Э. Архитектура компьютера.
- http://www.3dnews.ru/offsyanka/613751

Темы лекций

1. Основные понятия и определения
2. История развития ЭВМ
3. Архитектура ЭВМ
4. Архитектура ЭВМ
5. Архитектура ЭВМ
6. Представление информации в ЭВМ
7. Представление информации в ЭВМ
8. Представление информации в ЭВМ
9. Логические основы построения ЭВМ
10. Телекоммуникационные сети
11. Телекоммуникационные сети
12. Дисковые массивы RAID
13. Виды угроз при работе в сети
14. Способы защиты информации

- Контрольная работа №1
- Контрольная работа №2

Аттестация по дисциплине

Для аттестации студентов по дисциплине используется балльно-рейтинговая система. Сумма баллов за текущую деятельность составляет не более 80 баллов, количество баллов по итоговой аттестации (зачёт) не превышает 20 баллов. В течение 1-го семестра необходимо представить и защитить 4 лабораторных работы, самостоятельную работу, выполнить две контрольные работы по материалам лекций в сроки, установленные учебным графиком.

№	Вид учебной работы	Максимальное количество баллов	Диапазон баллов	Срок выполнения (неделя семестра)
1	Лабораторная работа №1	5	3-5	4
2	Лабораторная работа №2	5	3-5	8
3	Лабораторная работа №3	5	3-5	12
4	Лабораторная работа №4	5	3-5	16
5	Самостоятельная работа (Реферат)	30	5-30	15
6	Самостоятельная работа студента	10	0-10	15
7	Контрольная работа №1	10	0-10	6
8	Контрольная работа №2	10	0-10	11
9	Зачёт		5-20	17

Аттестация по дисциплине

Рейтинговая система

Характеристика работы студента	Диапазон баллов рейтинга	Оценка ECTS	Традиционная шкала оценки	
«Отлично» – работа высокого качества, уровень выполнения отвечает всем требованиям, теоретическое содержание курса освоено полностью, без пробелов, необходимые практические навыки работы с освоенным материалом сформированы, все предусмотренные программой обучения учебные задания выполнены, качество их выполнения оценено числом баллов, близким к максимальному	90-100	A+	отлично	зачтено
		A		
		A-		
«Очень хорошо» – работа хорошая, уровень выполнения отвечает большинству требований, теоретическое содержание курса освоено полностью, без пробелов, необходимые практические навыки работы с освоенным материалом в основном сформированы, все предусмотренные программой обучения учебные задания выполнены, качество выполнения большинства из них оценено числом баллов, близким к максимальному	80-89	B+		
		B	хорошо	
		B-		
«Хорошо» – уровень выполнения работы отвечает всем основным требованиям, теоретическое содержание курса освоено полностью, без пробелов, некоторые практические навыки работы с освоенным материалом сформированы недостаточно, все предусмотренные программой обучения учебные задания выполнены, качество выполнения ни одного из них не оценено минимальным числом баллов, некоторые из выполненных заданий, возможно, содержат ошибки	70-79	C+		
		C		
		C-	удов.	
«Удовлетворительно» – уровень выполнения работы отвечает большинству основных требований, теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые практические навыки работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые виды заданий выполнены с ошибками	60-69	D+	удов.	зачтено
		D		
		D-		
«Посредственно» – работа слабая, уровень выполнения не отвечает большинству требований, теоретическое содержание курса освоено частично, некоторые практические навыки работы не сформированы, многие предусмотренные программой обучения учебные задания не выполнены, либо качество выполнения некоторых из них оценено числом баллов, близким к минимальному	50-59	E		
«Неудовлетворительно» (с возможностью пересдачи) – теоретическое содержание курса освоено частично, необходимые практические навыки работы не сформированы, большинство предусмотренных программой обучения учебных заданий не выполнено, либо качество их выполнения оценено числом баллов, близким к минимальному; при дополнительной самостоятельной работе над материалом курса возможно повышение качества выполнения учебных заданий	25-49	FX	неуд.	не зачтено
«Неудовлетворительно» (без возможности пересдачи) – теоретическое содержание курса не освоено, необходимые практические навыки работы не сформированы, все выполненные учебные задания содержат грубые ошибки, дополнительная самостоятельная работа над материалом курса не приведет к какому-либо значимому повышению качества выполнения учебных заданий	0-24	F		

Правила аттестации

Правила текущей аттестации

- К защите лабораторной работы допускается студент, выполнивший задания в полном объеме и представивший отчет по приведенной на сетевом диске кафедры форме.
- К защите самостоятельной работы допускается студент, выполнивший соответствующее задание в полном объеме и представивший отчет в соответствие с предъявленными требованиями.
- На защите лабораторной работы, студент должен ответить на 2-5 вопроса по порядку выполнения работы, выполнить 1-4 практических задания.
- На защите самостоятельной работы студент должен ответить на 3-5 теоретических вопроса.
- Пересдача лабораторной работы или самостоятельной работы назначается в случае, если студент не ориентируется в учебном материале, не может объяснить ход и результаты выполнения работы. Пересдача, как и невыполнение учебного графика, сопровождается потерей 10-30% баллов.
- Контрольные работы по дисциплине проводится по материалам лекций в виде теста, содержащего 10 вопросов, на которые нужно ответить в течение 45 минут. Балльная оценка тестов приведена в таблице.

Правила итоговой аттестации

- К зачёту допускаются студенты, защитившие все лабораторные работы, самостоятельную работу и набравшие не менее 30 баллов по результатам текущего рейтинга.
- Зачёт проводится в письменном виде. В билет входит 3 теоретических вопроса и 2-3 задачи.
- 15-20 баллов выставляется, если все задания выполнены полностью, без серьезных замечаний. 10-14 баллов – если выполнены 3 задания из 5, 5-10 баллов – если выполнены два задания из трех, но с замечаниями.

Правила аттестации

Самостоятельная работа

Написать и оформить в соответствие с требованиями реферат на одну из тем:

- **Архитектура ЭВМ**. Архитектура и принцип работы компьютеров Z4, ENIAC, x86, аналоговых, нейрокомпьютеров, …
- **Квантовый компьютер**. Квантовый компьютер, квантовые вычисления, квантовая элементная база, языки программирования для квантовых компьютеров, …
- **Передача данных**. Понятие данные, информация. Способы передачи данных. Обнаружение и исправление ошибок при передаче. Коллизия и способы ее разрешения. OSI. Протоколы передачи данных. Аппаратура передачи данных.
- **Компьютерные вирусы**. Классификация, способы заражения и обезвреживания. Способы защиты от компьютерных вирусов.
- **Запоминающие устройства**. Способы записи, хранения, считывания и удаления данных на различных носителях. HDD, Flash, DRAM, SRAM, FRAM, EPROM, EEPROM, на ферритовых сердечниках (*core storage*), оптические диски (HD DVD, Blu-Ray, CD), молекулярная память, …
- **Архитектура вычислительных сетей**. Классификация, топология компьютерных сетей. Способы передачи данных. Коллизии и способы их разрешения. Протоколы TCP-IP. OSI, DNS, NetBios, … Аппаратура передачи данных.
- **Алгебра логики**. Двоичная, троичная логика. Математические операции над целыми и вещественными числами. Представление информации в ЭВМ.
- **Операционные системы**. Операционные системы реального времени. Объекты ядра операционной системы. Принципы многопоточного программирования. Архитектура операционной системы Windows, Linux.
- **Аппаратные платформы компьютеров**. Аппаратные платформы i386, x86-64, IA64, Core 2 Duo, Xeon.
- **Алгоритмы сжатия**. Сжатие без потерь. Сжатие с потерями. Преобразование Барроуза-Уилера (*BWT*), Шиндлера (*ST*). Алгоритм Лемпеля — Зива, Prediction by Partial Matching (PPM), вейвлетное сжатие, … Сжатие видео (mpeg, …), аудио (mp3, …), изображений (jpeg, …).
- **Алгоритмы сортировки**. Прямое включение, прямой выбор, прямой обмен, с помощью дерева, с помощью разделения, пирамидальная сортировка, Шелла, Бентли-Седжвика, метод пузырька. Прямое слияние, естественное слияние, сбалансированное многопутевое слияние, многофазная сортировка.

Правила аттестации

Самостоятельная работа

Написать и оформить в соответствие с требованиями реферат на одну из тем:

- **Алгоритмы поиска**. Линейный поиск, двоичный поиск, поиск в таблице, прямой поиск строки. Алгоритм Кнута, Морриса и Пратта, алгоритм Боуера и Мура, Ахо — Корасик, Битапа (Baeza-Yates-Gonnet).
- **Безопасность**. Безопасность современных электронных подписей. Основы защиты информации и сведений, составляющих государственную тайну. Методы защиты информации. ГОСТы.
- **Беспроводные технологии передачи данных.** Wi-Fi, радио-Ethernet, Zig-Bee, Bluetooth, GSM.
- **Технологии производства панелей мониторов.** Преимущества и недостатки, технология изготовления ЖК (TN, PVA, MVA, IPS матриц), стерео, плазменных мониторов.
- **Криптография**. Симметричные ключи (DES, AES, RC4, IDEA, …), асимметричные ключи (RSA, Elgamal, …), хеш функции (MD, SHA, …). Область применения. Безопасность.
- **Цифровое телевидение.** Стандарты цифрового телевидения (DVB, ATSC, ISDB, …). Способы распространения цифрового сигнала. HDTV, DVB-T, HDMI, HDCP, TriplePlay, IPTV, Voice over IP, … Оборудование приема и передачи цифрового вещания.
- **Методы моделирования.** Непрерывно-детерминированные, дискретно-детерминированные, дискретно-стохастические, непрерывно-стохастические, сетевые модели.
- **История развития ЭВМ. Понятие и основные виды архитектуры ЭВМ.** Основные этапы развития вычислительной техники. Принципы работы вычислительной системы. Архитектуры ЭВМ. Состав и назначение основных элементов персонального компьютера. Центральный процессор. Системные шины и слоты расширения.
- **Способы представления информации в ЭВМ.** Способы представления целых и вещественных чисел в ЭВМ. Способы представления графической информации в ЭВМ. Способы представления видео информации в ЭВМ. Способы представления аудио информации в ЭВМ.
- **Печать графической информации.** Способы лазерной, струйной печати графической информации.

Правила аттестации

Самостоятельная работа

Написать и оформить в соответствие с требованиями реферат на одну из тем:

- **Логические основы ЭВМ.** Основные понятия алгебры логики. Логические величины: истина (логическая единица) и ложь (логический ноль). Логические операции: инверсия, дизъюнкция и конъюнкция. Основные законы булевой алгебры. Техническая реализация логических величин. Бистабильная ячейка - триггер. Регистры. Сумматор. Выполнение операций сложения, вычитания и умножения целых чисел. Арифметико-логическое устройство.
- **Основные этапы развития вычислительной техники.** Докомпьютерный период. Создание первого компьютера. Ламповые и транзисторные ЭВМ. Микроэлектронная база ВТ: интегральные схемы малой степени интеграции, БИС и СБИС. Микропроцессоры. Персональные компьютеры (ПК). Классификация ПК. Принципы работы вычислительной системы. Принципы фон Неймана: программного управления, однородности и адресации памяти. Архитектуры ЭВМ. Понятие архитектуры. Процессор, запоминающее устройство (ЗУ). Шина данных, адресная шина и шина команд. Архитектуры с фиксированным набором устройств. Открытые архитектуры.
- **Состав и назначение основных элементов персонального компьютера.** Системный блок и его состав: системная плата, жесткий диск, дисководы, блок питания и другие устройства. Системы ввода-вывода информации: дисплей, клавиатура, мышь, принтер и другие. Устройства на системной плате. Центральный процессор. Основные характеристики микропроцессора. Эволюция микропроцессоров. Процессоры с расширенной и сокращенной системами команд. Характеристики современных микропроцессоров. Системные шины и слоты расширения. Технические характеристики шин.

Правила аттестации

Самостоятельная работа

Написать и оформить в соответствие с требованиями реферат на одну из тем:

- **Запоминающие устройства. Классификация, принцип работы, основные характеристики.** Оперативные и постоянные ЗУ. Адресное пространство ЗУ. Ячейка памяти (ЯП), статические и динамические ЯП. Основные характеристики ЗУ: объем, разрядность, время записи и считывания Техническая реализация модулей памяти. Накопитель на жестком магнитном диске. Принцип работы, основные характеристики. Низкоуровневая структура дисков: дорожки, сектора, цилиндры. Логические диски. Загрузочный сектор, таблицы размещения файлов. Другие накопители на магнитных дисках Накопители на оптических дисках, их технические характеристики. Принципы записи на оптических дисках, их разновидности. Flash-запоминающие устройства.
- **Устройства ввода-вывода данных**. Мониторы. Принципы работы мониторов различных типов. Основные характеристики мониторов. Видеоадаптер: назначение, основные характеристики. Клавиатура, разновидности клавиатур. Манипулятор типа «мышь». Принтеры и сканнеры. Мультимедийный проектор.
- **Программное обеспечение**. Виды программного обеспечения и их характеристики. Системное (базовое, служебное) и прикладное программное обеспечение (ПО). Пакеты прикладных программ (ППП). Общие и специализированные ППП. Универсальные пакеты инженерных и научных расчетов. Отраслевые специализированные пакеты. Системы автоматизированного проектирования.

Правила аттестации

Самостоятельная работа

Написать и оформить в соответствие с требованиями реферат на одну из тем:

- **Понятие и назначение операционной системы**. Определение операционной системы (ОС). Функции ОС. Классификация ОС. Эволюция ОС Windows. Концепции графического интерфейса Windows: рабочий стол, окно, объект. Стандартные программы и служебные утилиты: восстановление системы, очистка и дефрагментация дисков, архивация данных. Антивирусные программы. Использование справки. Другие операционные системы.
- **Файловые структуры**. Понятие файловой системы. Функции файловой системы. Примеры файловых систем: FAT, NTFS. Имена и расширения файлов, каталоги и подкаталоги (папки). Форматы и атрибуты файлов. Файловые менеджеры. Копирование, перенос, удаление и переименование файлов средствами Windows и файловыми менеджерами. Архивация файлов.
- **Основы машинной графики.** Представление графической информации. Векторная и растровая графика. Цветовые модели RGB и CMYK. Программные пакеты для работы с векторной и растровой графикой. Средства технической и научной графики. Форматы графических файлов.
- **Моделирование как метод познания**. Классификация и формы представления моделей. Методы и технологии моделирования. Информационная модель объекта.
- **Принципы построения сетей**. Компоненты вычислительных сетей. Коммуникационное оборудование. Средства использования сетевых сервисов.

Правила аттестации

Самостоятельная работа

Написать и оформить в соответствие с требованиями реферат на одну из тем:

- **Защита информации в локальных и глобальных компьютерных сетях**. Основные понятия информационной безопасности: конфиденциальность, целостность, достоверность информации; доступ, санкционированный и несанкционированный. Угрозы безопасности информации и их классификация. Юридические основы информационной безопасности: понятие компьютерного преступления, соответствующие статьи УК. Объекты нападения; виды компьютерных преступлений. Компьютерные вирусы: классификация, каналы распространения, локализация, проявления действий. Критерии защищенности компьютерных систем. Организационные, инженерно-технические и другие меры защиты информации. Брандмауэр. Методы ограничения доступа. Мониторинг несанкционированных действий.
- **Шифрование данных.** Криптографические методы защиты данных. Методы шифрования: заменой, перестановкой, с использованием ключей и хеш-функций. Шифрование данных в Windows. Электронная цифровая подпись электронных документов. Электронная сертификация.

Альтернатива реферату:

- Разработать сайт группы. Рассказать о каждом студенте группы. Фото со встреч. …

Правила аттестации

Самостоятельная работа

Требования к оформлению реферата:

- Формат текста: Microsoft Word.
- Формат страницы: А 4 (210*297 мм).
- Поля: сверху: 20мм, снизу: 20мм, справа: 15мм, слева: 30мм;
- Шрифт: размер (кегль) – 14; тип – Times New Roman; абзацные отступы 1.5.
- Выравнивание: по ширине
- Межстрочный интервал – 1.5.
- В тексте допускаются рисунки, графики, таблицы.
- Рисунки, графики, схемы должны выполняться в графических редакторах, поддерживающих векторную графику; таблица - в режиме таблиц.
- Все рисунки, графики, таблицы должны быть пронумерованы и иметь название.
- Название глав печатается прописными буквами, шрифт – жирный.
- После отступа в 1.5 интервала следует текст, печатаемый через полуторный интервал.
- В заголовке недопустимы переносы, точка в конце не ставится.
- Список использованной литературы озаглавливается словом литература, набранным жирным шрифтом 14 кеглем и расположенным посередине.

Содержание должно оформиться автоматически средствами Microsoft Word. Все рисунки, таблицы, схемы должны быть пронумерованы и иметь название. Ссылки на рисунки, таблицы, схемы и список литературы должны быть построены с применением возможностей Microsoft Word.

Основные понятия и определения

Термин информатика возник в 60-х гг. во Франции для названия области, занимающейся автоматизированной обработкой информации с помощью электронных вычислительных машин. Французский термин informatigue (информатика) образован путем слияния слов information (информация) и automatigue (автоматика) и означает "информационная автоматика или автоматизированная переработка информации". В англоязычных странах этому термину соответствует синоним computer science (наука о компьютерной технике).

- ***Информация*** — сведения об объектах и явлениях окружающей среды, их параметрах, свойствах и состоянии, которые уменьшают имеющуюся о них степень неопределенности, неполноты знаний.
- ***Информатика*** — это область человеческой деятельности, связанная с процессами преобразования информации с помощью компьютеров и их взаимодействием со средой применения.
- ***Информационная культура*** — умение целенаправленно работать с информацией и использовать для ее получения, обработки и передачи компьютерную информационную технологию, современные технические средства и методы.
- ***Информационные ресурсы*** — отдельные документы и отдельные массивы документов, документы и массивы документов в информационных системах (библиотеках, архивах, фондах, банках данных и пр.).
- ***Информационный продукт*** — совокупность данных, сформированная производителем для распространения в вещественной или невещественной форме.
- ***Услуга*** — результат непроизводственной деятельности предприятия или лица, направленный на удовлетворение потребности человека или организации в использовании различных продуктов.
- ***Информационная услуга*** — получение и предоставление в распоряжение пользователя информационных продуктов.
- ***База данных*** — совокупность связанных данных, правила организации которых основаны на общих принципах описания, хранения и манипулирования данными.
- ***Информационный рынок*** — система экономических, правовых и организационных отношений по торговле продуктами интеллектуального труда на коммерческой основе.
- ***Инфраструктура информационного рынка*** — совокупность секторов, каждый из которых объединяет группу людей или организаций, предлагающих однородные информационные продукты и услуги.

Основные понятия и определения

- **Меры информации**

Для измерения информации используются два параметра:

- количество информации
- объем данных.

Эти параметры имеют разные выражения и интерпретацию в зависимости от рассматриваемой формы адекватности. Каждой форме адекватности соответствует своя мера количества информации и объема данных.

Различают следующие меры информации:

- ✔ **Синтаксическая мера информации;**
- ✔ **Семантическая мера информации;**
- ✔ **Прагматическая мера информации.**

Мера информации

✔ Синтаксическая мера информации

Синтаксическая мера информации – это мера количества информации, которая оперирует с обезличенной информацией, не выражающей смыслового отношения к объекту.

- **Объем данных Vд** в сообщении измеряется количеством символов (разрядов) в этом сообщении. В различных системах счисления один разряд имеет различный вес и соответственно меняется единица измерения данных:
- **Количество информации на синтаксическом** уровне невозможно определить без рассмотрения понятия неопределенности состояния системы (энтропии системы). Действительно, получение информации о какой-либо системе всегда связано с изменением степени неосведомленности получателя о состоянии этой системы. Рассмотрим это понятие.

Пример: Пусть до получения информации потребитель имеет некоторые предварительные (априорные) сведения о системе a. Мерой его неосведомленности о системе является функция H(a), которая в то же время служит и мерой неопределенности состояния системы. После получения некоторого сообщения b получатель приобрел некоторую дополнительную информацию Ib(a), уменьшившую его априорную неосведомленность так, что апостериорная (после получения сообщения b) неопределенность состояния системы стала Hb(a). Тогда количество информации Ib(a) о системе, полученной в сообщении b, определится как Ib(a)=H(a)-Hb(a)

Мера информации

✔ Семантическая мера информации

Семантическая мера - отображает смысловое содержания информации. Для измерения смыслового содержания информации, т.е. ее количества на семантическом уровне, наибольшее признание получила *тезаурусная мера*, которая связывает семантические свойства информации со способностью пользователя принимать поступившее сообщение. Для этого используется понятие тезаурус пользователя.

Тезаурус — это совокупность сведений, которыми располагает пользователь или система.

Пример. Количество семантической информации в сообщении, количество новых знаний, получаемых пользователем, является величиной относительной. Одно и то же сообщение может иметь смысловое содержание для компетентного пользователя и быть бессмысленным (семантический шум) для пользователя некомпетентного. Относительной мерой количества семантической информации может служить коэффициент содержательности С, который определяется как отношение количества семантической информации к ее объему: C=Ic/Vd

Мера информации

✔ Прагматическая мера информации

Прагматическая мера информации - эта мера определяет полезность информации (ценность) для достижения пользователем поставленной цели. Эта мера также величина относительная, обусловленная особенностями использования этой информации в той или иной системе. Ценность информации целесообразно измерять в тех же самых единицах (или близких к ним), в которых измеряется целевая функция.

Показатели качества информации

Репрезентативность информации связана с правильностью ее отбора и формирования в целях адекватного отражения свойств объекта. Важнейшее значение здесь имеют:

- правильность концепции, на базе которой сформулировано исходное понятие;
- обоснованность отбора существенных признаков и связей отображаемого явления.

Содержательность информации отражает семантическую емкость, равную отношению количества семантической информации в сообщении к объему обрабатываемых данных, т.е. $C = Ic/Vd$

С увеличением содержательности информации растет семантическая пропускная способность информационной системы, так как для получения одних и тех же сведений требуется преобразовать меньший объем данных.

Коэффициент информативности, характеризующийся отношением количества синтаксической информации (по Шеннону) к объему данных $Y = I/Vd$.

Достаточность (полнота) информации означает, что она содержит минимальный, но достаточный для принятия правильного решения состав (набор показателей). Понятие полноты информации связано с ее смысловым содержанием (семантикой) и прагматикой. Как неполная, т.е. недостаточная для принятия правильного решения, так и избыточная информация снижает эффективность принимаемых пользователем решений.

Доступность информации восприятию пользователя обеспечивается выполнением соответствующих процедур ее получения и преобразования. Например, в информационной системе информация преобразовывается к доступной и удобной для восприятия пользователя форме. Это достигается, в частности, и путем согласования ее семантической формы с тезаурусом пользователя.

Актуальность информации определяется степенью сохранения ценности информации для управления в момент ее использования и зависит от динамики изменения ее характеристик и от интервала времени, прошедшего с момента возникновения данной информации.

Своевременность информации означает ее поступление не позже заранее назначенного момента времени, согласованного с временем решения поставленной задачи.

Показатели качества информации

Точность информации определяется степенью близости получаемой информации к реальному состоянию объекта, процесса, явления и т.п. Для информации, отображаемой цифровым кодом, известны четыре классификационных понятия точности:

- формальная точность, измеряемая значением единицы младшего разряда числа;
- реальная точность, определяемая значением единицы последнего разряда числа, верность которого гарантируется;
- максимальная точность, которую можно получить в конкретных условиях функционирования системы;
- необходимая точность, определяемая функциональным назначением показателя.

Достоверность информации определяется ее свойством отражать реально существующие объекты с необходимой точностью. Измеряется достоверность информации доверительной вероятностью необходимой точности, т.е. вероятностью того, что отображаемое информацией значение параметра отличается от истинного значения этого параметра в пределах необходимой точности.

Устойчивость информации отражает ее способность реагировать на изменения исходных данных без нарушения необходимой точности. Устойчивость информации, как и репрезентативность, обусловлена выбранной методикой ее отбора и формирования.

Такие параметры качества *информации*, как *репрезентативность*, *содержательность*, *достаточность*, *доступность*, *устойчивость*, целиком определяются на методическом уровне разработки информационных систем.
Параметры *актуальности*, *своевременности*, *точности* и *достоверности* обусловливаются в большей степени также на методическом уровне, однако на их величину существенно влияет и характер функционирования системы, в первую очередь ее надежность. При этом параметры актуальности и точности жестко связаны соответственно с параметрами своевременности и до стоверности.

История развития компьютеров

- **4000 - 1300 до н.э.** Представители первой известной шумерской цивилизации записывают информацию на покрытых глиной дощечках
- **3000 до н. э.** В Вавилоне изобретены счеты
- **1612 - 1614** Джон Непер ввел десятичную точку, сформулировал понятие логарифма и использовал пронумерованные палочки ("кости Непера") для вычислений
- **1623** Вильгельм Шикард изобрел "вычисляющие часы" с зубчатым колесным механизмом, которые помогают умножать многоразрядные числа
- **1642 - 1643** Блез Паскаль создал первый механический сумматор на основе зубчатых колес.
- **1666 -** В Англии Сэмюэль Морланд изобрел механический калькулятор, который мог складывать и вычитать.
- **1674** - Годфрид Лейбниц построил «Шаговый счетчик» - калькулятор, использующий ступенчатую передачу.
- **1774** - Филип Мэтьюз построил и продал несколько вычислительных устройств, обладавших точностью до 12 знаков.
- **1777 -** Граф Стэхоупский III изобрел умножающий калькулятор.
- **1786 -** Дж. Мюллер придумал и описал «Разностную машину", но не смог найти средств, чтобы построить ее.
- **1801 -** В ткацком станке Джозефа-Марии Жакарда для управления процессом изготовления ткани использовалась связанная последовательность перфорированных карт.
- **1820 -** Арифмометр Томаса, созданный на основе разработанного Лейбницем принципа шагового барабана, продемонстрирован на заседании Французской Академии Наук, после чего стал первым калькулятором, имевшим коммерческий успех.
- **1822 -** Чарльз Беббидж приступил к созданию «Разностной машины"
- **1829 -** Уильям Остин Барт запатентовал еще довольно громоздкую, но работающую пишущую машинку, первую в Америке.
- **1834-35** Беббидж приступил к проектированию «Аналитической машины».
- **1842-43** Августа Ада, графиня Лавлейс, транслировала памфлет Луиджи Менабра на «Аналитической машине», присовокупив собственный комментарий
- **1854 -** Джордж Буль опубликовал "Исследование законов мышления", где привел описание системы символических и логических рассуждений, ставшей фундаментом для компьютерного проектирования
- **1858** - Проложен телеграфный кабель через Атлантический океан с обеспечением сервиса в течение нескольких дней
- **1861** - Трансконтинентальная телеграфная линия соединила восточное и западное побережья Америки.

История развития компьютеров

- **1876** Александер Грэм Белл изобрел и запатентовал телефон
- **1882** Уильям Бэрроуз оставил работу банковского клерка и посвятил себя изобретению счетной машины
- **1889** По итогам специального конкурса, электрические табуляторы Германа Холлеритабыли выбраны для использования в переписи населения 1890 года
- **1893** Изобретен первый калькулятор с четырьмя функциями
- **1901** Появился клавишный перфоратор
- **1904** Джон Флеминг запатентовал диодную электровакуумную лампу
- **1906** На основе диода Флеминга Ли де Форест создал триодную электровакуумную лампу
- **1908** Британский ученый Кемпбел Свинтон описал метод электронной развертки, что знаменовало использование электронно-лучевой трубки в телевизионных приемниках
- **1915** Физик Мэнсон Бенедикс обнаружил, что германиевый кристалл может быть использован для преобразования переменного тока в постоянный
- **1920-1921** Слово «Робот» (производное от чешского термина для обозначения принудительного труда) впервые использовано Карелом Чапеком в его пьесе "РУР - Универсальные Роботы Россума"
- **1929** Переданы первые сигналы цветного телевещания
- **1931** Преподаватель из Мичигана Рейнолд Джонсон изобрел способ подсчета ответов на «Программируемые» тесты с помощью отметок проводящего карандаша на таблице-панели; вскоре IBM купила эту технологию
- **1935** IBM разработала счетную умножающую машину "601" на перфокартах и электрическую пишущую машинку
- **1936** В Германии Конрад Зусе приходит к выводу, что программы, состоящие из битовых комбинаций, можно запоминать; он подает заявку на патентование метода автоматического выполнения вычислений с использованием "памяти комбинаций"
- **1937** Клод Шеннон публикует работу о принципах построения двоичного электрического сумматора. Джордж Стибитц разработал двоичную схему на основе Булевой алгебры. Говард Эйкен предложил IBM разработать цифровую вычислительную машину, способную выполнять четыре фундаментальных арифметических операции в определенном порядке. В статье Алана Тьюринга "О вычислимых числах" введено понятие Машины Тьюринга.

История развития компьютеров

- **1938** Уильям Хьюлетт и Дэвид Паккард основали Hewlett-Packard в гараже в Пало-Альто К. Зусе завершил построение электромеханического двоичного компьютера Z1
- **1939** Джон Винсент Атанасофф вместе со своим студентом Клиффордом Берри построил прототип электронного цифрового компьютера с использованием двоичной арифметики.
- **1940** К.Зусе завершил разработку Z2, который использовал телефонные реле вместо механических логических схем
- **1941** К.Зусе завершил разработку Z3: первая полностью функциональная программа исполнялась на электромеханическом цифровом компьютере
- **1943** В Moore School of Electrical Engineering в Филадельфии 31 мая началась работа по конструированию «ENIAC»
- **1944** В Гарвардском университете 7 августа открыта построенная Говардом Эйкеном машина Howard Mark (также известная как IBM ASCC - Automatic Sequence Controlled Calculator)
- **1945** «ENIAC» построена и начала работать Джон фон Нейман (John von Neumann) в своем докладе по проектированию «EDVAC» (Electronic Discrete Variable Automatic Computer) ввел понятие запоминаемой программ. В процессе работы над прототипом машины Mark II Грейс Мюррей Хоппер обнаружила первую компьютерную ошибку (bug), вызвавшую сбой в работе реле
- **1946** В Университете Пенсильвании введена в действие «ENIA» Макли, Артур Баркс, Герман Голдстайн и Джон фон Нейман опубликовали работу "Предварительное обсуждение логического проектирования электронного вычислительного инструмента". В Американском институте инженеров электротехники (IEE) основан подкомитет по большим вычислительным устройствам
- **1947** Джон Бардин и Уолтер Бреттейн сообщили руководству «Bell Labs», что они вместе с Уильямом Шокли разработали первый транзистор
- **1948** В качестве компьютерного запоминающего устройства введены накопители на магнитном барабане Клод Шеннон опубликовал "Математическую теорию связи", заложив таким образом основу современного понимания коммуникационных процессов Начал работать SSEC (Selective Sequence Electronic Calculator) - компьютер, использующий как электронику, так и реле Введен в действие Manchester Mark I (известный как "baby machine") - первый цифровой компьютер с запоминаемой программой на основе электровакуумных ламп. Ричард Хемминг сформулировал способ обнаружения и корректировки ошибок в блоках данных

История развития компьютеров

- **1949** Whirlwind ("Вихрь"), первый компьютер для решения задач реального времени, был разработан под руководством Джея Форрестера в Массачусетском технологическом институте (MIT); он содержал около 5000 электровакуумных ламп Разработанный Джоном Макли "Short Order Code" можно считать первым языком программирования высокого уровня
- **1951** Первый компьютер Univac I установлен в Американском агентстве по переписи населения. Джей Форрестер подал заявку на патентование матричной памяти на магнитных сердечниках Уильям Шокли изобрел транзистор с p-n переходом (junction transistor) Дэвид Уилер, Морис Уилкс и Стенли Джилл ввели понятие подпрограммы и предложили "Wheeler jump" в качестве средства для реализации этого понятия Бетти Холбертон создает генератор "сортировка/слияние" - предшественник компилятора
- **1952** Грейс Мюррей Хоппер разработал "A-O" - первый компилятор. Введены в действие два компьютера на основе фон-неймановской архитектуры: Illiac I в Университете Иллинойса и Ordvac, специально построенный для военных нужд В ходе телетрансляции компьютер Univac I предсказал исход американских президентских выборов, что вызвало всплеск интереса к компьютерным технологиям. Выпущен IBM 701, получивший известность под названием Defence Calculator
- **1953** Начинает выпускаться «IBM 650», известный как «Magnetic Drum Calculator». Получил признание как первый коммерческий компьютер
- **1954** UniPrinter, разработанный Эрлом Мастерсоном, печатает со скоростью 600 строк в минут Texas Instrument объявила о выпуске кремниевого транзистора. Univac 1103A стала первой коммерческой вычислительной машиной с памятью на ферритовых сердечниках
- **1957** Джон Бэкус и его сотрудники установили в IBM (Westinghouse) первый компилятор языка Фортран Один из первых компьютеров на транзисторах Atlas Guidance Computer, выпущенный фирмой Burroughs, нашел применение при управлении запуском ракеты Atlas Гордон Мур, Роберт Нойс и другие основали компанию Fairchild Semiconductor Зарегистрирована фирма Control Data 4 В России запущен первый спутник: началась "космическая гонка"
- **1958** Основана Digital Equipment Corp
- **1959** С целью создания языка Cobol (Common Business Oriented Language) основан Комитет Codasyl (The Committee on Data Systems Languages Джон Маккарти разработал LISP (list processing) - язык для использования в задачах искусственного интеллекта На выставке в Париже продемонстрирован первый японский коммерческий компьютер на транзисторах NEAC 2201, выпущенный компанией NEC Роберт Нойси Гордон Мур) от имени Fairchild Semiconductor подали заявку на патентование технологии интегральных схем ЮНЕСКО спонсировала первую крупномасштабную международную конференцию по компьютерным технологиям

