**Вопросы к зачету   
по дисциплине «Архитектура вычислительных систем   
(часть 1)»**

1. Понятие архитектуры компьютера (с примерами архитектур)

Компьютер представляет собой универсальное устройство, которое используется для автоматической обработки информации, осуществляемой по заранее составленному плану. Такой план, обладающий необходимыми для получения требуемого результата свойствами, принято называть алгоритмом, а его запись в определенной, понятной компьютеру форме называется программой.

Термин «архитектура» трактуется самым различным образом и вообще-то не имеет общепризнанного определения.

Термин «архитектура» в применении к компьютерам относительно нов. Почти до конца 70-х гг. применялись более узкие термины «устройство» или «структура» компьютера. Но, как правило, после рассмотрения собственно устройства машины следовало изучение способов представления программ и данных в компьютере, особенностей различных устройств компьютера, организации обмена данных и т.д. Впоследствии всю эту совокупность сведений объединили под одним общим названием – **архитектура компьютера**.

Под архитектурой компьютера понимается совокупность взаимосвязанных сведений о способах представления в компьютере программ и данных, о назначении, структуре и особенностях функционирования отдельных его устройств, а также об организации компьютера в целом.

В «Толковом словаре по вычислительным системам» архитектура компьютера определяется следующим образом: **Архитектура** – это описание цифровой вычислительной системы на некотором общем уровне, включая описание пользовательских возможностей программирования, системы команд и средств пользовательского интерфейса, организации памяти и системы адресации, операций ввода–вывода и управления и т.д.

Основными функциями компьютера является хранение, обработка, прием и передача данных. Для выполнения каждой из этих функций в компьютере предусмотрены специальные устройства.

1. **Память** – группа устройств, которые обеспечивают хранение программ и данных;

2. **Процессор** (от process – обработка) – одно или несколько устройств, которые обеспечивают задаваемую программой обработку данных;

3. **Устройства ввода/вывода** – группа устройств, которые обеспечивают обмен, то есть прием и передачу данных между пользователем и компьютером или между двумя или более компьютерами.

Различные устройства компьютера присоединяются друг к другу с помощью стандартизированных и унифицированных (т.е. единообразных) аппаратных средств – кабелей, разъемов и т.д. При этом устройства обмениваются друг с другом информацией и управляющими сигналами, которые также приводятся к некоторым стандартным формам. Совокупность этих стандартных средств и форм образует конкретный **интерфейс** (от interface – взаимный вид) того или иного устройства или программы.

Изучение архитектуры компьютера естественно начать с обсуждения логического строения его памяти.

Память компьютера имеет сложную многоуровневую структуру, реализованную в виде взаимодействующих устройств, которые могут использовать различные физические принципы для хранения данных. К этим устройствам относятся интегральные схемы, магнитные и оптические диски т.д. Многоуровневый подход к реализации памяти вытекает из необходимости обеспечения эффективной работы компьютера при решении задач, точнее при выполнении соответствующих им программ. Но в любом случае, при любой физической реализации памяти ее базовыми функциональными элементами являются **бит** и **байт**.[N1]

Не рассматривая способы физической реализации бита рассмотрим его основные функциональные возможности.

1. Бит может находится только в одном из двух возможных состояниях, одно из которых принято считать «0», а другое – «1».

2. В любой момент времени можно узнать, в каком из двух состояний находится бит, при этом текущее состояние бита остается неизменным;

3. Всегда, когда в этом возникает необходимость и вне зависимости от текущего состояния, можно перевести из одного состояния в другое. Иначе говоря, в бит можно записать информацию.

**Бит** обеспечивает необходимую основу для реализации функции **хранения данных**.

Бит – очень маленькая порция данных. Для представления хранения, например, одной цифры десятичного числа требуется 4 бита. Для хранения двоичных машинных кодов используются несколько битов, которые совместно образуют устройство, которое называется **ячейка памяти**. В общем случае ячейки различных компьютеров могут состоять из различного числа битов, однако это создает значительные сложности при обмене данными между компьютерами различных типов. Поэтому а настоящее время стандартными являются ячейки, состоящие из 8-ми битов – байт.

Самостоятельный элемент памяти компьютера, состоящий из восьми битов, называется **байтом.**

Байт может содержать произвольную комбинацию (двоичный код) из 8-ми нулей и единиц; количество различных кодов, представимых при помощи байта – 256.

***Основная (оперативная) память ЭВМ обычно является адресной. Это означает, что каждый хранимый в ОП единице информации ставится в соответствие определенное число – адрес.***

Более крупные единицы информации – основная машинная единица и ее производные – слово, полуслово, двойное слово и т.п. образуются из целого числа байт. Обычно слово соответствует формату данных, наиболее часто встречающееся в данной машине в качестве операнда.

Нумерация бит в байте и байт в слове может быть разной. Рассмотрим некоторые примеры.

Отметим следующие особенности:

1. Нумерация бит в байте идет слева направо от 0 до 7. Таким же образом идет нумерация бит в слове, полуслове и т.д.

2. Адресом единицы информации (слова, например) является самый младший из номеров, составляющих ее байтов.

3. Старшие разряды слов, полуслов и т.д. записываются в байты с меньшим номером. Т.е. более значащие разряды хранятся по меньшим адресам.

Для организации памяти IBM 360 характерна одна особенность, называемая выравниванием; она требует, чтобы любая единица информации хранилась по адресу, кратному размеру единицы информации в байтах. Например, слово можно хранить только по адресам 0, 4, 8, 12, и т.д.

Такая организация памяти отражалась на языках программирования. Например, FORTRAN IV для IBM 360/370 переменные в т.н. COMMON-блоках (глобальные переменные, extern и C) программист должен был обязательно размещать в порядке убывания их размеров в байтах.

Важнейшей характеристикой любых устройств памяти является объем. Объем равен количеству байтов, из которых он состоит.

1. Преобразование информации и представление данных в компьютерных системах, числовые, буквенные данные, изображения (с примерами)

ПРЕДСТАВЛЕНИЕ ДАННЫХ

При компьютерной обработке информации приходится иметь дело с текстовыми, графическими, числовыми, звуковыми и другими данными. Для хранения данных различной природы применяются различные способы их представления в двоичном алфавите – различные способы кодировки. Кроме того, для одной и той же разновидности данных также могут использоваться различные способы кодировки, которые отличаются друг от друга эффективностью и различными требованиями к ресурсам компьютера.

Конкретный способ кодирования той или иной разновидности информации в компьютере принято называть форматом данных.

В общем случае термин «формат» понимается как строго определенный, исчерпывающе полный набор правил. Следовательно, в приведенном ранее определении речь идет об исчерпывающем наборе правил кодирования в компьютере той или иной разновидности данных.

ТЕСТОВЫЕ ДАННЫЕ

При хранении в компьютере любой текст рассматривается как линейная последовательность символов. Каждому символу в последовательности ставится в соответствие конкретный двоичный код, состоящий ровно из восьми двоичных разрядов (одного байта).

Алфавит, используемый для представления текстов на естественном языке должен содержать как минимум 52 латинские буквы (строчные и прописные), десятичные цифры, знаки препинания, математические знаки, специальные знаки и т.д., всего примерно 150 символов. Исходы из теоретических соображений, это требует при равномерном алфавитном двоичном кодировании для представления любого знака исходного алфавита log2150 = 7,2, т.е. восьми двоичных цифр.

Списки всех используемых при записи текстов символов и однозначно соответствующих им двоичных кодов образуют так называемые **кодовые таблицы**. В практике программирования применяются различные кодовые таблицы. Одной из наиболее часто используемых является кодовая таблица **ASCII** (American Standard Code for Information Interchange – американский стандартный код для обмена информацией). В этой таблице зафиксированы коды для 128 различных символов. Их список и соответствующие им восьмиразрядные (то есть состоящие из восьми двоичных цифр, разрядов) двоичные коды образуют основную (базовую) кодовую таблицу ASCII. Но один байт может содержать 256 различных восьмиразрядных кодов. Это означает, что в стандарте ASCII задействована только половина возможностей 8-битного кодирования. Имеются различные расширения основной кодовой таблицы ASCII, в которых задаются коды еще для 128 символов, в том числе и для символов различных национальных алфавитов.

Если учесть все возможные буквы, входящие в национальные алфавиты различных стран, все возможные символы, которые встречаются в математических и других специальных текстах, то их общее количество окажется значительно больше 256 символов, кодирование которых обеспечивает один байт. Поэтому было разработано несколько десятков различных кодовых таблиц. При этом в разных кодовых таблицах один и тот же код может соответствовать разным символам.

Учитывая недостаточность возможностей обсуждавшихся кодовых таблиц, в последнее время все шире используется кодовая таблица **UNICODE** (UNIversal CODE – универсальный код), в которой для кода одного символа отводится два байта, а не один, как в рассмотренных ранее таблицах. С помощью двух байтов, то есть 16 битов, можно закодировать уже 216 = 65536 различных символов. Такого количества кодов вполне достаточно, чтобы представить большинство из встречающихся во всевозможных текстах символов.

Основные элементарные действия, которые должен уметь выполнять компьютер при обработке текстовых данных – сравнение кодов двух символов и замена одного кода другим.

Для рассмотрения общих принципов кодирования графических данных рассмотрим изображение на экране компьютера. Изображение состоит из некоторого количества горизонтальных линий (строк). Каждая строка состоит из элементарных точек – пикселов (PICture’S Element). Весь массив называют растром. Разрешающая способность (разрешение) – определяется количеством пикселей.

Строки, из которых состоит изображение, можно просматривать сверху вниз друг за другом. Такой способ работы с изображением называется строчной разверткой или сканированием. Таким образом, все изображение можно представить как линейную последовательность пикселов.

Рассмотрим принципы кодирования монохромного изображения (например, черно=белого). Каждый пиксел изображения может быть представлен либо «0», либо «1», а все изображение – двоичной последовательностью. Длина последовательности определяется разрешаюшей способностью. НапримерФ, при разрешающей способности 600 х 800, необходимое количество пикселов 480 000, которые могут быть закодированы при помощи 480 000 бит или 60 000 байт.