История развития компьютеров

- **1960** В результате совместной работы европейских и американских ученых разработан стандарт для Algol 60 В Корнелльском Университете Фрэнк Розенблатт построил компьютерное устройство Perceptron, способное к обучению на основе метода проб и ошибок с использованием нейронных сетей Компания DEC объявила о выпуске PDP-1, первого коммерческого компьютера с монитором и клавиатурным вводом
- **1961** В Массачусетсском технологическом институте (MIT) Фернандо Корбато (Fernando Corbato) разработал принципы разделения времени при коллективном доступе пользователей к вычислительной системе Компьютер IBM 7030 показал в 30 раз большую производительность по сравнению с моделью 704, что подстегнуло работы в области суперкомпьютеров
- **1962** Первые кафедры информатики (Computer Science) основаны в Университетах Стенфорда и Пэдью) Росс Перо основал Electronic Data Systems - компанию, которой суждено было стать крупнейшей в мире в области компьютерного сервиса Первая видеоигра изобретена студентом выпускного курса Массачусетсского технологического института (MIT) Стивом Расселом; очень скоро в нее стали играть во всех компьютерных лабораториях США В Англии создан суперкомпьютер Atlas - самый мощный в мире; среди задействованных в нем новых решений - виртуальная память и конвейерные операции
- **1963** В MIT Иван Сазерленд разработал систему Sketchpad, положившую начало эре компьютерной графики ANSI - Американский институт национальных стандартов - сертифицировал код ASCII 7
- **1964** IBM объявила о производстве серии компьютеров "третьего поколения" - System/360 В Дартмутском колледже Джон Кемени) и Томас Куртц разработали BASIC (Beginner's All Purpose Symbolic Instruction Code) IBM завершила семилетний проект Sabre, предназначенный для автоматизации бронирования мест при авиаперевозках Разработанный Сеймуром Креем и выпущенный компанией Control Data, CDC 6600 стал первым суперкомпьютером, который имел успех на рынке IBM разработала систему автоматизированного проектирования Даг Энгелбарт изобрел компьютерную мышь
- **1965** Компания DEC выпустила PDP-8 - первый миникомпьютер на базе транзисторных схем Project MAC - проект по созданию системы разделения времени - привел к построению операционной системы Multics Морис Уилкс на основе идеи Гордона Скеротта предложил технологию кэш-памяти

История развития компьютеров

- **1967** В Норвежском вычислительном центре Оле-Йохан Дал и Кристен Нигаард завершили работу над Simula - первым объектно-ориентированным языком Джек Килби, Джерри Меримен и Джеймс ван Тассел изобрели карманный калькулятор с четырьмя функциями Дональд Кнут в своем фундаментальном труде впервые систематически описывает алгоритмы и структуры данных отдельно от программ, в которых они могут быть использованы
- **1968** Конференция под эгидой Научного комитета НАТО идентифицирует кризис разработки ПО (Software crisis) и вводит термин программной инженерии (software engineering) Эдсгар Дейкстра обосновал вредоносность оператора "goto", что явилось началом эры структурного программирования Фирма Burroughs представила B2500 и B3500 - первые компьютеры на интегральных схемах Введен в действие Федеральный стандарт обработки информации, использующий шестизначный формат данных YYMMDD; так была заложена основа проблемы 2000 года Сеймур Крей разработал CDC 7600 - суперкомпьютер с производительностью 40 MFLOPS Роберт Нойс, Энди Гроув и Гордон Мур (Gordon Moore) основали корпорацию Intel
- **1969** Bell Labs прекратила работу в рамках Project MAC и приступила к созданию Unix. Введен в действие стандарт RS-232-C, облегчающий обмен данными между компьютером и периферийными устройствами Министерство обороны США начинает финансирование программы исследований сетевых технологий в рамках сети ARPANET с четырьмя первыми узлами в Университете UCLA, Калифорнийском университете в Санта-Барбара, Университете Юта и SRI
- **1970** Деннис Ричи и Кеннет Томсон в Bell Labs разработали Unix Информационный бюллетень The Computer Group News преобразован в Computer - ежемесячный журнал для всех членов IEEE Computer Society, которое формально образовалось из IEEE Computer Group В Стэнфордском университете фирма Xerox основала Palo Alto Research Center (PARC), для проведения исследований в области компьютерных технологий Е.Ф. Кодд выполнил описание реляционной модели. Появились гибкие ("floppy") диски
- **1971** Тед Хофф, С. Мэзор и Ф. Фэджин разработали микропроцессор Intel 4004 - первый "компьютер на чипе" Рой Томлисон из "Bolt Beranek and Newman" послал первое сообщение по электронной почте Никлаус Вирт разработал язык Pascal

История развития компьютеров

- **1972** Появился Intel 8008, первый 8-разрядный микропроцессор Видеоигра Pong, разработанная Ноланом Бушнеллом, стала настолько популярной, что он основал Atari - первую фирму для коммерческого производства компьютерных игр На основе идей Алана Кея в Xerox PARC разработан язык Smalltalk. Деннисом Ритчи был создан язык Си (его предшественник носил имя "B"). В Университете Марселя Элейн Колмерю разработал Prolog, ставший первым популярным языком логического программирования. Появились компьютеры PDP 11/45 фирмы DEC. Стив Возняк построил и принялся продавать прямо в общежитии Университета Беркли свой нашумевший "Голубой ящик" (blue box) - тоновый генератор, который позволял делать бесплатные телефонные звонки
- **1973** В Xerox PARC разработан экспериментальный ПК, названный Alto, в котором применялась мышь, Ethernet и графический интерфейс пользователя В Стэнфордском Университете под руководством Винтона Серфа началась работа над Transmission Control Protocol (TCP)
- **1974** Чарльз Симони в центре Xerox PARC написал первое приложение с использованием принципа WYSIWYG (What You See Is What You Get)
- **1975** Первый ПК - Altair 8800, доступный для потребителя в виде комплекта компонентов, появился на обложке январского выпуска журнала Popular Electronic В IBM Джон Кокке начал работу над проектом 801 с целью создания миникомпьютера на архитектуре, названной позже RISC Фредерик Брукс опубликовал книгу "Мифический Человеко-месяц" ("The Mythical Man Month"), в которой охарактеризовал разработку ПО как "жестокую схватку с ужасным чудовищем в яме со смолой". Один из выводов - "если проект не укладывается в сроки, то добавление новых разработчиков лишь усугубит положение"
- **1976** Фирма Cray Research выпустила Cray-1 - первый суперкомпьютер с векторной архитектурой Гэри Килдэлл разработал операционную систему CP/M для 8-разрядных ПК Стив Джобс и Стив Возняк спроектировали и построили - на основе монтажных плат - компьютер Apple 1
- **1977** Стив Джобс и Стив Возняк 3 января учредили Apple Computer; весной объявлено о компьютере Apple II, который стал своего рода эталоном для персональных компьютеров Билл Гейтс и Пол Аллен основали Microsoft
- **1978** DEC выпустила VAX 11/780 - 32-разрядный компьютер, ставший популярным в научных и технических приложениях
- **1979** Motorola выпустила микропроцессор 68000, который позднее поддержал компьютеры Macintosh В результате разработок Sony и Philips появились цифровые видеодиски

История развития компьютеров

- **1980** IBM выбрала MS DOS от Microsoft в качестве операционной системы для своих ПК В результате разработок по заказу Министерства обороны США создан язык Ada, предназначенный для встроенных приложений и управления процессами в реальном времени Уэйн Рэтлифф разработал dBase II - первую СУБД для ПК Создан первый "портативный" переносной компьютер Osborne 1 весом 24 фунта и размером с чемодан. В Университете Беркли Дэвид Паттерсон ввел в употребление термин "сокращенный набор команд" (reduced-instruction set)
- **1981** Началось массовое производство IBM PC с открытой архитектурой, что послужило корпоративной Америке сигналом о начале рыночной экспансии настольных компьютеров
- **1982** Columbia Data Products выпустила первый "клон" компьютеров IBM PC; вскоре по тому же пути последовала Compaq Журнал Time назвал компьютер "Человеком Года". Cray X-MP - система из двух параллельно работающих компьютеров Cray-1 - оказалась в три раза быстрее, чем Cray-1 Японские разработчики приступили к проекту "компьютерные системы пятого поколения", ориентируясь на принципы искусственного интеллекта Коммерческий сервис электронной почты охватил 25 городов США Compaq выпустила портативный IBM PC
- **1983** Переход к TCP/IP ознаменовал начало эпохи создания глобальной "сети сетей" Как провозвестники грядущего бума, связанного с системами параллельных вычислений, основаны фирмы Thinking Machines и Ncube В Bell Labs концерна AT&T Бьерн Страуструп разработал язык C++ - объектное расширение Си
- **1984** Устройства CD-ROM, введенные компаниями Sony и Philips, обеспечили значительное увеличение объема памяти для цифровых данных. Motorola выпустила MC68020 на 250 000 транзисторах. Писатель Уильям Гибсон в романе "Neuromancer" впервые использовал термин "киберпространство" (cyberspace). Intel приступила к производству чипов 16-bit 80286
- **1985** С выпуском компьютера Cray 2 и параллельного процессора Connection Machine фирмы Thinking Machine производительность суперкомпьютеров достигла 1 млрд. операций в секунду. Фирма Inmos представила транспьютеры на основе специальной архитектуры параллельной обработки. С разработкой Windows 1.0 компания Microsoft привнесла в DOS-совместимые компьютеры особенности, до того присущие только компьютерам Macintosh. Intel начала выпускать чип 80386 с 32-битовой обработкой и встроенным управлением памятью
- **1986** Статья в Wall Street Journal сыграла свою роль в популяризации понятия (и термина!) CASE (Computer-Aided Software Engineering)

История развития компьютеров

- **1988** Выпущенная фирмой Motorola серия 88000 32-разрядных RISC микропроцессоров позволила довести скорость обработки до 17 млн. операций в секунду. Студент выпускного курса Роберт Моррис младший, запустив разработанную им вирусную (worm) программу в Internet, обратил всеобщее внимание на проблему обеспечения безопасности сетей
- **1989** Тим Бернерс-Ли предложил CERN (Европейскому совету ядерных исследований) проект World Wide Web. Intel представила чип 80486 на 1,2 млн. транзисторов. Сеймур Крей основал фирму Cray Computer и приступил к разработке Cray3, использующей чипы на арсениде галлия. Выпущен первый набор эталонных тестов SPEC benchmarks, который облегчил сравнение производительности компьютеров для вычислительных задач в научных приложениях
- **1990** В мае Microsoft представила Windows 3.0, что дало новый импульс юридическому спору с Apple относительно "сходства" реализованных принципов "look and feel" с решениями в операционной системе Macintosh. Hewlett-Packard и IBM приступили к производству компьютеров с RISC-архитектурой. Тим Бернерс-Ли написал прототип для World Wide Web с использованием других его инноваций - URL, HTML и HTTP
- **1991** Японское Министерство труда и промышленности принимает решение о прекращении программы по компьютерам пятого поколения; вместо этого запланировано приступить к разработке компьютеров шестого поколения на основе нейронных сетей. Cray Research представила компьютер Cray Y-MP C90 производительностью в 16 GFLOPS на 16 процессорах. Альянс IBM, Motorola и Apple по разработке микропроцессора на PowerPC
- **1993** Intel представила Pentium. В Национальном Центре суперкомпьютерных приложений Иллинойского университета студенты и инженеры создали NCSA Mosaic - графический интерфейс пользователя для навигации в WWW
- **1994** В апреле Джим Кларк и Марк Андриссен основали Netscape Communications (первоначально - Mosaic Communications). В университете Южной Калифорнии Леонард Адлеман продемонстрировал, что ДНК может быть использована как вычислительное средство Стал доступным первый браузер Netscape, и число пользователей WWW начало стремительно расти
- **1995** На экраны вышла "Игрушечная история" - первый полнометражный фильм, целиком сделанный при помощи компьютеров Появился язык Java, призванный обеспечить создание платформенно-независимых приложений; первым Java-апплетом стал Duke

История развития компьютеров

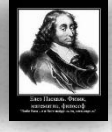
- **1996** Intel объявил о выпуске Pentium Pro. Выход ОС Windows NT 4.
- **1997** Первая победа в шахматах компьютерной программы в матче с сильнейшим гроссмейстером, Deep Blue побеждает Гарри Каспарова (2 победы, 3 ничьих и 1 поражение). Выход ОС Mac OS 8
- **1998** Выпуск моноблочного ПК iMac; выход ОС Windows 98.
- **1999** выход версии 2.2 ядра ОС Linux.
- **2000** выход ОС Windows 2000 и ОС Windows Me. Запуск проектов UDDI и ebXML, направленных на интеграцию электронного бизнеса в мировом масштабе.
- **2001** Выход ОС Windows XP. Появились гибкие дисплеи.
- 2002 Выпуск серверной ОС Windows Server 2003. Разработан Earth Simulator — самый быстрый суперкомпьютер с 2002 по 2004 год: NEC для японского агентства аэрокосмических исследований
- **2003** Разработан стереоскопический 3D-дисплей.
- **2004** Создан полевой транзистор на углеродной нанотрубке: Infineon.
- **2005** Разработан прототип полевого транзистора на одной молекуле.
- **2006** Разработан терагерцовый транзистор; Разработан эмиссионный дисплей на углеродных нанотрубках;
- **2007** Суперкомпьютер Blue Gene/P производительностью 1 петафлопс (квадриллион операций в секунду). Появились компьютерные системы распознавания лиц, превосходящие возможности человека.
- **2008** выход ультрапортативного ноутбука MacBook Air и цифрового сетевого мультимедийного проигрывателя Apple TV. Выпуск ОС Windows Server 2008. Суперкомпьютер IBM Roadrunner превысил производительность в 1 петафлоп (квадриллион операций в секунду) и стал самым быстрым компьютером в мире.
- **2009** Корпорация Oracle покупает Sun Microsystems. Microsoft выпускает ОС Windows 7. Суперкомпьютер Cray XT5 (Jaguar) стал самой производительной в мире компьютерной системой.

Основные этапы развития компьютеров

Год	Название	Создатель	Примечание
1834	Аналитическая машина	Бэббидж	Первая попытка построить цифровой компьютер
1936	Z1	Зус	Первая релейная вычислительная машина
1943	COLOSSUS	Британия	Первый электронный компьютер
1944	Mark1	Айкен	Первый американский многоцелевой компьютер
1946	ENIAC I	Экерт/Моушли	Отсюда начинается история современных компьютеров
1949	EDSAC	Уилкс	Первый компьютер с программами, хранящимися в памяти
1951	Whirlwind I	МТИ	Первый компьютер реального времени
1952	IAS	Фон Нейман	Этот проект используется в большинстве современных компьютеров
1960	PDP-1	DEC	Первый мини-компьютер (продано 50 экземпляров)
1961	1401	IBM	Очень популярный маленький компьютер
1962	7094	IBM	Очень популярная небольшая вычислительная машина
1963	B5000	Burroughs	Первая машина, разработанная для языка высокого уровня
1964	360	IBM	Первое семейство компьютеров
1964	6600	CDC	Первый суперкомпьютер для научных расчетов
1965	PDP-8	DEC	Первый мини-компьютер массового потребления (продано 50 000 экземпляров)
1970	PDP-11	DEC	Эти мини-компьютеры доминировали на компьютерном рынке в 70-е годы.
1974	8080	Intel	Первый универсальный 8-битный компьютер на микросхеме
1974	CRAY-1	Cray	Первый векторный супер-компьютер
1978	VAX	DEC	Первый 32-битный суперминикомпьютер
1981	IBM PC	IBM	Началась эра современных персональных Компьютеров
1985	MIPS	MIPS	Первый компьютер RISC
1987	SPARC	Sun	Первая рабочая станция RISC на основе процессора SPARC
1990	RS6000	IBM	Первый суперскалярный компьютер

Поколения компьютеров

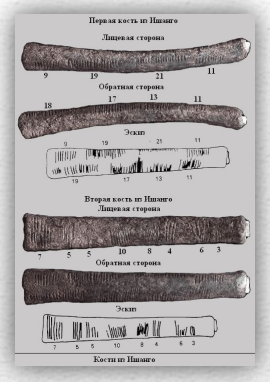
- **«Нулевое»** – до 1940 года.
  - ✔ Вычислительный элемент – механический.
  - ✔ Арифмометры, механические счетные машины.
  - ✔ Простые арифметические операции.
- **«Первое»** – 1940-1950.
  - ✔ Вычислительный элемент – электронные лампы.
  - ✔ Быстродействие – до нескольких десятков тысяч операций в секунду.
  - ✔ «Большие» ЭВМ.
- **«Второе»** –1950-1964.
  - ✔ Вычислительный элемент – транзисторы.
  - ✔ Быстродействие – до 1-2 млн. операций в секунду.
  - ✔ Мини-ЭВМ.
- **«Третье»** – 1964-1971.
  - ✔ Вычислительный элемент – сверхинтегральные схемы.
  - ✔ Быстродействие – до 300 млн. операций в секунду. Микро-ЭВМ, предназначенные для работы с одним пользователем. Первые операционные системы.
- **«Четвертое»** – 1971-…
  - ✔ Вычислительный элемент – микропроцессоры.
  - ✔ Быстродействие миллиарды операций в секунду.
  - ✔ Персональные ЭВМ. Готовые прикладные программы, графический интерфейс, использование технологий мультимедиа. Глобальные компьютерные сети.
- **«Пятое»** – (?) –
  - ✔ Нанотехнологии.
  - ✔ Компьютеры на основе отдельных молекул и даже атомов. Нейросети, моделирующие структуру нервной системы человека. «Биологические компьютеры».

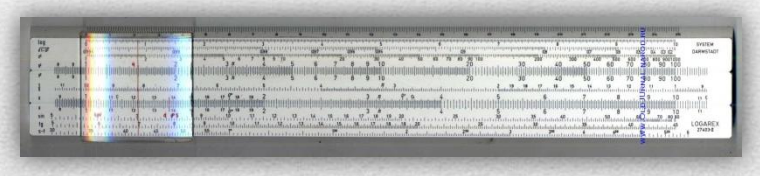



Поколения компьютеров

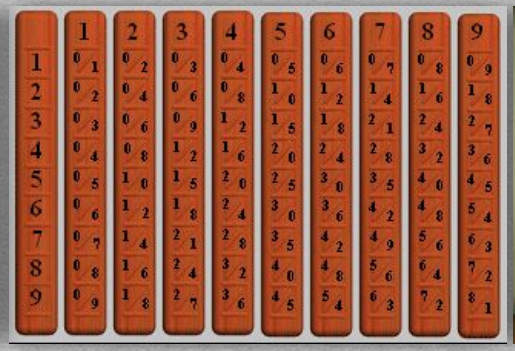
- **«Нулевое» – (с … до 1940)**
  - ✔ Вычислительный элемент – механический.
  - ✔ Простые арифметические операции.
  - ✔ Арифмометры, механические счетные машины.


Модель связки бирок из Красноярского края




«Логарифмическая линейка»


«Счеты»


«Палочки Непера»


«Машина Бэббиджа»


«Машина Блеза Паскаля»

Поколения компьютеров

- **«Первое» (1940-1950)**

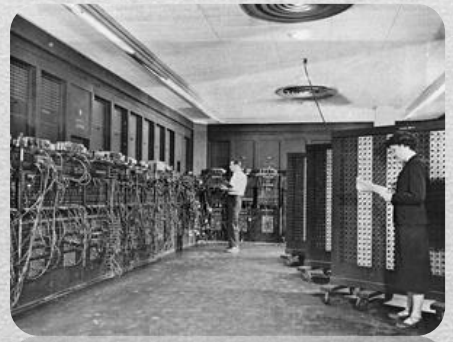
✔ Вычислительный элемент – электронные лампы.

✔ Быстродействие – до нескольких десятков тысяч операций в секунду.

✔ Ввод информации - с перфоленты и кинопленки.

✔ «Большие» ЭВМ.

- **«Colossus»** – секретная разработка британского правительства (1500 электронных ламп).
- **«ENIAC» (Electronic Numerical Integrator And Calculator)**. Создатели: *Джон Моушли и Дж. Преспер Экерт*. Вес машины: *30 тонн*. Минусы: *использование десятичной системы счисления*; *множество переключателей и кабелей*.
- **«EDVAC Computer».** Достижение: *первая машина с программой в памяти*.
- **«Whirlwind I»**. *Слова малой длины, работа в реальном времени*.
- **«IBM 701»**. *Первый компьютер, лидирующий на рынке в течение 10 лет*.
- **«Урал-2».**
- **«Марк-1».**



«ENIAC»

«EDVAC»

«Whirlwind I»

«IBM 701»

Поколения компьютеров

- **«Второе» (1950-1964)**

- ✔ Вычислительный элемент – транзисторы.
- ✔ Быстродействие – до 1-2 млн. операций в секунду.
- ✔ Мини-ЭВМ.
  - Развиваются способы хранения информации: широко используется магнитная лента, позже появляются диски. В этот период была замечена первая компьютерная игра.
  - «CDC6600» - имел возможность параллельного выполнения команд.
  - «NEAC-2203» 1960 год. Произведенный компанией «Nippon Electric Company», эта машина, основанная на барабане, одна из первых транзисторных японских компьютеров.
  - «М-222».
  - «БЭСМ-6».

«CDC 6600»

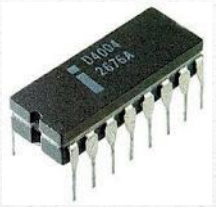
«БЭСМ-6»

«Z11»

«NEAC-2203»

Поколения компьютеров

- **«Третье» (1964-1971)**
  - ✔ Вычислительный элемент – интегральные схемы.
  - ✔ Быстродействие – до 300 млн. операций в секунду.
  - ✔ Микро-ЭВМ, предназначенные для работы с одним пользователем.
    - Появились первые операционные системы.
    - Появились серии компьютеров «PDP».
    - «IBM System-360».
    - «ЕС-1012», «ЕС-1032», «ЕС-1033», «ЕС-1022», «ЕС-1060».



«IBM System-360»

«ЕС-1060»

«PDP-8»

«PDP-11»

Поколения компьютеров

- **«Четвёртое» (1971-…)**
- ✔ Вычислительный элемент – микропроцессоры.
- ✔ Быстродействие миллиарды операций в секунду.
  - Персональные ЭВМ.
  - Готовые прикладные программы, графический интерфейс, использование технологий мультимедиа.
  - Глобальные компьютерные сети.

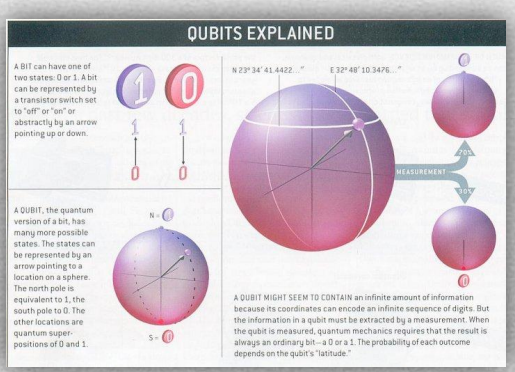
«ДВК»

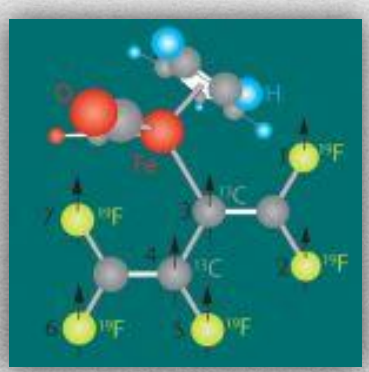
Ноутбук

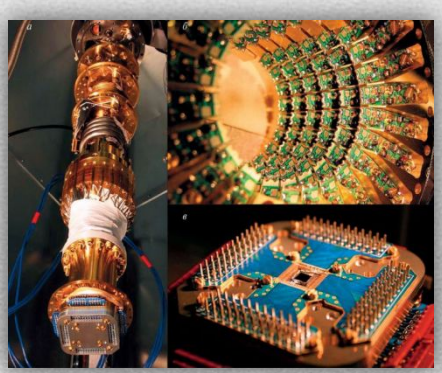
Настольный ПК

Поколения компьютеров

- **«Пятое» (? Будущее)**
- ✔ Нанотехнологии. Компьютеры на основе отдельных молекул и даже атомов. Нейросети, моделирующие структуру нервной системы человека. «Биологические компьютеры».







«Кубит»

Архитектура ЭВМ

Машина Тьюринга

Машина Тьюринга (МТ) — абстрактный исполнитель (абстрактная вычислительная машина). Была предложена Аланом Тьюрингом в 1936 году для формализации понятия алгоритма.

В состав машины Тьюринга входит бесконечная в обе стороны лента (возможны машины Тьюринга, которые имеют несколько бесконечных лент), разделённая на ячейки, и управляющее устройство, способное находиться в одном из множества состояний. Число возможных состояний управляющего устройства конечно и точно задано.

Управляющее устройство может перемещаться влево и вправо по ленте, читать и записывать в ячейки ленты символы некоторого конечного алфавита. Выделяется особый пустой символ, заполняющий все клетки ленты, кроме тех из них (конечного числа), на которых записаны входные данные.

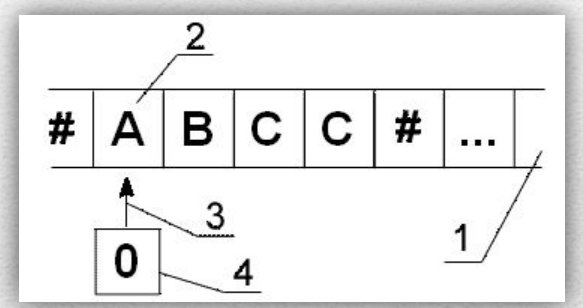
Управляющее устройство работает согласно правилам перехода, которые представляют алгоритм, реализуемый данной машиной Тьюринга. Каждое правило перехода предписывает машине, в зависимости от текущего состояния и наблюдаемого в текущей клетке символа, записать в эту клетку новый символ, перейти в новое состояние и переместиться на одну клетку влево или вправо. Некоторые состояния машины Тьюринга могут быть помечены как терминальные, и переход в любое из них означает конец работы, остановку алгоритма.

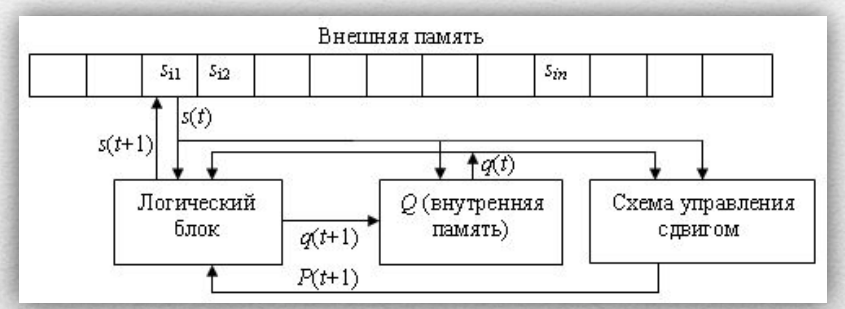
Машина Тьюринга называется детерминированной, если каждой комбинации состояния и ленточного символа в таблице соответствует не более одного правила. Если существует пара «ленточный символ — состояние», для которой существует 2 и более команд, такая машина Тьюринга называется недетерминированной.

Архитектура ЭВМ

Машина Тьюринга

Машина Тьюринга — автоматическое устройство, которое работает с лентой (1) потенциально неограниченной длины. Лента разделена на ячейки, каждая из которых хранит один символ. Одна из ячеек называется текущей (2). В любой момент времени машина Тьюринга находится в некотором состоянии, которое записано в управляющем устройстве (4). Кроме того, головка чтения/записи (3) управляющего устройства указывает на текущую ячейку.





Управляющее устройство выполняет одно действие за единицу времени (шаг). Действие включает возможное изменение состояния, возможное изменение символа в текущей ячейке и возможное движение головки чтения/записи в соседнюю ячейку. Эти действия определены в специальной таблице, называемой управляющей таблицей. Обозначим движение вдоль ленты следующими символами: "<" — влево, ">" — вправо, "=" — движение отсутствует.

Управляющая таблица является программой для машины Тьюринга. Работа машины Тьюринга считается завершённой, когда ни одна строка управляющей таблицы не содержит комбинации текущего символа и текущего состояния.

Архитектура ЭВМ

Машина Тьюринга

Пример управляющей таблицы

Текущее состояние	Текущий символ	Новое состояние	Новый символ	Движение
1	-	2	-	>
2	-	3	+	>
3	#	4	#	<
4	+	4	+	<
4	-	5	-	=

Исходные данные для машины Тьюринга заранее помещаются в клетки ленты. Результат записывается на ту же ленту. Будем считать, что исходное состояние машины Тьюринга равно 1, а входные данные на ленте ограничены символами '#' с обоих концов. (Все ячейки ленты, кроме заполненных минусами, заполнены символами '#'.) Головка управляющего устройства указывает на самый левый символ '–' входных данных. В начале работы лента содержит знак '–' (минус), повторённый n раз ($1 \le n \le 200$), а ввод вашей программы содержит целое число k.

Представьте, что минусы расположены по кругу. Начиная с первого, каждый k-й незачёркнутый минус зачёркивается, то есть превращается в '+' (плюс). Исполнение останавливается, когда остаётся только один незачёркнутый минус. Ваша задача — описать управляющую таблицу для машины Тьюринга, которая зачеркнёт все минусы, кроме одного (его позиция определена в соответствии с описанным выше алгоритмом, но вы можете использовать любой способ нахождения позиции) для заданного k. Например, для n = 10 и k = 3 четвёртый минус останется незачёркнутым.

Архитектура ЭВМ

- **Машина Тьюринга**

Разрешается записывать на ленту следующие символы: '+', '#', 'A'..'Z'. Клетки, изначально заполненные минусами, могут содержать только символы '–' и '+'. Клетка, в которой останется минус, не должна меняться. После выполнения головка чтения/записи должна указывать на незачёркнутый минус. Количество шагов s не должно превосходить 1 000 000. Количество строк p в таблице управления не должно превосходить 10 000. Размер ленты ограничен 10 001 ячейкой (5000 с каждой стороны от начального положения головки чтения/записи).

Архитектура ЭВМ

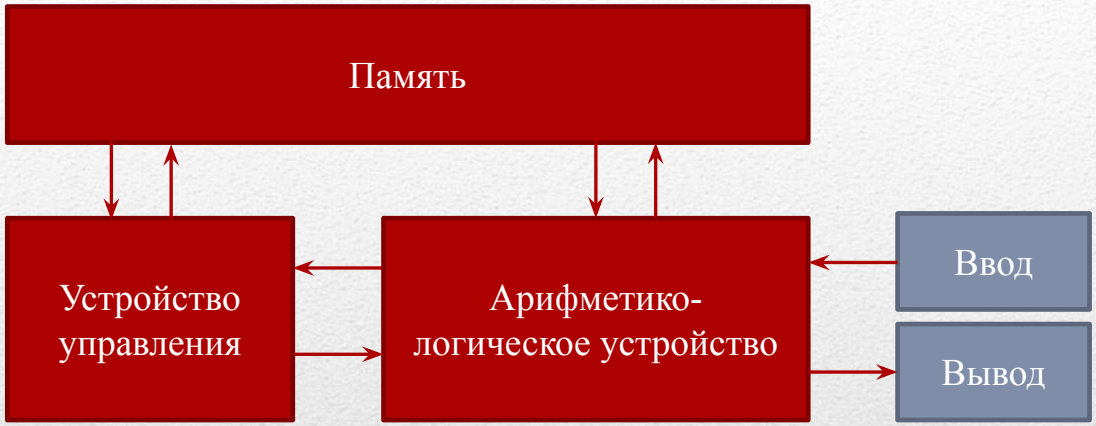
Под «Архитектурой» понимается совокупность свойств и характер ВМ, рассматриваемая с точки зрения пользователя.

Различаю два вида архитектур компьютеров:

- «Гарвардская»;
- «фон Неймана» («Принстонская»).

Архитектура ЭВМ

Архитектура «фон Неймана»



«Принципы фон Неймана»:

- ✔ **Принцип двоичного кодирования**
  Согласно этому принципу, вся информация, поступающая в ЭВМ, кодируется с помощью двоичных сигналов (двоичных цифр, битов) и разделяется на единицы, называемые словами.
- ✔ **Принцип однородности памяти**
  Программы и данные хранятся в одной и той же памяти. Поэтому ЭВМ не различает, что хранится в данной ячейке памяти — число, текст или команда. Над командами можно выполнять такие же действия, как и над данными.
- ✔ **Принцип адресуемости памяти**
  Структурно основная память состоит из пронумерованных ячеек; процессору в произвольный момент времени доступна любая ячейка. Отсюда следует возможность давать имена областям памяти, так, чтобы к хранящимся в них значениям можно было бы впоследствии обращаться или менять их в процессе выполнения программы с использованием присвоенных имен.
- ✔ **Принцип последовательного программного управления**
  Предполагает, что программа состоит из набора команд, которые выполняются процессором автоматически друг за другом в определенной последовательности.
- ✔ **Принцип жесткости архитектуры**
  Неизменяемость в процессе работы топологии, архитектуры, списка команд.

Архитектура ЭВМ

Архитектура «Гарвардская»

Отличительными признаками являются:

1. Хранилище инструкций и хранилище данных представляют собой разные физические устройства;
2. Канал инструкций и канал данных также физически разделены.

✔ **Классическая гарвардская архитектура**

Физическое разделение команд и данных, доступ к которым осуществляется через шину инструкций и шину данных. В некоторых системах инструкции могут хранится в памяти только для чтения, в то время как, для сохранения данных обычно требуется память с возможностью чтения и записи.

✔ **Модифицированная гарвардская архитектура**

При разделении каналов передачи команд и данных на кристалле процессора последний должен иметь почти вдвое больше выводов, так как шина адреса и шина данных составляют основную часть выводов микропроцессора. Способом решения этой проблемы стала идея использовать **общие шину данных и шину адреса для всех внешних данных, а внутри процессора использовать шину данных, шину команд и две шины адреса.** Разделение осуществляется при помощи раздельных управляющих сигналов: чтения, записи или выбора области памяти.

✔ **Расширенная гарвардская архитектура**

Часто требуется выбрать три составляющие: два операнда и инструкцию. Для этого существует кэш-память. В ней может храниться инструкция — следовательно, обе шины остаются свободными и появляется возможность передать два операнда одновременно. **Использование кэш-памяти вместе с разделёнными шинами** получило название «Super Harvard Architecture» («SHARC») — расширенная Гарвардская архитектура.

✔ **Гибридные модификации с архитектурой «фон Неймана»**

Существуют гибридные архитектуры, сочетающие достоинства как «Гарвардской» так и «фон Неймановской» архитектур. Современные CISC-процессоры обладают раздельной кэш-памятью 1-го уровня для инструкций и данных, что позволяет им за один рабочий такт получать одновременно и команду, и данные для её выполнения. То есть процессорное ядро, формально, является гарвардским, но программно оно фон Неймановское, что упрощает написание программ. Обычно в данных процессорах одна шина используется и для передачи команд, и для передачи данных, что упрощает конструкцию системы. Современные варианты таких процессоров могут иногда содержать встроенные контроллеры сразу нескольких разнотипных шин для работы с различными типами памяти — например, DDR RAM и Flash.