Однако полученное таким образом изображение будет чрезмерно контрастным. Реальное черно-белое изображение состоит не только из белого и черного цветов. В него входит множество различных промежуточных оттенков – серый, светло-серый, темно-серый и т.д. Если кроме белого и черного цветов использовать только две дополнительные градации, скажем, светло-серый и темно-серый цвета, то чтобы закодировать цветовое состояние одного пиксела, потребуется уже два бита. При этом кодировка может быть, например, такой: черный цвет – 002, темно-серый – 012, светло-серый – 102, белый – 112.

Общепринятым на сегодняшний день и дающим достаточно реалистичные монохромные изображения считается кодирования состояния одного пиксела с помощью восьми битов, то есть одного байта, которое позволяет передавать 256 различных оттенков серого цвета, от полностью белого до полностью черного. В этом случае для передачи всего растра из 800х600 пикселов потребуется уже не 60 000, а 480 000 байтов.

Цветное изображение может изображаться различными способами. Один из них – метод RGB (Red, Green, Blue), который опирается на то, что человек воспринимает все цвета как сумму трех основных цветов. В таком представлении для кодирования каждого пиксела надо минимум три бита, тогда количество различных цветов пиксела равно восьми. Если каждый из цветов кодировать с помощью байта, то можно передать 256 оттенков каждого цвета или 16 777 216 различных цветов. Такой способ принято называть True Color (точный цвет) или полноцветным режимом.

Полноцветный режим требует очень много памяти. Например, при разрешении экрана 600 х 800 метод RGB требует 1 440 000 байт, что соответствует 1,37 Мбайт. Отсюда возникает необходимость разработки различных режимов работы и графических форматов, которые позволяют сократить объем памяти.

Большинство действий, которые приходится выполнять над графическими данными в пиксельном формате, сводятся к замене текущего цвета пиксела другим цветом. Например, стирание какого-либо участка изображения – это не что иное как замена цветов всех пикселов стираемого участка цветом фона рисунка. Следовательно, и для графических данных элементарные действия, которые должен «уметь» выполнять компьютер, сводятся к сравнению двух двоичных кодов и замене одного кода другим.

ЧИСЛОВЫЕ ДАННЫЕ

Вообще говоря, запись любого числа может рассматриваться как часть текста и в этом случае цифры – это обычные символы алфавита, для которых существенным являются только действия сравнения и замены.

Однако, в задачах обработки информации требуется выполнять различные математические операции над числами. Учитывая это принципиальное отличие, данные над которыми должны выполнятся математические преобразования, выделяют в особую группу числовых данных и для них применяются специальные способы кодирования.

ФОРМАТЫ ПРЕДСТАВЛЕНИЯ ЧИСЕЛ В КОМПЬЮТЕРЕ

Для представления числовых данных в компьютерах используются два принципиально разных формата: формат с **фиксированной точкой** (запятой) и формат с **плавающей точкой** (запятой).

Формат с фиксированной точкой предназначен для абсолютно точного представления целых чисел. В программировании эти числа относятся к целому типу, в то время как формат с плавающей точкой используется для представления только нецелых, приближенных чисел. В программировании такие числа относятся к вещественному типу. Напомним, что вещественные числа возникают в задачах в результате различных измерений (например, измерений веса тела или его длины), которые, как известно, всегда выполняются с некоторой погрешностью, приближенно.

ФОРМАТЫ ЦЕЛЫХ ЧИСЕЛ

Существуют две модификации формата с фиксированной точкой, которые принято называть его беззнаковым и знаковым представлениями. Беззнаковое представление используется для работы с целыми неотрицательными числами, а существующее в нескольких вариантах знаковое – для работы как с положительными, так и с отрицательными числами

**Беззнаковое представление формата с фиксированной точкой**

В беззнаковом представлении целого числа используется прямой двоичных код, который представляет собой запись этого числа в двоичной системе счисления. При этом все разряды занятого числом поля содержат его значащие цифры. Точка, отделяющая целую часть от дробной считается расположенной, фиксированной справа от крайнего правого разряда. Следовательно, под дробную часть числа отводится нулевое количество разрядов, и в данном варианте кодировки возможна работа только с целыми числами. Постоянное расположение, фиксация позиции точки дала название формату – с фиксированной точкой.

Для кодирования чисел в формате с фиксированной точкой используются поля длиной 1, 2 или 4 байта, поэтому N может быть равно 8, 16, 32. В таблице 2.1 приведены обычно используемые в программировании названия соответствующих этим полям целых типов и диапазоны их возможных значений.

**Знаковые представления формата с фиксированной точкой**

В истории развития архитектуры компьютеров использовались четыре различных варианта представления знаковых чисел:

1. Система со знаком;

2. Обратный код, поразрядное дополнение или код с дополнением до единицы;

3. Дополнительный, комплементарный код или код с дополнением до двух;

4. Система со смещением.

В настоящее время первые две системы устарели и практически вышли из употребления. Тем не менее по ходу изложения материала мы затронем и эти устаревшие системы.

Очевидно, что для машинного представления знаковых чисел нужно определенным образом закодировать знак числа и его модуль. Так как существует всего два знака чисел, «+» и «–», то самое простое напрашивающееся решение состоит в том, чтобы представить код знака одной двоичной цифрой и выделить под него один из разрядов поля, а во всех остальных разрядах записывать модуль числа, кодируя его, например, с помощью прямого двоичного кода, так же как кодируются беззнаковые целые числа. Такой способ называется **системой** **кодирования со знаком**. Код знака числа принято размещать в самом левом разряде поля, который в связи с этим принято называть **знаковым битом**. По традиции знак «плюс» кодируется нулем, а знак «минус» – единицей. Таким образом, если в знаковом бите находится нуль, это означает, что остальные биты поля содержат модуль положительного числа, а если знаковый бит занят единицей, то в них находится модуль отрицательного числа. Заметим, что можно было бы договориться и о другом способе кодирования знака, но по ряду причин выбран именно этот способ.

Обратите внимание на то, что в знаковом представлении различные биты кода играют различную роль, в то время как во всех рассматривавшихся ранее вариантах кодирования все цифры кода играют одну и ту же роль, например, образуют код отдельного символа текста или являются значащими цифрами беззнакового кода целого числа. Закрепление за разрядами поля конкретных функций хранения различных элементов кода принято называть **разрядной сеткой**.

Поскольку один из N битов поля отводится под знак числа, под запись кода модуля остается N-1 битов. Следовательно, в таком поле могут быть закодированы целые числа в диапазоне

**Такое представление обладает следующими особенностями.**

1. 010 соответсвует два кода 00…..0 (+010) и 10…0 (-010), что приводит к дополнительным аппаратным затратам при анализе кода.

2. Во-вторых, и это самое главное, возникает специальная арифметика с совершенно непривычными правилами выполнения самых обычных действий. Так по правилам обычной арифметики, сложение чисел +410 и -410 дает в результате 010. А теперь выполним сложение для полученных ранее кодов этих чисел: 000001002 + 100001002 = 100010002­. Как видим, получен совершенно неожиданный результат: вместо ожидавшегося кода числа 010 сложение дало код числа -810. Этот результат является следствием неудачного выбора способа кодирования. Его нужно выбирать исходя из логики использования кода, а не из «напрашивающегося» или «очевидного» на первый взгляд подхода. Основное требование при выборе системы кодирования чисел состоит в том, что полученный код должен удовлетворять правилам выполнения сложения и вычитания в двоичной системе счисления. В связи с этим код каждого следующего положительного числа должен получаться прибавлением единицы к коду текущего числа, а код каждого следующего отрицательного числа должен получаться вычитанием единицы из кода текущего числа. Построенный таким образом код принято называть **дополнительным**, **с** **дополнением до двух** или **комплементарным** (от complementary – дополняющий).

Получение дополнительных кодов

Способ кодирования знаковых чисел, основанный на использовании дополнительного кода устраняет все отмеченные ранее недостатки применения системы кодирования со знаком. Во-первых, необходимые арифметические свойства удовлетворяются автоматически по способу построения кода. Например, при сложении кодов 01002 (+410) и 11002(–410) получается код 100002­, старшая единица которого не помещается в используемые четыре разряда и отбрасывается. Таким образом, остается код 00002­, который в точности соответствует нужному результату. Отметим, что это общий технический прием при выполнении действий со знаковыми числами. Он аналогичен приему, использованному в рассуждениях при получении дополнительного кода числа –1. Можно считать, что во время выполнения аналогичных операций возвращается заем, выполненный ранее из воображаемого дополнительного разряда.

Во-вторых, исчезла неоднозначность кодирования нуля. В самом деле, код 10002, который раньше вместе с кодом 00002 использовался для кодирования нуля, оказался закрепленным за кодом максимального по модулю отрицательного числа -810. Это привело к тому, что диапазоны представления положительных и отрицательных чисел стали несимметричными. В самом деле, если для записи кода используется только 4 бита, то при выборе любого способа кодирования возможно формирование

всего 24 = 16 различных кодов. Из них при применении системы со знаком 14 кодов закреплено за ненулевыми числами из симметричного диапазона от ‑710 до +710, и еще два кода, 00002 и 1000­2, соответствуют нулю. Применение дополнительного кода позволяет изобразить шестнадцать целых чисел (вместе с нулем) из несимметричного диапазона от -810 до +710. При этом под нуль занят всего один код 00002, а закрепление кода 10002 за числом -810 как раз и приводит к появлению несимметричности диапазона.

При использовании дополнительного кода в поле длиной N бит можно закодировать числа в диапазоне

Получение дополнительного кода целых чисел.

1. Прямой код записывается в выбранное поле длиной 1, 2 или 4 байта.

2. Прямой код инвертируется (обращается), то есть каждая цифра 1 кода заменяется цифрой 0. Заметим, что полученный таким образом код называется **обратным**, **с дополнением до единицы** или **с поразрядным дополнением**. К обратному коду прибавляется единица.

ФОРМАТ ВЕЩЕСТВЕННЫХ ЧИСЕЛ

Обычно используемая запись числа в виде ±a, b, содержащем целую (а) и дробную (b) части, считается основной формой записи вещественных чисел. В естественных науках довольно часто вместо основной формы числа используется его запись в виде произведения двух сомножителей, один из которых является основанием системы счисления в некоторой степени, например: 2,9 ∙ 1018 или 0,91 ∙ 10-31. Чаще всего такая запись используется во время работы с очень большими или очень маленькими по модулю числами. При этом достигается значительная экономия во времени, наглядности, в простоте восприятия содержащего такие числа текста. Сравните, например, способы записи одного и того же числа: 2 900 000 000 000 000 000 и 2,9 ∙ 1018, а также 0,000000000000000000000000000000091 и 0,091 ∙ 10-30.

Запись вида ±m×p±q называется **формой с порядком** или **экспоненциальной формой** вещественного числа. Некоторые авторы используют также название **полулогарифмическая форма**. Сомножитель m принято называть **мантиссой**, а степень ±q, в которую возводится основание p системы счисления, – **порядком** числа. Так, для числа 2,9 ∙ 1018 мантисса m = 2,9, основание p = 10, а порядок q = 18.

**Нормализованные числа**

Обсуждаемая форма представлени чисел не дает однозначной записи одного и того же числа. Например

1234,5 =123,45 ∙10^1=12,345∙10^2=1,2345∙10^3 ….

Чтобы избавиться от указанной неоднозначности, на значение мантиссы m накладывают какое-либо ограничение, например, 0,1 ≤ m < 1 или 1 ≤ m < 10. Отметим, что в стандартах последнего времени закрепилось использование ограничения 1 ≤ m < 10, которое в случае двоичной системы счисления означает, что целая часть мантиссы всегда должна быть равна единице.

**Выполнение операций над нормализованными числами**

Операций на числами в основной форме выполняются поразрядно. При представлении числа в нормализованном виде так делать нельзя, т.к. «вес» одно и того же разряда мантиссы зависит от величины порядка числа.

Поэтому перед выполнением операции производят выравнивание порядков, т.е. производится процедура приведения чисел к одинаковому порядку. Это делается при помощи денормализации числа с меньшим порядком. Для этого порядок числа увеличивается на единицу, а мантисса уменьшается в десять раз. Так выполняется до тех пор, пока порядки чисел не станут равными.

1,23456=0,123456∙10^1=0,0123456∙10^2=0,0123456∙10^3=….

После выполнения действия результат может быть ненормализованным. В этом случае, следует выполнить процедуру нормализации числа.

Обнаруженное усложнение выполнения действий над нормализованными числами является платой за возможность работать с числами из гораздо более широкого диапазона, чем в формате с фиксированной точкой. Заметим также, что используемый для представления вещественных чисел формат с плавающей точкой получил свое название в связи с необходимостью перемещать во время выполнения арифметических операций текущее положение точки (запятой), отделяющей целую часть числа от дробной.

**Общие принципы кодирования чисел в формате с плавающей точкой**

Для кодирования вещественных чисел в формате с плавающей точкой используется их нормализованная форма ±m×p±q, 1 ≤ m < 10. Анализ этой формы показывает, что в системе счисления с заданным основанием p код нормализованного числа должен содержать:

1. Код знака числа;

2. Код нормализованной мантиссы 1 ≤ m < 10;

3. Код знака порядка;

4. Код порядка p.

В ранний период развития вычислительной техники разные разработчики компьютеров использовали различные аппаратные реализации формата с плавающей точкой. Его техническая реализация имеет много тонкостей, связанных с реакцией процессора на возникающие во время вычислений ситуации исчезновения порядка и переполнения. Неаккуратный подход к реализации иногда приводил к возникновению грубых ошибок в вычислениях. Кроме того, различные представления формата с плавающей точкой препятствовали переносу программ с компьютеров одной модели на компьютеры других моделей. Такой разнобой продолжался до начала 1980-х гг., когда одной из комиссий **IEEE** (Institute of Electrical and Electronics Engineers – Международный институт инженеров по электротехнике и электронике) было поручено создание стандарта для реализации формата с плавающей точкой. В результате был создан стандарт на представление формата с плавающей точкой в компьютерах, который под номером **IEEE** **754** бы принят в 1985 г. в качестве международного. В настоящее время процессоры основных мировых разработчиков (Intel, SPARC, Power и т.д.) придерживаются этого стандарта.

Стандарт IEEE 754 предусматривает, что двоичные коды чисел в формате с плавающей точкой могут занимать поля длиной 4, 8 и 10 байтов. В соответствии с общими принципами кодирования чисел в этом формате поля содержат код знака числа, код порядка и его знака, а также код мантиссы.

Код знака числа как всегда занимает один, самый левый разряд поля. Кодируется знак также стандартным образом: 0 для положительных чисел и 1 – для отрицательных. Разработчики стандарта предложили использовать для кодирования порядка со знаком систему со смещением, константа которой равна K = 2N-1 – 1, где N – количество разрядов поля, выделенных под хранение порядка и его знака. Если истинный порядок нормализованного числа равен Pн, то в указанных битах поля записывается прямой двоичных (беззнаковый) код суммы Рм = Рн + К, который принято называть машинным порядком исходного вещественного числа. Количество разрядов N, которое выделяется под **машинный порядок** равно 8, 11 и 15 для 4-, 8- и 10-байтовых полей соответственно. Таким образом, в обсуждаемом стандарте минимально возможный N-битовый машинный порядок у обычных чисел – 000…012, а максимально возможный – 111...102. Более подробно это вопрос обсуждается далее.

Оставшиеся от знакового бита и кода порядка разряды полей отводятся под код мантиссы, который, таким образом, занимает 23, 52 и 64 бита для 4-, 8- и 10-байтовых полей соответственно. В стандарте IEEE 754 мантисса всегда считается нормализованной, за исключением уже упоминавшихся специальных значений, которые имеют порядки 000…002 и 111…112. Но у нормализованной мантиссы в двоичной системе счисления целая часть всегда равна единице. Поэтому разработчики стандарта предложили при хранении числа эту всегда равную единице целую часть не указывать в разрядной сетке поля. При хранении кодов чисел она только подразумевается, но во время выполнения действий целая часть автоматически восстанавливается. С помощью этого приема обеспечивается увеличение точности дробной части мантиссы, так как сэкономленный на неявно задаваемой целой части разряд отводится под дополнительную цифру дробной части мантиссы.

Имеется три формата с плавающей точкой (одинарной точности ОТ, двойной точности ДТ, расширенной точности РТ). Значащие числа находятся в поле мантиссы, поле порядка показывает фактическое положение десятичной точки в разрядах мантиссы, а бит знака определяет знак числа.

Устройство FPU обычно поддерживает представление мантиссы в нормализованной форме, т.е. старший бит равен 1. Следовательно, за исключением числа нуль мантисса состоит из целой части и дроби в следующем виде: 1.(F1)(F2)(F3) … (Fn), где Fi = 0 или 1.

В форматах ОТ и ДТ бит F0 при передаче чисел и хранении их в памяти не фигурирует. Это т.н. скрытый и неявный бит, который содержит 1. Следовательно, в этих форматах невозможно представить ненормализованные числа (за некоторым исключением). Кроме того, скрытый бит не позволяет представить в этих форматах нуль, и он должен кодироваться как специальное значение.

Числа в формате РТ имеют явный бит F0. Напомним, что внутри FPU числа хранятся только в этом формате. Перевод в другие форматы из них осуществляется автоматически.

В качестве примера рассмотрим представление числа –247.375 в вещественных форматах FPU.

-247.375 =>-11110111.011

Определяем истинный порядок. Он равен –7.

Смещенный порядок:

· для обычной точности: 7+127=134 è 1000 0110

· для двойной точности: 7+1023=1030 è100 0000 0110

· для расширенной точности: 7+16383=16390 è 100 0000 0000 0110.

Значения чисел:

· ОТ: 1 1000 0110 1110 1110 0110 00 ….00

· ДТ: 1 100 0000 0110 1110 1110 0110 00 ….00

· РТ:1 110 0000 0000 0110 1111 0111 0011 00…00.

Специальные значения. Форматы чисел допускают представление специальных объектов и значений. Для их представления зарезервированы значения смещенного порядка 000 … 0 и 111 … 1.

Сравнивая логику вычислений с данными, представленными в формате с фиксированной точкой, и логику выполнения действий над данными в формате с плавающей точкой, несложно заметить существенно более сложный характер вычислений с плавающей точкой. В последнем случае необходимо, по крайней мере, уметь выделять одиночные биты и некоторые группы битов – участки полей, которые соответствуют кодам порядка и мантиссы. Дальнейшая раздельная обработка этих кодов уже близка к действиям над целочисленными данными. Для выделения некоторых групп битов процессор должен в дополнение к вышеперечисленным операциям сравнения, сложения и сдвига «уметь» выполнять над двоичными кодами такие операции, как дизъюнкция, конъюнкция и отрицание.

Минимальный набор действий, которыми владеть процессор компьютера, включает операции сравнения, сложения, дизъюнкции и конъюнкции, а также пересылки, отрицания и сдвига одиночного кода на некоторое количество разрядов вправо и влево.

ОСОБЕННОСТИ КОМПЬЮТЕРНОЙ АРИФМЕТИКИ

Завершая обсуждение принципов кодирования числовой информации, необходимо особо подчеркнуть, что, несмотря на используемые в современных компьютерах изощренные приемы кодирования, достичь полного соответствия между обычно арифметикой и компьютерной обработкой числовых данных принципиально невозможно. Этот вывод является следствием того, что в любом случае код числа в памяти компьютера, представляющего собой реальное техническое устройство, занимает конечное количество разрядов поля. Это количество теоретически можно сделать сколь угодно большим, но оно принципиально не может быть бесконечным, в то время как в математике числа в общем случае изображаются бесконечным количеством значащих цифр.

Конечность количество разрядов поля, используемого для записи кода числа, приводит, во-первых, к ограниченности диапазонов кодируемых целых и вещественных чисел, а во-вторых, к конечной точности представления вещественных чисел. Если результат выполнения некоторой операции над данными с фиксированной точкой выйдет за границы диапазона представимых в данном поле чисел, то либо лишние биты окажутся утерянными, либо будет сформировано сообщение о возникновении ошибки и вычисления будут остановлены. Если для представления чисел используется формат с плавающей точкой, то результат может быть заменен машинным нулем или бесконечностью. В этом случае в зависимости от конкретной ситуации дальнейшие вычисления могут вернуть приемлемый приближенный результат либо так же, как и в предыдущем случае, может быть сформировано сообщение об ошибке с прекращением вычислений.