Архитектура ЭВМ

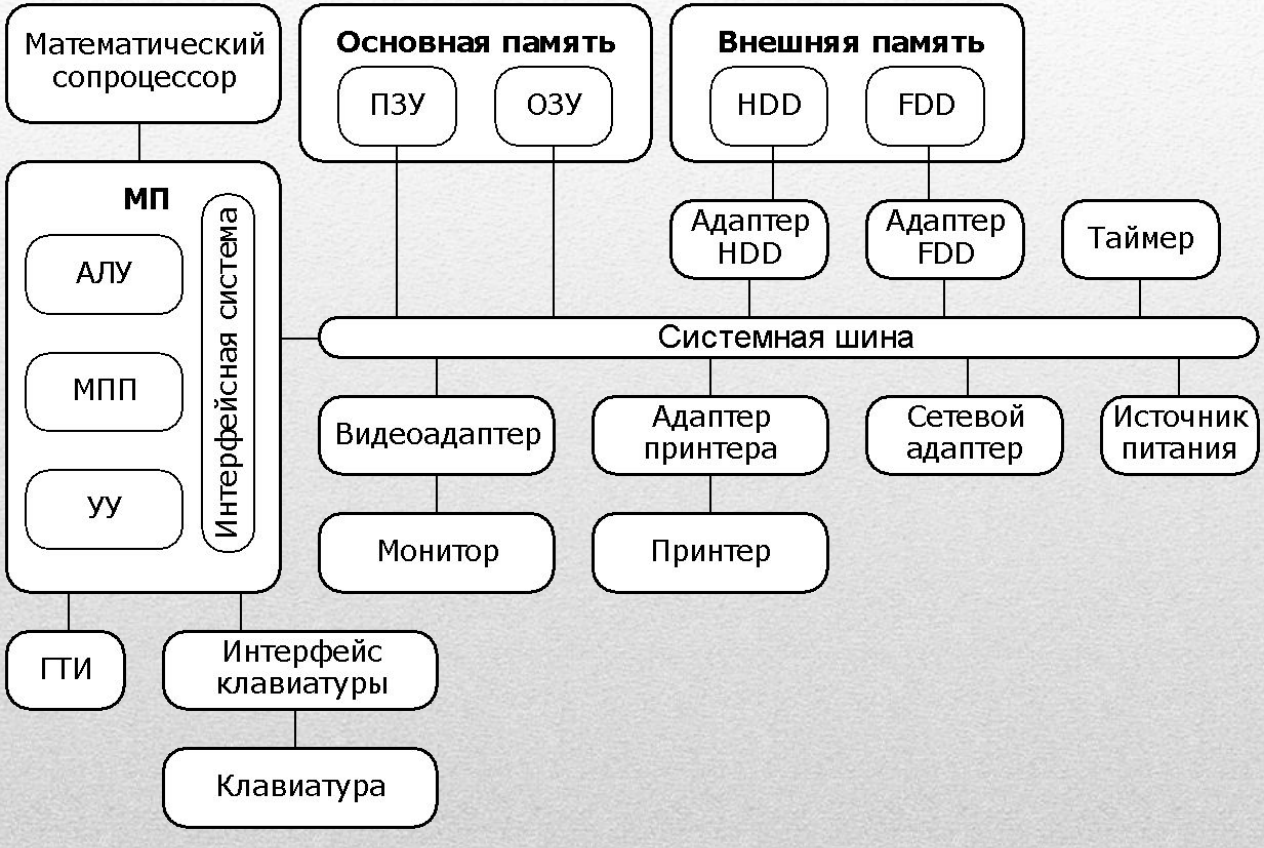
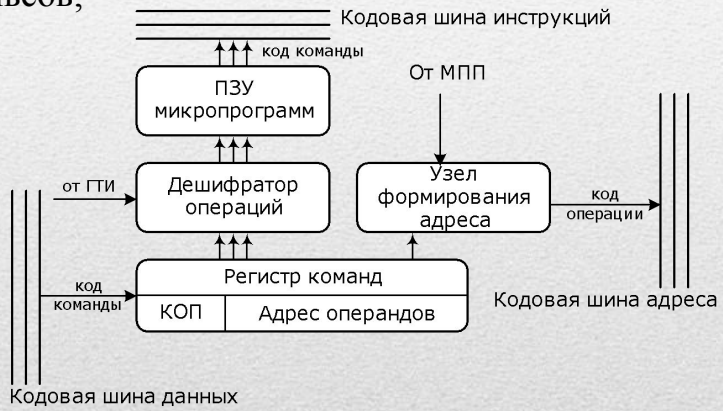


Рис. Структурная схема персонального компьютера

Архитектура ЭВМ

Устройство управления

Устройство управления (УУ) - формирует и подает во все блоки машины в нужные моменты времени определенные сигналы управления (управляющие импульсы), обусловленные спецификой выполняемой операции и результатами предыдущих операций; формирует адреса ячеек памяти, используемых выполняемой операцией, и передает эти адреса в соответствующие блоки ЭВМ; опорную последовательность импульсов устройство управления получает от генератора тактовых импульсов;

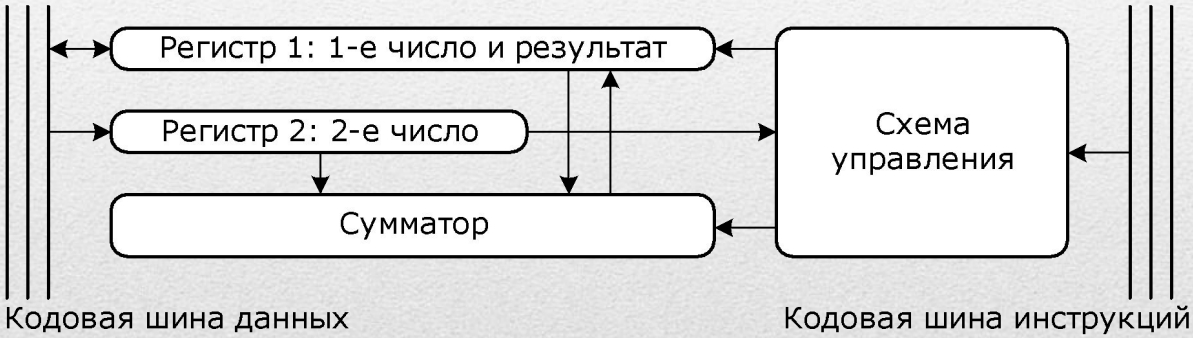


- Регистр команд – запоминающий регистр, в котором хранится код команды: код выполняемой операции (КОП) и адреса операндов, участвующих в операциях. Регистр команд расположен в интерфейсной части МП, в блоке регистров команд.
- Дешифратор операций – логический блок, выбирающий в соответствии с поступающим из регистра команд кодом операции (КОП) один из множества имеющихся у него выходов.
- ПЗУ микропрограмм – хранит в своих ячейках управляющие сигналы, необходимые для выполнения в блоках ПК операций обработки информации. Импульс по выбираемому дешифратором операций в соответствии с кодом операции считывает из ПЗУ микропрограмм необходимую последовательность.
- Узел формирования адреса – устройство, вычисляющее полный адрес ячейки памяти (регистра) по реквизитам поступающим из регистра команд и регистров МПП. Узел формирования адреса расположен в интерфейсной части МП.
- Кодовая шина данных, адреса и инструкций – часть внутренней интерфейсной шины МП.

Архитектура ЭВМ

Арифметико-логическое устройство

Арифметико-логическое устройство (АЛУ) - предназначено для выполнения всех арифметических и логических операций над числовой и символьной информацией (в некоторых моделях ПК для ускорения выполнения операций к АЛУ подключается дополнительный «Математический сопроцессор»);



Функционально АЛУ состоит из двух регистров, сумматора и схем управления.

- ✔ Сумматор – вычислительная схема, выполняющая процедуру сложения поступающих на ее вход двоичных кодов; сумматор имеет разрядность двойного слова.
- ✔ Регистры – быстродействующие ячейки памяти различной длины: регистр 1 имеет разрядность двойного слова, а регистр 2 имеет разрядность слова.
- ✔ Схема управления – принимает по кодовой шине инструкций управляющие сигналы от устройства управления и преобразует их в сигналы для управления работой регистров и сумматора АЛУ.

Архитектура ЭВМ

Микропроцессорная память

Микропроцессорная память (МПП) - служит для кратковременного хранения, записи и выдачи информации, непосредственно используемой в вычислениях в ближайшие такты работы машины. МПП строится на регистрах и используется для обеспечения высокого быстродействия машины, ибо основная память (ОП) не всегда обеспечивает скорость записи, поиска и считывания информации, необходимую для эффективной работы быстродействующего микропроцессора.

Регистры - быстродействующие ячейки памяти различной длины (в отличие от ячеек ОП, имеющих стандартную длину 1 байт и более низкое быстродействие);

- **Регистры общего назначения (РОН)** - представляющие собой часть регистров процессора, использующихся без ограничения в арифметических операциях, но имеющие определенные ограничения.

- **Регистры специального назначения (РСП)** - содержат данные, необходимые для работы процессора - смещения базовых таблиц, уровни доступа.

Архитектура ЭВМ

Интерфейсная система микропроцессора

Интерфейсная система микропроцессора

Интерфейсная система микропроцессора - Реализует сопряжение и связь с другими устройствами ПК; включает в себя внутренний интерфейс МП, буферные запоминающие регистры и схемы управления портами ввода-вывода (ПВВ) и системной шиной.

Интерфейс (interface)

Интерфейс - Совокупность средств сопряжения и связи устройств компьютера, обеспечивающая их эффективное взаимодействие.

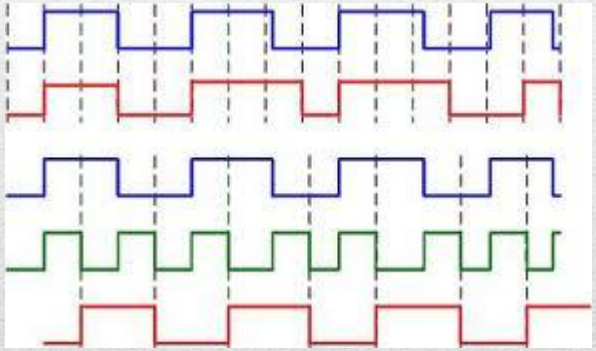
Порт ввода-вывода (I/O ≈ Input/Output port)

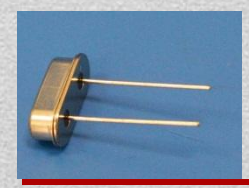
Порт ввода-вывода - аппаратура сопряжения, позволяющая подключить к микропроцессору другое устройство ПК.

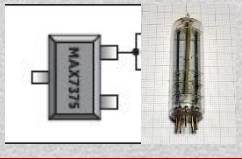
Архитектура ЭВМ

Генератор тактовых импульсов

Генератор тактовых импульсов - генерирует последовательность электрических импульсов; частота генерируемых импульсов определяет тактовую частоту машины. Промежуток времени между соседними импульсами определяет время одного такта работы машины или просто такт работы машины.







Архитектура ЭВМ

Системная шина

Системная шина - это основная интерфейсная система компьютера, обеспечивающая сопряжение и связь всех его устройств между собой.

Системная шина включает в себя:

- ✔ **кодовую шину данных (КШД)**, содержащую провода и схемы сопряжения для параллельной передачи всех разрядов числового кода (машинного слова) операнда;
- ✔ **кодовую шину адреса (КША)**, включающую провода и схемы сопряжения для параллельной передачи всех разрядов кода адреса ячейки основной памяти или порта ввода-вывода внешнего устройства;
- ✔ **кодовую шину инструкций (КШИ)**, содержащую провода и схемы сопряжения для передачи инструкций (управляющих сигналов, импульсов) во все блоки машины;
- ✔ **шину питания (ШП)**, имеющую провода и схемы сопряжения для подключения блоков ПК к системе энергопитания.

Системная шина обеспечивает три направления передачи информации:

- между микропроцессором и основной памятью;
- между микропроцессором и портами ввода-вывода внешних устройств;
- между основной памятью и портами ввода-вывода внешних устройств (в режиме прямого доступа к памяти).

Архитектура ЭВМ

Микропроцессор

Микропроцессор - процессор (устройство, отвечающее за выполнение арифметических, логических операций и операций управления, записанных в машинном коде), реализованный в виде одной микросхемы или комплекта из нескольких специализированных микросхем (в отличие от реализации процессора в виде электрической схемы на элементной базе общего назначения или в виде программной модели). Первые микропроцессоры появились в 1970-х годах и применялись в электронных калькуляторах, в них использовалась двоично-десятичная арифметика 4-битных слов. Вскоре их стали встраивать и в другие устройства, например терминалы, принтеры и различную автоматику. Доступные 8-битные микропроцессоры с 16-битной адресацией позволили в середине 1970-х годах создать первые бытовые микрокомпьютеры.

- CISC;
- RISC;
- MISC;
- URISC;
- VLIW.

Архитектура ЭВМ

Микропроцессор

CISC (Complex instruction set computing)

CISC - концепция проектирования процессоров, которая характеризуется следующим набором свойств:

- нефиксированное значение длины команды;
- арифметические действия кодируются в одной команде;
- небольшое число регистров, каждый из которых выполняет строго определённую функцию.

RISC (Restricted (reduced) instruction set computer)

RISC - архитектура процессора, в которой быстродействие увеличивается за счёт упрощения инструкций, чтобы их декодирование было более простым, а время выполнения - короче. Характерные особенности RISC-процессоров:

- Фиксированная длина машинных инструкций (например, 32 бита) и простой формат команды.
- Специализированные команды для операций с памятью — чтения или записи. Операции вида «прочитать-изменить-записать» отсутствуют. Любые операции «изменить» выполняются только над содержимым регистров (т. н. архитектура load-and-store).
- Большое количество регистров общего назначения (32 и более).
- Отсутствие поддержки операций вида «изменить» над укороченными типами данных — байт, 16-битное слов.
- Отсутствие микропрограмм внутри самого процессора. То, что в CISC процессоре исполняется микропрограммами, в RISC процессоре исполняется как обыкновенный машинный код, не отличающийся принципиально от кода ядра ОС и приложений.

MISC (Minimal instruction set computer)

MISC - процессорная архитектура с минимальным набор команд.

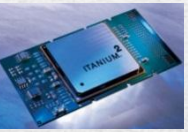
Архитектура ЭВМ

Микропроцессор

URISC (Ultimate RISC)

URISC - предельный случай процессора типа RISC (компьютер с сокращённым набором инструкций), в котором выполняется только один тип инструкций: обычно это «reverse-subtract and skip if borrow», что означает «вычесть и пропустить следующую инструкцию, если вычитаемое было больше уменьшаемого» соответственно. Аналогичная концепция, основанная именно на «subtract and branch unless positive» — «вычесть и перейти, если результат не положительный», называется SUBLEQ.

VLIW-процессоры (Very long instruction word)



VLIW - сверхдлинное командное слово. Архитектура процессоров с явно выраженным параллелизмом вычислений, заложенным в систему команд процессора. Ключевым отличием от суперскалярных CISC-процессоров является то, что для них загрузкой исполнительных устройств занимается часть процессора (планировщик), на что отводится достаточно малое время, в то время как загрузкой вычислительных устройств для VLIW-процессора занимается компилятор, на что отводится существенно больше времени (качество загрузки и, соответственно, производительность теоретически должны быть выше). Примером VLIW-процессора является Intel Itanium.

DSP (Digital signal processor)

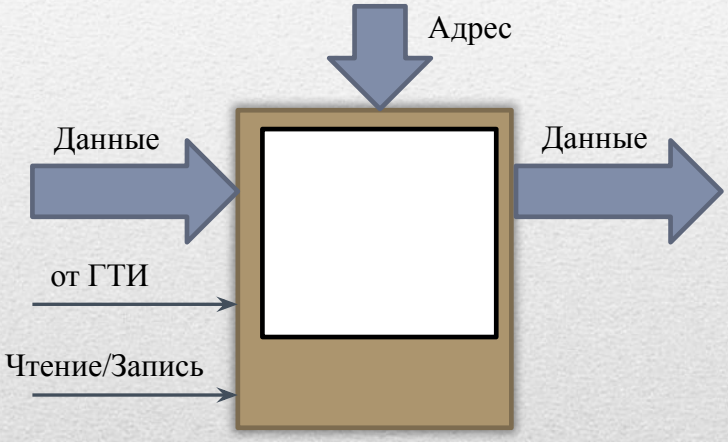
DSP- специализированный микропроцессор, предназначенный для цифровой обработки сигналов (обычно в реальном масштабе времени). Особенности:

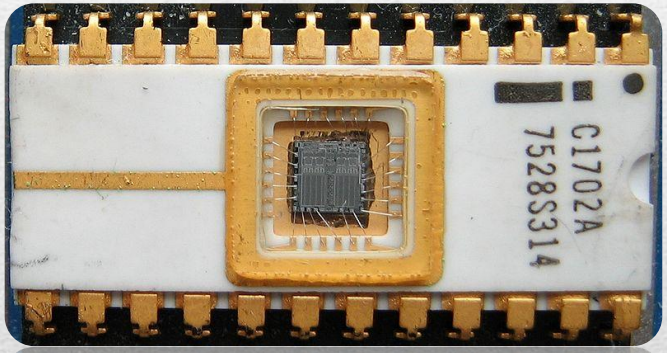
- Гарвардская архитектура (разделение памяти команд и данных), как правило модифицированная;
- Аппаратное ускорение сложных вычислительных инструкций, то есть быстрое выполнение операций, характерных для цифровой обработки сигналов, например, операция «умножение с накоплением» обычно исполняется за один такт.
- «Бесплатные» по времени циклы с заранее известной длиной. Поддержка векторно-конвейерной обработки с помощью генераторов адресных последовательностей.
- Детерминированная работа с известными временами выполнения команд, что позволяет выполнять планирование работы в реальном времени.

Архитектура ЭВМ

Основная память

Основная память (ОП) - предназначена для хранения и оперативного обмена информацией с прочими блоками машины. ОП содержит два вида запоминающих устройств: постоянное запоминающее устройство (ПЗУ) и оперативное запоминающее устройство (ОЗУ).

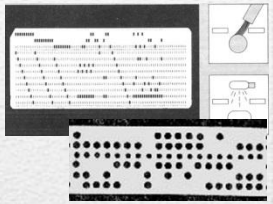




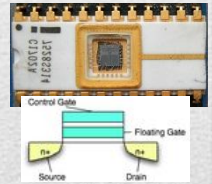
Основная память

Постоянное запоминающее устройство

ПЗУ служит для хранения неизменяемой (постоянной) программной и справочной информации, позволяет оперативно только считывать хранящуюся в нем информацию (изменить информацию в ПЗУ нельзя).



- **перфорационная карта** - сделана из тонкого картона, представляет информацию наличием или отсутствием отверстий в определённых позициях карты.
- **перфолента** - бумажная или целлулоидная пленка, на которую информация наносилась перфоратором в виде совокупности отверстий
- **ROM** — масочное ПЗУ, изготавливается фабричным методом.
- **PROM** - класс полупроводниковых запоминающих устройств, постоянная память с пережигаемыми перемычками



- **EPROM** - перепрограммируемое ПЗУ. Например, содержимое микросхемы К537РФ1 стиралось при помощи ультрафиолетовой лампы.
- **EEPROM -** электрически стираемое перепрограммируемое ПЗУ. Память может стираться и заполняться данными до миллиона раз. (Flash)
- **ПЗУ на магнитных доменах** - устройства, в которых используются электромагнитные процессы на доменном уровне.
- **NVRAM** non-volatile memory — «неразрушающаяся» память, строго говоря, не является ПЗУ.

Основная память

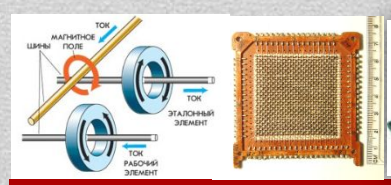
Оперативное запоминающее устройство

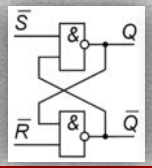
ОЗУ - энергозависимая память, предназначеная для оперативной записи, хранения и считывания информации (программ и данных), непосредственно участвующей в информационно-вычислительном процессе, выполняемом ПК в текущий период времени. Главными достоинствами оперативной памяти являются ее высокое быстродействие и возможность обращения к каждой ячейке памяти отдельно (прямой адресный доступ к ячейке).

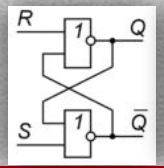
- ✔ **DRAM** (Dynamic random access memory) - динамическая память с произвольным доступом.
- ✔ **SRAM** (static random access memory) - статическая оперативная память с произвольным доступом
- ✔ **SDR SDRAM** (Single Data Rate Synchronous Dynamic Random Access Memory) - синхронная память с тактовым генератором для синхронизации всех сигналов и с использованием конвейерной обработки информации.
- ✔ **FRAM (Ferroelectric RAM) –** сегнетоэлектрическая память, использующая слой сегнетоэлектрика вместо диэлектрического слоя для обеспечения энергонезависимости.

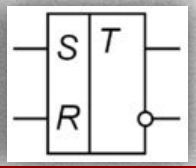
DDR - Double Data Rate

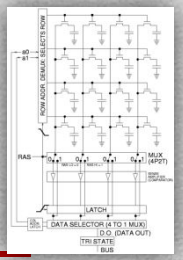
Триггер - класс электронных устройств, обладающих способностью длительно находиться в одном из двух устойчивых состояний и чередовать их под воздействием внешних сигналов.





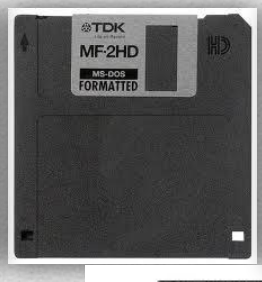


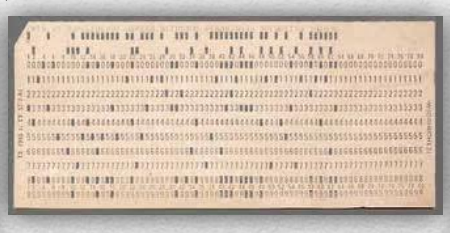
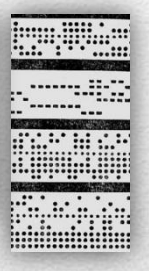


Внешняя память

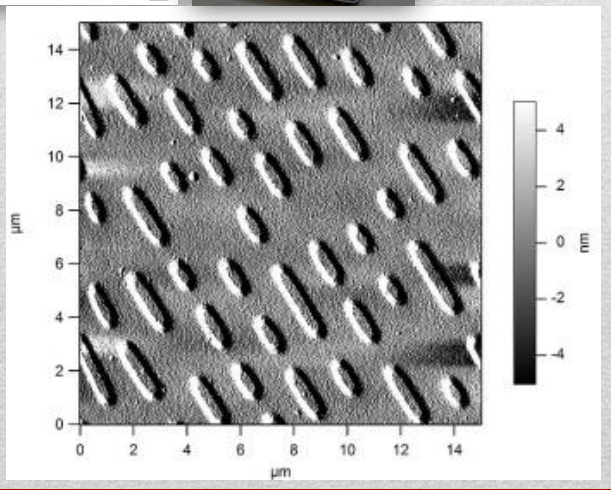
Внешняя память

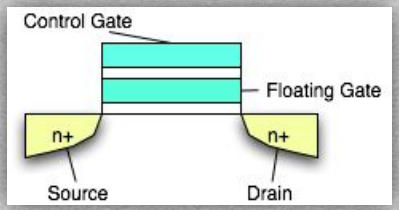
Внешняя память. Она относится к внешним устройствам ПК и используется для долговременного хранения любой информации, которая может когда-либо потребоваться для решения задач. В частности, во внешней памяти хранится все программное обеспечение компьютера. Внешняя память содержит разнообразные виды запоминающих устройств, но наиболее распространенными, имеющимися практически на любом компьютере, являются накопители на жестких (НЖМД) и гибких (НГМД) магнитных дисках.





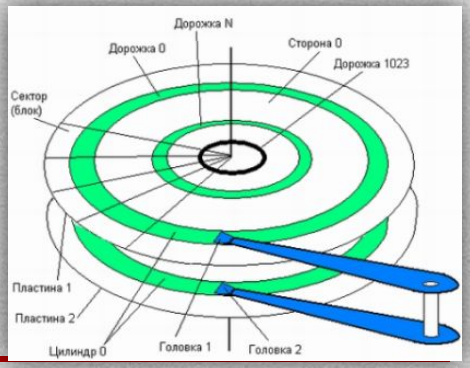


Файловая система

Файловая система - порядок, определяющий способ организации, хранения и именования данных на носителях информации. Файловая система определяет формат содержимого и способ физического хранения информации, которую принято группировать в виде файлов. Конкретная файловая система определяет размер имени файла (папки), максимальный возможный размер файла и раздела, набор атрибутов файла. Некоторые файловые системы предоставляют сервисные возможности, например, разграничение доступа или шифрование файлов.

Секторы - аппаратно адресуемые блоки носителя. Размер секторов на жестких дисках в x86-системах почти всегда равен 512 байтам. Таким образом, если операционная система должна модифицировать 632-й байт диска, она записывает 512-байтовый блок данных во второй сектор диска.

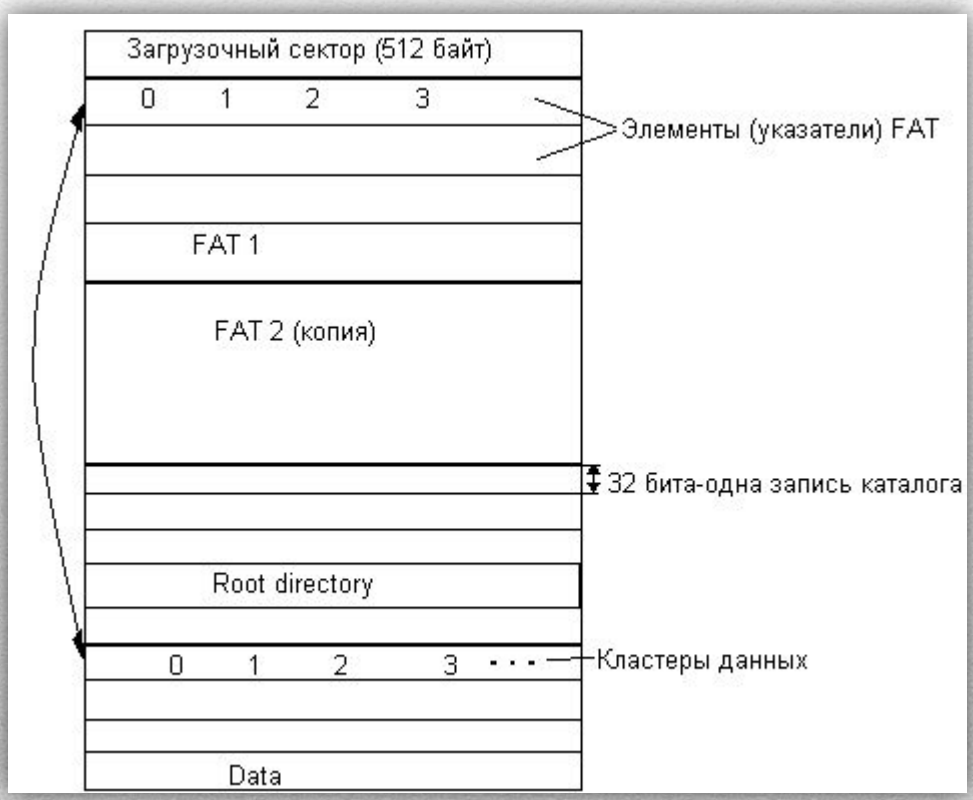
Кластеры — адресуемые блоки, используемые многими файловыми системами. Размер кластера всегда кратен размеру сектора. Файловая система использует кластеры для более эффективного управления дисковым пространством: кластеры, размер которых превышает размер сектора, позволяют разбить диск на блоки меньшей длины — управлять такими блоками легче, чем секторами. Потенциальный недостаток кластеров большего размера — менее эффективное использование дискового пространства, или внутренняя фрагментация, которая возникает из-за того, что размеры файлов редко бывают кратны размеру кластера.



Файловая система

- **FAT**

FAT (File Allocation Table) – файловая система, разработана Биллом Гейтсом и Марком МакДональдом. Использовалась в качестве основной файловой системы в операционных системах семейств DOS и Windows.



Файловая система

- **NTFS**

NTFS (*New Technology File System*) - файловая система для семейства операционных систем Microsoft Windows NT. имеет встроенные возможности разграничивать доступ к данным для различных пользователей и групп пользователей (списки контроля доступа — Access Control Lists (ACL)), а также назначать квоты (ограничения на максимальный объём дискового пространства, занимаемый теми или иными пользователями). NTFS использует систему журналирования USN для повышения надёжности файловой системы.

NTFS разработана на основе файловой системы HPFS (*High Performance File System* — высокопроизводительная файловая система), создававшейся Microsoft совместно с IBM для операционной системы OS/2. Но, получив такие несомненно полезные новшества, как квотирование, журналируемость, разграничение доступа и аудит, в значительной степени утратила присущую прародительнице (HPFS) весьма высокую производительность файловых операций.

Файловая система

Файловые системы Linux

- ext — первая файловая система Linux, использовалась в ранних версиях Linux.
- ext2 — стандартная, но уже устаревшая файловая система Linux.
- ext3 — модифицированная версия файловой системы ext2, но с поддержкой журнала, существенно повышающего надежность файловой системы.
- ext4 — новейшая файловая система Linux. Поддержка ext4 как стабильной файловой системы появилась в ядре Linux версии 2.6.28 — это самая новая версия ядра на момент написания этих строк. Если сравнивать эту файловую систему с ext3, то производительность и надежность новой файловой системы существенно увеличена, а максимальный размер раздела теперь равен 1024 Пбайт2 (1 Эбайт3) . Максимальный размер файла больше 2 Тбайт. ext4 почти в два раза превзошла файловые системы ext3, XFS, JFS и ReiserFS.
- ReiserFS — основная особенность этой файловой системы заключается в хранении в одном блоке нескольких маленьких файлов. Например, если у вас размер блока 4 Кбайт, то в него поместится до четырех файлов по одному килобайту каждый. Если у вас много маленьких файлов, то такая файловая система — просто находка, ведь она позволяет экономить дисковое пространство. Однако с большими файлами эта файловая система работает медленно.
- JFS — разработка IBM, обладает высокой производительностью, но оптимизирована под сервер баз данных, поскольку размер блока небольшой — от 512 байт до 4 Кбайт. Медленно работает с большими файлами.
- XFS — обладает относительно высокой производительностью — она быстрее, чем ext3, ReiserFS и JFS, но медленнее, чем ext4. Устанавливает большой размер блока — до 64 Кбайт, что позволяет ее использовать на графических станциях для обработки видео.

Представление информации в ЭВМ

Единицы измерения количества информации

512кБ≠512КБ

Байт определяется для конкретного компьютера как минимальный шаг адресации памяти, который на старых машинах не обязательно был равен 8 битам

- В соответствии с международным стандартом МЭК 60027-2 единицы «бит» и «байт» применяют с приставками СИ.
- Исторически сложилась такая ситуация, что с наименованием «байт» некорректно (вместо $1000=10^3$ принято $1024=2^{10}$) используют приставки СИ: 1Кбайт=1024байт, 1Мбайт=1024Кбайт, 1Гбайт=1024Мбайт и т.д. При этом обозначение Кбайт начинают с прописной буквы в отличие от строчной буквы «к» для обозначения множителя 10^3.

Измерения в байтах

ГОСТ 8.417-2002			Приставки СИ		Приставки МЭК			
Название	Символ	Степень	Название	Степень	Название	Символ		Степень
байт	Б	2^0	-	10^0	байт	B	Б	2^0
килобайт	КБ	2^{10}	кило-	10^3	кибибайт	KiB	КиБ	2^{10}
мегабайт	МБ	2^{20}	мега-	10^6	мебибайт	MiB	МиБ	2^{20}
гигабайт	ГБ	2^{30}	гига-	10^9	гибибайт	GiB	ГиБ	2^{30}
терабайт	ТБ	2^{40}	тера-	10^{12}	тебибайт	TiB	ТиБ	2^{40}
петабайт	ПБ	2^{50}	пета-	10^{15}	пебибайт	PiB	ПиБ	2^{50}
эксабайт	ЭБ	2^{60}	экса-	10^{18}	эксбибайт	EiB	ЭиБ	2^{60}
зеттабайт	ЗБ	2^{70}	зетта-	10^{21}	зебибайт	ZiB	ЗиБ	2^{70}
йоттабайт	ЙБ	2^{80}	йотта-	10^{24}	йобибайт	YiB	ЙиБ	2^{80}

Представление информации в ЭВМ

Единицы измерения объема данных

Машинное слово — машиннозависимая и платформозависимая величина, измеряемая в битах или байтах (тритах или трайтах), равная разрядности регистров процессора и/или разрядности шины данных (обычно некоторая степень двойки).

Поля постоянной длины:

- слово - 2 байта;
- полуслово - 1 байт;
- двойное слово - 4 байта;
- расширенное слово - 8 байт;
- слово длиной 10 байт- 10 байт.

Информационно-логические основы построения ЭВМ

Представление информации в ЭВМ

✔ Системы счисления и формы представления чисел

Система счисления - это способ наименования и изображения чисел с помощью символов, имеющих определенные количественные значения.

В зависимости от способа изображения чисел системы счисления делятся на *позиционные* и *непозиционные*.

- В **позиционной системе счисления** количественное значение каждой цифры зависит от ее места (позиции) в числе.
- В **непозиционной системе счисления** цифры не меняют своего количественного значения при изменении их расположения в числе.

Пример: Позиционная система счисления - арабская десятичная система, в которой: основание P=10, для изображения чисел используются 10 цифр (от 0 до 9). Непозиционная система счисления - римская, в которой для каждого числа используется специфическое сочетание символов (XIV, CXXVII и т.п.).

Представление информации в ЭВМ

Позиционная система счисления

Количество (*P*) различных цифр, используемых для изображения числа в позиционной системе счисления, называется основанием системы счисления. Значения цифр лежат в пределах от 0 до *P*-1. В общем случае запись любого смешанного числа в системе счисления с основанием *P* будет представлять собой ряд вида:

$$a_{m-1} \cdot P^{m-1} + a_{m-2} \cdot P^{m-2} + \ldots + a_1 \cdot P^1 + a_0 \cdot P^0 + a_{-1} \cdot P^{-1} + \ldots + a_{-s} \cdot P^{-s}$$

где:

- *нижние индексы* определяют местоположение цифры в числе (разряд):
- *положительные значения индексов* - для целой части числа (*m* разрядов);
- *отрицательные значения* - для дробной (*s* разрядов).

Максимальное целое число, которое может быть представлено в *m* разрядах:

$$N_{\max} = P^m - 1$$

Минимальное значащее (не равное 0) число, которое можно записать в *s* разрядах дробной части:

$$N_{\min} = P^{-s}$$

Пример:

$101110{,}101_{(2)} = 1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 + 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3} = 46{,}625_{(10)}$

т.е. двоичное число 101110,101 равно десятичному числу 46,625.

Представление информации в ЭВМ

Система счисления

Наиболее употребляемыми в настоящее время позиционными системами являются:
2 — двоичная;
3 — троичная;
8 — восьмеричная;
10 — десятичная (используется повсеместно);
12 — двенадцатеричная (счёт дюжинами);
13 — тринадцатеричная;
16 — шестнадцатеричная (используется в программировании, информатике);
60 — шестидесятеричная (единицы измерения времени, измерение углов и, в частности, координат, долготы и широты).

10	2	8	16	10	2	8	16
1	0001	01	1	9	1001	11	9
2	0010	02	2	10	1010	12	A
3	0011	03	3	11	1011	13	B
4	0100	04	4	12	1100	14	C
5	0101	05	5	13	1101	15	D
6	0110	06	6	14	1110	16	E
7	0111	07	7	15	1111	17	F
8	1000	10	8	0	0000	00	0

Представление информации в ЭВМ

Правила перевода чисел из одной системы счисления в другую

- Для перевода **двоичного числа в десятичное** необходимо его записать в виде многочлена, состоящего из произведений цифр числа и соответствующей степени числа 2, и вычислить по правилам десятичной арифметики.
- Для перевода **восьмеричного числа в десятичное** необходимо его записать в виде многочлена, состоящего из произведений цифр числа и соответствующей степени числа 8, и вычислить по правилам десятичной арифметики
- Для перевода **шестнадцатеричного числа в десятичное** необходимо его записать в виде многочлена, состоящего из произведений цифр числа и соответствующей степени числа 16, и вычислить по правилам десятичной арифметики
- Для перевода **десятичного числа в двоичную** систему его необходимо последовательно делить на 2 до тех пор, пока не останется остаток, меньший или равный 1. Число в двоичной системе записывается как последовательность последнего результата деления и остатков от деления в обратном порядке
- Для перевода **десятичного числа в восьмеричную** систему его необходимо последовательно делить на 8 до тех пор, пока не останется остаток, меньший или равный 7. Число в восьмеричной системе записывается как последовательность цифр последнего результата деления и остатков от деления в обратном порядке

Представление информации в ЭВМ

Правила перевода чисел из одной системы счисления в другую

- Для перевода **десятичного числа в шестнадцатеричную** систему его необходимо последовательно делить на 16 до тех пор, пока не останется остаток, меньший или равный 15. Число в шестнадцатеричной системе записывается как последовательность цифр последнего результата деления и остатков от деления в обратном порядке
- Чтобы перевести число из **двоичной системы в восьмеричную**, его нужно разбить на триады (тройки цифр), начиная с младшего разряда, в случае необходимости дополнив старшую триаду нулями, и каждую триаду заменить соответствующей восьмеричной цифрой
- Чтобы перевести число из **двоичной системы в шестнадцатеричную**, его нужно разбить на тетрады (четверки цифр), начиная с младшего разряда, в случае необходимости дополнив старшую тетраду нулями, и каждую тетраду заменить соответствующей восьмеричной цифрой.
- Для перевода **восьмеричного числа в двоичное** необходимо каждую цифру заменить эквивалентной ей двоичной триадой.
- Для перевода **шестнадцатеричного числа в двоичное** необходимо каждую цифру заменить эквивалентной ей двоичной тетрадой
- При переходе из **восьмеричной системы счисления в шестнадцатеричную** и обратно, необходим промежуточный перевод чисел в двоичную систему

Представление информации в ЭВМ

Примеры перевода чисел из одной системы счисления в другую

Пример: $11101000_{(2)}=1\cdot2^7+1\cdot2^6+1\cdot2^5+0\cdot2^4+1\cdot2^3+0\cdot2^2+0\cdot2^1+0\cdot2^0=232_{(10)}$

Пример: $75013_{(8)}=7\cdot8^4+5\cdot8^3+0\cdot8^2+0\cdot8^1+3\cdot8^0=31243_{(10)}$

Пример: $FDA1_{(16)}=15\cdot16^3+13\cdot16^2+10\cdot16^1+1\cdot16^0=64929_{(10)}$

Пример: $001001011_{(2)}=001\ 001\ 011_{(2)}=113_{(8)}$

Пример: $001011100011_{(2)}=0010\ 1110\ 0011_{(2)}=2E3_{(16)}$

Пример: $531_{(8)}=101\ 011\ 001_{(2)}$

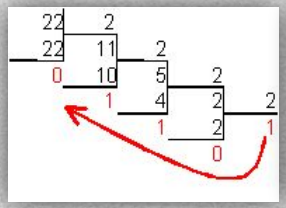
Пример: $EE8_{(16)}=1110\ 1110\ 1000_{(2)}$

Пример: $FEA_{(16)}=1111\ 1110\ 1010_{(2)}=111\ 111\ 101\ 010_{(2)}=7752_{(8)}$

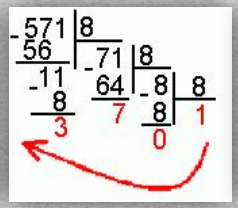
Пример: $6635_{(8)}=110\ 110\ 011\ 101_{(2)}=1101\ 1001\ 1101_{(2)}=D9D_{(16)}$

Пример:
Число $33_{(10)}$ - $x_{(2)}$
33 : 2 = 16 остаток 1;
16 : 2 = 8 остаток 0;
8 : 2 = 4 остаток 0;
4 : 2 = 2 остаток 0;
2 : 2 = 1 остаток 0;
1 : 2 = 0 остаток 1;
Получили $100001_{(2)}$.

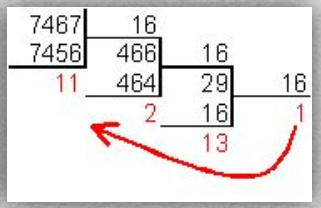
Пример:
Число $55_{(10)}$ – $_{(2)}$
55 : 2 = 27 остаток 1;
27 : 2 = 13 остаток 1;
13 : 2 = 6 остаток 1;
6 : 2 = 3 остаток 0;
3 : 2 = 1 остаток 1;
1 : 2 = 0 остаток 1.
Получили $110111_{(2)}$.



$22_{(10)}=10110_{(2)}$



$571_{(10)}=1073_{(8)}$



$7467_{(10)}=1D2B_{(16)}$

Представление информации в ЭВМ

Представление отрицательных чисел

✔ Прямой код

Прямой код – это представление числа в двоичной системе счисления, при котором первый (старший) разряд отводится под знак числа. Если число положительное, то в левый разряд записывается 0; если число отрицательное, то в левый разряд записывается 1.

Пример:
000001010 - положительное число (10)
100001010 - отрицательное число (-10)

Операция сложения положительного числа и отрицательного числа, представленного в прямом коде
Прямой код числа 5: 0000 0101
Прямой код числа -7: 1000 0111
Два исходных числа сравниваются. В разряд знака результата записывается знак большего исходного числа.
Если числа имеют разные знаки, то вместо операции сложения используется операция вычитания из большего по модулю значения меньшего. При этом первый (знаковый) разряд в операции не участвует.

```
  000 0111
_ 000 0101
-----------
  000 0010
```

После выполнения операции учитывается первый разряд.
Ответ: Результат операции 1000 0010, или (-2_{10}).

Представление информации в ЭВМ

Представление отрицательных чисел

✔ Дополнительный код (дополнение до двух)

В **дополнительном коде**, также как и прямом, первый разряд отводится для представления знака числа. Поэтому, если в старшем разряде находится 1, то мы имеем дело с отрицательным числом. Для получения числа, записанного в дополнительном коде, необходимо: *все разряды числа в дополнительном коде инвертировать, кроме старшего и прибавить 1.*

Пример:
Формирование дополнительного кода числа (-7)
Прямой код : **1**000 0111
Инверсия : **1**111 1000
Добавление единицы: **1**111 100**1**

Пример: Операция сложения положительного числа и отрицательного числа, представленного в дополнительном коде
Прямой код числа 5: 0 000 0101
Прямой код числа -7: 1 000 0111
Формирование дополнительного кода числа -7.
Прямой код: 1000 0111
Инверсия: 1111 1000
Добавление единицы: 1111 1001
Операция сложения.
0000 0101 + 1111 1001 = 1111 1110
Проверка: проверка результата путем преобразования к прямому коду.
Дополнительный код: 1111 1110
Вычитание единицы: 1111 1101
Инверсия: **1000 0010** (или -2_{10})

Представление информации в ЭВМ

- **Представление отрицательных чисел**

✔ **Обратный код**

Обратный код положительного двоичного числа совпадает с прямым кодом. Для отрицательного числа все цифры числа заменяются на противоположные (1 на 0, 0 на 1), а в знаковый разряд заносится единица.

Пример:
Число «-1»
Код модуля числа: 0 000 0001
Обратный код числа: 1 111 1110

Пример:
Число «-127»
Код модуля числа: 0 111 1111
Обратный код числа: 1 000 0000

Представление информации в ЭВМ

Представление вещественных чисел

В вычислительных машинах применяются следующие формы представления двоичных вещественных чисел:

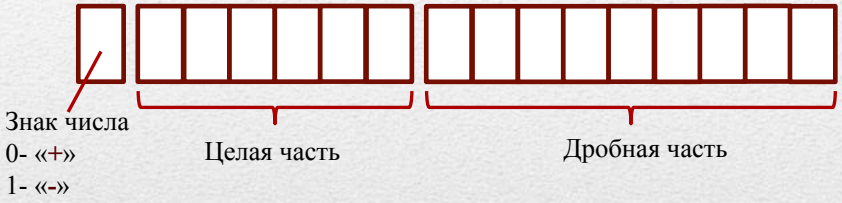
- ✔ Естественная форма или форма с фиксированной запятой (точкой);
- ✔ Нормальная форма или форма с плавающей запятой (точкой).

Представление информации в ЭВМ

Представление вещественных чисел

✔ С фиксированной запятой (естественная форма)

С фиксированной запятой - все числа изображаются в виде последовательности цифр с постоянным для всех чисел положением запятой, отделяющей целую часть от дробной.



Диапазон значащих чисел (N) в системе счисления с основанием P при наличии m разрядов в целой части и s разрядов в дробной части числа (без учета знака числа) будет:

$$P^{-s} \leq N \leq P^{m} + P^{-s}$$

Представление информации в ЭВМ

Представление вещественных чисел

✔ С плавающей запятой

С плавающей запятой - каждое число изображается в виде двух групп цифр. Первая группа цифр называется *мантиссой*, вторая- *порядком*, причем абсолютная величина мантиссы должна быть меньше 1, а порядок - целым числом. В общем виде число в форме с плавающей запятой может быть представлено так:

$$N = \pm MP^{\pm r}$$

где:

M - *мантисса* числа (|M| < 1);

r - *порядок* числа (r- целое число);

P - *основание* системы счисления.

Пример. $3.141592 = 0.3141592 \cdot 10^{1}$

0.3141592 – мантисса;

10 – основание системы счисления;

1 – порядок.

Пример:

В десятичной системе счисления имеются 5 разрядов в целой части числа (до запятой) и 5 разрядов в дробной части числа (после запятой); числа, записанные в такую разрядную сетку, имеют вид:

+00721,35500	$+0{,}721355 \cdot 10^{3}$
+00000,00328	$+0{,}328 \cdot 10^{-3}$
-10301,20260	$-0{,}103012026 \cdot 10^{5}$

Представление информации в ЭВМ

- **Представление вещественных чисел**

✔ **С фиксированной запятой (естественная форма)**

Алгоритм представления числа с плавающей запятой:

1. Перевести число из *p*-ичной системы счисления в двоичную;
2. Представить двоичное число в нормализованной экспоненциальной форме;
3. Рассчитать смещённый порядок числа;
4. Разместить знак, порядок и мантиссу в соответствующие разряды сетки.

Представление информации в ЭВМ

Двоично-десятичная система счисления

Двоично-десятичная система счисления получила большое распространение в современных ЭВМ ввиду легкости перевода в десятичную систему и обратно. Она используется там, где основное внимание уделяется не простоте технического построения машины, а удобству работы пользователя. В этой системе счисления все десятичные цифры отдельно кодируются четырьмя двоичными цифрами и в таком виде записываются последовательно друг за другом.

Таблица. Таблица двоичных кодов десятичных и шестнадцатеричных цифр.

Цифра	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Код	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111

Пример:
Десятичное число 9703 в двоично-десятичной системе выглядит так: 1001011100000011

Пример:
Шестнадцатеричное число F17B в двоичной системе выглядит так: 1111000101111011.

Представление информации в ЭВМ

Представление целых чисел

Двоично-кодированные десятичные числа могут быть представлены в ПК полями переменной длины в так называемых *упакованном* и *распакованном* форматах.

✔ **В упакованном формате** для каждой десятичной цифры отводится по 4 двоичных разряда (полбайта), при этом знак числа кодируется в крайнем правом полубайте числа (1100 - знак "+" и 1101 - знак "-"). Структура поля упакованного формата:

Цф	Цф	Цф	Цф	. . .	Цф	Знак

где: **Цф** - цифра, **Знак** - знак числа.

Упакованный формат используется обычно в ПК при выполнении операций сложения и вычитания двоично-десятичных чисел.

✔ **В распакованном формате** для каждой десятичной цифры отводится по целому байту, при этом старшие полубайты (зона) каждого байта (кроме самого младшего) в ПК заполняются кодом 0011. (в соответствии с ASCII-кодом), а в младших (левых) полубайтах обычным образом кодируются десятичные цифры. Старший полубайт (зона) самого младшего (правого) байта используется для кодирования знака числа.

Структура поля распакованного формата:

Зона	Цф	Зона	Цф	. . .	Зона	Цф	Знак	Цф

Представление информации в ЭВМ

Распакованный формат используется в ПК при вводе-выводе информации в ПК, а также при выполнении операций умножения и деления двоично-десятичных чисел.

Пример. Число $-193_{(10)} = -000110010011_{(2\text{-}10)}$ в ПК будет представлено:

- в упакованном формате:

0001	1001	0011	1101

- в распакованном формате:

0011	0001	0011	1001	1101	0011

Представление информации в ЭВМ

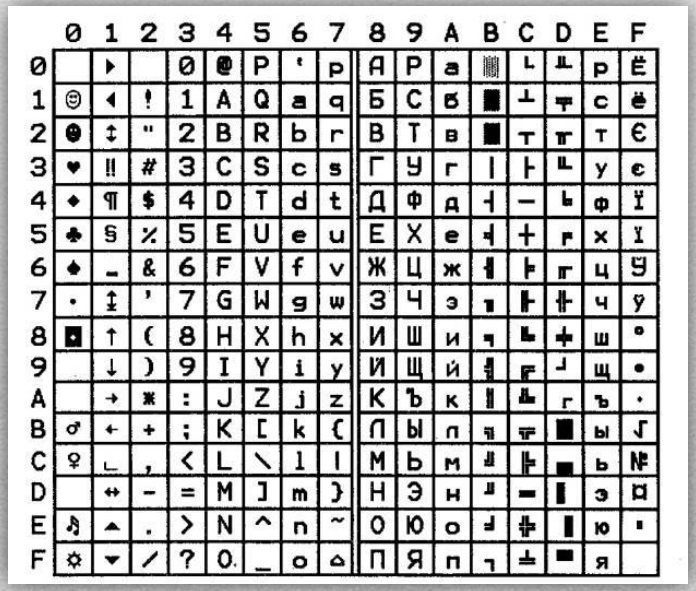
Коды ASCII

Распакованный формат представления двоично-десятичных чисел (иногда его называют "зонный") является следствием использования в ПК ASCII-кода для представления символьной информации.

Код ASCII (American Standard Code for Information Interchange - Американский стандартный код для обмена информацией) имеет *основной стандарт* и его *расширение*.

00-7F - основной стандарт является международным и используется для кодирования управляющих символов, цифр и букв латинского алфавита;

80-FF - расширение стандарта кодирует символы псевдографики и буквы национального алфавита (естественно, в разных странах разные).



Представление информации в ЭВМ

Коды UNICODE

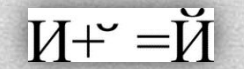
UNICODE - стандарт кодирования символов, позволяющий представить знаки практически всех письменных языков.

- 0000-007F - код ASCII;
- 0100-017F - европейские и латинские символы;
- 0400-04FF - кириллица.

…

Графические символы в Юникоде подразделяются на *протяжённые* и *непротяжённые* (бесширинные). Непротяжённые символы при отображении не занимают места в строке. К ним относятся, в частности, знаки ударения и прочие диакритические знаки. Как протяжённые, так и непротяжённые символы имеют собственные коды.

Особый тип модифицирующих символов — **селекторы** варианта начертания (variation selectors). Они действуют только на те символы, для которых такие варианты определены. В версии 5.0 варианты начертания определены для ряда математических символов.



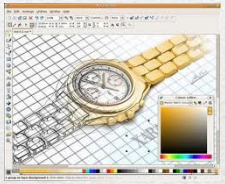
Пример: символ «á» может быть представлен как последовательность базового символа «a» (U+0061) и модифицирующего символа «◌́» (U+0301) или как монолитный символ «á» (U+00C1).

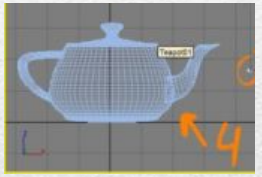
Представление информации в ЭВМ

Представление графической информации в ЭВМ

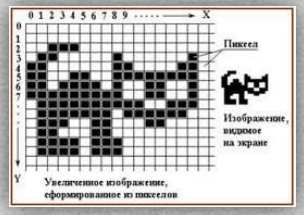
Графическая информация в ЭВМ представляется в **растровом** и **векторном** форматах.

В **векторном** формате изображение разделяется на примитивы - прямые линии, многоугольники, окружности и сегменты окружностей, параметрические кривые, залитые определенным цветом или шаблоном, связные области, набранные определенным шрифтом отрывки текста и т. д.





В **растровом** формате изображение разбивается на прямоугольную матрицу элементов, называемых пикселами (слегка искаженное PICture ELement - элемент картинки). Матрица называется растром. Для каждого пиксела определяется его яркость и, если изображение цветное, цвет. Если, как это часто бывает при оцифровке реальных сцен или преобразовании в растровый формат (растеризации) векторных изображений, в один пиксел попали несколько элементов, их яркость и цвет усредняются с учетом занимаемой площади. При оцифровке усреднение выполняется аналоговыми контурами аналого-цифрового преобразователя, при растеризации - алгоритмами анти-алиасинга.



Представление информации в ЭВМ

Представление графической информации в ЭВМ

Разрешение – количество точек, приходящихся на единицу длины.

Пиксель – минимальный участок изображения, цвет которого можно задавать независимым способом

Представление информации в ЭВМ

Представление звуковой информации в ЭВМ

В основе представления звуковой информации в ЭВМ лежит два направления:

- **Метод FM (Frequency Modulation)** основан та том, что теоретически любой сложный звук можно разложить на последовательность простейших гармонических сигналов разных частот, каждый из которых представляет собой правильную синусоиду, а, следовательно, может быть описан числовыми параметрами, т.е. кодом.
- **Метод таблично волнового синтеза (Wave-Table).** В заранее подготовленных таблицах хранятся образцы звуков для множества различных музыкальных инструментах. В технике такие образцы называют сэмплами. Числовые коды выражают тип инструмента, номер его модели, высоту тона, продолжительность и интенсивность звука, динамику его изменения, некоторые параметры среды, в которой происходит звучание, а также прочие параметры, характеризующие особенности звучания.

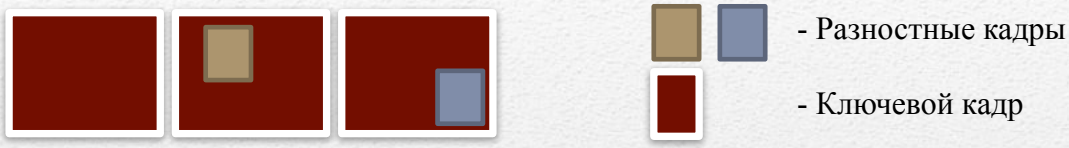
Способы звукозаписи:

- цифровая запись, когда реальные звуковые волны преобразуются в цифровую информацию путем измерения звука тысячи раз в секунду;
- MIDI-запись, которая, вообще говоря, является не реальным звуком, а записью определенных команд-указаний (какие клавиши надо нажимать, например, на синтезаторе). MIDI-запись является электронным эквивалентом записи игры на фортепиано.

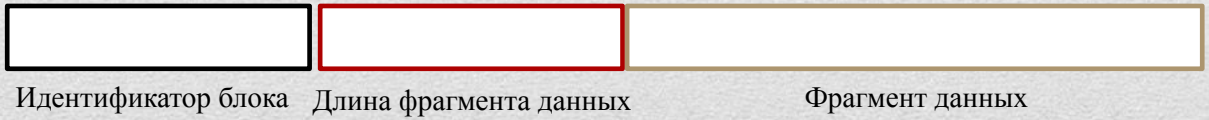
Представление информации в ЭВМ

Представление видео информации в ЭВМ

При представлении видеоинформации в ЭВМ первый кадр сохраняется целиком (ключевой), а в следующих сохранять лишь отличия от начального кадра (разностные кадры).



- **AVI формат** (Audio Video Interleave - чередование аудио и видео) - состоит в хранении структур произвольных мультимедийных данных, каждая из которых имеет простой вид. Файл как таковой представляет собой единый блок, причем в него, как и в любой другой, могут быть вложены новые блоки. Идентификатор блока определяет тип информации, которая хранится в блоке.



- **MPEG (Motion Picture Expert Group)** – состоит из сжатого RGB-представления с равноправными компонентами преобразуя в яркость и две "координаты" цветности. Сжатие происходит за счёт специальных математических преобразований (DCT - дискретно-косинусное преобразование), несколько загрубляющее изображение в мелких деталях.
- **DivX** (Digital Video Express) – способ сжатия.

ЛОГИЧЕСКИЕ ОСНОВЫ ПОСТРОЕНИЯ ПК

Основы алгебры логики

Алгебра логики - это раздел математической логики, значения всех элементов (функций и аргументов) которой определены в двухэлементном множестве: 0 и 1.

Алгебра логики оперирует с логическими *высказываниями*.

Высказывание - это любое предложение, в отношении которого имеет смысл утверждение о его истинности или ложности. При этом считается, что высказывание удовлетворяет закону исключенного третьего, т.е. каждое высказывание или истинно, или ложно и не может быть одновременно и истинным, и ложным.

Наименьшим элементом алгебры логики является 0, наибольшим элементом-1.

В алгебре логики операция отрицания (операция НЕ, операция инверсии), обозначается чертой над элементом $(\overline{a})$

Пример. Высказывания: "Сейчас идет снег"- это утверждение может быть истинным или ложным; "Вашингтон столица США" истинное утверждение; "Частное от деления 10 на 2 равно 3" -ложное утверждение.

В алгебре логики все высказывания обозначают буквами a, b, c и т.д. Содержание высказываний учитывается только при введении их буквенных обозначений, и в дальнейшем над ними можно производить любые действия, предусмотренные данной алгеброй. Причем если над исходными элементами алгебры выполнены некоторые разрешенные в алгебре логики операции, то результаты операций также будут элементами этой алгебры.

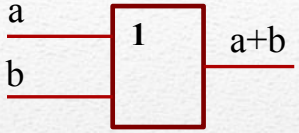
Простейшими операциями в алгебре логики являются операции логического сложения (иначе, операция ИЛИ, операция дизъюнкции) и логического умножения (иначе, операция И, операция конъюнкции). Для обозначения операции логического сложения используют символы + или V, а логического умножения - символы * или L .

Правила выполнения операций в алгебре логики определяются рядом аксиом, теорем и следствий.

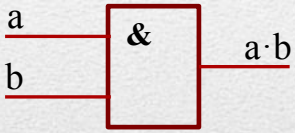
ЛОГИЧЕСКИЕ ОСНОВЫ ПОСТРОЕНИЯ ПК

Логические элементы

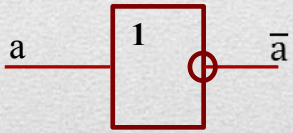
✔ Логический элемент ИЛИ, реализующая операцию логического сложения



✔ Логический элемент И, реализующая операцию логического умножения



✔ Логический элемент НЕ, реализующая операцию инверсии



Примечание

1. В ряде случаев перед построением логической схемы устройства по логической функции последнюю, пользуясь соотношениями алгебры логики следует преобразовать к более простому виду (минимизировать).
2. Для логических схем ИЛИ, И и НЕ существуют типовые технические схемы, реализующие их на реле, электронных лампах, дискретных полупроводниковых элементах. Для построения современных ЭВМ обычно применяются системы интегральных элементов, у которых с целью большей унификации в качестве базовой логической схемы используется всего одна из схем: И НЕ (штрих Шеффера), ИЛИ НЕ (стрелка Пирса) или И ИЛИ НЕ.

ЛОГИЧЕСКИЕ ОСНОВЫ ПОСТРОЕНИЯ ПК

- **Аксиомы и свойства логических операций**

Коммутативность: $a \odot b = b \odot a;\ \forall \odot \in \{+, \cdot, -\}$

Идемпотентность: $a \odot a = a;\ \forall \odot \in \{+, \cdot\}$

Ассоциативность: $(a \odot b) \odot c = a \odot (b \odot c);\ \forall \odot \in \{+, \cdot\}$

$$\overline{\overline{a}} = a$$

$$a + a = a$$

$$a \cdot a = a$$

$$a + a \cdot b = a$$

$$a \oplus b = \overline{a} \cdot b + a \cdot \overline{b} = (a + b) \cdot (\overline{a} + \overline{b})$$

$$a + b = a, \quad \text{если } a \le b \qquad a + b = b, \quad \text{если } a \ge b$$

$$a \cdot b = a, \quad \text{если } a \le b \qquad a \cdot b = b, \quad \text{если } a \ge b$$

$$a + \overline{a} = 1 \qquad \overline{0} = 1$$

$$a \cdot \overline{a} = 1 \qquad \overline{1} = 0$$

ЛОГИЧЕСКИЕ ОСНОВЫ ПОСТРОЕНИЯ ПК

Законы алгебры логики

✔ **Сочетательный:**

$$(a + b) + c = a + (b + c)$$
$$(a \cdot b) \cdot c = a \cdot (b \cdot c)$$

✔ **Переместительный:**

$$a + b = b + a$$
$$a \cdot b = b \cdot a$$

✔ **Дистрибутивность:**

$$a \cdot (b + c) = a \cdot b + a \cdot c$$
$$a + b \cdot c = (a + b) \cdot (a + c)$$
$$a \cdot (b \oplus c) = a \cdot b \oplus a \cdot c$$

✔ **де Моргана:**

$$\overline{a + b} = \overline{a} \cdot \overline{b}$$
$$\overline{a \cdot b} = \overline{a} + \overline{b}$$

✔ **Законы поглощения:**

$$a \cdot (a + b) = a$$
$$a + a \cdot b = a$$

ЛОГИЧЕСКИЕ ОСНОВЫ ПОСТРОЕНИЯ ПК

Таблица истинности

Таблица истинности – это таблица, описывающая логическую функцию.

a	b	·	+	Å
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

a	
0	1
1	0

« » – сложение по модулю 2 («XOR», «Искл. ИЛИ»);
«·» – логическое умножение («И», «AND», «&», «∧», «Конъюнкция»);
«+» – логическое сложение («ИЛИ», «OR», «|», «∨», «Дизъюнкция», «max(a,b)»);
$\overline{a}$ – логическое отрицание (¬).

ЛОГИЧЕСКИЕ ОСНОВЫ ПОСТРОЕНИЯ ПК

Функция в алгебре логики это алгебраическое выражение, содержащее элементы алгебры логики

$$a, b, c, \ldots,$$

связанные между собой операциями, определенными в этой алгебре.

Пример. Примеры логических функций.

$$f(a,b,c) = a + a \cdot b \cdot n + a + c$$
$$f(a,b,c) = a \cdot b + a \cdot c + a \cdot b \cdot c$$

Согласно *теоремам разложения функций на конституанты* (составляющие) любая функция может быть разложена на конституанты "1":

$$f(a) = f(1) \cdot a + f(0) \cdot \bar{a}$$

$$f(a,b) = f(1,b) \cdot a + f(0,b) \cdot \bar{a} = f(1,1) \cdot a \cdot a + f(1,0) \cdot a \cdot \bar{b} + f(0,1) \cdot \bar{a} \cdot b + f(0,0) \cdot \bar{a} \cdot \bar{b}$$

Эти соотношения используются для синтеза логических функций и вычислительных схем.

Логический синтез вычислительных схем

Рассмотрим логический синтез (создание) вычислительных схем на примере одноразрядного двоичного сумматора, имеющего два входа ("a" и "b") и два выхода ("S" и "P") и выполняющего операцию сложения в соответствии с заданной таблицей:

a	b	$f_1(a, b)=S$	$f_2(a, b)=P$
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

где:

$f_1(a, b)=S$ - значение цифры суммы в данном разряде;

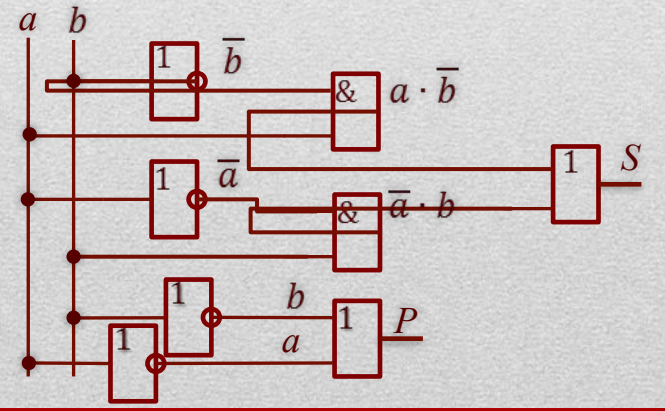
$f_2(a, b)=P$ - цифра переноса в следующий (старший) разряд.

Согласно правилам, можно записать:

$$S = f_1(a,b) = 0 \cdot a \cdot b + 1 \cdot a \cdot \overline{b} + 1 \cdot \overline{a} \cdot \mathrm{b} + 0 \cdot \overline{a} \cdot \overline{b} = a \cdot \overline{b} + \overline{a} \cdot \mathrm{b}$$

$$P = f_2(a,b) = 0 \cdot a \cdot b + 0 \cdot a \cdot \overline{b} + 0 \cdot \overline{a} \cdot \mathrm{b} + 1 \cdot \overline{a} \cdot \overline{b} = \overline{a} \cdot \overline{b}$$

Логическая схема, реализующая полученную функцию:



Телекоммуникационные сети

Определения

Сеть – это совокупность программных, аппаратных и коммуникационных средств, обеспечивающих эффективное распределение вычислительных ресурсов.

Сети условно делят на 3 категории:

- ✔ Локальные сети (LAN, Local Area Network);
- ✔ Городские сети (MAN, Metropolitan Area Network);
- ✔ Глобальные сети (WAN, Wide Area Network).

Телекоммуникационные сети

Коммутационное оборудование

Концентратор (хаб) - сетевое устройство, предназначенное для объединения нескольких устройств Ethernet в общий сегмен сети.

Коммутатор (свич) - устройство, предназначенное для соединения нескольких узлов компьютерной сети в пределах одного или нескольких сегментов сети.

Повторитель (репитер) - предназначено для увеличения расстояния сетевого соединения путём повторения электрического сигнала «один в один».

Маршрутизатор (роутер) – устройство, соединяющее сети разного типа, но использующее одну операционную систему. Маршрутизатор выполняет свои функции на сетевом уровне, поэтому он зависит от протоколов обмена данными, но не зависит от типа сети. С помощью двух адресов сети и адреса узла маршрутизатора однозначно выбирает определенную станцию сети. Маршрутизатор также может выбрать наилучший путь для передачи сообщения абоненту сети, фильтрует информацию, проходящую через него, направляя в одну из сетей только ту информацию, которая ей адресована. Кроме того, маршрутизатор обеспечивает балансировку нагрузки в сети, перенаправляя потоки сообщений по свободным каналам связи.

Мост – устройство, соединяющее две сети, использующие одинаковые методы передачи данных. Сети, которые объединяет мост, должны иметь одинаковые сетевые уровни модели взаимодействия открытых систем, нижние уровни могут иметь некоторые отличия. Для сети персональных компьютеров мост – отдельная ЭВМ со специальным программным обеспечением и дополнительной аппаратурой. Мост может соединять сети разных топологий, но работающие под управлением однотипных сетевых операционных систем.

Шлюз – устройство, позволяющее организовать обмен данными между двумя сетями, использующими различные протоколы взаимодействия. Шлюз служит для объединения ЛВС совершенно различных типов, работающих по существенно отличающимся друг от друга протоколами. Он выполняет свои функции на уровнях выше сетевого. Он не зависит от используемой передающей среды, но зависит от используемых протоколов обмена данными. Обычно шлюз выполняет преобразование между двумя протоколами. С помощью шлюзов можно подключить локальную вычислительную сеть к главному компьютеру, а также локальную сеть подключить к глобальной.

Основные топологии ЛВС

Топология ЛВС – это усредненная геометрическая схема соединений узлов сети.
Для локальных вычислительных сетей типичными топологиями являются: *кольцо, шина, звезда*.
Любую компьютерную сеть можно рассматривать как совокупность узлов.
Узел – любе устройство, непосредственно подключенное к передающей среде сети.

Все компьютеры в локальной сети соединены линиями связи. Геометрическое расположение линий связи относительно узлов сети и физическое подключение узлов к сети называется физической топологией. В зависимости от топологии различают сети: шинной, кольцевой, звездной, иерархической и произвольной структуры.

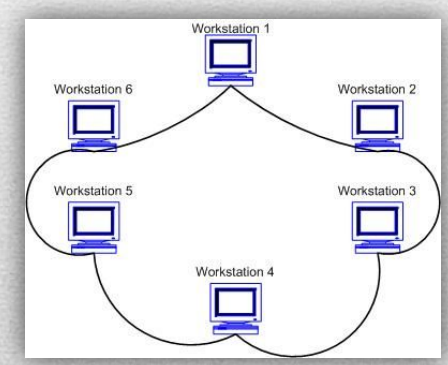
В настоящее время в локальных сетях используются следующие физические топологии:

- физическая «Шина» (bus);
- физическая «Звезда» (star);
- физическое «Кольцо» (ring);
- физическая «Звезда» и логическое «Кольцо» (Token Ring).

Топологии ЛВС

Кольцевая топология

Кольцевая топология – соединение узлов сети замкнутой кривой – кабелем передающей среды. Выход одного узла сети соединяется со входом другого. Информация по кольцу передается от узла к узлу. Каждый промежуточный узел между передатчиком и приемником ретранслирует посланное сообщение. Принимающий узел распознает и получает только адресованные ему сообщения. Кольцевая топология является идеальной для сетей, занимающих сравнительно небольшое пространство. В ней отсутствует центральный узел, что повышает надежность сети. Ретрансляция информации позволяет использовать в качестве передающей среды любые типы кабелей. Последовательность обслуживания узлов – снижает быстродействие такой сети, а выход из строя одного из узлов нарушает целостность кольца и требует принятия специальных мер для сохранения тракта передачи информации.