Пример нарушения ассоциативного закона в компьютерной арифметике рассматривался ранее. В точки зрения обычной арифметики, значение выражения (y – x) + x всегда одно и то же и равно у, в то время как получаемый на компьютере результат существенно зависит от значений величин x и у, от используемого формата кодирования – с фиксированной или с плавающей точкой, а если используется формат с плавающей точкой, то и от реализованной в компьютере реакции на исчезновение порядка.

Теперь рассмотрим пример нарушения дистрибутивного закона. Пусть требуется вычислить значение выражения (x + y)z. Если значения х и у велики, то их суммирование может вызвать переполнение, и тогда результат не будет получен вообще или же он может трактоваться как бесконечно большой. С другой стороны, если сомножитель z мал и выражение реализовано в эквивалентном с точки зрения обычной арифметики виде xz + yz, то велика вероятность того, что вычисления вернут правильный результат. Итак, мы опять сталкиваемся с ситуацией, когда эквивалентные с точки зрения обычной математики выражения в компьютерном варианте вычислений могут иметь разные значения.

Пример нарушения ассоциативного закона в компьютерной арифметике рассматривался ранее. В точки зрения обычной арифметики, значение выражения (y – x) + x всегда одно и то же и равно у, в то время как получаемый на компьютере результат существенно зависит от значений величин x и у, от используемого формата кодирования – с фиксированной или с плавающей точкой, а если используется формат с плавающей точкой, то и от реализованной в компьютере реакции на исчезновение порядка.

Теперь рассмотрим пример нарушения дистрибутивного закона. Пусть требуется вычислить значение выражения (x + y)z. Если значения х и у велики, то их суммирование может вызвать переполнение, и тогда результат не будет получен вообще или же он может трактоваться как бесконечно большой. С другой стороны, если сомножитель z мал и выражение реализовано в эквивалентном с точки зрения обычной арифметики виде xz + yz, то велика вероятность того, что вычисления вернут правильный результат. Итак, мы опять сталкиваемся с ситуацией, когда эквивалентные с точки зрения обычной математики выражения в компьютерном варианте вычислений могут иметь разные значения.

Известно, что на числовой прямой любой отрезок или интервал содержит несчетное множество действительных чисел, в то время как любое поле для представления чисел в формате с плавающей точкой всегда может быть использовано для записи лишь конечного количества различных кодов. Это влечет за собой следующую особенность, связанную с отсутствием свойства плотности у множества чисел с плавающей точкой. Свойство плотности множества действительных чисел состоит в том, что для любых двух действительных чисел α и β существует число χ – такое, что α < χ < β. В компьютерной математике из-за отсутствия этого свойства между двумя кодами не всегда можно найти код, обладающий таким свойством. И если в результате вычислений получается не представимое в разрядной сетке число, то его приходится округлять в ту или иную сторону. Соответственно на результат начинает влиять выбранный в компьютере способ округления, и при определенных обстоятельствах ошибки округления могут накапливаться, существенно корректируя окончательный результат.

1. Преобразование информации и представление данных в компьютерных системах, аудио и видеоданные, надежность кодирования, базовые принципы кодирования (с примерами)

АУДИО- И ВИДЕОДАННЫЕ

Развитие способов кодирования звуковой информации, а также движущихся изображений – **анимации** и видеозаписей – происходило с запаздыванием относительно способов кодирования рассмотренных выше разновидностей информации. Приемлемые способы хранения и воспроизведения с помощью компьютера звуковых и видеозаписей появились только в 1990-х годах. Эти способы работы со звуком и видео получили название **мультимедийных технологий**.

Звук представляет собой довольно сложное, непрерывное колебание воздуха. Основной подход к кодированию звука, который называется преобрзованием в **цифровую форму**, **оцифровыванием** или **дискретизацией**, основан на том, что непрерывный звуковой сигнал заменяется дискретным (то есть состоящим из обособленных, раздельных элементов) набором значений сигнала – **отсчетов сигнала** – в некоторые последовательные моменты времени. Каждый отсчет кодируется в одном из рассмотренных ранее числовых форматов. Таким образом, кодирование и обработка звуковых данных фактически сводятся к кодированию и обработке числовых данных.

В соответствии с этим определением количество отсчетов сигнала в единицу времени называется **частотой дискретизации**. Так, на рис. 2.5 сигнал, длящийся 2 с, заменяется 100 отсчетами (жирные точки на графике амплитуды сигнала). Следовательно, в данном случае частота дискретизации равна 50 Гц. При записи звука в мультимедийных технологиях применяются частоты 8, 11, 22 и 44 кГц. Частота дискретизации 44 кГц означает, что одна секунда непрерывного звучания заменяется набором из сорока четырех тысяч отдельных отсчетов сигнала. Чем выше частота дискретизации, тем лучше качество оцифрованного звука. В последних разработках частота дискретизации достигает 192 кГц.

Каждый отдельный отсчет представляет собой число, которое затем можно представить в виде некоторого двоичного кода. Качество преобразования звука в цифровую форму определяется не только частотой дискретизации, но и количеством битов памяти, отводимых на запись кода одного отсчета. Этот параметр принято называть **разрядностью преобразования**. В настоящее время обычно используются разрядности 8, 16, 24 бит. На описанных принципах основывается формат **WAV** (от WAVeform-audio – волновая форма аудио) кодирования звука.

Кодирование видеоинформации – еще более сложная проблема, чем кодирование звуковой информации, так как нужно позаботиться не только о дискретизации непрерывных движений, но и о синхронизации изображения со звуковым сопровождением. В настоящее время для этого используется формат, который называется **AVI** (Audio-Video Interleaved – чередующиеся аудио и видео). Отметим, что основные мультимедийные форматы AVI и WAV очень требовательны к памяти. Поэтому на практике применяются различные способы компрессии (сжатия) звуковых и видеокодов.

ПРИНЦИП ОБЕЗЛИЧИВАНИЯ КОДА

Для компьютера не существует разницы между кодами данных различной природы. Интерпретация (истолкование смысла) машинного кода может быть самой разной. Один и тот же код разыми программами может рассматриваться и как число, и как текст, и как изображение, и как звук. Другими словами, как именно трактуется, понимается тот или иной машинный код, определяется обрабатывающей этот код программой.

Принцип обезличивания машинного кода состоит в том, что один и тот же код может восприниматься процессором компьютера в любом из используемых в аппаратных и программных средствах способов кодирования. Конкретная трактовка кода определяется обращающейся к коду программой.

НАДЕЖНОСТЬ КОДИРОВАНИЯ

Одним из важнейших аспектов организации храения и передачи кодов данных является обеспечение их надежности и безошибочности. Существует множество различных факторов, воздействие которых на аппаратные средства компьютера приводит к искажению кодов данных. К ним относятся, например, влияние сильных электромагнитных полей от источников, находящихся вблизи запоминающих, передающих устройств и линий связи, влияние температурного расширения материалов, из которых состоят запоминающие устройства, сбои в работе аппаратуры, механические удар и т.д. Для защиты данных от большинства таких факторов к настоящему времени уже разработан и внедрены достаточно надежные средства.

Вместе с тем, применяя специальные методы кодирования, можно обеспечить контроль за появлением ошибки и даже восстановление исходного кода после ее обнаружения. Применение этих методов позволяет снизить вероятность ошибки до 10-9 и ниже.

Общим методом обеспечения надежности храения и передчи данных в компьютере и по линиям связи является включение в код дополнительных контрольных разрядов (битов). Существует много различных вариантов этого метода.

Во многих компьютерах одним из уровней контроля за появлением ошибок является включение в байт дополнительного, контрольного девятого разряда, который формируется аппаратурой автоматически. А те восемь битов, из которых, как мы до сих пор считали, состоит байт, в дальнейшем будем называть **информационными**. Значение контрольного разряда определяется так, чтобы общее количество разрядов байта (всех информационных и контрольного), которые содержат 1, было нечетным. Рассмотрим, например, байт с информационными битами 1001 11102. Количество единиц в нем нечетно, следовательно, аппаратура сформирует значение контрольного бита, равное 0. При этом общее количество единиц в девяти битах байта останется нечетным. А, например, для байта с информационными битами 1001 01102­, количество единиц в котором четное, контрольный бит окажется равным 1 – в результате образуется нужное нечетное количество единиц.

Кроме введения контрольного разряда существуют и более развитые способы кодирования, которые не только позволяют установить факт появления ошибок большой кратности, но и обеспечивают их исправление. К таким способам относятся коды, принцип построения которых в 1948 г. предложил Р. Хемминг.

Теперь рассмотрим предложенный Р. Хеммингом общий порядок построения кодов, обеспечивающий однозначное обнаружение и исправление ошибок в кодах данных. Он включает в себя следующие действия.

1. Контрольные биты вставляются в исходный код и нумеруются совместно с информационными битами слева направо начиная с 1.

2. Контрольные биты располагаются в позициях с номерами n = 2k, k = 0, 1, 2, 3… объединенного кода, то есть в позициях с номерами n = 1, 2, 4, 8, 16 …

3. Для каждого контрольного разряда с номером n весь код делится на группы, состоящие из 2n битов.

4. Контрольный бит с номером n контролирует в группе первые n подряд расположенных битов кода (для первой группы, включая контрольный) с пропуском следующих n битов.

5. В каждой группе контрольный бит формируется так, чтобы общее количество единиц в ней было нечетным.

Для определения ошибочного бита важно знать не какие информационные биты относятся к данному контрольному, а какие контрольные биты закреплены за данным информационным. Анализ показывает, что в общем случае информационный бит с номером b проверяется контрольными битами с номерами b1, b2 … bk, такими, что b = b1 + b2 + … + bk. Это значит, что номер ошибочного бита равен сумме номеров контрольных битов, обнаруживших ошибку.

Методы построения кодов Хемминга и некоторых других, более мощных способов кодирования, используются в различных устройствах памяти компьютера. В частности, на этих методах основан аппаратных механизм контроля оперативной памяти ECC (Error Checking and Correcting – контроль и исправление ошибок), который является дополнительным средством обеспечения высокой ее помехоустойчивости.

1. Понятие такта данных, базовое понятие синхронизации (с примерами)

ПОНЯТИЕ ТАКТА

В выполнении действий над данными участвуют несколько устройств. Эти действия должны быть согласованы и синхронизированы друг с другом. Для синхронизации работы компьютера используется специальное устройство – тактовый генератор, который через равные промежутки времени вырабатывает импульсы синхронизации – синхроимпульсы.