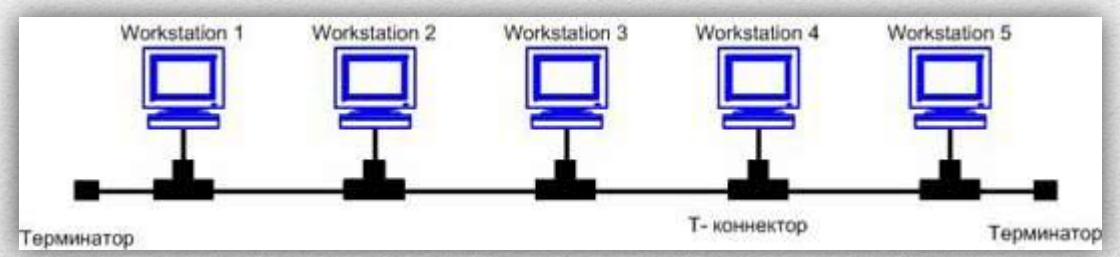
Данную сеть очень легко создавать и настраивать. К основному недостатку сетей топологии кольцо является то, что повреждение линии связи в одном месте или отказ ПК приводит к неработоспособности всей сети.

Как правило, в чистом виде топология “кольцо” не применяется из-за своей ненадёжности, поэтому на практике применяются различные модификации кольцевой топологии.

Топологии ЛВС

Шинная топология

Шинная топология – связана с использованием в качестве передающей среды коаксиального кабеля. Данные от передающего узла сети распространяются по шине в обе стороны. Промежуточные узлы не транслируют поступающих сообщений. Информация поступает на все узлы, но принимает сообщение только тот, которому оно адресовано. Такая топология обеспечивает высокое быстродействие ЛВС, легко наращивается, конфигурируется и адаптирована к различным системам, а так же устойчива к неисправностям отдельных узлов. Недостаток: нельзя использовать различные типы кабеля в пределах одной сети.



Преимущества сетей шинной топологии:

- отказ одного из узлов не влияет на работу сети в целом;
- сеть легко настраивать и конфигурировать;
- сеть устойчива к неисправностям отдельных узлов.

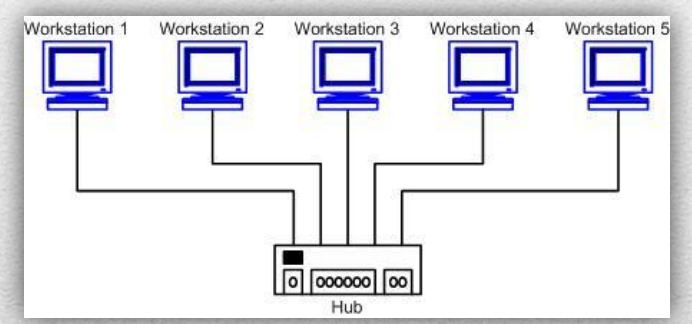
Недостатки сетей шинной топологии:

- разрыв кабеля может повлиять на работу всей сети;
- ограниченная длина кабеля и количество рабочих станций;
- трудно определить дефекты соединений

Топологии ЛВС

Звездообразная топология

Звездообразная топология – имеет центральный узел, к которому подключаются периферийные узлы. Каждый периферийный узел имеет свою отдельную линию связи с центральным узлом. Вся информация передается через центральный узел, который ретранслирует, переключает и маршрутизирует информационные потоки в сети. Такая топология значительно упрощает взаимодействие узлов ЛВС друг с другом, позволяет использовать более простые сетевые адаптеры, работоспособность ЛВС целиком зависит от центрального узла.



Преимущества сетей топологии звезда:

- легко подключить новый ПК;
- имеется возможность централизованного управления;
- сеть устойчива к неисправностям отдельных ПК и к разрывам соединения отдельных ПК.

Недостатки сетей топологии звезда:

- отказ хаба влияет на работу всей сети;
- большой расход кабеля;

Топологии ЛВС

Token Ring

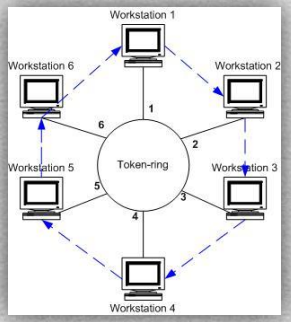
Token-Ring – топология основана на топологии "физическое кольцо с подключением типа звезда". В данной топологии все рабочие станции подключаются к центральному концентратору (Token Ring) как в топологии физическая звезда. Центральный концентратор - это интеллектуальное устройство, которое с помощью перемычек обеспечивает последовательное соединение выхода одной станции со входом другой станции. Другими словами с помощью концентратора каждая станция соединяется только с двумя другими станциями (предыдущей и последующей станциями). Таким образом, рабочие станции связаны петлей кабеля, по которой пакеты данных передаются от одной станции к другой и каждая станция ретранслирует эти посланные пакеты. В каждой рабочей станции имеется для этого приемо-передающее устройство, которое позволяет управлять прохождением данных в сети. Физически такая сеть построена по типу топологии “звезда”. Концентратор создаёт первичное (основное) и резервное кольца. Если в основном кольце произойдёт обрыв, то его можно обойти, воспользовавшись резервным кольцом, так как используется четырёхжильный кабель. Отказ станции или обрыв линии связи рабочей станции не вличет за собой отказ сети как в топологии кольцо, потому что концентратор отключет неисправную станцию и замкнет кольцо передачи данных.

Преимущества сетей топологии :

- топология обеспечивает равный доступ ко всем рабочим станциям;
- высокая надежность, так как сеть устойчива к неисправностям отдельных станций и к разрывам соединения отдельных станций.

Недостатки сетей топологии:

- большой расход кабеля и соответственно дорогостоящая разводка линий связи.



Топологии ЛВС

Методы доступа к передающей среде

Передающая среда является общим ресурсом для вех узлов сети. Чтобы получить возможность доступа к этому ресурсу из узла сети, необходимы специальные механизмы – методы доступа.

Метод доступа к передающей среде – метод, обеспечивающий выполнение совокупности правил, по которым узлы сети поучают доступ к ресурсу.

Существуют два основных класса методов доступа: *детерминированные* и *недетерминированные*.

При ***детерминированных*** методах доступа передающая среда распределяется между узлами с помощью специального механизма управления, гарантирующего передачу данных узла в течение некоторого, достаточно малого интервала времени.

Наиболее распространенными ***детерминированными*** методами доступа являются *метод опроса* и *метод передачи права*. Метод доступа используется преимущественно в сетях звездообразной топологии.

Метод передачи права применяется в сетях с кольцевой топологией. Он основан на передаче по сети специального сообщения – *маркера*.

Маркер – служебное сообщение определенного формата, в которое абоненты сети могут помещать свои информационные пакеты.

Маркер циркулирует по кольцу, и любой узел, имеющие данные для передачи, помещает их в собственный маркер, устанавливает признак занятости маркера и передает его по кольцу. Узел, которому было адресовано сообщение, принимает его, устанавливает признак подтверждения приема информации и отправляет маркер в кольцо.

Передающий маркер циркулирует по кольцу, и любой узел, имеющий данные для передачи, помещает их в свободный маркер, устанавливает признак занятости маркера и отправляет его в сеть. Существуют методы доступа, использующие несколько маркеров.

Недетерминированные – случайные методы доступа предусматривают конкуренцию всех узлов сети за право передачи. Возможны одновременные попытки передачи со стороны нескольких узлов, в результате чего возникают коллизии.

Наиболее распространенным недетерминированным методом доступа является множественный метод доступа с контролем несущей частоты и обнаружением коллизий (CSMA/CD). Контроль несущей частоты заключается в том, что узел, желающий передать сообщение, «прослушивает» передающую среду, ожидая ее освобождения. Если сред свободна, узел начинает передачу.

Сетевая модель OSI

Сетевая модель OSI (ЭМВОС) (базовая *эталонная модель* взаимодействия открытых систем, Open Systems Interconnection Basic Reference Model, 1978 г.) — абстрактная сетевая модель для коммуникаций и разработки сетевых протоколов. Предлагает взгляд на компьютерную сеть с точки зрения измерений. Каждое измерение обслуживает свою часть процесса взаимодействия. Благодаря такой структуре совместная работа сетевого оборудования и программного обеспечения становится гораздо проще и прозрачнее.

В настоящее время основным используемым стеком протоколов является TCP/IP, разработка которого не была связана с моделью OSI и к тому же была совершена до её принятия.

Сетевая модель OSI

Модель состоит из семи уровней, расположенных друг над другом. Уровни взаимодействуют друг с другом (по «вертикали») посредством интерфейсов, и могут взаимодействовать с параллельным уровнем другой системы (по «горизонтали») с помощью протоколов. Каждый уровень может взаимодействовать только со своими соседями и выполнять отведённые только ему функции. Подробнее можно посмотреть на рисунке.

Тип данных	Уровень	Функции
Данные	7. Прикладной уровень	Доступ к сетевым службам
	6. Уровень представления	Представление и кодирование данных
	5. Сеансовый уровень	Управление сеансом связи
Сегменты	4. Транспортный	Прямая связь между конечными пунктами и надежность
Пакеты	3. Сетевой	Определение маршрута и логическая адресация
Кадры	2. Канальный	Физическая адресация
Биты	1. Физический уровень	Работа со средой передачи, сигналами и двоичными данными

Сетевая модель OSI

- **Прикладной (приложений) уровень (Application layer)**

Верхний уровень модели, обеспечивает взаимодействие пользовательских приложений с сетью. Этот уровень позволяет приложениям использовать сетевые службы, такие как удалённый доступ к файлам и базам данных, пересылка электронной почты. Также отвечает за передачу служебной информации, предоставляет приложениям информацию об ошибках и формирует запросы к уровню представления.

Пример: HTTP, POP3, SMTP, FTP, XMPP, OSCAR, Modbus, SIP, TELNET

Сетевая модель OSI

Уровень представления (Presentation layer)

Этот уровень отвечает за преобразование протоколов и кодирование/декодирование данных. Запросы приложений, полученные с прикладного уровня, он преобразует в формат для передачи по сети, а полученные из сети данные преобразует в формат, понятный приложениям. На этом уровне может осуществляться сжатие/распаковка или кодирование/декодирование данных, а также перенаправление запросов другому сетевому ресурсу, если они не могут быть обработаны локально.

Другой функцией, выполняемой на уровне представлений, является шифрование данных, которое применяется в тех случаях, когда необходимо защитить передаваемую информацию от приема несанкционированными получателями. Чтобы решить эту задачу, процессы и коды, находящиеся на уровне представлений, должны выполнить преобразование данных. На этом уровне существуют и другие подпрограммы, которые сжимают тексты и преобразовывают графические изображения в битовые потоки, так что они могут передаваться по сети.

Пример: AFP — Apple Filing Protocol, ICA — Independent Computing Architecture, LPP — Lightweight Presentation Protocol, NCP — NetWare Core Protocol NDR — Network Data Representation RDP — Remote Desktop Protocol, XDR — eXternal Data Representation, X.25 PAD — Packet Assembler/Disassembler Protocol

Пример: Чтобы понять, как это работает, представим, что имеются две системы. Одна использует для представления данных расширенный двоичный код обмена информацией EBCDIC, например, это может быть мейнфрейм компании IBM, а другая — американский стандартный код обмена информацией ASCII (его используют большинство других производителей компьютеров). Если этим двум системам необходимо обменяться информацией, то нужен уровень представлений, который выполнит преобразование и осуществит перевод между двумя различными форматами.

Сетевая модель OSI

Сеансовый уровень (Session layer)

Сеансовый уровень - 5-й уровень модели отвечает за поддержание сеанса связи, позволяя приложениям взаимодействовать между собой длительное время. Уровень управляет созданием/завершением сеанса, обменом информацией, синхронизацией задач, определением права на передачу данных и поддержанием сеанса в периоды неактивности приложений. Синхронизация передачи обеспечивается помещением в поток данных контрольных точек, начиная с которых возобновляется процесс при нарушении взаимодействия.

Пример: ADSP (AppleTalk Data Stream Protocol), ASP (AppleTalk Session Protocol), H.245 (Call Control Protocol for Multimedia Communication), ISO-SP (OSI Session Layer Protocol (X.225, ISO 8327)), iSNS (Internet Storage Name Service), L2F (Layer 2 Forwarding Protocol), L2TP (Layer 2 Tunneling Protocol), NetBIOS (Network Basic Input Output System), PAP (Password Authentication Protocol), PPTP (Point-to-Point Tunneling Protocol), RPC (Remote Procedure Call Protocol), RTCP (Real-time Transport Control Protocol), SMPP (Short Message Peer-to-Peer), SCP (Secure Copy Protocol), ZIP (Zone Information Protocol), SDP (Sockets Direct Protocol).

Сетевая модель OSI

Транспортный уровень (Transport layer)

Транспортный уровень - 4-й уровень модели предназначен для доставки данных без ошибок, потерь и дублирования в той последовательности, как они были переданы. При этом не важно, какие данные передаются, откуда и куда, то есть он предоставляет сам механизм передачи. Блоки данных он разделяет на фрагменты (UDP-датаграмма, TCP-сегмент), размер которых зависит от протокола, короткие объединяет в один, а длинные разбивает. Пример: TCP, UDP.

Существует множество классов протоколов транспортного уровня, начиная от протоколов, предоставляющих только основные транспортные функции (например, функции передачи данных без подтверждения приема), и заканчивая протоколами, которые гарантируют доставку в пункт назначения нескольких пакетов данных в надлежащей последовательности, мультиплексируют несколько потоков данных, обеспечивают механизм управления потоками данных и гарантируют достоверность принятых данных.

Пример: ATP (AppleTalk Transaction Protocol), CUDP (Cyclic UDP), DCCP (Datagram Congestion Control Protocol), FCP (Fiber Channel Protocol), IL (IL Protocol), NBF (NetBIOS Frames protocol), NCP (NetWare Core Protocol), SCTP (Stream Control Transmission Protocol), SPX (Sequenced Packet Exchange), SST (Structured Stream Transport), TCP (Transmission Control Protocol), UDP (User Datagram Protocol).

Сетевая модель OSI

Сетевой уровень (Network layer)

Сетевой уровень - 3-й уровень сетевой модели OSI предназначен для определения пути передачи данных. Отвечает за трансляцию логических адресов и имён в физические, определение кратчайших маршрутов, коммутацию и маршрутизацию, отслеживание неполадок и «заторов» в сети. Протоколы сетевого уровня маршрутизируют данные от источника к получателю. На этом уровне работает маршрутизатор (роутер).

Пример: IP/IPv4/IPv6 (Internet Protocol), IPX (Internetwork Packet Exchange, протокол межсетевого обмена), X.25 (частично этот протокол реализован на уровне 2) CLNP (сетевой протокол без организации соединений), IPsec (Internet Protocol Security), ICMP (Internet Control Message Protocol), RIP (Routing Information Protocol), OSPF (Open Shortest Path First), ARP (Address Resolution Protocol).

Сетевая модель OSI

Канальный уровень (*Data Link layer*)

Канальный уровень - уровень предназначен для обеспечения взаимодействия сетей на физическом уровне и контроля за ошибками, которые могут возникнуть. Полученные с физического уровня данные он упаковывает во фреймы, проверяет на целостность, если нужно, исправляет ошибки (посылает повторный запрос поврежденного кадра) и отправляет на сетевой уровень. Канальный уровень может взаимодействовать с одним или несколькими физическими уровнями, контролируя и управляя этим взаимодействием.

Спецификация IEEE 802 разделяет этот уровень на два подуровня — MAC (Media Access Control) регулирует доступ к разделяемой физической среде, LLC (Logical Link Control) обеспечивает обслуживание сетевого уровня.

На этом уровне работают коммутаторы, мосты.

В программировании этот уровень представляет драйвер сетевой платы, в операционных системах имеется программный интерфейс взаимодействия канального и сетевого уровней между собой, это не новый уровень, а просто реализация модели для конкретной ОС. Примеры таких интерфейсов: ODI, NDIS

Протоколы: ARCnet, ATM, Cisco Discovery Protocol (CDP), Controller Area Network (CAN), Econet, Ethernet, Ethernet Automatic Protection Switching (EAPS), Fiber Distributed Data Interface (FDDI), Frame Relay, High-Level Data Link Control (HDLC), IEEE 802.2 (provides LLC functions to IEEE 802 MAC layers), IEEE 802.11 wireless LAN, Link Access Procedures, D channel (LAPD), LocalTalk, Multiprotocol Label Switching (MPLS), Point-to-Point Protocol (PPP), Serial Line Internet Protocol (SLIP) (obsolete), Spanning tree protocol, StarLan, Token ring, Unidirectional Link Detection (UDLD), x.25.

Сетевая модель OSI

Физический уровень (Physical layer)

Самый нижний уровень модели предназначен непосредственно для передачи потока данных. Осуществляет передачу электрических или оптических сигналов в кабель или в радиоэфир и, соответственно, их приём и преобразование в биты данных в соответствии с методами кодирования цифровых сигналов. Другими словами, осуществляет интерфейс между сетевым носителем и сетевым устройством.

Определяемые на данном уровне параметры: тип передающей среды, тип модуляции сигнала, уровни логических «0» и «1» и т. д.

На этом уровне работают концентраторы (хабы), повторители (ретрансляторы) сигнала и медиаконверторы.

Функции физического уровня реализуются на всех устройствах, подключенных к сети. Со стороны компьютера функции физического уровня выполняются сетевым адаптером или последовательным портом. К физическому уровню относятся физические, электрические и механические интерфейсы между двумя системами. Физический уровень определяет такие виды среды передачи данных как оптоволокно, витая пара, коаксиальный кабель, спутниковый канал передач данных и т. п. Стандартными типами сетевых интерфейсов, относящимися к физическому уровню, являются: V.35, RS-232C, RS-485, RJ-11, RJ-45, разъемы AUI и BNC.

Протоколы: IRDA, USB, EIA RS-232, EIA-422, EIA-423, RS-449, RS-485, Ethernet (включая 10BASE-T, 10BASE2, 10BASE5, 100BASE-TX, 100BASE-FX, 100BASE-T, 1000BASE-T, 1000BASE-SX и другие), 802.11Wi-Fi, DSL, ISDN, SONET/SDH, GSM Um radio interface, IEEE 802.15, ITU и ITU-T, Firewire, TransferJet, Etherloop, ARINC 818, G.hn/G.9960.

Семейство TCP/IP

Семейство TCP/IP имеет три транспортных протокола: TCP, полностью соответствующий OSI, обеспечивающий проверку получения данных; UDP, отвечающий транспортному уровню только наличием порта, обеспечивающий обмен датаграммами между приложениями, не гарантирующий получения данных; и SCTP, разработанный для устранения некоторых недостатков TCP, в который добавлены некоторые новшества. (В семействе TCP/IP есть ещё около двухсот протоколов, самым известным из которых является служебный протокол ICMP, используемый для внутренних нужд обеспечения работы; остальные также не являются транспортными протоколами.)

5	Прикладной «7 уровень»	напр., HTTP, RTP, FTP, DNS (RIP, работающий поверх UDP, и BGP, работающий поверх TCP, являются частью сетевого уровня)
4	Транспортный	напр., TCP, UDP, SCTP, DCCP (протоколы маршрутизации, подобные OSPF, что работают поверх IP, являются частью сетевого уровня)
3	Сетевой	Для TCP/IP это IP (IP) (вспомогательные протоколы, вроде ICMP и IGMP, работают поверх IP, но тоже относятся к сетевому уровню; протокол ARP является самостоятельным вспомогательным протоколом, работающим поверх физического уровня)
2	Канальный	Ethernet, IEEE 802.11 Wireless Ethernet, SLIP, Token Ring, ATM и MPLS
1	Физический	напр., физическая среда и принципы кодирования информации, T1, E1

Семейство IPX/SPX

В семействе IPX/SPX порты (называемые «сокеты» или «гнёзда») появляются в протоколе сетевого уровня IPX, обеспечивая обмен датаграммами между приложениями (операционная система резервирует часть сокетов для себя). Протокол SPX, в свою очередь, дополняет IPX всеми остальными возможностями транспортного уровня в полном соответствии с OSI.

В качестве адреса хоста IPX использует идентификатор, образованный из четырёхбайтного номера сети (назначаемого маршрутизаторами) и MAC-адреса сетевого адаптера.

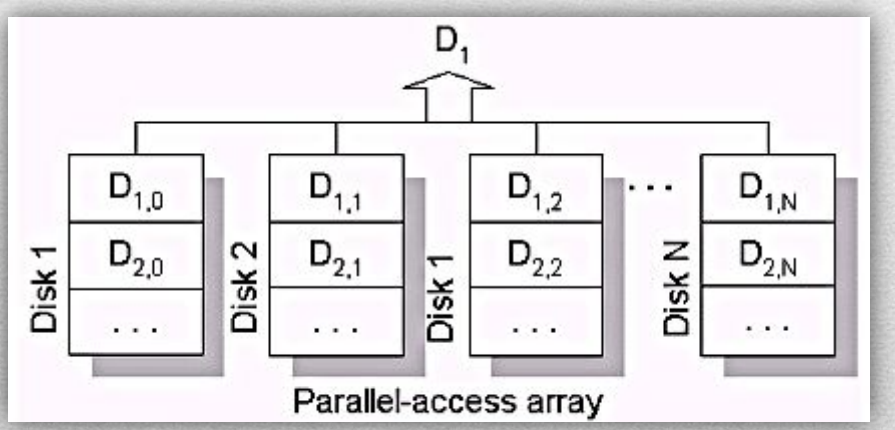
RAID Levels

В современной компьютерной индустрии в качестве вторичной системы хранения данных повсеместно используются магнитные диски, ибо, несмотря на все свои недостатки, они обладают наилучшими характеристиками для соответствующего типа устройств при доступной цене.

RAID Levels

Увеличиваем быстродействие

Невозможность значительного увеличения технологических параметров магнитных дисков влечет за собой необходимость поиска других путей, одним из которых является параллельная обработка.
Если расположить блок данных по N дискам некоторого массива и организовать это размещение так, чтобы существовала возможность одновременного считывания информации, то этот блок можно будет считать в N раз быстрее, (без учёта времени формирования блока). Поскольку все данные передаются параллельно, это архитектурное решение называется parallel-access array (массив с параллельным доступом).

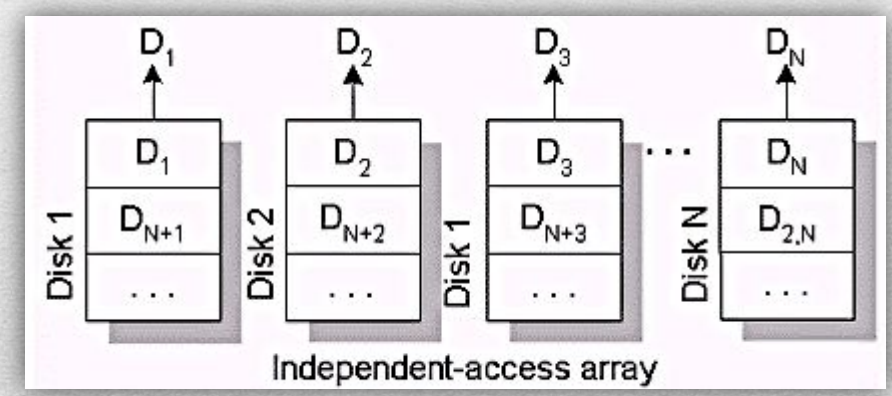


RAID Levels

Распределение нагрузки

Массивы с параллельным доступом обычно используются для приложений, требующих передачи данных большого размера.

Некоторые задачи, наоборот, характерны большим количеством малых запросов. К таким задачам относятся, например, задачи обработки баз данных. Располагая записи базы данных по дискам массива, можно распределить загрузку, независимо позиционируя диски. Такую архитектуру принято называть independent-access array (массив с независимым доступом).



RAID Levels

Увеличение отказоустойчивости

К сожалению, при увеличении количества дисков в массиве, надежность всего массива уменьшается. При независимых отказах и экспоненциальном законе распределения наработки на отказ, MTTF всего массива (mean time to failure - среднее время безотказной работы) вычисляется по формуле MTTFarray = MMTFhdd/Nhdd (MMTFhdd - среднее время безотказной работы одного диска; NHDD - количество дисков).

Таким образом, возникает необходимость повышения отказоустойчивости дисковых массивов. Для повышения отказоустойчивости массивов используют избыточное кодирование. Существует два основных типа кодирования, которые применяются в избыточных дисковых массивах - это дублирование и четность.

Дублирование, или зеркализация - наиболее часто используются в дисковых массивах. Простые зеркальные системы используют две копии данных, каждая копия размещается на отдельных дисках. Это схема достаточно проста и не требует дополнительных аппаратных затрат, но имеет один существенный недостаток - она использует 50% дискового пространства для хранения копии информации.

Второй способ реализации избыточных дисковых массивов - использование избыточного кодирования с помощью вычисления четности. Четность вычисляется как операция XOR всех символов в слове данных. Использование четности в избыточных дисковых массивах уменьшает накладные расходы до величины, исчисляемой формулой: HPhdd=1/Nhdd (HPhdd - накладные расходы; Nhdd - количество дисков в массиве).

RAID Levels

История и развитие RAID

Несмотря на то, что системы хранения данных, основанные на магнитных дисках, производятся уже 50 лет, массовое производство отказоустойчивых систем началось совсем недавно. Дисковые массивы с избыточностью данных, которые принято называть RAID (redundant arrays of inexpensive disks - избыточный массив недорогих дисков) были представлены исследователями (Петтерсон, Гибсон и Катц) из Калифорнийского университета в Беркли в 1987 году. Но широкое распространение RAID системы получили только тогда, когда диски, которые подходят для использования в избыточных массивах стали доступны и достаточно производительны. Со времени представления официального доклада о RAID в 1988 году, исследования в сфере избыточных дисковых массивов начали бурно развиваться, в попытке обеспечить широкий спектр решений в сфере компромисса - цена-производительность-надежность.

RAID 0 был представлен индустрией как определение не отказоустойчивого дискового массива. RAID 1 был определен как зеркальный дисковый массив. RAID 2 зарезервирован для массивов, которые применяют код Хемминга. Уровни RAID 3, 4, 5 используют четность для защиты данных от одиночных неисправностей. Именно эти уровни, включительно по 5-й были представлены в Беркли, и эта систематика RAID была принята как стандарт де-факто.

Для стандартизации продуктов RAID в 1992 году был организован промышленный консорциум - RAID Advisory Board.

Уровни RAID 3,4,5 достаточно популярны, имеют хороший коэффициент использования дискового пространства, но у них есть один существенный недостаток - они устойчивы только к одиночным неисправностям. Особенно это актуально при использовании большого количества дисков, когда вероятность одновременного простоя более чем одного устройства увеличивается. Кроме того, для них характерно длительное восстановление, что также накладывает некоторые ограничения для их использования.

На сегодняшний день разработано достаточно большое количество архитектур, которые обеспечивают работоспособность массива при одновременном отказе любых двух дисков без потери данных. Среди всего множества стоит отметить two-dimensional parity (двухпространственная четность) и EVENODD, которые для кодирования используют четность, и RAID 6, в котором используется кодирование Reed-Solomon.

В схеме использующей двухпространственную четность, каждый блок данных участвует в построении двух независимых кодовых слов. Таким образом, если из строя выходит второй диск в том же кодовом слове, для реконструкции данных используется другое кодовое слово.

RAID Levels

Минимальная избыточность в таком массиве достигается при равном количестве столбцов и строчек. И равна: 2 x Square (NDisk) (в "квадрат").

Если же двухпространственный массив не будет организован в "квадрат", то при реализации вышеуказанной схемы избыточность будет выше.

Архитектура EVENODD имеет похожую на двухпространственную четность схему отказоустойчивости, но другое размещение информационных блоков, которое гарантирует минимальное избыточное использование емкостей. Так же как и в двухпространственной четности каждый блок данных участвует в построении двух независимый кодовых слов, но слова размещены таким образом, что коэффициент избыточности постоянен (в отличие от предыдущей схемы) и равен: 2 x Square (NDisk).

Используя два символа для проверки, четность и недвоичные коды, слово данных может быть сконструировано таким образом, чтобы обеспечить отказоустойчивость при возникновении двойной неисправности. Такая схема известна как RAID 6. Недвоичный код, построенный на основе Reed-Solomon кодирования, обычно вычисляется с использованием таблиц или как итерационный процесс с использованием линейных регистров с обратной связью, а это - относительно сложная операция, требующая специализированных аппаратных средств.

Учитывая то, что применение классических вариантов RAID, реализующих для многих приложений достаточную отказоустойчивость, имеет часто недопустимо низкое быстродействие, исследователи время от времени реализуют различные ходы, которые помогают увеличить быстродействие RAID систем.

В 1996 г. Саведж и Вилкс предложили AFRAID - часто избыточный массив независимых дисков (A Frequently Redundant Array of Independent Disks). Эта архитектура в некоторой степени приносит отказоустойчивость в жертву быстродействию. Делая попытку компенсировать проблему малой записи (small-write problem), характерную для массивов RAID 5-го уровня, разрешается оставлять стрипинг без вычисления четности на некоторый период времени. Если диск, предназначенный для записи четности, занят, то ее запись откладывается. Теоретически доказано, что 25% уменьшение отказоустойчивости может увеличить быстродействие на 97%. AFRAID фактически изменяет модель отказов массивов устойчивых к одиночным неисправностям, поскольку кодовое слово, которое не имеет обновленной четности, восприимчиво к отказам дисков.

Вместо того чтобы приносить в жертву отказоустойчивость, можно использовать такие традиционные способы увеличения быстродействия, как кэширование. Учитывая то, что дисковый трафик имеет пульсирующий характер, можно использовать кеш память с обратной записью (writeback cache) для хранения данных в момент, когда диски заняты. И если кеш-память будет выполнена в виде энергонезависимой памяти, тогда, в случае исчезновения питания, данные будут сохранены. Кроме того, отложенные дисковые операции, дают возможность объединить в произвольном порядке малые блоки для выполнения более эффективных дисковых операций.

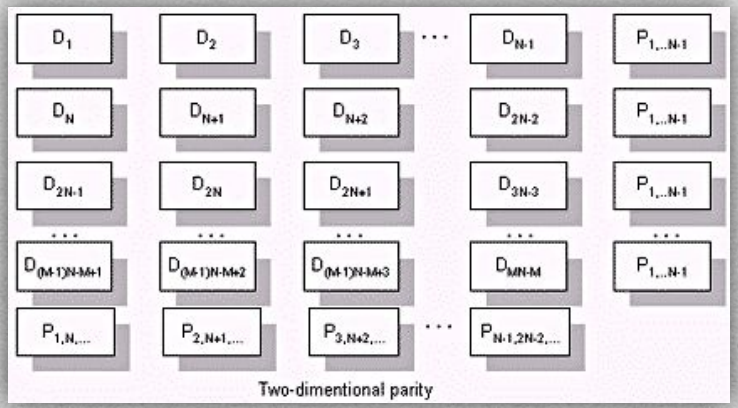
RAID Levels

Существует также множество архитектур, которые, принося в жертву объем, увеличивают быстродействие. Среди них - отложенная модификация на log диск и разнообразные схемы модификации логического размещение данных в физическое, которые позволяют распределять операции в массиве более эффективно.

Один из вариантов - parity logging (регистрация четности), который предполагает решение проблемы малой записи (small-write problem) и более эффективного использования дисков. Регистрация четности предполагает отложение изменения четности в RAID 5, записывая ее в FIFO log (журнал регистраций типа FIFO), который размещен частично в памяти контроллера и частично на диске. Учитывая то, что доступ к полному треку в среднем в 10 раз более эффективен, чем доступ к сектору, с помощью регистрации четности собираются большие количества данных модифицированной четности, которые потом все вместе записываются на диск, предназначенный для хранения четности по всему треку.

Архитектура floating data and parity (плавающие данные и четность), которая разрешает перераспределить физическое размещение дисковых блоков. Свободные сектора размещаются на каждом цилиндре для уменьшения rotational latency (задержки вращения), данные и четность размещаются на этих свободных местах. Для того, чтобы обеспечить работоспособность при исчезновении питания, карту четности и данных нужно сохранять в энергонезависимой памяти. Если потерять карту размещения все данные в массиве будут потеряны.

Virtual stripping - представляет собой архитектуру floating data and parity с использованием writeback cache. Естественно реализуя положительные стороны обеих.



RAID Levels

Кроме того, существуют и другие способы повышения быстродействия, например распределение RAID операций. В свое время фирма Seagate встроила поддержку RAID операций в свои диски с интерфейсом Fibre Chanel и SCSI. Что дало возможность уменьшить трафик между центральным контроллером и дисками в массиве для систем RAID 5. Это было кардинальным новшеством в сфере реализаций RAID, но технология не получила путевки в жизнь, так как некоторые особенности Fibre Chanel и SCSI стандартов ослабляют модель отказов для дисковых массивов.

Для того же RAID 5 была представлена архитектура TickerTAIP. Выглядит она следующим образом - центральный механизм управления originator node (узел-инициатор) получает запросы пользователя, выбирает алгоритм обработки и затем передает работу с диском и четность worker node (рабочий узел). Каждый рабочий узел обрабатывает некоторое подмножество дисков в массиве. Как и в модели фирмы Seagate, рабочие узлы передают данные между собой без участия узла-инициатора. В случае отказа рабочего узла, диски, которые он обслуживал, становятся недоступными. Но если кодовое слово построено так, что каждый его символ обрабатывается отдельным рабочим узлом, то схема отказоустойчивости повторяет RAID 5. Для предупреждения отказов узла-инициатора он дублируется, таким образом, мы получаем архитектуру, устойчивую к отказам любого ее узла. При всех своих положительных чертах эта архитектура страдает от проблемы "ошибки записи" ("write hole"). Что подразумевает возникновение ошибки при одновременном изменении кодового слова несколькими пользователями и отказа узла.

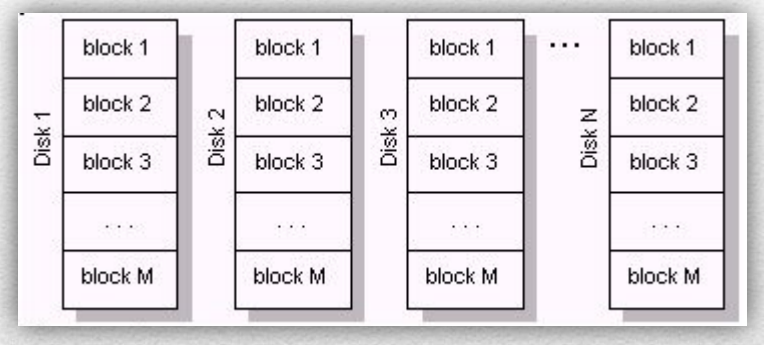
Следует также упомянуть достаточно популярный способ быстрого восстановления RAID - использование свободного диска (spare). При отказе одного из дисков массива, RAID может быть восстановлен с использованием свободного диска вместо вышедшего из строя. Основной особенностью такой реализации есть то, что система переходит в свое предыдущее (отказоустойчивое состояние без внешнего вмешательства). При использовании архитектуры распределения свободного диска (distributed sparing), логические блоки spare диска распределяются физически по всем дискам массива, снимая необходимость перестройки массива при отказе диска.