Промежуток времени от начала одного импульса синхронизации до начала следующего за ним импульса называется тактом. Такты обладают равными длительностями с высокой степенью точности.

Выполнение процессором любых действий всегда происходит во время некоторой части такта, а за ней следует пауза, в течение которой ничего не происходит. Наличие такой паузы является принципиальным фактором, обусловленным физическими законами, которые управляют работой процессора.

Рассматривая выполнение процессором некоторой последовательности действий, можно считать, что любые действия, происходящие внутри такта, происходят мгновенно, в моменты времени t0, t1, t2, …, соответствующие границам тактов.

С понятием такта связана тактовая частота – одна из важнейших технических характеристик различных устройств компьютера.

Тактовая частота представляет собой техническую характеристику отдельных устройств компьютера, которая равна количеству тактов, управляющих работой устройства, в единицу времени. Единицей измерения тактовой частоты является герц, равный одному такту в секунду.

Тактовая частота является одним из главных факторов, определяющих скорость обработки данных компьютером. Первые персональные компьютеры работали на частотах 5 – 10 МГц. В настоящее время процессоры компьютеров работают с тактовыми частотами от сотен мегагерц до 3 – 4 ГГц. Возможно, в недалеком будущем появятся процессоры с тактовой частотой порядка 10 ГГц. Однако следует заметить, что, согласно теоретическим оценкам, процессоры, выполненные по современным технологиям, не смогут превзойти частот 30 – 40 ГГц.

Совокупность, состоящая из двоичного кода выполняемой программы, значений обрабатываемых дискретных данных, а также набора дискретных значений и признаков, которые используются для управления процессом обработки, образует внутреннее состояние компьютера.

С точки зрения введенного понятия внутреннего состояния компьютера, можно считать, что в моменты времени t0, t1, t2… происходит мгновенный переход компьютера из одного внутреннего состояния в другое – из состояния, соответствующего значениям обрабатываемых данных и признаков до выполнения действия, в состояние, соответствующее значениям данных и признаков после выполнения этого действия. Таким образом, множество внутренних состояний компьютера также является дискретным. В связи с этим говорят, что компьютер в целом является дискретным устройством.

Устройства, обеспечивающие работу с дискретными данными или сигналами, обладающие дискретным множеством внутренних состояний, а также выполняющие действия в дискретные моменты времени, называются дискретными. Компьютер в целом относится к группе дискретных устройств.

1. Базовые логические элементы, построение схем на базовых логических элементах, комбинаторика (с примерами)

ВЕНТИЛИ И КОМБИНАЦИОННЫЕ СХЕМЫ

Основными базовыми операциями, которые должен выполнять процессор компьютера, являются логические операции И (отрицание), ИЛИ (дизъюнкция), ИЛИ (конъюнкция), арифметическое сложение, сдвиг. Используемые для реализаций этих и других операций устройства принято называть вентилями (от немецкого Ventil – клапан)

Вентилем называется физическое устройство, реализующее одну из базовых логических операций: отрицание, дизъюнкцию, конъюнкцию, исключающую дизъюнкцию и т.д. Вентили, входящие в состав процессоров компьютера, называют также логическими элементами.

ОСНОВНЫЕ ЛОГИЧЕСКИЕ ЭЛЕМЕНТЫ



РЕАЛИЗАЦИЯ ВЕНТИЛЕЙ

В 1938 г. известный специалист в области теории информации Клод Шеннон предложил использовать для моделирования основных логических операций релейно-контактные электрические схемы. Этот подход был использован в электромеханических релейных вычислительных машинах Z-3 (Германия, Конрад Цузе, 1939), «Марк 2» (США, Говард Айкен, 1947), РВМ‑1 (СССР, Н.И. Бессонов, 1951) и в целом ряде других машин.

Знаки двоичного алфавита 0 и 1 моделируются отсутствием или наличием тока в цепи соответственно. Заметим: переключатели называются релейными потому, что управление ими обычно осуществляется с помощью электромагнитных реле.

Уже в 1950-е годы релейные вычислительные машины были вытеснены сначала ламповыми, а затем транзисторными компьютерами.

Все современные вентили реализуются на основе полупроводниковых устройств – **транзисторов** или их аналогов в интегральных схемах, характерное время срабатывания которых в настоящее время приближается к долям наносекунды (1 нс = 10-9 с).

В вентильных схемах на базе транзисторов двоичному знаку 0 соответствует низкое напряжение с уровнем от 0 до 1 В, а двоичному знаку 1 – высокое напряжение с уровнем от 2 до 5 В. Могут применяться и другие конкретные значения напряжений, однако в любом случае используется два четко различимых его уровня.

Вентили «НЕ», «НЕ И» и «НЕ ИЛИ», используемые для построения других вентилей и произвольных схем, считаются базовыми, а схемы, которые получаются с помощью всевозможных комбинаций базовых вентилей, принято называть **цифровыми логическими схемами**. Важным частным случаем цифровых схем являются **комбинационные схемы**, в которых значения, получаемые на выходах схемы, зависят только от значений, поступающих на ее входы. Такие схемы классифицируются также как **схемы без памяти**.

Для построения комбинационной схемы необходимо представить логическую функцию в виде логической формулы, провести минимизацию и получить правило соединения логических элементов. Эти вопросы были рассмотрены в других курсах.

Существуют типовые комбинационные схемы, например, компаратор, дешифратор, шифратор, мультиплексор, демультиплексор и т.д.

АРИФМЕТИЧЕСКИ-ЛОГИЧЕСКОЕ УСТРОЙСТВО

Все функции процессора, связанные с выполнением тех или иных действий над данными, сосредоточены в его внутреннем блоке, который принято называть **арифметико-логическим устройством** (АЛУ).

Часть процессора, выполняющая арифметические, логические и иные операции над данными, называется арифметико-логическим устройством.

СХЕМА ПАМЯТИ НА БАЗОВЫХ ВЕНТИЛЯХ

Схемы, состоящие из базовых вентилей, применяются не только для создания устройств, выполняющих действия над данными. Они используются также и для реализации одной из разновидностей памяти в компьютере. Но схемы памяти не могут быть отнесены к группе комбинационных схем, так как получаемый на их выходах результат зависит не только от поступивших на вход данных, но и от текущего состояния схемы. Собственно говоря, эта зависимость и обеспечивает принципиальную возможность запоминания данных. Схемы, обладающие такими свойствами, относятся к группе последовательных схем. Кроме того, для них используется название **«автоматы с памятью».**

Устройство на базовых вентилях, которое можно использовать для хранения одного бита данных, называется **триггером**, или **SR-защелкой**.

Триггером (от trigger – защелка, спусковой крючок) называется устройство, которое может сколь угодно долго находиться в одном из двух состояний устойчивого равновесия и переключается из одного состояния в другое скачком по сигналу или воздействию извне. Триггер может быть реализован, в частности, на электровакуумных лампах накаливания и на полупроводниковых устройствах.

Входы функционируют согласно названиям: S – set, R – reset. Устойчивыми при S = 0 и R = 0 являются два состояния, когда на входах/выходах V и находятся различные значения, и неустойчивым – одно состояние, когда на них находятся одинаковые значения. Значение, находящееся на основном выходе V любом из устойчивых состояний, считается значением, хранящимся в триггера.

При любом исходном состоянии триггера появление 1 на входе S (при сохранении R = 0) переводит его в состояние 1. Можно считать, что эта ситуация эквивалентна записи значения 1 в триггер.

При любом исходном состоянии триггера появление 1 на входе R (при сохранении S = 0) переводит триггер в состояние 0. Можно считать, что эта ситуация эквивалентна записи 0 в триггер. Итак, триггер запоминает, на каком его входе S или R в последний раз было единичное значение. На основе этого свойства конструируются некоторые реальные схемы памяти компьютера.

ИНТЕГРАЛЬНЫЕ СХЕМЫ

Интегральная схема (ИС), или микросхема, представляет собой микроминиатюрную электрическую цепь, которая содержит некоторое количество элементов, эквивалентных транзисторам, резисторам и т.д. Вся схема с помощью специальной технологии размещается на очень маленькой кремниевой или какой-либо другой подходящей по свойствам пластинке, точнее на поверхности или же внутри полупроводникового кристалла.

Микросхема обычно помещается в прямоугольный пластиковый, керамический или иной корпус, снабженный контактными ножками, которые служат входами и выходами схемы. Длина корпусов микросхем колеблется от 0,5 до 5 см.

Интегральные схемы характеризуются **степенью интеграции**. Она численно равна количеству транзисторов (иногда вентилей), из которых состоит схема. Степень интеграции используется для деления микросхем на четыре группы:

1. Малые интегральные схемы (МИС) – до 102 транзисторов;

2. Средние интегральные схемы (СИС) – от 102 до 103 транзисторов;

3. Большие интегральные схемы (БИС) – от 103 до 105 транзисторов;

4. Сверхбольшие интегральные схемы (СБИС) – свыше 105 транзисторов.

Кроме того, часто используемыми параметрами микросхем являются линейны размер вентиля и уже упоминавшаяся ранее площадь микросхемы (не следует путать с площадью корпуса микросхемы). Современные вентили имеют линейные размеры, измеряемые микрометрами (1 мкм = 10-6 м) и нанометрами (1 нм = 10-9 м).

Интегральные схемы обычно специально создаются для решения конкретных задач, например, таких, как хранение или обработка данных. Название микросхемы включает ее специализацию: микросхема памяти или микросхема процессора. Микросхемы процессоров обычно называют еще более кратко – **микропроцессор**.

**Микропроцессором** называется интегральная схема, обеспечивающая выполнение всех функций процессора компьютера.

1. Понятие многоуровневой компьютерной организации (примеры организации уровней)

Цифровой компьютер – это машина, которая может решать задачи, выполняя поданные ей команды. Последовательность команд, описывающих решение определенной задачи, называется программой. Электронные схемы каждого компьютера могут распознавать и выполнять ограниченный набор простых команд. Поэтому все программы перед выполнением должны быть превращены в последовательность таких команд, которые обычно не сложнее чем

1. Сложить два числа;

2. Проверить, не является ли число нулем;

3. Скопировать кусок данных из одной части памяти в другую.

Это примитивные команды и являются тем языком, которым человек общается с компьютером. Такой язык называется машинным. Обычно разработчик старается сделать машинные команды как можно проще, что бы избежать дополнительных сложностей при конструировании компьютера, снизить затраты на электронику. Поэтому большинство машинных языков очень примитивны и их использование трудно и утомительно.

Это привело к тому, что с течением времени появились ряд уровней абстракций, каждая из которых настраивается над абстракцией более низкого уровня. Такой поход называется многоуровневой компьютерной организацией. Такой подход в рассмотрении архитектуры компьютера принят в классической книге «Архитектура компьютера» Эндрю Таненбаума, ведущего специалиста в области разработки компьютеров из МТИ.

СОВРЕМЕННЫЕ МНОГОУРОВНЕВЫЕ МАШИНЫ

Существует разница между тем, что удобно людям и тем, что удобно для компьютера (или что удобно программисту и что удобно разработчику электроники). В литературе это получило название семантического разрыва. Семантический разрыв определяет различие принципов, лежащих в основе языков программирования высокого уровня и тех принципов, которые определяют архитектуру ЭВМ. В этом курсе мы рассмотрим, каким образом можно решить эти проблемы.

ЯЗЫКИ, УРОВНИ И ВИРТУАЛЬНЫЕ МАШИНЫ

Обозначим через Я1 – язык программирования, удобный для человека. Машинные команды также образуют некоторый язык, понятный компьютеру, который обозначим как Я0. Для того, чтобы перевести программы с языка Я1 на понятный компьютеру язык Я0 можно использовать следующие подходы.

Первый способ – замена каждой команды, написанной на языке Я1 эквивалентным набором команд, написанных на языке Я0. В этом случае компьютер выполняет новую программу, написанную на языке Я0 вместо старой, записанной на языке Я1. Такая технология называется трансляцией.

Второй способ – написание программы на языке Я0, которая берет команды, написанные на языке Я1 в качестве исходных данных, рассматривает каждую команду по очереди и сразу выполняет эквивалентный набор команд языка Я0. Эта технология не требует составления новой программы на языке Я0, Эта технология называется интерпретацией, а программа, осуществляющая интерпретацию называется интерпретатором.

Различия между трансляцией и интерпретацией заключается в том, что при трансляции все программа Я1 переделывается в программу Я0 и программа Я1 может быть отброшена, а в память компьютера загружается программа Я0. При интерпретации каждая команда перекодируется и тут же выполняется.

На практике проще представить себе существование некоторого гипотетического компьютера, или виртуальной машины, для которой машинным языком является удобный для нас язык Я1, чем думать о трансляции и интерпретации. Назовем такую машину М1, а машину, понимающую язык Я0, машиной МО.

Хотелось бы сконструировать машину, для которой машинным языком был бы язык Я1, однако эта машина была бы слишком дорогой.

Для целесообразности трансляции и интерпретации необходимо, чтобы языки Я0 и Я1 не очень отличались друг от друга. Казалось бы, это требование находится в некотором противоречии с целью создания языка Я1, удобного для человека. Однако это противоречие разрешается следующим очевидным образом.

Создается следующий набор команд Я2, который в большей степени ориентирован на человека. Этому языку соответствует виртуальная машина М2, соответствующие трансляторы или интерпретаторы Я2-Я1 и т.д. Изобретение ряда языков, каждый из которых более удобен для человека, чем предыдущий может продолжаться до тех пор, пока мы не дойдем до подходящего. Каждый последующий язык рассматривает предыдущий как основу и поэтому можно представить компьютер в виде уровней.

Человеку, пишущему программы на языке Яn не обязательно знать об интерпретаторах и трансляторах более низких уровней. Большинство пользователей, использующих машину уровня n интересуются только этим уровнем. Однако для изучения работы компьютера необходимо иметь представление обо всех уровнях.

Деление на уровни является в некотором роде условным. Можно говорить о шести уровнях представления современного компьютера.

На рисунке 4.2 отсутствует уровень физических устройств, расположенный ниже уровня 0. На этом уровне находятся электронные схемы, которые не являются предметом рассмотрения курса.

На самом нижнем уровне – цифровом логическом – объекты состоят из вентилей и хотя вентили являются электронными схемами, их работа может быть описана как цифровые средства. Вентили образуют биты памяти, которые в свою очередь, образуют регистры.

Следующий уровень – микроархитектурный. На этом уровне рассматриваются совокупности регистров, которые образуют локальную (или регистровую) память и арифметико-логическое устройство (АЛУ), предназначенное для выполнения простых операций. Регистры вместе с АЛУ образуют тракт данных. Основная операция состоит в следующем: выбирается один или два регистра, АЛУ производит над ними некоторую операцию (например, сложение), результат помещается в один из регистров. В некоторых машинах тракт данных контролируется микропрограммой. На других контроль осуществляется аппаратными средствами.

Второй уровень – уровень архитектуры системы команд. Руководства по машинному языку, выпускаемые фирмами-производителями содержат информацию именно об этом уровне. Когда там описывается система команд, то описываются команды, выполняемые программой-интерпретатором или аппаратными средствами.

Второй уровень – уровень архитектуры системы команд. Руководства по машинному языку, выпускаемые фирмами-производителями содержат информацию именно об этом уровне. Когда там описывается система команд, то описываются команды, выполняемые программой-интерпретатором или аппаратными средствами.

Следующий третий уровень – уровень операционной системы – обычно гибридный. Большинство команд этого уровня имеется также и на уровне архитектуры системы команд (команды, имеющиеся на одном уровне, также могут иметься и на других). Особенности уровня: новые команды, иная организация памяти, способность выполнять две и более программ одновременно и некоторые другие. Команды третьего уровня, идентичные командам второго, выполняются микропрограммой или аппаратными средствами, а не операционной системой. Часть команд (не имеющихся на втором уровне), выполняются операционной системой.

Нижние уровни конструируются не для того, что бы с ними работал обычный программист. Они предназначены для работы интерпретаторов и трансляторов и поддерживаются системными программистами. Уровни с четвертого и выше предназначены для работы прикладных программистов. Следующая особенность: уровни 2 и 3 обычно интерпретируются, а уровни 4 и выше как правило, поддерживаются транслятором. Уровни 1, 2 и 3 – цифровые, т.е. программы, написанные на языках этих уровней, состоят из наборов цифр. Начиная с четвертого уровня, языки содержат слова и сокращения, понятные человеку.

Четвертый уровень представляет собой символьную запись языка более низкого уровня. Программа, которая выполняет трансляцию называется ассемблером.

Пятый уровень обычно состоит из языков, разработанных для прикладных программистов. Такие языки называются языками высокого уровня.

Выводы: компьютер проектируется как иерархическая структура уровней, каждый из которых надстраива6ется над предыдущим. Каждый уровень представляет собой абстракцию с различными объектами и операциями.

Набор типов данных, операций и особенностей каждого уровня называется архитектурой. Архитектура связана с аспектами, которые видны программисту. Аспекты разработки, технологии и т.д. не являются частью архитектуры. Термины компьютерная архитектура и компьютерная организация в сущности означают одно и то же.

РАЗВИТИЕ МНОГОУРОВНЕВЫХ МАШИН

*Аппаратное обеспечение* состоит из электронных схем, памяти, устройств ввода-вывода, т.е. из осязаемых объектов.

*Программное обеспечение* состоит из алгоритмов и программ. В первых вычислительных машинах разница между программными и аппаратными средствами была очевидной. Со временем эта грань стала размываться. Сейчас можно говорить о том, что аппаратное и программное обеспечения логически эквивалентны. Карен Панетта Ленц говорил: «Аппаратное обеспечение – это всего лишь окаменевшее программное обеспечение». Разделение функций программного и аппаратного обеспечения определяется такими факторами как стоимость, скорость, надежность, а также частота ожидаемых изменений.

ИЗОБРЕТЕНИЕ МИКРОПРОГРАММИРОВАНИЯ

У первых компьютеров было только два уровня уровень архитектуры набора команд и цифровой логический уровень.

В 1951 году Морис Уилкс (Кембриджский университет) предложил идею трехуровневого компьютера. Такой компьютер должен иметь встроенный неизменяемый интерпретатор (микропрограмму), функция которого заключалась в выполнении программ посредством интерпретатора. Таким образом, аппаратное обеспечение должно было выполнять только микропрограммы с ограниченным набором команд. Электронные схемы существенно упростились, цена уменьшилась, а надежность возросла. К 70-м годам идея микропрограммирования стала преобладающей.

ИЗОБРЕТЕНИЕ ОПЕРАЦИОННОЙ СИСТЕМЫ

Первые операционные системы появились в 60-е годы. Придумана она была для того, чтобы автоматизировать работу оператора. Однако создание операционной системы было первым шагом на пути в развитии новой виртуальной машины. К уровню архитектуры команд добавлялись новые команды и в итоге сформировался новый уровень. Некоторые команды нового уровня были идентичны командам предыдущего, но появились и другие команды, которые полностью отличались. Эти команды тогда назывались *макросами ОС* или *вызовами супервизора*. Сейчас используют термин системный вызов. Первые операционные системы были ориентированы на работу в *пакетном режиме*. В начале 60-х годов в МТИ разработали операционную систему, которая позволяла одновременно работать нескольким пользователям. В такой системе ресурсы центрального процессора разделялись между несколькими пользователями. Такие системы назывались и сейчас называются системами с *разделением времени*.

ПЕРЕМЕЩЕНИЕ ФУНКЦИОНАЛЬНОСТИ НА УРОВЕНЬ МИКРОКОМАНД

С 1970 г. микропрограммирование стало обычным, производители вводили новые команды путем расширения микропрограммы, т.е. программными методами. Многие новые команды не представляли особой ценности, т.к. эти же действия можно было сделать уже имеющимися средствами. Однако новые команды могли выполнять операции быстрее или предоставляли дополнительные удобства пользователю. Например:

1. Ускорение работы с массивами (индексная и косвенная адресация)

2. Системы прерывания, которые дают команду процессору, как только закончилась операция ввода или вывода;

3. Способность приостановить одну программу и начать другую, используя небольшое количество команд (переключение процесса);

4. Специальные команды для обработки изображений, мультимедийных данных.