Для того чтобы избежать проблемы восстановления, характерной для классических уровней RAID, используется также архитектура, которая носит название parity declustering (распределение четности). Она предполагает размещение меньшего количества логических дисков с большим объемом на физические диски меньшего объема, но большего количества. При использовании этой технологии время реакции системы на запрос во время реконструкции улучшается более чем вдвое, а время реконструкции - значительно уменьшается.

RAID Levels

Архитектура основных уровней RAID

Перед рассмотрением примем некоторые допущения. Для демонстрации принципов построения RAID систем рассмотрим набор из N дисков (для упрощения N будем считать четным числом), каждый из которых состоит из M блоков.



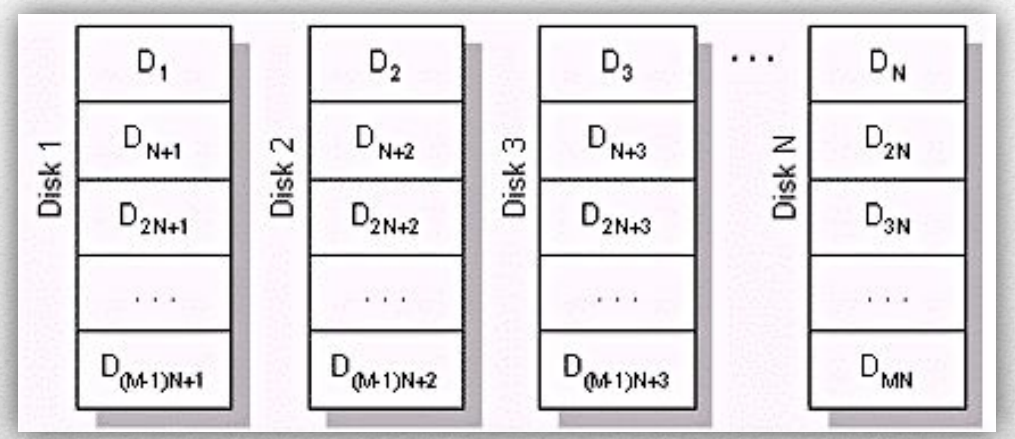
Данные будем обозначать - Dm,n, где m - число блоков данных, n - число подблоков, на которые разбивается блок данных D.

Диски могут подключаться как к одному, так и к нескольким каналам передачи данных. Использование большего количества каналов увеличивает пропускную способность системы.

RAID Levels

RAID 0. Дисковый массив без отказоустойчивости (Striped Disk Array without Fault Tolerance)

Представляет собой дисковый массив, в котором данные разбиваются на блоки, и каждый блок записываются (или же считывается) на отдельный диск. Таким образом, можно осуществлять несколько операций ввода-вывода одновременно.



Преимущества:

- наивысшая производительность для приложений требующих интенсивной обработки запросов ввода/вывода и данных большого объема;
- простота реализации;
- низкая стоимость на единицу объема.

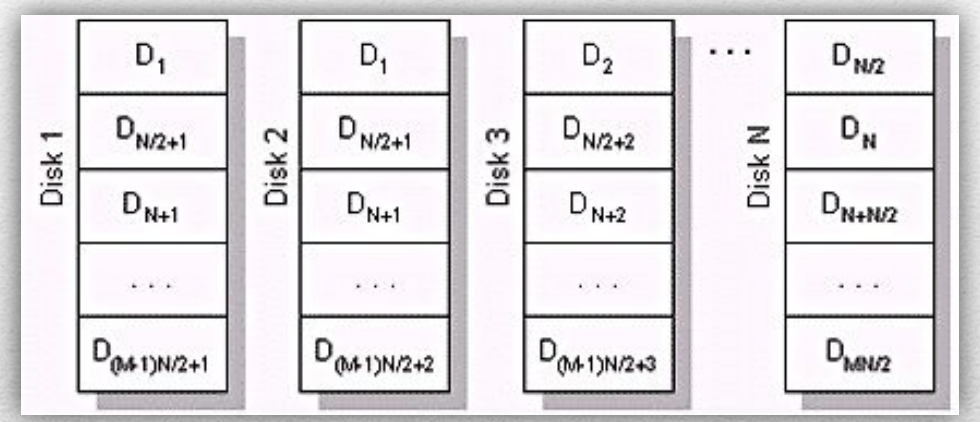
Недостатки:

- не отказоустойчивое решение;
- отказ одного диска влечет за собой потерю всех данных массива.

RAID Levels

RAID 1. Дисковый массив с дублированием или зеркалка (mirroring)

Зеркалирование - традиционный способ для повышения надежности дискового массива небольшого объема. В простейшем варианте используется два диска, на которые записывается одинаковая информация, и в случае отказа одного из них остается его дубль, который продолжает работать в прежнем режиме.



Преимущества:

- простота реализации;
- простота восстановления массива в случае отказа (копирование);
- достаточно высокое быстродействие для приложений с большой интенсивностью запросов.

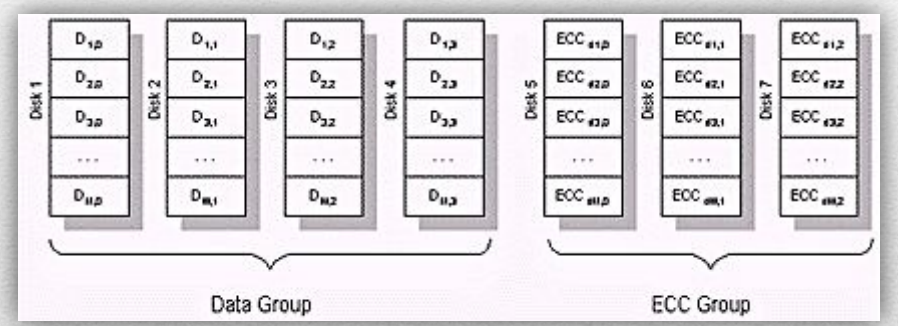
Недостатки:

- высокая стоимость на единицу объема - 100% избыточность;
- невысокая скорость передачи данных.

RAID Levels

RAID 2. Отказоустойчивый дисковый массив с использованием кода Хемминга (Hamming Code ECC).

Избыточное кодирование, которое используется в RAID 2, носит название кода Хемминга. Код Хемминга позволяет исправлять одиночные и обнаруживать двойные неисправности. Сегодня активно используется в технологии кодирования данных в оперативной памяти типа ECC. И кодировании данных на магнитных дисках.



Преимущества:

- быстрая коррекция ошибок ("на лету");
- очень высокая скорость передачи данных больших объемов;
- при увеличении количества дисков, накладные расходы уменьшаются;
- достаточно простая реализация.

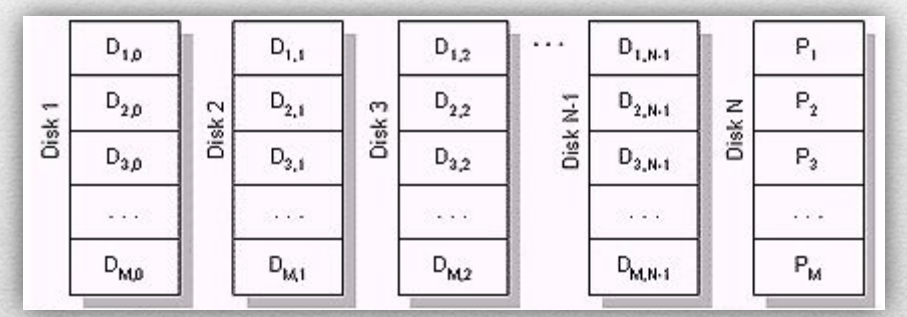
Недостатки:

- высокая стоимость при малом количестве дисков;
- низкая скорость обработки запросов (не подходит для систем ориентированных на обработку транзакций).

RAID Levels

RAID 3. Отказоустойчивый массив с параллельной передачей данных и четностью (Parallel Transfer Disks with Parity)

Данные разбиваются на подблоки на уровне байт и записываются одновременно на все диски массива кроме одного, который используется для четности. Использование RAID 3 решает проблему большой избыточности в RAID 2. Большинство контрольных дисков, используемых в RAID уровня 2, нужны для определения положения неисправного разряда. Но в этом нет нужды, так как большинство контроллеров в состоянии определить, когда диск отказал при помощи специальных сигналов, или дополнительного кодирования информации, записанной на диск и используемой для исправления случайных сбоев.



Преимущества:

- очень высокая скорость передачи данных;
- отказ диска мало влияет на скорость работы массива;
- малые накладные расходы для реализации избыточности.

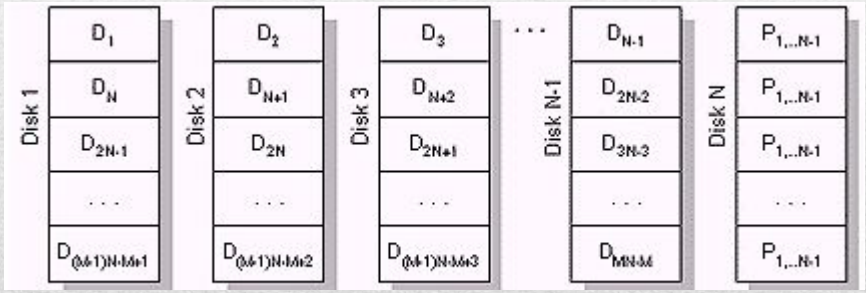
Недостатки:

- непростая реализация;
- низкая производительность при большой интенсивности запросов данных небольшого объема.

RAID Levels

RAID 4. Отказоустойчивый массив независимых дисков с разделяемым диском четности (Independent Data disks with shared Parity disk)

Данные разбиваются на блочном уровне. Каждый блок данных записывается на отдельный диск и может быть прочитан отдельно. Четность для группы блоков генерируется при записи и проверяется при чтении. RAID уровня 4 повышает производительность передачи небольших объемов данных за счет параллелизма, давая возможность выполнять более одного обращения по вводу/выводу одновременно. Главное отличие между RAID 3 и 4 состоит в том, что в последнем, расслоение данных выполняется на уровне секторов, а не на уровне битов или байтов.



Преимущества:

- очень высокая скорость чтения данных больших объемов;
- высокая производительность при большой интенсивности запросов чтения данных;
- малые накладные расходы для реализации избыточности.

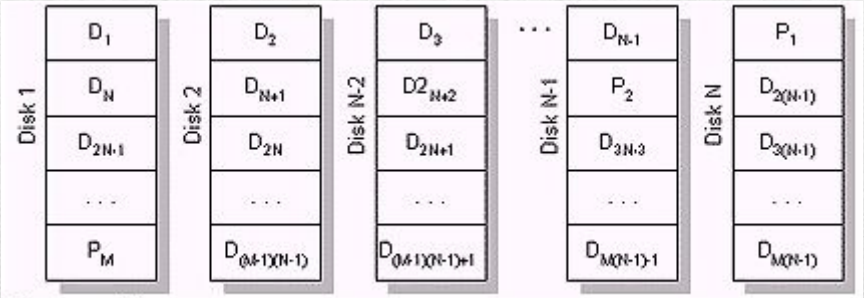
Недостатки:

- достаточно сложная реализация;
- очень низкая производительность при записи данных;
- сложное восстановление данных;
- низкая скорость чтения данных малого объема при единичных запросах;
- асимметричность быстродействия относительно чтения и записи.

RAID Levels

RAID 5. Отказоустойчивый массив независимых дисков с распределенной четностью (Independent Data disks with distributed parity blocks)

Этот уровень похож на RAID 4, но в отличие от предыдущего четность распределяется циклически по всем дискам массива. Это изменение позволяет увеличить производительность записи небольших объемов данных в многозадачных системах. Если операции записи спланировать должным образом, то, возможно, параллельно обрабатывать до N/2 блоков, где N - число дисков в группе.



Преимущества:

- высокая скорость записи данных;
- достаточно высокая скорость чтения данных;
- высокая производительность при большой интенсивности запросов чтения/записи данных;
- малые накладные расходы для реализации избыточности.

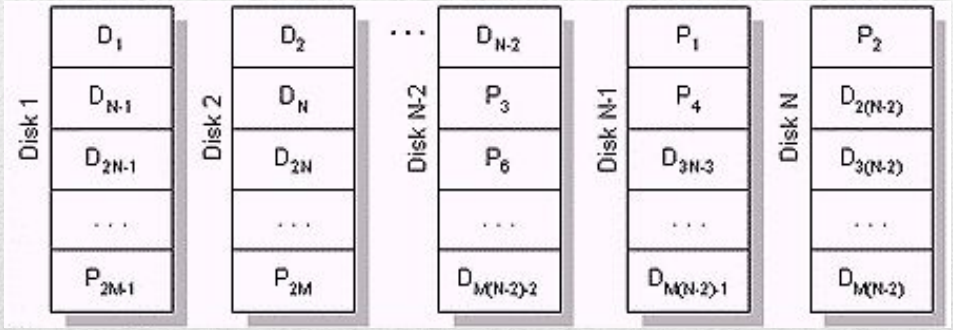
Недостатки:

- скорость чтения данных ниже, чем в RAID 4;
- низкая скорость чтения/записи данных малого объема при единичных запросах;
- достаточно сложная реализация;
- сложное восстановление данных.

RAID Levels

RAID 6. Отказоустойчивый массив независимых дисков с двумя независимыми распределенными схемами четности (Independent Data disks with two independent distributed parity schemes)

Данные разбиваются на блочном уровне, аналогично RAID 5, но в дополнение к предыдущей архитектуре используется вторая схема для повышения отказоустойчивости. Эта архитектура является устойчивой к двойным отказам. Однако при выполнении логической записи реально происходит шесть обращений к диску, что сильно увеличивает время обработки одного запроса.



Преимущества:

- высокая отказоустойчивость;
- достаточно высокая скорость обработки запросов;
- относительно малые накладные расходы для реализации избыточности.

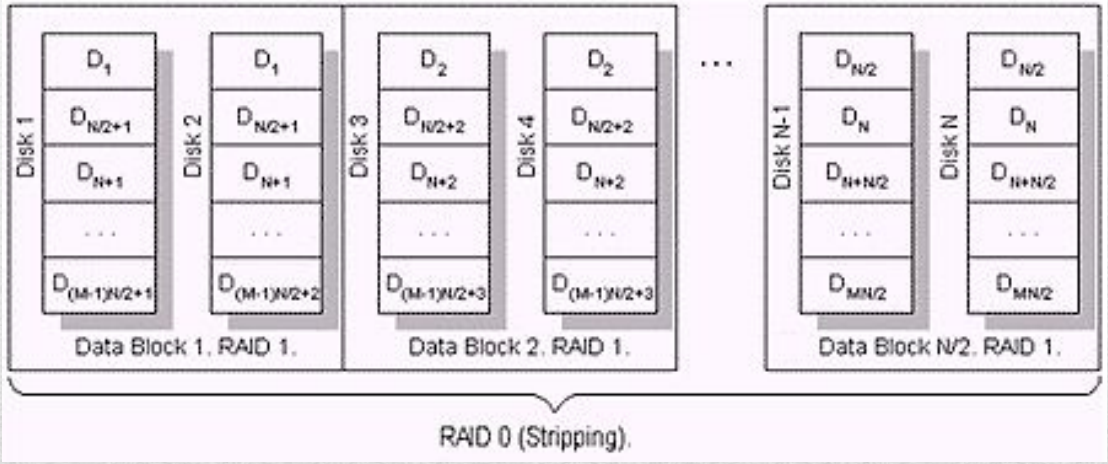
Недостатки:

- очень сложная реализация;
- сложное восстановление данных;
- очень низкая скорость записи данных.
- Современные RAID контроллеры позволяют комбинировать различные уровни RAID. Таким образом, можно реализовать системы, которые объединяют в себе достоинства различных уровней, а также системы с большим количеством дисков. Обычно это комбинация нулевого уровня (stripping) и какого либо отказоустойчивого уровня.

RAID Levels

RAID 10. Отказоустойчивый массив с дублированием и параллельной обработкой

Эта архитектура являет собой массив типа RAID 0, сегментами которого являются массивы RAID 1. Он объединяет в себе очень высокую отказоустойчивость и производительность.



Преимущества:

- высокая отказоустойчивость;
- высокая производительность.

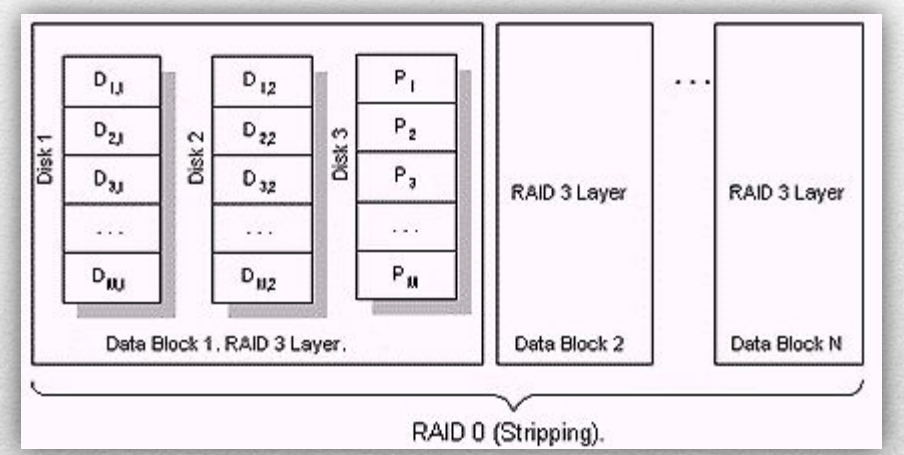
Недостатки:

- очень высокая стоимость;
- ограниченное масштабирование.

RAID Levels

RAID 30. Отказоустойчивый массив с параллельной передачей данных и повышенной производительностью.

Представляет собой массив типа RAID 0, сегментами которого являются массивы RAID 3. Он объединяет в себе отказоустойчивость и высокую производительность. Обычно используется для приложений требующих последовательной передачи данных больших объемов.



Преимущества:

- высокая отказоустойчивость;
- высокая производительность.

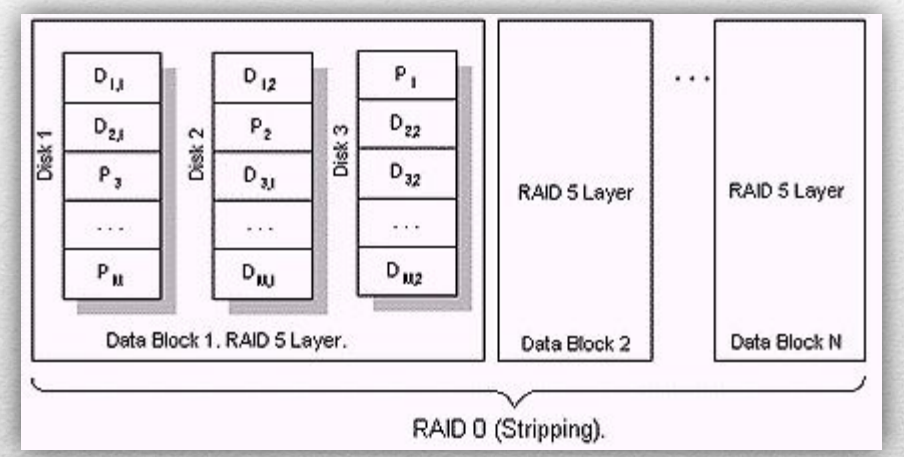
Недостатки:

- высокая стоимость;
- ограниченное масштабирование.

RAID Levels

RAID 50. Отказоустойчивый массив с распределенной четностью и повышенной производительностью

Являет собой массив типа RAID 0, сегментами которого являются массивы RAID 5. Он объединяет в себе отказоустойчивость и высокую производительность для приложений с большой интенсивностью запросов и высокую скорость передачи данных.



Преимущества:

- высокая отказоустойчивость;
- высокая скорость передачи данных;
- высокая скорость обработки запросов.

Недостатки:

- высокая стоимость;
- ограниченное масштабирование.

RAID Levels

RAID 7. Отказоустойчивый массив, оптимизированный для повышения производительности. (Optimized Asynchrony for High I/O Rates as well as High Data Transfer Rates). RAID 7® является зарегистрированной торговой маркой Storage Computer Corporation (SCC)

Для понимания архитектуры RAID 7 рассмотрим ее особенности:

- Все запросы на передачу данных обрабатываются асинхронно и независимо.
- Все операции чтения/записи кэшируются через высокоскоростную шину x-bus.
- Диск четности может быть размещен на любом канале.
- В микропроцессоре контроллера массива используется операционная система реального времени ориентированная на обработку процессов.
- Система имеет хорошую масштабируемость: до 12-ти host-интерфейсов, и до 48-ми дисков.
- Операционная система контролирует коммуникационные каналы.
- Используются стандартные SCSI диски, шины, материнские платы и модули памяти.
- Используется высокоскоростная шина X-bus для работы с внутренней кеш памятью.
- Процедура генерации четности интегрирована в кеш.
- Диски, присоединенные к системе, могут быть задекларированы как отдельно стоящие.
- Для управления и мониторинга системы можно использовать SNMP агент.

Преимущества:

- высокая скорость передачи данных и высокая скорость обработки запросов (1.5 - 6 раз выше других стандартных уровней RAID);
- высокая масштабируемость хост интерфейсов;
- скорость записи данных увеличивается с увеличением количества дисков в массиве;
- для вычисления четности нет необходимости в дополнительной передаче данных.

Недостатки:

- собственность одного производителя;
- очень высокая стоимость на единицу объема;
- короткий гарантийный срок;
- не может обслуживаться пользователем;
- нужно использовать блок бесперебойного питания для предотвращения потери данных из кеш памяти.

RAID Levels

Сравнение дисковых массивов

RAID	Минимум дисков	Потребность в дисках	Отказо-устойчивость	Скорость передачи данных	Интенсивность обработки запросов	Практическое использование
0	2	N	< 1 диск	< RAID 3	очень высокая до N x 1 диск	Графика, видео
1	2	2N*	< RAID 6	R > 1 диск W = 1 диск	до 2 x 1 диск W = 1 диск	малые файл-серверы
2	7	2N<X<N+1	< RAID 1	~ RAID 3	Низкая	мейнфреймы
3	3	N+1	< RAID 1	< RAID 7	Низкая	Графика, видео
4	3	N+1	< RAID 1	R < RAID 3 W < RAID 5	R = RAID 0 W << 1 диск	файл-серверы
5	3	N+1	< RAID 1	R < RAID 4 W < RAID 3	R = RAID 0 W < 1 диск	серверы баз данных
6	4	N+2	самая высокая	низкая	R > 1 диск W < RAID 4	используется крайне редко
7	12	N+1	< RAID 1	самая высокая	самая высокая	разные типы приложений

* - рассматривается обычно используемый вариант;
k - количество подсегментов;
R - чтение;
W - запись.

RAID Levels

Некоторые аспекты реализации RAID систем

Рассмотрим три основных варианта реализации RAID систем:

- программная (software-based);
- аппаратная - шинно-ориентированная (bus-based);
- аппаратная - автономная подсистема (subsystem-based).

Нельзя однозначно сказать, что какая-либо реализация лучше, чем другая. Каждый вариант организации массива удовлетворяет тем или иным потребностям пользователя в зависимости от финансовых возможностей, количества пользователей и используемых приложений.

Каждая из вышеперечисленных реализаций базируется на исполнении программного кода. Отличаются они фактически тем, где этот код исполняется: в центральном процессоре компьютера (программная реализация) или в специализированном процессоре на RAID контроллере (аппаратная реализация).

Главное преимущество программной реализации - низкая стоимость. Но при этом у нее много недостатков: низкая производительность, загрузка дополнительной работой центрального процессора, увеличение шинного трафика. Программно обычно реализуют простые уровни RAID - 0 и 1, так как они не требуют значительных вычислений. Учитывая эти особенности, RAID системы с программной реализацией используются в серверах начального уровня.

Аппаратные реализации RAID соответственно стоят больше чем программные, так как используют дополнительную аппаратуру для выполнения операций ввода вывода. При этом они разгружают или освобождают центральный процессор и системную шину и соответственно позволяют увеличить быстродействие.

Некоторые аспекты реализации RAID систем

Шинно-ориентированные реализации представляют собой RAID контроллеры, которые используют скоростную шину компьютера, в который они устанавливаются (в последнее время обычно используется шина PCI). В свою очередь шинно-ориентированные реализации можно разделить на низкоуровневые и высокоуровневые. Первые обычно не имеют SCSI чипов и используют так называемый RAID порт на материнской плате со встроенным SCSI контроллером. При этом функции обработки кода RAID и операций ввода/вывода распределяются между процессором на RAID контроллере и чипами SCSI на материнской плате. Таким образом, центральный процессор освобождается от обработки дополнительного кода и уменьшается шинный трафик по сравнению с программным вариантом. Стоимость таких плат обычно небольшая, особенно если они ориентированы на системы RAID - 0 или 1 (есть также реализации RAID 3,5,10,30,50, но они дороже), благодаря чему они понемногу вытесняют программные реализации с рынка серверов начального уровня. Высокоуровневые контроллеры с шинной реализацией имеют несколько другую структуру, чем их младшие братья. Они берут на себя все функции, связанные с вводом/выводом и исполнением RAID кода. Кроме того, они не так зависимы от реализации материнской платы и, как правило, имеют больше возможностей (например, возможность подключения модуля для хранения информации в кеш в случае отказа материнской платы или исчезновения питания). Такие контроллеры обычно стоят дороже низкоуровневых и используются в серверах среднего и высокого уровня. Они, как правило, реализуют RAID уровней 0,1,3,5,10,30,50. Учитывая то, что шинно-ориентированные реализации подключаются прямо к внутренней PCI шине компьютера, они являются наиболее производительными среди рассматриваемых систем (при организации одно-хостовых систем). Максимальное быстродействие таких систем может достигать 132 Мбайт/с (32bit PCI) или же 264 Мбайт/с (64bit PCI) при частоте шины 33MHz.

Вместе с перечисленными преимуществами шинно-ориентированная архитектура имеет следующие недостатки:

- зависимость от операционной системы и платформы;
- ограниченная масштабируемость;
- ограниченные возможности по организации отказоустойчивых систем.

RAID Levels

Некоторые аспекты реализации RAID систем

Всех этих недостатков можно избежать, используя автономные подсистемы. Эти системы имеют полностью автономную внешнюю организацию и в принципе являют собой отдельный компьютер, который используется для организации систем хранения информации. Кроме того, в случае удачного развития технологии оптоволоконных каналов быстродействие автономных систем ни в чем не будет уступать шинно-ориентированным системам.

Обычно внешний контроллер ставится в отдельную стойку и в отличие от систем с шинной организацией может иметь большое количество каналов ввода/вывода, в том числе и хост-каналов, что дает возможность подключать к системе несколько хост-компьютеров и организовывать кластерные системы. В системах с автономным контроллером можно реализовать горячее резервирование контроллеров.

Одним из недостатков автономных систем остается их большая стоимость.

Учитывая вышесказанное, отметим, что автономные контроллеры обычно используются для реализации высокоемких хранилищ данных и кластерных систем.

Виды сетевых угроз

Основные понятия и определения

- Угроза информационной безопасности – потенциальная возможность определенным образом нарушить информационную безопасность.
- Попытка реализации угрозы называется атакой.
- Показатель, характеризующий безопасность информации при воздействии различных факторов опасности – критерий безопасности.
- Промежуток времени от момента, когда появляется возможность использовать слабое место, и до момента, когда пробел ликвидируется, называется окном опасности, ассоциированным с данным уязвимым местом.

Виды сетевых угроз

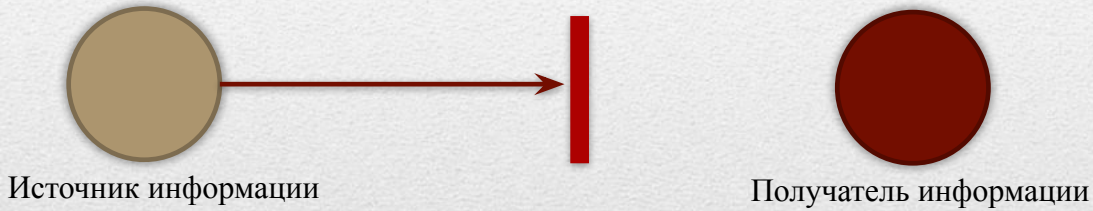
Основные понятия и определения

- Для ликвидации окна опасности должны произойти следующие события:
  1. должно стать известно о средствах использования пробела в защите;
  2. должны быть выпущены соответствующие заплаты;
  3. заплаты должны быть установлены в защищаемой ИС

Виды сетевых угроз

- **Прерывание**

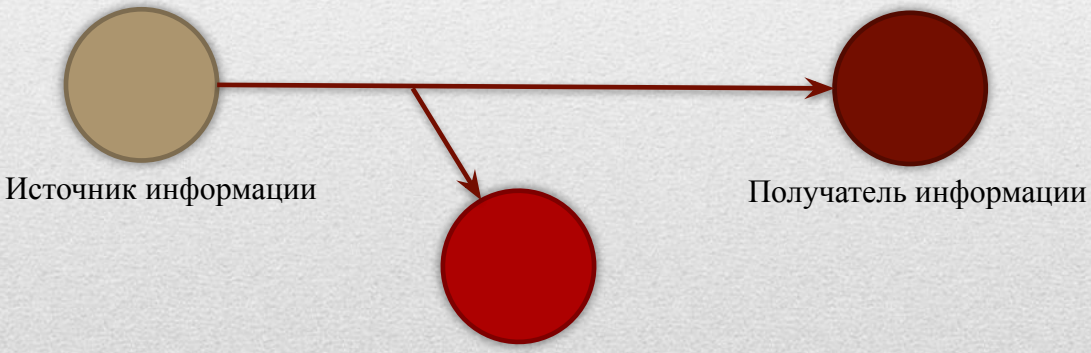
Прерывание - компоненты системы выходят из строя, становятся недоступными или непригодными. Эта атака, целью которой является нарушение доступности.



Виды сетевых угроз

- **Перехват**

Перехват - это атака, целью которой является нарушение конфиденциальности, в результате чего доступ к компонентам системы получают несанкционированные стороны. В роли несанкционированной стороны может выступать лицо, программа компьютер.

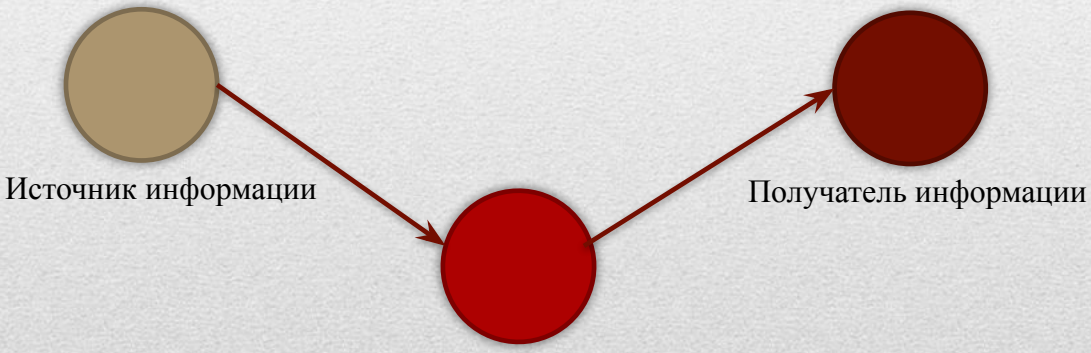


Виды сетевых угроз

- **Изменение**

Изменение - несанкционированная сторона не только получает доступ к системе, но и вмешивается в работу ее компонентов.

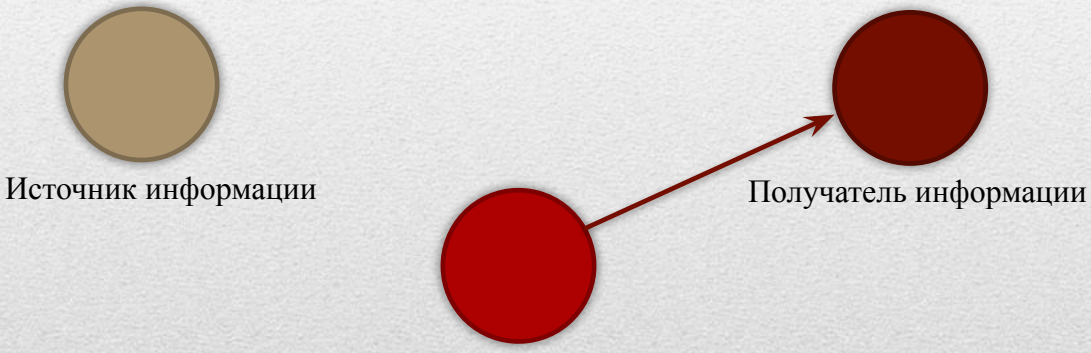
Целью этой атаки является нарушение целостности.



Виды сетевых угроз

- **Подделка**

Подделка - несанкционированная сторона помещает в систему поддельные объекты. Целью этой атаки является нарушение аутентичности (компьютерная система должна иметь возможность проверять идентичность пользователя).



Виды сетевых угроз

Классификация угроз информационной безопасности

Классификация угроз ИБ можно выполнить по нескольким критериям:

- по аспекту ИБ (доступность, целостность, конфиденциальность);
- по компонентам ИС, на которые угрозы нацелены (данные, программа, аппаратура, поддерживающая инфраструктура);
- по способу осуществления (случайные или преднамеренные действия природного или техногенного характера);
- по расположению источника угроз (внутри или вне рассматриваемой ИС).

Виды сетевых угроз

Классификация угроз ИБ по базовым свойствам информации

Вне зависимости от конкретных видов угроз информационная система должна обеспечивать базовые свойства информации и систем ее обработки. Для автоматизированных ИС рассматриваются три основных вида угроз:

- угроза нарушения конфиденциальности;
- угроза нарушения целостности;
- угроза отказа служб (отказа в доступе).

Виды сетевых угроз

- **Примеры реализации угроз (угроза нарушения конфиденциальности)**

Часть информации, хранящейся и обрабатываемой в ИС, должна быть сокрыта от посторонних. Передача данной информации может нанести ущерб как организации, так и самой информационной системе.

Виды сетевых угроз

- **Примеры реализации угроз (угроза нарушения конфиденциальности)**

Средствами атаки могут служить различные технические средства (подслушивание разговоров, сети), другие способы (несанкционированная передача паролей доступа и т.п.).

Важный аспект при предотвращении угрозы конфиденциальности – непрерывность защиты данных на всем жизненном цикле ее хранения и обработки. Пример реализации угрозы – доступное хранение резервных копий данных.

Виды сетевых угроз

Примеры реализации угроз (угроза нарушения целостности данных)

Одними из наиболее часто реализуемых угроз ИБ являются кражи и подлоги. Целостность информации может быть разделена на статическую и динамическую.

Примерами нарушения статической целостности являются:

- ввод неверных данных;
- несанкционированное изменение данных;
- изменение программного модуля вирусом;

Примеры нарушения динамической целостности:

- нарушение атомарности транзакций;
- дублирование данных;
- внесение дополнительных пакетов в сетевой трафик.