УСТРАНЕНИЕ МИКРОПРОГРАММИРОВАНИЯ

В 60-х – 70-х годах количество микропрограмм увеличивалось, но они работали все медленнее и медленнее, т.к. требовали значительного объема памяти. Пришло понимание того, что с устранением микропрограмм резко сократится количество команд и компьютеры станут работать быстрее. Таким образом, компьютеры вернулись к тому состоянию, в котором они были до изобретения микропрограммирования.

Вывод: граница между аппаратным и программным обеспечением постоянно перемещается. Так же обстоит дело с уровнями – между ними нет четких границ.

РАЗВИТИЕ КОМПЬЮТЕРНОЙ АРХИТЕКТУРЫ

НУЛЕВОЕ ПОКАЛЕНИЕ – МЕХАНИЧЕСКИЕ КОМПЬЮТЕРЫ

Первую счетную машину создал французский ученый Блез Паскаль в 1642 году. Ему тогда было 19 лет, он создал машину для своего отца, сборщика налогов. Машина могла выполнять только операции сложения и вычитания.

Тридцать лет спустя великий немецкий математик Готфрид Вильгельм Лейбниц (1646-1716) построил счетную машину, которая помимо операций сложения и вычитания могла выполнять операции умножения и деления.

Еще через 150 лет профессор математики Кембриджского университета Чарльз Бэбидж (1792-1871) (изобретатель спидометра) разработал и сконструировал разностную машину, предназначенную для подсчета таблиц чисел морской навигации. Машина могла выполнять только один алгоритм – метод конечных разностей с использованием полиномов. Машина, выполнявшая только один алгоритм, Бэбиджу вскоре наскучила и он начал разрабатывать (и потратил на это очень много средств) аналитическую машину. У аналитической машины было запоминающее устройство, вычислительное устройство, устройство ввода (для перфокарт), устройство вывода (перфоратор и печатающее устройство).

Машина могла выполнять разные задачи. Она считывала команды с перфокарт и выполняла их. Поскольку аналитическая машина программировалась на ассемблере, то ей было необходимо программное обеспечение. Первый программист – племянница поэта Байрона Ада Ловлейс. В ее честь назван язык программирования АДА. Аналитическая машина была механической. Идеи Бэбиджа опередили эпоху в том смысле, что технологическая база не позволяла создавать устройства такой сложности с приемлемой надежностью.

В конце 30-х -- начале 40-х годов ХХ века счетные машины были сконструированы в Германии и Америке, в которых были использованы электромагнитные реле. Машина Джона Атанасова была очень развита для своего времени. В ней использовалась бинарная арифметика, информационные емкости, которые периодически обновлялись. К сожалению, эта машина так и не заработала.

Говард Айкен, опираясь на исследования Бэбиджа, решил создать такой же компьютер, но на основе реле. Работа над первым компьютером была закончена в 1944 году. Называлась машина «MARK 1». Затем началась эра электроники.

ПЕРВОЕ ПОКОЛЕНИЕ – ЭЛЕКТРОННЫЕ ЛАМПЫ

Стимулом для разработки электронного компьютера стала Вторая мировая война. Машина создавалась для шифровки и дешифровки в Великобритании. Одним из создателей этой машины был Алан Тьюринг.

В Америке Джон Моушли со своим студентом Дж. Преспером Экертом начали конструировать компьютер, предназначенный в первую очередь для составления таблиц для нацеливания тяжелой артиллерии. К моменту завершения разработки война закончилась, машина стала не нужна для военных целей, и разработчикам было разрешено организовать школу, где они рассказывали о своей работе. Эта машина – ENIAC. Патент на ЦВМ они не получили, т.к. приоритет был отдан Атанасову.

В это же время в Институт специальных исследований в Принстоне приехал один из участников проекта ENIAC Джон фон Нейман, чтобы сконструировать свою версию компьютера.

Он предложил размещать программу вместе с данными в оперативной памяти и использовать бинарную арифметику. Основной проект известен теперь как фон-неймановская вычислительная машина. Он был использован в машине EDSAC. Практически все современные компьютеры являются фон-неймановскими машинами. Схема архитектуры этой машины приведена на рисунке В3. Машина не имела операций с плавающей точкой. Нейман полагал, что любой сведущий математик способен держать плавающую точку в голове.

Приблизительно в это же время в МТИ был создан компьютер Whirlwind-1. Особен­нос­ти компьютера: слова неболь­шой длины (16 бит) и работа в РМВ. Он является прототипом мини-компьютера.

В 1953 году IBM создала свой первый компьютер IBM-704.

ВТОРОЕ ПОКОЛЕНИЕ – ТРАНЗИСТОРЫ

Транзисторы были изобретены в лаборатории Bell Джоном Бардином, Уолтером Браттейном и Уильямом Шокли, за что в 1956 году им была присуждена Нобелевская премия. Первый компьютер на транзисторах был построен в МТИ и назывался ТХ-1, а затем ТХ-2. Практического значения эти компьютеры не имели, но один из разработчиков, Кеннет Ольсен в 1957 году основал фирму DEC и произвели первую серийную машину на транзисторах PDP-1 (1961 г.). Эта была самая быстродействующая машина того времени. Время цикла – 5 микросекунд. Это в два раза меньше, чем у IBM-7090 (транзисторного аналога IBM-709). Стоил PDP-1 $120 000, а IBM – миллионы. Компания DEC продала десятки компьютеров PDP и так возникла компьютерная промышленность. Один из компьютеров был отдан в МТИ, где был создан первый графический дисплей, а студенты написали первую компьютерную игру – «Война миров».

Затем была создана машина PDP-8, которая была 12-разрядной, стоила $16 000, а главное нововведение – одна шина. Шина – это набор параллельно соединенных проводов для связи компонентов компьютера. Структура компьютера с общей шиной приведена на рис. В4. Такая структура с тех пор используется во всех компьютерах.

В 964 году компания CDC выпустила машину CDC-6600, которая имела производительность на порядок выше, чем IBM-7090 и ее более дешевый аналог IBM-1401. Высокая производительность обеспечивалась за счет того, что внутри центрального процессора находилась машина с высокой степенью параллелизма. Разработчиком этого компьютера был Сеймур Крей. Он посвятил свою жизнь созданию мощных компьютеров, которые сейчас называются суперкомпьютерами. Это компьютеры CDC-6600, CDC-7600, Crey-1.

Разработчики упомянутых выше компьютеров занимались в первую очередь аппаратным обеспечением, стремясь повысить его надежность, быстродействие и снизить стоимость.

Следует отметить еще один проект – Burroughs B50000. Разработчики создавали компьютер с намерением программировать ее на языке Algol 60 (предшественник языка Pascal), сконструировав аппаратное обеспечение так, что бы упростить работу компилятору. Так появилась идея, что программное обеспечение тоже надо учитывать при разработке компьютера.

ТРЕТЬЕ ПОКОЛЕНИЕ – ИНТЕГРАЛЬНЫЕ СХЕМЫ

В 1958 году была изобретена кремниевая технология (изобретатель – Роберт Нойс). Компьютеры на интегральных схемах были меньшего размера, работали быстрее, стоили дешевле. Наиболее значительные следующие.

К 1964 г. Фирма IBM лидировала на рынке, но выпускаемые ей компьютеры были программно несовместимы. Компания сделала решительный шаг. Она выпустила серию компьютеров на транзисторах System 360, которые были предназначены как для научных, так и для коммерческих расчетов. System 360 содержала много нововведений. Это было семейство компьютеров с одним и тем де ассемблером. Каждая новая модель была больше и мощнее предыдущей. Идея создания семейств компьютеров вскоре стала популярной и в течении нескольких лет большинство компьютерных компаний выпустило целые серии сходных машин.

Еще одно нововведение – мультипрограммирование. В памяти располагалось несколько программ и пока одна программа ждала окончания ввода-вывода, другая выполнялась.

Мир микрокомпьютеров сделал также большой шаг вперед вместе с производством компьютеров PDP-11. Во многих отношениях PDP-11 была младшим братом IBM 360 по организации компьютера и наличию в семействе машин разной стоимости и производительности.

ЧЕТВЕРТОЕ ПОКОЛЕНИЕ – СВЕРХБОЛЬШИЕ ИНТЕГРАЛЬНЫЕ СХЕМЫ

Появление СБИС в 80-х годах позволило помещать на одну плату сначала десятки тысяч, а затем и миллионы транзисторов. К 80-м годам цены на компьютеры упали на столько, что приобретать компьютеры смогли не только организации, но и отдельные люди. Началась эра персональных компьютеров. Первые персональные компьютеры продавались в виде комплектов, как правило, на базе Intel 8080. Программное обеспечение пользователь писал сам. Затем появилась операционная система CP/M. Эта ОС помещалась на дискету, включала систему управления файлами и интерпретатор для выполнения пользовательских команд, которые набирались на клавиатуре.

Компания IBM, лидирующая в то время на рынке компьютеров, тоже решила заняться производством персоналок. Для ускорения процесса разработки копания IBM предоставила одному своему сотруднику, Филипу Эстриджу крупную сумму денег на создание персонального компьютера. Компьютер IBM PC появился в 1981 году и стал самым покупаемым в истории.

Но IBM вместо того, чтобы держать проект в секрете или защитить его патентами, опубликовала полные проекты, включая электронные схемы. Многие компания тут же начали делать клоны, которые продавали дешевле. Из других компаний, производивших персональные компьютеры на базе своих процессоров, выжить удалось только некоторым, и то только потому, что они работали в узких областях.

Первая версия IBM PC была оснащена операционной системой MS-DOS, которую выпускала крошечная компания Microsoft. Эта компания разработала также собственную ОС Windows, которая работала на базе MS-DOS. Успех процессора 8088 воодушевил Intel на усовершенствования. Модель 386 – первый представитель линейки Pentium.

В середине 80-х годов на CISC-архитектурой на RISC-компьютеры, которые проще и работают быстрее. В 90-х годах появились суперскалярные компьютеры. Первый 64-разрядный компьютер был выпущен в 1992 году (Alpha, DEC), но коммерческий успех был скромным – 64-разрядные компьютеры приобрели популярность только в спустя десятилетие и в качестве персональных серверов.