Виды сетевых угроз

Примеры реализации угроз (угроза нарушения целостности данных)

Отказ служб (отказа в доступе к ИС) относится к одним из наиболее часто реализуемых угроз ИБ. Относительно компонент ИС данный класс угроз может быть разбит на следующие типы:

- отказ пользователей (нежелание, неумение работать с ИС);
- внутренний отказ информационной системы (ошибки при переконфигурировании системы, отказы программного и аппаратного обеспечения, разрушение данных);
- отказ поддерживающей инфраструктуры (нарушение работы систем связи, электропитания, разрушение и повреждение помещений).

Виды сетевых угроз

Основные принципы обеспечения информационной безопасности

Информационная безопасность может быть обеспечена при соблюдении следующих принципов:

- Системности;
- Комплексности;
- Непрерывности защиты;
- Разумной достаточности;
- Гибкости управления и применения;
- Открытости алгоритмов и механизмов защиты;
- Простоты применения защитных мер и средств.

Виды сетевых угроз

Основные принципы обеспечения информационной безопасности

Информационная безопасность может быть обеспечена при соблюдении следующих принципов:

- Системности;
- Комплексности;
- Непрерывности защиты;
- Разумной достаточности;
- Гибкости управления и применения;
- Открытости алгоритмов и механизмов защиты;
- Простоты применения защитных мер и средств.

Вирусы

Вирусы

Вирусы

Устройства ввода-вывода данных

Восстановление данных

Управление правильностью (помехозащищенностью) передачи информации выполняется с помощью помехоустойчивого кодирования. Различают коды, обнаруживающие ошибки, и корректирующие коды, которые дополнительно к обнаружению еще и исправляют ошибки.

Помехозащищенность достигается с помощью введения избыточности. Устранение ошибок с помощью корректирующих кодов (такое управление называют Forward Error Control) реализуют в симплексных каналах связи. В дуплексных каналах достаточно применения кодов, обнаруживающих ошибки (Feedback or Backward Error Control), так как сигнализация об ошибке вызывает повторную передачу от источника. Это основные методы, используемые в информационных сетях.

Простейшими способами обнаружения ошибок являются контрольное суммирование, проверка на нечетность. Однако они недостаточно надежны, особенно при появлении пачек ошибок. Поэтому в качестве надежных обнаруживающих кодов применяют циклические коды. Примером корректирующего кода является код Хемминга.

Восстановление данных

Коды Хемминга

Коды Хемминга — простейшие линейные коды с минимальным расстоянием 3, то есть способные исправить одну ошибку. В коде Хемминга вводится понятие кодового расстояния d(*,*) (расстояния между двумя кодами), равного числу разрядов с неодинаковыми значениями. Возможности исправления ошибок связаны с минимальным кодовым расстоянием $d_{mi}n$. Исправляются ошибки кратности r=(ent(d_{min}-1)/2) и обнаруживаются ошибки кратности $d_{min} - 1$ (здесь ent означает «целая часть»). Так, при контроле на нечетность $d_{min} = 2$ и обнаруживаются одиночные ошибки. В коде Хемминга $d_{min} = 3$. Дополнительно к информационным разрядам вводится L = log2K избыточных контролирующих разрядов, где K- число информационных разрядов, L округляется до ближайшего большего целого значения. L - разрядный контролирующий код есть инвертированный результат поразрядного сложения (то есть сложения по модулю 2) номеров тех информационных разрядов, значения которых равны 1.

Пример №1. Пусть имеем основной код 100110, то есть K = 6. Следовательно, L = 3 и дополнительный код равен
010 # 011 # 110 = 111,
где # — символ операции поразрядного сложения, и после инвертирования имеем 000. Теперь вместе с основным кодом будет передан и дополнительный. На приемном конце вновь рассчитывается дополнительный код и сравнивается с переданным. Фиксируется код сравнения (поразрядная операция отрицания равнозначности), и если он отличен от нуля, то его значение есть номер ошибочно принятого разряда основного кода. Так, если принят код 100010, то рассчитанный в приемнике дополнительный код равен инверсии от 010 # 110 = 100, то есть 011, что означает ошибку в 3-м разряде.

Пример №2. Основной код 1100000, дополнительный код 110 (результат инверсии кода 110 # 111 = 001). Пусть принятый код 1101000, его дополнительный код 010, код сравнения 100, то есть ошибка в четвертом разряде.

Восстановление данных

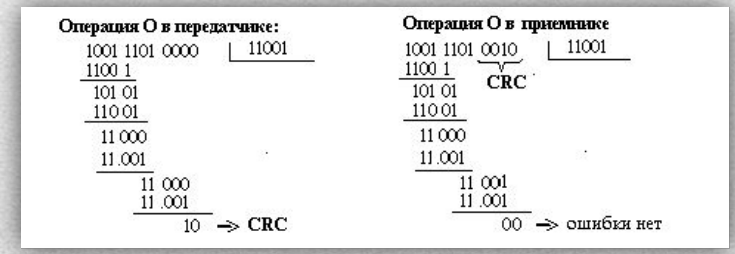
Циклические коды

К числу эффективных кодов, обнаруживающих одиночные, кратные ошибки и пачки ошибок, относятся *циклические коды* (CRC - Cyclic Redundance Code). Они высоконадежны и могут применяться при блочной синхронизации, при которой выделение, например, бита нечетности было бы затруднительно.

Один из вариантов циклического кодирования заключается в умножении исходного кода на образующий полином $g(x)$, а декодирование - в делении на $g(x)$. Если остаток от деления не равен нулю, то произошла ошибка. Сигнал об ошибке поступает на передатчик, что вызывает повторную передачу.

Пример. Пусть A = 1001 1101, образующий полином 11001. Так как K = 4, то A(2K)=100111010000.

Число	Циклический код	Число	Циклический код
0	0000000000	13	0010001111
1	0000001011	14	0010011010
2	0000010110	15	0010100101
3	0000100001	16	0011000110
5	0000110111	18	0011000110
6	0001000010	19	0011010001
.....	.....	.....	.....



Положительными свойствами циклических кодов являются малая вероятность необнаружения ошибки и сравнительно небольшое число избыточных разрядов.

Программирование в операционной системе Windows

Операционная система в наибольшей степени определяет облик всей вычислительной системы в целом. Несмотря на это, пользователи, активно использующие вычислительную технику, зачастую испытывают затруднения при попытке дать определение операционной системе. Частично это связано с тем, что ОС выполняет две по существу мало связанные функции: обеспечение пользователю-программисту удобств посредством предоставления для него расширенной машины и повышение эффективности использования компьютера путем рационального управления его ресурсами.

Программирование в операционной системе Windows

- В 1988 году фирмой IBM создан проект под названием SAA (System Application Architecture (перевод: архитектура системных приложений)), призванный определить все будущие программные разработки. Часть SAA, касающаяся взаимодействия программы и пользователя, называется CUA (Common User Access). Если программа написана с соблюдением норм CUA, пользователь сможет уверенно применять при работе с ней знакомые приемы и потратит самое минимальное время на ее освоение. Например, согласно спецификации CUA, нажатие клавиши «F1» должно вызывать на экран контекстно-зависимую подсказку. При этом пользователь знает, и что экран подсказки будет совершенно определенного формата, и что имеется возможность просмотреть индексный указатель и список команд. Сходный облик различных программ значительно облегчает обучение новым программным средствам.
- Все современное операционные системы и программное обеспечение придерживаются спецификации SAA. Например Widows полностью соответствует CUA. Другая привлекательность Windows – ее независимость по отношению к периферии, которая касается, например, печати: если система Windows поддерживает ваш принтер, то вы сможете писать документ, включающий иллюстрации, не заботясь об управляющих кодах принтера.
- С точки зрения программиста, Windows обладает и другими преимуществами, среди которых можно назвать: способность поддерживать параллельные процессы, наличие системы динамического распределения памяти, обеспечивающей параллельное выполнение нескольких больших задач, а также возможность автоматически задействовать защищенный режим.

Архитектура системы Windows NT

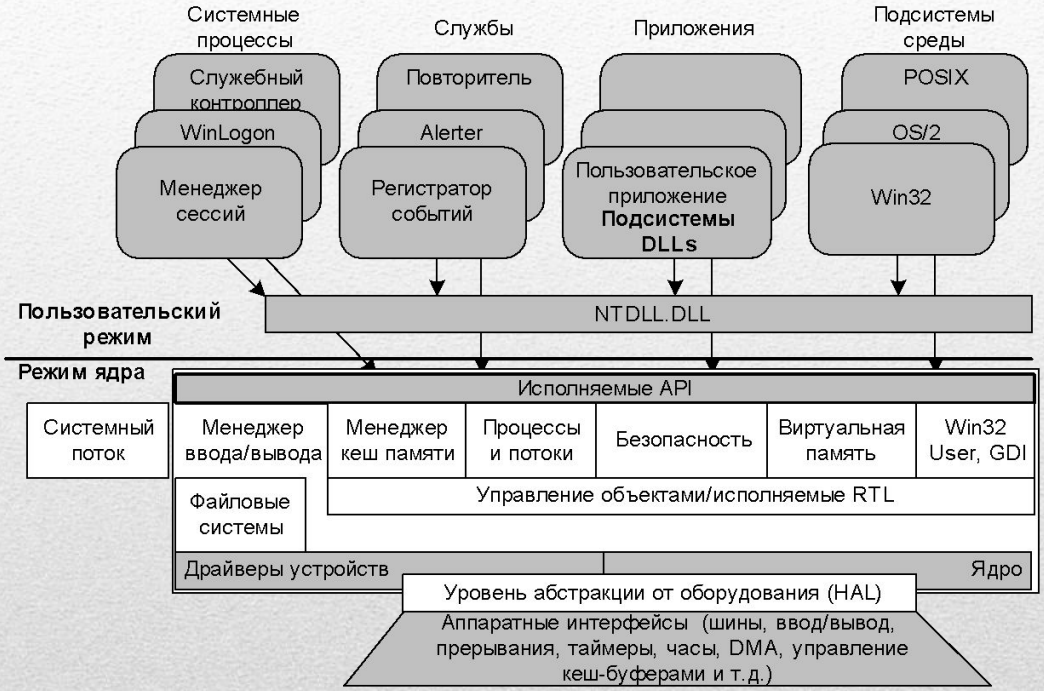


Рис. 1 Архитектура Windows NT и ее компонентов

Элементы над разделительной линией представляют собой процессы пользовательского режима, а под ней располагаются процессы операционной системы, выполняемые ядром. Потоки пользовательского режима выполняются в защищенном адресном пространстве. Однако, во время их выполнения в режиме ядра, они получают доступ к системному пространству. Таким образом, системные процессы, процессы сервера (службы), подсистема среды или пользовательское приложение имеют свое собственное адресное пространство.

Архитектура системы Windows NT

✔ Режим ядра

В режиме ядра выполняются следующие компоненты ОС:

- ✔ **исполняемая часть NT** которая включает управление памятью, процессами, потоками, безопасностью, вводом/выводом, межпроцессорными обменами;
- ✔ **ядро Windows NT** выполняет низкоуровневые функции операционной системы: диспетчеризация потоков, прерываний и исключений, синхронизация процессоров. Ядро также включает набор процедур и базовых объектов, используемый исполняемой частью для создания высокоуровневых конструкций;
- ✔ **слой абстракции от оборудования** (HAL - Hardware Abstraction Layer), изолирует ядро, драйверы устройств и исполняемую часть NT от аппаратных платформ, на которых должна работать операционная система;
- ✔ **драйверы устройств** включают как файловую систему, так и аппаратные драйверы, которые транслируют пользовательские вызовы функций ввода/вывода в запросы физических устройств ввода/вывода;
- ✔ **функции графического интерфейса** пользователя работают с окнами, элементами управления и рисунками.

Архитектура системы Windows NT

✔ Исполняемая часть

Исполняемая часть Windows NT - верхний слой программы - ядра NTOSKRNL.EXE. (Само ядро - это нижний слой). Исполняемая часть содержит следующие компоненты.

- ✔ Менеджер процессов и потоков управляет процессами и потоками. Фактически потоки и процессы поддерживаются в NT нижележащим слоем. Исполняемая часть добавляет дополнительную семантику и функции к этим объектам нижнего уровня.
- ✔ Менеджер виртуальной памяти использует схему управления, при которой каждый процесс получает собственное достаточно большое адресное пространство, защищенное от воздействия других процессов. Менеджер памяти также обеспечивает низкоуровневую поддержку для менеджера кэш-памяти.
- ✔ Монитор безопасности проводит политику обеспечения мер безопасности на локальном компьютере, охраняя системные ресурсы и выполняя процедуры аудита и защиты объектов.
- ✔ Система ввода/вывода использует независимый от устройств ввод/вывод и отвечает за пересылку данных соответствующим драйверам для дальнейшей обработки.
- ✔ Менеджер кэш-памяти улучшает производительность системы ввода/вывода файлов, размещая читаемые с диска данные в основной памяти для ускорения доступа к ним, а также откладывая на короткое время запись измененных данных на диск.

Кроме того, исполняемая часть включает четыре главных группы функций, используемых только что перечисленными компонентами.

- ✔ Менеджер объектов, который создает, удаляет объекты и абстрактные типы данных, а также управляет ими. Объекты используются в Windows NT для представления таких ресурсов операционной системы, как процессы, потоки и объекты синхронизации.
- ✔ LPC передает сообщения между клиентским процессом и процессом сервера на том же самом компьютере. По сути, LPC - это оптимизированная версия известной процедуры удаленного вызова RPC (Remote Procedure Call), стандарта для организации взаимодействия процессов в архитектуре клиент/сервер.
- ✔ Широкий набор библиотечных функций общего типа: обработка строк, арифметические операции, преобразование типов данных, обработка структур.
- ✔ Процедуры распределения памяти, взаимообмен между процессами через память, два специальных типа объектов синхронизации - ресурсы и объекты fast mutex.

Архитектура системы Windows NT

✔ Ядро

Ядро NTOSKRNL.EXE выполняет большинство основных операций NT, определяющих порядок использования процессора: диспетчеризация потоков; диспетчеризация и обработка исключений; синхронизация работы процессоров; обеспечение базовых объектов ядра, которые используются исполняемой частью (и в некоторых случаях экспортируются в режим пользователя).

В отличие от остальной исполняемой части операционной системы, ядро никогда не выгружается из оперативной памяти, его выполнение никогда не прерывается другими потоками. Код ядра написан в основном на Си, а части, дающие наибольшую нагрузку на процессор, на языке Ассемблере.

Объекты ядра. Одна из функций ядра - обеспечение низкоуровневой базы для хорошо определенных примитивов операционной системы, которые обеспечивают работу компонентов высшего уровня. Ядро изолирует само себя от остальной части ОС, что позволяет вынести принятие политических решений из ядра, за исключением диспетчеризации потоков. Ядро использует набор простейших объектов, называемых объектами ядра, позволяющих управлять работой центрального процессора и порядком создания вычисляемых объектов. Большинство вычисляемых объектов включает в себя один или более объектов ядра, включая определенные ядром атрибуты. Один из наборов объектов называется объектами управления и включает объект процесса ядра, объект APC, объект процедуры отложенного вызова DPC (Deferred Procedure Call) и несколько объектов, используемых системой ввода/вывода (например, объект обработки прерывания).

Другой набор объектов ядра - объекты диспетчеризации, включает объекты синхронизации потоков, поток ядра, mutex, объекты события, семафора, таймера, таймера ожидания и ряд других.

Поддержка оборудования. Другой главнейшей задачей ядра является абстрагирование (или изоляция) исполняемой части и драйверов устройств от различий микропроцессорных платформ, на которых способна работать Windows NT: x86 и Alpha AXP. Специфичные для архитектуры функции (такие, как контекстное переключение потока) реализованы в ядре. Функции, которые могут отличаться от машины к машине, реализованы в составе HAL.

Ядро поддерживает набор интерфейсов, семантически идентичных для всех архитектур. Некоторые из интерфейсов реализованы по-разному для разных архитектур, однако, и идентичны внешне интерфейсы реализованы с помощью специфичного для архитектуры кода. Независимый от архитектуры интерфейс может быть вызван на любой машине, и его семантика будет той же, несмотря на то, зависит ли код от архитектуры или нет. Некоторые интерфейсы ядра (например, процедуры синхронизации SMP) реализованы в HAL, поскольку их реализация может изменяться даже внутри одного семейства компьютеров. В качестве примера зависящего от архитектуры кода можно назвать также поддержку кэша центрального процессора.

Архитектура системы Windows NT

✔ Абстракция от оборудования

Загружаемый модуль ядра HAL обеспечивает низкоуровневый интерфейс с аппаратной платформой, что позволяет скрыть такие зависимые от аппаратуры детали, как интерфейс ввода/вывода, контроллеры прерываний, механизм обмена данными между процессорами - любые аппаратно-зависимые и специфические для архитектуры функции.

Драйверы устройств. Драйверы устройств - это загружаемые модули, которые работают в режиме ядра, обеспечивая интерфейс между системой ввода/ вывода и соответствующим оборудованием. Названия этих модулей обычно имеют расширение .SYS. Все они, как правило, написаны на Си (иногда C++) с использованием вызовов процедур HAL и могут быть переносимыми на уровне двоичного кода между платформами, поддерживаемыми NT. Имеется несколько типов драйверов устройств:

- **Драйверы, манипулирующие устройствами** (с использованием HAL) для записи выходных данных или получения входных данных от физических устройств или через сеть.
- **Драйверы файловой системы,** которые принимают запросы на файловый ввод/вывод и транслируют их в запросы ввода/вывода, связанные с конкретными устройствами.
- **Драйверы фильтров.** Примером могут быть драйверы поддержки зеркальных дисков, шифрования данных, перехвата ввода/вывода для дополнительной обработки данных перед передачей их на следующий уровень и т.д.
- **Сетевые драйверы,** которые передают и принимают удаленные запросы на ввод/вывод.

Поскольку установка драйверов устройств является единственным способом добавить к системе пользовательский код, работающий в режиме ядра, то некоторые программисты могут рассматривать написание драйверов устройств как способ доступа к внутренним функциям и структурам данных операционной системы, недоступным из пользовательского режима.

Архитектура системы Windows NT

✔ Пользовательские процессы

Имеется четыре базовых типа пользовательских процессов.

- **Специальные процессы поддержки системы,** например, процесс регистрации пользователя и менеджер сессий, которые не являются службами NT.
- **Процессы сервера**, которые являются службами NT (аналог демонов в ОС Unix). Примером может быть регистратор событий (Event Logger). Многие дополнительно устанавливаемые приложения, такие как Microsoft SQL Server и Exchange Server, также включают компоненты, работающие как службы NT.
- **Подсистемы среды,** которые обеспечивают пользовательским приложениям среду других операционных систем. Windows NT поставляется с тремя подсистемами: Win32, Posix и OS/2 2.1.
- **Пользовательские приложения** одного из пяти типов: Win32, Windows 3.1, MS-DOS, Posix или OS/2 1.2.

Архитектура системы Windows NT

✔ Подсистемы среды и библиотеки DLL

Каждая из подсистем обеспечивает пользовательским приложениям доступ к разным поднаборам служб Windows NT. Это означает, что некоторые вещи могут быть сделаны из приложения, построенного на одной подсистеме, и не возможны из приложения, построенного в другой подсистеме. Так, приложение для Win32 не может использовать функцию fork подсистемы Posix.

Каждый исполняемый модуль связывается с одной и только одной подсистемой. Когда начинается выполнение модуля, изучается тип кода его заголовка, что позволяет определить подсистему среды для создания новых процессов.

Пользовательские процессы не вызывают службы NT напрямую, а используют библиотеки динамических связей (DLL) соответствующей подсистемы среды. Роль библиотек, принадлежащих подсистеме среды, в том, чтобы транслировать документированные функции среды в соответствующие вызовы недокументированных служб NT. Эти библиотеки DLL экспортируют документированный интерфейс, который могут вызывать связанные с подсистемой программы. Например, библиотеки DLL подсистемы Win32 используют функции Win32 API. Библиотека DLL подсистемы Posix использует функции Posix 1003.1 API.

Архитектура системы Windows NT

✔ Подсистема Win32

Главные компоненты подсистемы Win32 - процесс подсистемы среды и драйвер режима ядра. Процесс подсистемы среды поддерживает:

- консольные (текстовые) окна;
- создание и удаление процессов и потоков;
- работу виртуальной 16-разрядной DOS машины;
- иные функции (GetTempFile, DefineDosDevice, ExitWindowsEx и др.).

Драйвер режима ядра поддерживает:

- менеджер окон, который управляет отображением окон, выводом на экран, вводом с клавиатуры, от мыши и других устройств, а также передачей пользовательских сообщений приложениям;
- интерфейс графических устройств GDI (Graphical Device Interface), библиотека функций для вывода на графические устройства, для рисования текста, линий, фигур и манипуляций графическими объектами;
- зависимые от устройств драйверы графики, принтера и видеопорта;
- несколько библиотек DLL, которые транслируют документированные функции Win32 API в соответствующие недокументированные вызовы NTOSKRNL.EXE и WIN32K.SYS.

Приложения вызывают стандартные функции для создания окон и кнопок на дисплее. Менеджер окон передает эти запросы драйверам графических устройств через интерфейс графических устройств GDI, где они форматируются для вывода средствами конкретных устройств. GDI обеспечивает набор стандартных функций, позволяющих приложениям общаться с графическими устройствами, включая дисплеи и принтеры, без конкретных знаний о них. GDI интерпретирует запросы приложений на графический вывод и посылает их драйверам графических дисплеев. Этот интерфейс позволяет создавать код приложения, независимый от конкретных устройств и их драйверов.

Архитектура системы Windows NT

✔ NTDLL.DLL

NTDLL.DLL - это специальная система поддержки DLL - библиотек. Она содержит два типа функций.

- **Первая группа функций обеспечивает интерфейс к службам NT,** которые могут быть вызваны из пользовательского режима. Существует более 200 таких функций, например NtCreateFile, NtSetEvent и т.д. Для каждой из них имеется точка входа в NTDLL.DLL с тем же именем. Внутренний код функции содержит специфичные для архитектуры команды, которые вызывают переход в режим ядра для обращения к реальным службам NT, код которых содержится в NTOSKRNL.EXE.
- **Вторая группа функций содержит большое количество функций поддержки:** загрузчик исполняемых модулей, коммуникационные функции для процессов подсистемы Win32, библиотека функций реального времени пользовательского режима, диспетчер вызовов асинхронных процедур APC (Asynchronous Procedure Call) пользовательского режима, диспетчер исключений.

Основы программирования в Windows

Все программы для Windows разрабатываются на основе понятия передачи сообщений. В каждой программе необходимо иметь цикл ввода сообщений и для каждого из окон – свою процедуру обработки сообщений.

Windows генерирует сообщения всякий раз, когда происходит какое-либо событие или должно быть выполнено действие. Например, при перемещении «мыши» генерируется сообщение, где указывается программа, ответственная за реакцию на это перемещение, и координаты той точки, в которой оказался курсор. Точно так же с помощью сообщения Windows информирует прикладную программу о том, что в меню был выбран определенный пункт. Сущность программирования в Windows состоит в том, чтобы принять сообщение, переслать его в то окно, для которого оно предназначено, обработать его, после чего вернуться к чтению следующего сообщения.

Windows – многозадачная система, способная выполнять одновременно несколько программ. Поскольку каждой программе необходим ЦП, ни одна из них не должна захватывать его на слишком долгое время. Если процессор будет на долго занят одним приложением, это может повлечь ошибки в других, выполняемых параллельно и чувствительных к замедлению программ. В Unix, выполнение программы прерывается как только истечет априорно отведенное ей время, даже если ровно в этот момент заканчивалась важная операция.

Windows функционирует как многозадачная система всякий раз, когда прикладная программа анализирует очередь сообщений. Если сообщений, предназначенных данной программе, не оказывается, Windows начинает искать сообщения для других программ, активных в этот момент, и передает управление той, для которой сообщение есть. Эта программа, в свою очередь, анализирует и обрабатывает предназначенное ей сообщение и пытается прочесть следующее. В этот момент Windows возобновляет цикл.

При разработке программы, предназначенной для Windows, необходимо структурировать ее так, чтобы она могла поддерживать передачу управления по описанной схеме. Если соответствующих мер не принять, вновь созданная программа, несмотря на некорректность, будет способна выполняться, но другие программы, запущенные одновременно с ней, рискуют вовсе остановиться. Иными словами, вся коммуникация окажется заблокирована.

Основы программирования в Windows

✔ Сообщения

Для облегчения и ускорения обработки все сообщения в Windows строятся в соответствии со строго определенным компактным форматом. В котором передаются следующие параметры:

указатель на окно, для которого это сообщение предназначалось. Каждому окну присвоен свои идентификатор по которому система понимает, что сообщение предназначалось именно ему;

идентификатор сообщения, который сообщает программе, какова в точности природа происшедшего события;

тип сообщения, который определяет, какое сообщение было послано приложению;

время поступления сообщения в очередь;

координаты «мыши» в момент поступления сообщения.

На рис. 1 представлена организация сообщений в Windows. Windows ищет сообщение в очереди программы. Если сообщение не найдено, Windows исследует системную очередь, где распознает сообщения, поступившие от клавиатуры или «мыши». Если таковых не оказывается, прикладная программа переводится в состояние «сна».

Если в приложение поступило сообщение, то программа останавливается, обрабатываются по очереди все поступившие сообщения и возобновляется выполнение программы. Пока приложение ждет сообщения, Windows передает управление центральным процессором другим программам.

Основы программирования в Windows

✔ Сообщения

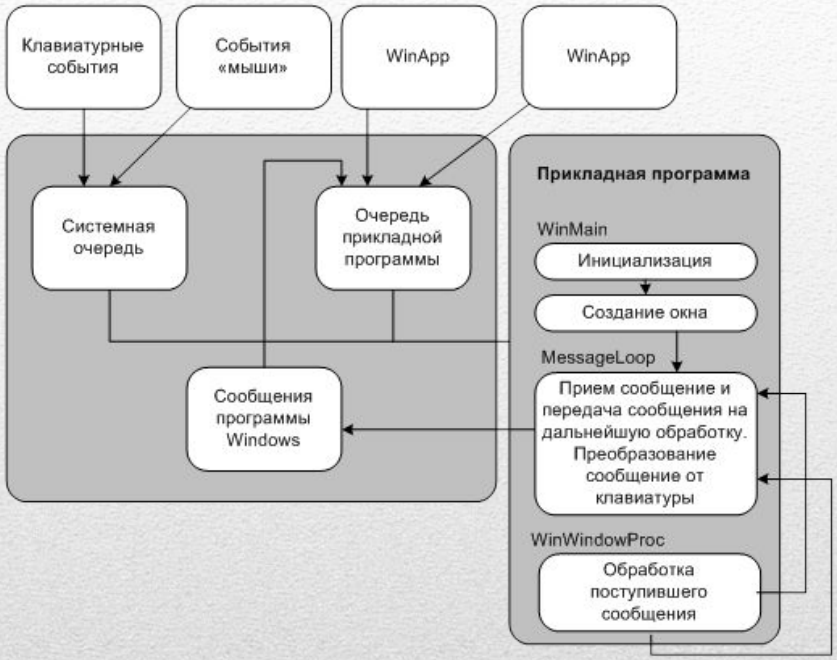


рис. Организация очереди сообщений в Windows

Сообщения может генерировать не только Windows, но и любая программа которая работает в системе. Для передачи сообщений в окно существуют два различных способа – непрямой, и прямой.

При непрямой передаче сообщение помещается в очередь окна-адресанта. Если очередь не пуста, окно получит переданное сообщение лишь после того, как обработает все предыдущие, а это может потребовать некоторого времени.

При прямой передаче происходит обращение непосредственно к процедуре окна, минуя очередь. Этот способ применяется в тех случаях, когда желательно, чтобы окно-адресант отреагировало на сообщение немедленно.

НЕПЕР ДЖОН

НЕПЕР (Нейпир) (Napier) Джон (1550-1617), шотландский математик, изобретатель логарифмов.

Потомок старинного воинственного шотландского рода. Изучал логику, теологию, право, физику, математику, этику. Увлекался алхимией и астрологией. Изобрел несколько полезных сельскохозяйственных орудий. В 1590-х годах пришел к идее логарифмических вычислений и составил первые таблицы логарифмов, однако свой знаменитый труд «Описание удивительных таблиц логарифмов» опубликовал лишь в 1614 году. В конце 1620-х годов была изобретена логарифмическая линейка, счетный инструмент, использующий таблицы Непера для упрощения вычислений. С помощью логарифмической линейки операции над числами заменяются операциями над логарифмами этих чисел.

В 1617 году, незадолго до своей смерти, Непер изобрел математический набор, призванный облегчить арифметические вычисления. Набор состоял из брусков с нанесенными на них цифрами от 0 до 9 и кратными им числами. Для умножения какого-либо числа бруски располагали рядом так, чтобы цифры на торцах составляли это число. Ответ можно было увидеть на боковых сторонах брусков. Помимо умножения, палочки Непера позволяли выполнять деление и извлечение квадратного корня.

ЛЕЙБНИЦ ГОТФРИД

ЛЕЙБНИЦ (Leibniz) Готфрид Вильгельм (1 июля 1646, Лейпциг — 14 ноября 1716, Ганновер), немецкий философ, математик, физик, языковед. Предвосхитил принципы современной математической логики («Об искусстве комбинаторики», 1666). Один из создателей дифференциального и интегрального исчислений. Создал первую механическую счетную машину, способную производить сложение, вычитание, умножение и деление. Независимо от Ньютона создал дифференциальное и интегральное исчисление и заложил основы двоичной системы счисления.

Закончил Лейпцигский университет, куда поступил в возрасте 15 лет. В 20 лет избрал дипломатическую карьеру, отказавшись от предложенной ему должности профессора. В 1673 году изготовил механический калькулятор, в частности, чтобы облегчить труд своего друга астронома Христиана Гюйгенса. В машине Лейбница использовался принцип связанных колец суммирующей машины Паскаля, но Лейбниц ввел в нее подвижный элемент (прообраз каретки настольного калькулятора), позволивший ускорить повторение операции сложения, необходимое при перемножении чисел. Вместо колесиков и приводов в машине Лейбница находились цилиндры с нанесенными на них цифрами. Каждый цилиндр имел девять рядов выступов или зубцов. При этом первый ряд содержал один выступ, второй ряд — два выступа и так вплоть до девятого ряда, который содержал соответственно девять выступов. Цилиндры с выступами были подвижными и приводились в определенные положения оператором.

Специально для своей машины Лейбниц применил систему счисления, использующую вместо обычных для человека десяти цифр две: 0 и 1. Принципы двоичной системы счисления Лейбниц объяснял на примере коробочки с отверстиями: открытое отверстие означало 1, закрытое – 0. Единица обозначалась выпавшим шаром, ноль – отсутствием выпавшего шара. Двоичная система счисления Лейбница нашла впоследствии применение в автоматических вычислительных устройствах.

Лейбниц основал Бранденбургское научное общество (позднее — Берлинская АН) и с 1700 являлся его президентом. По просьбе Петра I разработал проекты развития образования и государственного управления в России.

ПАСКАЛЬ БЛЕЗ

ПАСКАЛЬ (Pascal) Блез (19 июня 1623, Клермон-Ферран, Франция — 19 августа 1662), французский математик, физик, религиозный философ и писатель. Сформулировал одну из основных теорем проективной геометрии. Работы по арифметике, теории чисел, алгебре, теории вероятностей. Один из основоположников гидростатики, установил ее основной закон (закон Паскаля). Работы по теории воздушного давления. Сконструировал (1641, по другим сведениям — 1642) суммирующую машину.

Работу над суммирующей машиной начал в 19-летнем возрасте, наблюдая за утомительными расчетами своего отца — сборщика налогов. Суммирующая машина Паскаля представляла собой механическое устройство с многочисленными шестеренками. С ее помощью можно было складывать числа, вращая колесики с делениями от 0 до 9, связанные друг с другом таким образом, что избыток над девяткой переносился на следующее колесико, продвигая его на единицу вперед. Были отдельные колесики для единиц, десятков, сотен и т. д. К сожалению, машина не могла выполнять никаких других арифметических действий, кроме сложения. Вычитать, умножать или делить на ней можно было лишь путем многократного сложения (вычитания). Изобретенный Паскалем принцип связанных колес стал основой для вычислительных устройств следующих трех столетий.

ШИККАРД ВИЛЬГЕЛЬМ

ШИККАРД (Schickard) Вильгельм (1592-1635), немецкий математик и астроном. Изобрел и построил первую работающую модель 6-ти разрядного механического вычислительного устройства, которое могло складывать и вычитать числа. Описание машины Шиккарда, к сожалению, оказалось утраченным во время Тридцатилетней войны. Создал также первый механический планетарий, демонстрирующий положение Солнца, Земли и Луны согласно системе Коперника. Наблюдал метеоры из разных пунктов для определения их траектории.

ВИНЕР НОРБЕРТ

ВИНЕР Норберт (Wiener Norbert) (26 ноября 1894, Колумбия, шт. Миссури — 18 марта 1964, Стокгольм), американский математик.

В своем фундаментальном труде «Кибернетика» сформулировал основные ее положения. Винер — автор трудов по математическому анализу, теории вероятностей, электрическим сетям и вычислительной технике.

Учился в Тафтс-колледже, Корнуэльском, Гарвардском, Кембриджском, Геттинтенском и Колумбийском университетах. Одаренный математик, в 1919 году он стал ассистентом профессора математики Массачусетского технологического института, а с 1932 по 1960 год занимал должность профессора.

Во время Второй мировой войны, занимаясь исследованиями для целей противовоздушной обороны, он заинтересовался автоматическими расчетами и теорией обратной связи. Впоследствии Винер сформулировал основные положения новой науки — кибернетики, предметом изучения которой стали управление, связь и обработка информации в технике, живых организмах и человеческом обществе.

Лебедев Сергей Алексеевич

Лебедев С.А. – конструктор «Малой электронной счётной машины» (МЭСМ)

ШОКЛИ УИЛЬЯМ

ШОКЛИ (Chockley) Уильям Брэдфорд (1910, Лондон — 1989), американский физик. Труды по физике твердого тела и полупроводников.

Работал в исследовательской лаборатории фирмы Bell Telephone Laboratories, подразделении компании AT&T. В 1945 году возглавил группу физиков, изучавших свойства полупроводников. Задачей группы стала разработка приборов, способных заменить электронные лампы и электромеханические реле в системе телефонной связи. В 1947 году им удалось получить первую модель транзистора с точечными контактами — полупроводниковый усилитель. В 1951 году Шокли представил первый трехслойный германиевый транзистор, выполнявший те же функции, что и электронная лампа, но имевший гораздо меньшие размеры, более того, он был надежнее и экономичнее, но, увы, цена была слишком высокой. Но в 1954 году физику Гордону Тилу, перешедшему из Bell Telephone Laboratories в Texas Instruments, удалось изготовить транзисторы из дешевого кремния, что снизило их себестоимость и положило начало процессу миниатюризации в электронике.

В 1955 году Шокли покинул Bell Telephone Laboratories и создал собственную фирму по производству полупроводников близ Пало-Альто. За изобретение транзистора в 1956 году Шокли вместе с коллегами Дж. Бардином и У. Браттейном был удостоен Нобелевской премии.

ШЕННОН КЛОД

ШЕННОН Клод Элвуд (Shannon Claude Elwood) (р. 1916, Гэйлорд, шт. Мичиган), американский инженер и математик. Один из создателей математической теории информации. Основные труды по теории релейно-контактных схем, математической теории связи, кибернетике.

Учился в Мичиганском университете, где получил два диплома — по математике и по электротехнике. Затем перешел в Массачусетский технологический институт, где работал под руководством профессора Ванневара Буша на его дифференциальном анализаторе. В 1938 году защитил докторскую диссертацию, в которой разработал принципы логического устройства компьютера, соединив булеву алгебру с работой электрических схем. Эта работа стала поворотным пунктом в истории развития современной информатики и вычислительной техники. Позднее Шеннон работал в Bell Telephone Laboratories, где применил новые принципы разработки телефонных станций. В 1956 году стал членом ученого совета МТИ.

В 1948 году опубликовал работу «Математическая теория связи», в которой представил свою унифицированную теорию передачи и обработки информации. Информация в этом контексте включала все виды сообщений, включая те, что передаются по нервным волокнам в живых организмах. Шеннон предложил измерять информацию в математическом смысле, сводя ее к выбору между двумя значениями, или двоичными разрядами, — «да» или «нет», заложив таким образом фундамент современной теории связи, которая в настоящее время играет важную роль во многих областях.

НЕЙМАН ДЖОН

НЕЙМАН (Нойман) (Neumann) Джон (Янош) фон (3 декабря 1903, Будапешт — 8 февраля 1957, Вашингтон), американский математик и физик. Труды по функциональному анализу, квантовой механике, логике, метеорологии. Внес большой вклад в создание первых ЭВМ и разработку методов их применения. Его теория игр сыграла важную роль в экономике.

Изучал химию в Берлинском университете, в 1926 году получил диплом химика в Высшей технической школе в Цюрихе. В том же году в Будапештском университете стал доктором философии, защитив диссертацию по теории множеств.

В 1926-29 годах состоял приват-доцентом Берлинского, а в 1929-30 — Гамбургского университетов. Его основные работы того времени связаны с квантовой физикой и теорией операторов. Благодаря этим работам квантовая физика и теория операторов стали считаться двумя аспектами одного предмета.

В 1930 году фон Нейман посетил с лекциями Принстонский университет в Нью-Джерси, а в 1931 был принят туда профессором. В 1932 он дал точную формулировку и доказательство эргодической гипотезы в математической статистике. В 1932 году была опубликована его книга «Математические обоснования квантовой механики», которая стала классическим учебным пособием.

В 1933 году фон Нейман становится профессором вновь созданного Принстонского института перспективных научных исследований, в котором проработал до конца жизни.

Во второй половине 1930-х в совместно с Ф. Дж. Мюрреем фон Нейман опубликовал ряд работ по кольцам операторов, положив начало так называемой алгебре Неймана, которая впоследствии стала одним из главных инструментов для квантовых исследований.

В 1937 году фон Нейман принял гражданство США. Во время Второй мировой войны он служил консультантом в атомном центре в Лос-Аламосе, где рассчитал взрывной метод детонации ядерной бомбы и участвовал в разработке водородной бомбы. В марте 1955 года он стал членом американской комиссии по атомной энергии.

Из 150 трудов фон Неймана лишь 20 касаются проблем физики, остальные же равным образом распределены между чистой математикой и ее практическими приложениями, в том числе теорией игр и компьютерной теорией.

Фон Нейману принадлежат новаторские работы по компьютерной теории, связанные с логической организацией компьютеров, проблемами функционирования машинной памяти, имитацией случайности, проблемами самовоспроизводящихся систем. В 1944 году присоединился к группе Мокли и Эккерта, занятой созданием машины ENIAC, в качестве консультанта по математическим вопросам. Тем временем в группе началась разработка новой модели, EDVAC, которая, в отличие от предыдущей, могла бы хранить программы в своей внутренней памяти. В 1945 году Нейман опубликовал «Предварительный доклад о машине EDVAC», в котором описывалась сама машина и ее логические свойства. Описанная Нейманом архитектура компьютера получила название «фон Неймановской», и таким образом ему было приписано авторство всего проекта. Это вылилось впоследствии в судебное разбирательство о праве на патент и привело к тому, что Эккерт и Мокли покинули лабораторию и основали собственную фирму. Тем не менее, «архитектура фон Неймана» была положена в основу всех последующих моделей компьютеров. В 1952 году Нейман разработал первый компьютер, использующий программы, записанные на гибком носителе, MANIAC I.

В 1956 Комиссия по атомной энергии наградила его премией Энрико Ферми за выдающийся вклад в компьютерную теорию и практику.

Одной из его идей, для разработки которой Нейман предлагал использовать компьютерные расчеты, было потепление климата на Земле, которое можно вызвать, покрыв краской полярные льды и уменьшив таким образом отражение ими солнечной энергии, так что климат в Исландии должен был стать таким же, как на Гавайях.

Секретом успеха Неймана иногда считают его «аксиоматический метод». Он рассматривал предмет, сконцентрировавшись на его основных свойствах (аксиомах), из которых вытекает все остальное.

А.А. Бизяев

МОКЛИ ДЖОН

МОКЛИ Джон Уильям (Mauchly John William) (30 августа 1907, Цинциннатти, шт. Огайо — 8 января 1980, Эмблер, шт. Пенсильвания), американский физик и инженер, изобретатель (1946, совместно с Пр. Эккертом) первого универсального компьютера «Эниак» («ENIAC»).

Преподавал электротехнику в Пенсильванском университете в Филадельфии. Во время Второй Мировой войны вместе с Эккертом занялся проблемой ускорения пересчета артиллерийских огневых таблиц для вооруженных сил США.

В результате была предложена конструкция универсального цифрового компьютера, который мог оперировать закодированными данными. Использовав разработки Дж. Атанасоффа, коллеги к 1946 году завершили создание модели «Эниак», огромной машины, которая состояла из более 18 тысяч электронных ламп. Вес машины составлял 30 тонн, она требовала для размещения 170 м2. Машина оперировала двоичными числами и могла производить 5000 операций сложения или 300 операций умножения в секунду. Впервые эта машина была применена при баллистических военных исследованиях на Абердинском испытательном полигоне в 1947 году.

В 1948 Мокли и Эккерт основали компанию по производству компьютеров, которая через год представила бинарный автоматический вычислитель («BINAC»), в котором вместо перфокарт уже использовалась магнитная лента. Мокли предложил идею такой системы кодирования, которая позволяла бы машине воспринимать алгебраические уравнения, записанные в традиционной форме.

В 1950 году фирма Эккерта и Мокли была приобретена компанией «Ремингтон Рэнд» (позднее «Сперри Рэнд корпорейшн»).

Третьим компьютером Мокли и Эккерта стал UNIVAC I, созданный специально для коммерческих расчетов. Он мог свободно обрабатывать как цифровую, так и символьную информацию. Первый экземпляр машины был передан в Бюро переписи населения США. Затем было разработано много различных моделей UNIVAC, которые нашли применение в других сферах деятельности. Таким образом, UNIVAC стал первым серийным компьютером.

За достижения на компьютерном поприще Мокли был удостоен многих наград. В 1959-65 он занимал пост президента и в 1965-69 — председателя правления Mauchly Associates, Inc., в 1968-80 — президента Dynatrend Inc., в 1970-80 — президента Marketrend Inc.

НОЙС РОБЕРТ

НОЙС Роберт (Noyce Robert Norton) (12 декабря 1927, Берлингтон, шт. Айова — 3 июня 1990, Остин, шт. Техас), американский инженер, изобретатель (1959) интегральной схемы, системы взаимосвязанных транзисторов на единой кремниевой пластинке, основатель (1968, совместно с Г. Муром) корпорации Intel.

В 1949 году Нойс окончил Гриннелл-колледж в Айове со степенью бакалавра, а в 1953 году стал доктором философии Массачусетского технологического института. В 1956-57 годах работал в полупроводниковой лаборатории изобретателя транзисторов Уильяма Шокли, а затем вместе с семью коллегами уволился и основал одну из первых электронных фирм по производству кремниевых полупроводников — Fairchild Semiconductor (Фэрчайлд Семикондактор), которая дала название Силиконовой долине в Северной Калифорнии. Одновременно, но независимо друг от друга Нойс и Килби изобрели интегральную микросхему.

В 1968 году Нойс и его давний коллега Гордон Мур основали корпорацию Intel. Спустя два года они создали 1103-ю запоминающую микросхему из кремния и поликремния, которая заменила собой прежние малоэффективные керамические сердечники в запоминающих устройствах компьютеров. В 1971 Intel представила микропроцессор, объединяющий в одной микросхеме функции запоминающего устройства и процессора. Вскоре корпорация Intel стала лидером по производству микропроцессоров. В 1988 году Нойс стал президентом корпорации Sematech, исследовательского консорциума, совместно финансируемого промышленным капиталом и правительством США с целью развития передовых технологий в американской полупроводниковой промышленности.

ХЬЮЛЕТТ УИЛЬЯМ

ХЬЮЛЕТТ Уильям (William R. Hewlett) (р. 20 мая 1913, Энн-Арбор, шт. Мичиган), американский инженер, специалист в области электроники, а также предприниматель, соучредитель и почетный директор в отставке компании Hewlett-Packard, крупнейшего мирового производителя компьютерной и измерительной техники.

Билл Хьюлетт и Дэвид Паккард по праву считаются патриархами знаменитой Силиконовой долины. Они не только основали компанию, в которой сейчас работают свыше 100 тыс. человек, но в значительной степени создали компьютерную индустрию. В созданной ими компании всегда придавалось огромное внимание работе с людьми, а также проблемам повышения их благосостояния. Гибкий стиль управления компании Hewlett-Packard (HP) стал моделью для остальных предприятий Силиконовой долины, а впоследствии и для аналогичных производств за рубежом.

Хьюлетт родился в семье врача. Его отец умер, когда мальчику было 12 лет, и, возможно, только поэтому Билл стал не врачом, а инженером. В 1934 году он получил степень бакалавра искусств Стэнфордского университета, там же позднее в 1939 году был удостоен диплома инженера, а в 1936 — диплом электротехника знаменитого Массачусетского технологического института.

Хьюлетт и Паккард встретились в 1930 году, будучи студентами Стэнфордского университета. В 1937 году Хьюлетт, Паккард и два их друга под руководством профессора Фреда Термана создали свою первую компанию, оформив через два года свое партнерство. Хьюлетт выиграл по жребию право дать название новоиспеченной компании. Так фирма стала называться Hewlett-Packard.

Хьюлетт и Паккард сняли небольшой, в два этажа, дом в Пало-Альто, в гараже которого обосновалась компания, ее первоначальный капитал составлял 538 (!) долларов. Хьюлетт являлся генератором идей, а Паккард выполнял функции администратора новой компании. Сегодня на стене этого гаража красуется табличка «Здесь начиналась Силиконовая долина», а сам гараж стал неотъемлемой достопримечательностью Калифорнии.

Хьюлетт принимал активное участие в управлении компанией вплоть до 1987 года, за исключением того времени, когда он служил офицером во время Второй мировой войны. Его последним военным поручением было участие в работе специальной американской команды, которая инспектировала предприятия японской промышленности сразу после войны.

В 1947 году, вскоре после возвращения в Пало-Альто, Хьюлетт был избран вице-президентом HP, затем исполнительным вице-президентом (1957), президентом (1964), а в 1969 — главным исполнительным директором.

В 1978 году он оставил свой пост, чтобы обеспечить преемственность управления. С 1987 года Хьюлетт занимает почетный пост директора в отставке.

Переоценить вклад Хьюлетта в дело Силиконовой долины невозможно. Им Совместно с Паккардом была разработана целая шкала ценностей, названная «Путь Хьюлетта и Паккарда». Вектором этого пути является приверженность к новейшим исследованиям с учетом рыночной конъюнктуры, а также повышение уровня жизни сотрудников фирмы. Билл Хьюлетт стал живой легендой Америки и безупречной моделью менеджера на все времена.

В 1985 году президентом Р. Рейганом Уильяму Хьюлетту была вручена высшая научная награда США — национальная медаль науки.

А.А. Бизяев

ТЬЮРИНГ АЛАН МАТИСОН

ТЬЮРИНГ Алан Матисон (Turing Alan Mathison) (23 июня 1912, Лондон — 7 июня 1954, Уилмслоу, Великобритания), английский математик. Основные труды по математической логике, вычислительной математике. В 1936-37 годах ввел математическое понятие абстрактного эквивалента алгоритма, или вычислимой функции, получившее затем название «машины Тьюринга».

Родился в семье колониального чиновника в Индии. Обучался в Шерборнской школе и в Кингз-колледже в Кембридже.

Многие математики начала века были озабочены идеей исключения всех возможных математических ошибок путем создания алгоритма для установления истины. Однако математик Курт Гедель (1906-78) затруднил эти попытки, доказав свои т. н. теоремы о неполноте (теоремы Геделя), из которых, в частности, следует, что не существует полной формальной теории, где были бы доказуемы все истинные теоремы математики. Он показал, что любая математическая теория является неполной, поскольку должны существовать теоремы, истинность которых не может быть доказана в пределах данной теории. Под воздействием идей Геделя Тьюринг начал разрабатывать алгоритмический метод, способный определить, является ли данная задача не имеющей решения с целью исключить такие задачи из математики. Однако вместо этого в своей работе «О вычислимых числах» (1936) он доказал, что не существует такого универсального метода для определения вычислимости, и, следовательно, в математике всегда будут задачи, не имеющие решения (в отличие от пока неразрешимых). Работа Тьюринга опровергла мнение Дэвида Хилберта и его школы о том, что любая математическая теория может быть выражена через набор аксиом и теорем.

Чтобы проиллюстрировать свою точку зрения, Тьюринг предложил гипотетический механизм, названный «машиной Тюринга». Это устройство, состоявшее из бесконечной бумажной ленты с записанными на ней символами и считывающей головки, могло решать любые математические или логические задачи. Таким образом, она обладала основными свойствами современного компьютера: пошаговым выполнением математических операций, запрограммированных во внутренней памяти. Эта машина открыла дискуссию по теории автоматов и создала теоретическую базу для работы цифровых компьютеров, которые появились в 1940-е годы.

Тьюринг продолжил учебу в США — в Принстонском Университете, где под руководством американского математика Алонзо Черча в 1938 году получил степень доктора философии. Затем он вернулся в Великобританию, где был избран в совет Кингз-колледжа. Во время Второй мировой войны Тьюринг служил в правительственной шифровальной школе в Блетчли, где с помощью первых вычислительных машин пытались расшифровать германские послания, закодированные шифровальной машиной «Энигма». В конце 1943 года при участии Тьюринга была построена первая вычислительная машина, использовавшая вместо электромеханических реле 2000 электронных вакуумных ламп, — «Колосс», сыгравшая решающую роль в расшифровке шифров «Энигмы».

В 1945 году Тьюринг был принят в штат Национальной физической лаборатории в Лондоне, где возглавил разработку большого автоматического вычислительного устройства ACE (Automatic Computing Engine). В 1948 году Тьюринг был назначен заместителем Макса Ньюмена, директора вычислительной лаборатории Манчестерского университета, где создавался компьютер с самой большой по тому времени памятью — манчестерская автоматическая цифровая машина, или «Мадам» (Manchester Automatic Digital Machine), как ее называли в прессе. Тьюринг написал для нее несколько программ, пользуясь буквенно-цифровым кодом.

Работы Тьюринга по ранней технике программирования имели первостепенное значение. Ему также принадлежит мысль о том, что рано или поздно будет создан компьютер, способный мыслить, и предложил простой тест для определения этой способности у компьютера, названный «тестом Тьюринга». Эти работы Тьюринга считаются основополагающими в теории искусственного интеллекта. В 1952 году Тьюринг опубликовал первую часть своего учения о морфогенезе, развитии форм живых организмов. Эта работа осталась незаконченной, так как Тьюринг, впавший в депрессию в результате принудительного лечения от гомосексуализма, покончил с собой.

А.А. Бизяев

СКАЛЛИ ДЖОН

СКАЛЛИ Джон (Sculley John) (р. 1939, Нью-Йорк, США), американский бизнесмен, председатель и исполнительный директор компании Apple Computer в 1986-93 годах.

В 1961 году окончил университет Брауна со степенью бакалавра, а в 1963 — высшую финансовую школу Уортона при Пенсильванском университете со степенью магистра, после чего работал в области рекламы. В 1967 году был принят на работу в компанию PepsiCo, и к 1977 году стал ее президентом и исполнительным директором.

В 1983 году Стив Джобс лично пригласил Скалли возглавить Apple. В 1985 году из-за трений внутри компании Джобс вынужден был уйти в отставку, и Скалли через год стал исполнительным директором компании. Начиная с 1985 года, благодаря проводимой Скалли маркетинговой политике, позиции компании на рынке значительно укрепились. Особое внимание Скалли уделял разработке новых моделей компьютеров, совместимых с чужими компьютерными сетями. Он также много занимался распространением настольных издательских систем.

В 1991 году под руководством Скалли Apple вступила в стратегический альянс с IBM, в результате чего было создано совместное софтверное предприятие. Стремясь выйти на рынок бытовой электроники, Скалли энергично способствовал продвижению палмтопа Newton, представленного в 1993 году. Однако вскоре после того, как в июне 1993 года совет директоров Apple назначил на должность исполнительного директора президента компании Майкла Спиндлера, Скалли оставил фирму, заняв пост председателя и исполнительного директора радиокоммуникационной компании Spectrum Information Technologies, но уже в феврале 1994 ушел в отставку из-за разногласий с президентом компании.

ПАККАРД ДЭВИД

ПАККАРД Дэвид (Packard David) (7 сентября 1912, Пуэбло, шт. Колорадо — 26 марта 1996, Стэнфорд, шт. Калифорния), американский инженер и предприниматель, соучредитель компании Hewlett-Packard.

В 1934 году, получив степень бакалавра Стэнфордского университета (шт. Калифорния) был принят на работу в компанию Дженерал Электрик. В 1938 вернулся в Стэнфорд, где получил диплом инженера-электрика, а в 1939 вместе с Уильямом Хьюлеттом основал фирму, которая обосновалась в гараже небольшого двухэтажного дома в Пало-Альто. Первоначальный капитал компании составлял 538 (!) долларов. Сегодня на стене этого гаража красуется табличка «Здесь начиналась Силиконовая долина», а сам гараж стал неотъемлемой достопримечательностью Калифорнии.

Прошло много лет, и компания превратилась в мирового лидера по производству тестовых и измерительных систем, персональных компьютеров и принтеров. Паккард являлся президентом фирмы с 1947 по 1964 год, исполнительным директором с 1964 по 1968-й, председателем правления с 1964 по 1968 и с 1972 по 1993, после чего вплоть до своей смерти в 1996 году являлся почетным председателем в отставке.

В 1968 году президентом США Ричардом Никсоном Паккард был назначен заместителем министра обороны. На этом посту он прослужил до 1971 года, а в 1972 вернулся в Hewlett-Packard. В 1970-х и 1980-х годах Паккард был советником Белого дома по вопросам безопасности и менеджмента.

Хьюлеттом и Паккардом была разработана целая шкала ценностей «Путь Хьюлетта и Паккарда», в основе которой лежит основной принцип развития компании — постоянное движение вперед с учетом трансформаций рынка, а также неустанная забота о качестве жизни своих сотрудников.

Паккард активно участвовал в общественной жизни страны. Возглавлял многие фонды (в поддержку университетов, молодежных организаций, обществ планирования семьи и др.) и благотворительные организации. Большую роль сыграл в основании самого большого в мире Монтерейского Аквариума, на устройство которого семьей Паккарда было пожертвовано 55 млн. долларов. С полным правом можно сказать, что его вклад в дело социального и научно-технического развития страны беспрецедентен.

ОСБОРН АДАМ

ОСБОРН Адам (Osborne Adam) (р. 1940), американский предприниматель в области компьютеров, уроженец Таиланда. Создатель компании Osborne, производителя первых портативных персональных компьютеров.

Закончил Бирмингемский и Делавэрский университеты. В 1968 году защитил докторскую диссертацию по химической технологии. Начал работать с компьютерами в корпорации Shell Development в Калифорнии в конце 1960-х, а затем основал собственную компанию Osborne & Associates (1970) в Беркли (Калифорния). Работал издателем, консультантом-программистом с 1970 по 1979 годы. Пробовал себя и в области литературы. В 1979 Осборн продал свое издательство компании McGraw-Hil.

В 1980 он основал компьютерную фирму Osborne Computer Corporation, вложив в это 250 тысяч долларов собственных денег. Компания начала выпускать первый портативный персональный компьютер, имевший размер чемоданчика. Это был еще не ноутбук, так как весил он 11 кг, но уже мог умещаться под сиденьем самолета. Osborne I стал бестселлером при цене в 1795 долларов, включавшей стоимость всего оборудования, футляра, программ обработки текстов и электронных таблиц. Спрос на него был так велик, что спустя два с половиной года компания обанкротилась, не справившись со своими обязательствами. Тогда Осборн основал компанию Paperback Software, которая вскоре также развалилась после утраты авторских прав в пользу компании Lotus Development.

В 1990 году Осборн создал два новых предприятия — компанию по производству печатных плат в Индии для продажи их в США и компанию, занимающуюся искусственным интеллектом.

МАККРАКЕН ЭДВАРД

МАККРАКЕН Эдвард (McCracken Edward R.) (р. 1943), председатель и главный исполнительный директор (CEO) Silicon Graphics, Inc., крупнейшего мирового производителя средств компьютерной визуализации, высокопроизводительных рабочих станций и серверов.

Родился близ Фэрчайлда (шт. Айова) в небогатой фермерской семье. Посещал маленькую деревенскую школу. С десяти лет выращивал на ферме бычков, чтобы заработать на образование.

В 1966 году Маккракен окончил Государственный университет Айовы со степенью бакалавра точных наук и в 1968 году — Стэнфордский университет (по специальности электротехника). Вскоре он был принят на должность менеджера по продуктам и обучающим программам в компанию Hewlett-Packard, где он проработал 16 лет и дослужился до главного менеджера группы компьютерных систем.

В 1984 году Маккракен был приглашен в Silicon Graphics на должность председателя и главного исполнительного директора.

Невысокий, спокойный, он не соответствует привычному для стремительно развивающейся компьютерной индустрии образу грозного и решительного CEO. В своей деятельности Маккракен сделал ставку на медитативный метод стимулирования своей интуиции и творческих способностей. И это принесло свои плоды. За 10 лет, которые Маккракен провел у руля компании, доходы Silicon Graphics выросли с 5,3 млн. долларов до более чем 1,5 млрд., а общее количество работающих превысило 11 600 человек по всему миру, включая сотрудников поглощенной компании Cray Research, Inc. Доход компании за 1995 год составил 2,6 млрд. долларов, чистая прибыль — 216 млн. долларов. Около 35% продаж Silicon Graphics составили программы для технического конструирования, 20% — визуальное моделирование, 20% — научные цели и 20% — индустрия развлечений.

Значительную часть этого успеха приписывают комфортной атмосфере, царящей в Silicon Graphics. Когда в главном офисе компании проходят показы фильмов, в которых была использована технология SGI, повсюду бывают расставлены автоматы, продающие прохладительные напитки, и столы для пинг-понга.

Придерживаясь философии тесной взаимосвязи с потребителем, Маккракен считает, что нельзя следовать политике долгосрочного планирования. Это дает возможность оперативно вносить изменения в продукты компании и адаптировать их к быстро меняющимся условиям рынка визуальных компьютерных продуктов. Компания считает, что значительную часть мультимедиа-индустрии будущего составит интерактивное телевидение. Первый такой проект компании уже воплощен в жизнь.

Маккракен активно участвует в общественной жизни. В 1993 году он был назначен Биллом Клинтоном на должность сопредседателя Национального консультативного комитета по информационной инфраструктуре (The United States Advisory Council for the National Information Infrastructure Advisory Council). Маккракен также является директором нескольких корпораций, благотворительного фонда, заведует фондом Государственного университета Айовы.

Компания под руководством Маккракена привлекалась индустрией развлечений к созданию компьютерных эффектов в анимационных фильмах — таких хитах, как Toy Story, Apollo 13 и других. Предусмотрено активное участие Маккракена в работе созданной в 1997 году Стивеном Спилбергом электронной киностудии «DreamWork».

Среди многочисленных наград Маккракена — национальная медаль США за вклад в развитие технологии (1995), в частности за основополагающие работы в области трехмерной визуализации и суперкомпьютерных технологий, а также за его заслуги в превращении Silicon Graphics в глобальную мировую корпорацию передовых технологий. Эта медаль ежегодно вручается президентом США в качестве высшей награды за вклад в развитие современных технологий.

ЛЯПУНОВ АЛЕКСЕЙ АНДРЕЕВИЧ

ЛЯПУНОВ Алексей Андреевич (1911-73), российский математик, член-корреспондент АН СССР (1964). Автор трудов по теории множеств, математическим вопросам кибернетики, математической лингвистике.

Один из первых отечественных ученых, оценивших значение кибернетики. Под его руководством начались первые в нашей стране работы по кибернетике. В конце 1950-х годов Ляпунов сформулировал основные направления развития кибернетики, на основе которых в последующие десятилетия получили развитие общие и математические основы кибернетики, вычислительные машины, программирование и другие направления науки, разработал математическую теорию управляющих (кибернетических) систем. В 1954-64 годах Ляпунов организовал и вел семинар по кибернетике в МГУ.

Ляпунов создал первые учебные курсы программирования и разработал операторный метод программирования. Заложил основы машинного перевода и математической лингвистики, биологической кибернетики и математических методов в биологии.

С 1958 года руководил выходом сборника «Проблемы кибернетики», серии книг «Кибернетика в монографиях», выпуском на русском языке зарубежных работ по этой проблематике.

Последние годы жизни работал в Новосибирском Академгородке, участвовал в становлении Новосибирского университета, преподавал там кибернетику и математику.

ВОНГ ЧАРЛЬЗ

ВОНГ (Wang) Чарлз (р. 1941), американский предприниматель (китайского происхождения), президент и исполнительный директор компании Computer Associates (CA). Компания специализируется на выпуске программного обеспечения (программные продукты в области системного управления, СУБД, компьютерные средства экономического и финансового управления).

Родился в Шанхае в состоятельной семье юриста. После прихода коммунистов к власти в Китае в 1949 году семья была вынуждена эмигрировать в Соединенные Штаты, где осела в Нью-Йорке. Получив свободу, новоиспеченные иммигранты лишились прежнего достатка. Вонг пошел в американскую школу, ни слова не зная по-английски. В дальнейшем он положил немало сил, чтобы избавиться от «несмываемого» восточного акцента. Как многие переселенцы его поколения, он в полной мере испытал, что такое быть иммигрантом, да еще из Китая. Тем не менее благодаря своим незаурядным способностям, волевому характеру ему удалось поступить в колледж в Квинзе и успешно его закончить.

Там же в колледже в начале 1960-х годов он встретился с Руссом Артцем, неформальная связь с которым во многом определила планы Вонга в области бизнеса.

В 1976 году Вонг совместно с Артцем основал компанию Computer Associates, которой в недалеком будущем суждено было стать крупнейшей (второй после Microsoft) компьютерной фирмой в области программного обеспечения. Первым успешным продуктом компании была программа сортировки данных (CA Sort). Тонко уловив конъюнктуру (пользователям приходилось около 25% на эту процедуру), Вонг приобрел права на эту программу у швейцарской фирмы. Благодаря своим первым удачным шагам Computer Associates завоевала репутацию одного из лучших знатоков рынка софтвера.

С начала 1990-х годов компания становится лидером в области управления хранением и резервным копированием данных. Последним достижением в этой области является продукт ARCserve.

Принадлежащая CA технология ARCserve обеспечивает сквозное управление средствами хранения данных для всех ресурсов информационных технологий (ИТ), включая рабочие станции, серверы, базы данных, системы групповой работы, приложения и web-серверы. Эта технология позволяет осуществлять целостное управление хранилищами на любых платформах — от настольных ПК до мейнфреймов. Играя роль всеобъемлющего интегрированного решения по управлению средствами хранения, ARCserve обеспечивает не только простое резервное копирование, но также архивирование, восстановление сбоев, сетевую миграцию данных, реплицирование и управление носителями и устройствами.

Вонг, будучи удачливым предпринимателем, обладает репутацией «агрессивного» руководителя. Под его более чем двадцатилетним руководством компанией CA было поглощено более 60 фирм, сотни бывших сотрудников которых были выброшены на улицу. СМИ резко критиковали Вонга за бесчеловечное отношение к людям, тем не менее он остался непреклонен, считая выбранную стратегию единственно приемлемой.

Подбором сотрудников занимается специальное подразделение менеджеров, которое разработало целый комплекс интервью, распространяющихся абсолютно на всех приглашаемых в фирму людей. Сам Вонг играет не последнюю роль в этих обязательных мероприятиях.

При отборе людей Вонг исходит из очень простого принципа — прежде всего он считает, что руководителю, если он претендует на звание лидера, необходимо определить собственные достоинства и недостатки, чтобы добиться создания сбалансированного штата сотрудников. Работа с людьми остается приоритетным сегментом в его бизнесе.

БАРРЕТТ КРЕЙГ

БАРРЕТТ Крейг (Barrett Craig R.) (р. 29 августа 1939, Сан-Франциско, шт. Калифорния), американский ученый и предприниматель, глава корпорации Intel — мирового лидера в области микропроцессорных технологий.

С 1957 по 1964 год Барретт учился в Стэнфордском университете (шт. Калифорния), по окончании которого получил степень доктора физики по специальности материаловедение. В 1964-74 годах работал в должности адъюнкт-профессора на факультете материаловедения того же университета.

В 1969 году за свои исследования в области материаловедения Барретт был удостоен золотой медали имени Харди, присуждаемой Американским институтом инженеров горной и металлургической промышленности. Крейг Барретт — член американской Национальной академии инженерии, автор более 40 технических статей, в которых анализируется влияние микроструктуры на свойства материалов, а также учебника «Основы материаловедения», изданного в семидесятых годах и до сих пор широко используемого в университетах США.

В 1974 году поступил в корпорацию Intel в качестве менеджера по технологическому развитию. В 1984 году он занял должность вице-президента, в 1987 — старшего вице-президента, а в 1990 — исполнительного вице-президента. В 1992 году Крейг Барретт был избран в совет директоров корпорации Intel. В мае 1997 года доктор Барретт стал четвертым президентом Intel, а в 1998 году — исполнительным директором корпорации.

Барретт возглавляет Совет по полупроводниковым технологиям Департамента торговли США, является членом советов директоров корпорации Intel и ассоциации U. S. Semiconductor Industry Association.

ГЕЙТС УИЛЬЯМ

ГЕЙТС (Gates) Уильям (Билл) Генри III (р. 28 октября 1955, Сиэтл, шт. Вашингтон), американский предприниматель и изобретатель в области электронно-вычислительной техники, председатель и CEO ведущей компании в мире в области программного обеспечения Microsoft.

Родился в семье видного адвоката. Уже в средней школе проявил незаурядные математические способности. Будучи учеником старших классов, создал свою первую компанию Traf-O-Data, занимавшуюся продажей программ для определения интенсивности дорожного движения.

В 1975 году, бросив Гарвардский университет, где он готовился стать правоведом, как его отец, Гейтс совместно со своим школьным товарищем Полом Алленом основал компанию Microsoft. Первой задачей новой фирмы стала адаптация языка Бейсик для использования в одном из первых коммерческих микрокомпьютеров — «Альтаире» Эдварда Робертса.

В 1980 году Microsoft разработала операционную систему MS-DOS (Microsoft Disk Operation System) для первого IBM PC, ставшую к середине 1980-х годов основной операционной системой на американском рынке микрокомпьютеров. Затем Гейтс приступил к разработке прикладных программ — электронных таблиц Excel и текстового редактора Word, и к концу 1980-х Microsoft стала лидером и в этой области.

В 1986 году, выпустив акции компании в свободную продажу, Гейтс в возрасте 31 года стал миллиардером. В 1990 году компания представила оболочку Windows 3.0, в которой вербальные команды были заменены на пиктограммы, выбираемые с помощью «мыши», что значительно облегчило пользование компьютером. В начале 1990-х годов «Окна» продавались в количестве 1 миллиона копий в месяц. К концу 1990-х годов около 90% всех персональных компьютеров в мире были оснащены программным обеспечением Microsoft.

О работоспособности Билла Гейтса, а также его уникальном качестве эффективно включиться в работу на любом ее этапе ходят легенды. Безусловно, Гейтс принадлежит к когорте самых незаурядных бизнесменов новой генерации. В 1995 году он выпустил книгу «Дорога в будущее», которая стала бестселлером.

В 1997 возглавил список самых богатых людей в мире.

ЧЕМБЕРС ДЖОН

ЧЕМБЕРС (Chambers) Джон, американский предприниматель, президент и главный исполнительный директор компьютерной компании Cisco Systems, ведущего производителя оборудования для построения инфраструктуры глобальных сетей (Интернет), организации межсетевого взаимодействия, создания узлов для удаленного доступа и подключения корпоративных сетей к глобальным. В конце 1990-х годов компания превратилась в глобального рыночного лидера. Ее доход в 1998 году составил 8,5 млрд. долларов.

Чемберс окончил университет штата Индианы по специальности финансы и управление, позднее получил юридическое образование в области бизнеса. Свою профессиональную карьеру начал в компании Ванга, затем работал в IBM. Взлет его карьеры начался с переходом в Cisco, где проявились его незаурядные способности менеджера.

В 1996 году по опросу журнала «Business Week» Чемберс вошел в десятку лучших менеджеров года. Он сумел наладить плодотворное сотрудничество с компьютерными гигантами Microsoft и Intel, союз которых по выражению журнала «Fortune» был назван стратегическим альянсом.

В 1998 году Чемберс, единственный из американцев, был удостоен специальной премии японского премьер-министра в области бизнеса. Чемберс является советником президента Билла Клинтона в области бизнеса, вице-президента Альберта Гора и американского Конгресса в области производства.

Бизяев Алексей Анатольевич

Бизяев Алексей Анатольевич – разработал данные лекции для студентов НГТУ, факультета РЭФ.

По поводу использования данной презентации в своем курсе обращайтесь:
«Новосибирский государственный технический университет», факультет РЭФ, кафедра КТРС, IV-530а
Тел.: +7(383) 3460633
Сот.: +7-905-958-6134
E-mail: bizyaev@ngs.ru

Спасибо за внимание!