ПЯТОЕ ПОКОЛЕНИЕ – НЕВИДИМЫЕ КОМПЬЮТЕРЫ

В 1981 году правительство Японии объявило о намерении выделить национальным компаниям 500 миллионов долларов на разработку компьютеров пятого поколения на основе технологий искусственного интеллекта. Однако проект, в целом, оказался несостоятельным. Причина, скорее всего, заключается в том, что идея несколько опередила технологию.

Пятое поколение компьютеров материализовалось в виде малых по размерам компьютеров – карманных компьютеров и «невидимых компьютеров» – компьютеров, встраиваемых в бытовую технику, банковские карточки и т.п. Процессоры этого типа предусматривают большие функциональные возможности, широкий спектр применения за умеренную цену. Компьютеры пятого поколения ассоциируются не с некоторой архитектурой, а парадигмой использования. В настоящее время этому явлению применяется термин «всепроникающая компьютеризация».

1. Типы компьютеров и компьютерных систем (с примерами)

ТИПЫ КОМПЬЮТЕРОВ

ТЕХНОЛОГИЧЕСКИЕ АСПЕКТЫ

Закон технологического прогресса, известный как закон Мура (Гордон Мур – один из основателей Intel) утверждает, что количество транзисторов на одной микросхеме удваивается каждые 18 месяцев. По сути закон Мура не закон, а эмпирическое наблюдение за развитием технологий, и, по мнению специалистов, продержится еще лет 10, может больше. Однако, существует физический предел, определяемый принципом неопределенности Гейзенберга.

Закон Мура связан с так называемым **эффективным циклом**. Т.е. новые возможности порождают новые потребности, для удовлетворения которых требуются новые возможности и т.д.

Еще один фактор развития компьютерных технологий – первый закон программного обеспечения, названный в честь Натана Мирвольда (главный администратор Microsoft). Закон гласит: программное обеспечение это газ, который полностью заполняет резервуар, в котором находится. Т.е. программное обеспечение продолжает развиваться и требовать все больше ресурсов.

Шутка Г. Мура на тему темпов развития IT-технологий: если бы авиационные технологии развивались такими же темпами, как компьютерные, то самолеты стоили бы $500, облетали Землю за 20 минут на 20 литрах керосина и были бы размером с обувную коробку.

ШИРОКИЙ СПЕКТР КОМПЬЮТЕРОВ

Исследователь из лаборатории Bell Ричард Хамминг заметил, что количественное изменение характеристик на порядок ведет к качественному изменению. В компьютерных технологиях количественные характеристики изменились за 30 лет на 6 порядков. Компьютерные технологии развиваются как по пути увеличения мощности так и снижения цены на единицу мощности. Примерная классификация компьютеров приведена в таблице.

**Одноразовые компьютеры**

Наибольшим достижением в этой области можно считать появление микросхем RFID (Radio Frequency Identification – радиочастотная идентификация). Эти микросхемы без батареек, содержат приемо-передатчик, который по внешнему запросу выдает код. Это может быть использовано для снятия штрих-кодов, идентифицировать можно до конкретной единицы продукции, с увеличением объема памяти таких микросхем можно записывать и другие данные.

Микросхемы могут быть активными и пассивными, работать на разных частотах (чем выше частота, тем выше скорость передачи данных, но меньше радиус действия).

**Микроконтроллеры**

Микроконтроллеры выполняют функции управления устройствами и организации пользовательских интерфейсов. В отличие от RFID-микросхем, которые выполняют минимальный набор функций, микроконтроллеры представляют собой полноценные вычислительные устройства. Содержат процессор, память, устройства ввода-вывода. В ряде случаев программное обеспечение прошивается в памяти производителем.

Микроконтроллеры разделяются на универсальные и специализированные.

Особенности микроконтроллеров; низкие цены, работа в реальном масштабе времени, жесткие ограничения на размер и электропотребление.

**Игровые компьютеры**

Это обычные компьютеры, в которых расширенные возможности графических и звуковых контроллеров сочетаются с ограниченным ПО и пониженной расширяемостью. Они оптимизированы на конкретную область применения – выполнение трехмерных игр. Все остальное считается вторичным – отсюда и низкая, по сравнению с ПК, цена.

**Персональные компьютеры**

Некоторые специалисты называют ПК с процессорами Intel, отделяя их от компьютеров, оснащенных высокопроизводительными RISC-микросхемами (Sun UltraSPARC), которые называют рабочими станциями. Но особой разницы между этими компьютерами нет.

К персональным компьютерам близки карманные компьютеры PDA.

**Серверы**

Мощные персональные компьютеры и рабочие станции часто используются в качестве сетевых серверов как в пределах локальной сети, так и в Интернете. Серверы поставляются в однопроцессорной или мультипроцессорной конфигурациях. Некоторые серверы способны обрабатывать до миллиона транзакций в секунду.

С точки зрения архитектуры серверы ничем не отличаются от персональных компьютеров, только работают быстрее, имеют больше дискового пространства, устанавливают более быстрые сетевые соединения. Серверы работают под управлением тех же операционных систем, как правило Windows и UNIX.

**Комплексы рабочих станций**

COW-системы (Clusters of Workstations) – кластеры рабочих станций состоят из нескольких персональных компьютеров или рабочих станций. Эти компьютеры соединены высокоскоростной сетью и снабжены специальным ПО. Предназначены для решения «больших» задач.

В виде кластеров могут быть организованы веб-серверы. Если частота обращения в страницам сайта исчисляется тысячами в секунду, то дешевле организовать кластер из нескольких сотен (или тысяч) серверов и распределить между ними нагрузку. Такие кластеры называются серверными фермами (server farms).

**Мэйнфреймы**

Это большие компьютеры, размером с комнату. В большинстве случаев эти системы – потомки больших компьютеров IBM 360/370. Обычно они работают не намного быстрее, чем мощные серверы, но у них выше скорость процессов ввода/вывода, обладают большим дисковым пространством. Эти системы дорогие. Многие компании считают, что выгоднее вложить деньги в такую систему, чем переписывать все программное обеспечение под персональные компьютеры. Именно из-за этих систем и возникла проблема 2000 года.

В последнее время под влиянием Интернета наблюдается возрождение мэйнфреймов как полноценной категории компьютеров. Они занимают нишу мощных серверов Интернета, способных обрабатывать огромное количество транзакций в секунду.

До последнего времени существовала еще одна категория вычислительных машин – суперкомпьютеры. Это системы с высокопроизводительными процессорами, высокоскоростные диски, сетевые интерфейсы. Сейчас, когда вычислительные возможности, аналогичные тем, которые предлагают суперкомпьютеры реализуются в виде кластеров, эта категория компьютеров постепенно отмирает.

СЕМЕЙСТВА КОМПЬЮТЕРОВ

PENTIUM 4

В 1968 году Роберт Нойс, изобретатель кремниевой интегральной схемы, Гордон Мур и Артур Рок, капиталист из Сан-Франциско, основали корпорацию Intel для производства микросхем. Сначала дела шли не очень хорошо. В 60-х годах калькуляторы были размером с принтер и весили 20 кг.

В 1969 году японская фирма Buscom обратилась к компании Intel с просьбой выпустить 12 несерийных микросхем. Инженер Тэд Хофф решил, что можно поместить 4-х битный универсальный процессор на одну микросхему. Так в 1970 году появился первый процессор на одной микросхеме, процессор 4004. В 1972 году Intel выпустила 8-битный процессор 8008.

Новая микросхема вызвала большой интерес и Intel начала разработку новой микросхемы, у которой предел обращения к памяти в 16 Кбайт был преодолен. Так появился 8080, выпущенный в 1974 году. Этот процессор произвел революцию.

1978 год – процессор 8086. Он имел 16-разрядные внутренние регистры, 16-разрядные внутренние и внешние шины данных и мог адресовать до 1 Мб физической памяти. Быстродействие – 0,8 MIPS (миллион оп/с).

Затем появился 8088, с такой же архитектурой, как у 8086. Он имел шину не 16, а 8 бит, работал медленнее, но был дешевле. Когда фирма IBM выбрала 8088 для IBM PC, эта микросхема стала эталоном в производстве персональных компьютеров.

В начале 80-х годов Intel разработала 80286, совместимый с 8086. Он уже мог адресовать до 16 Мб физической памяти, имел специальный защищенный режим работы, который обеспечивал гибкий механизм адресации, управление доступом к памяти, управление привилегиями и т.п. Система команд также была расширена, а быстродействие достигало 2,7 MIPS.

Intel386 (1985) – это уже полностью 32-разрядный микропроцессор с 32-разрядными внутренними регистрами и шинами данных. Его адресное пространство – 4 Гб. Добавлены страничный механизм и добавлен новый режим работы V86, который обеспечивал совместимость данного микропроцессора с программами, написанными для предыдущих устройств серии. Внутренняя архитектура микропроцессора Intel386 позволила организовать параллельное выполнение нескольких операций (выборка команд, декодирование, исполнение команд) что позволило увеличить быстродействие до 6 MIPS. Приблизительно в это же время была разработана новая операционная система Windows.

В 1989 г. выпущен микропроцессор Intel486, который содержал интегрированное устройство вычислений с плавающей точкой (FPU) и внутреннюю кэш-память для данных и команд.

В 1993 году был создан первый микропроцессор семейства P5 – Pentium, характеризующийся значительно более совершенной архитектурой. В этих процессорах устройство целочисленных вычислений имело два практически идентичных блока, в которых различные команды могли выполняться параллельно. Были доработаны модули, отвечающие за выборку и декодирование команд: была предусмотрена возможность прогнозировать действия процессора, путем осуществления предвыборку и предекодирование команд еще до получения результатов предыдущих команд. Переработано FPU с целью повышения его быстродействия. Были внесены некоторые изменения в архитектуру процессов: появилась поддержка объемных страниц, более гибким стал режим V86, введены дополнительные средства внутренней самодиагностики. Быстродействие таких процессоров могло достигать 100 MIPS.

В 1993-1997 Intel продолжал работу по совершенствованию архитектуры своих процессоров. Был выпущен Pentium Pro – микропроцессор архитектуры P6, которая стала основной для более поздних микропроцессоров, вплоть до Pentium III. В этих процессорах: четыре модуля параллельно осуществляли выполнение команд, предвыборка дополнилась возможностями предсказания ветвлений, кэш-память стала двухуровневой, введена расширенная система обработки мультимедийной информации (MMX).

Pentium II/III – их быстродействие может достигать 1000 MIPS, объем оперативной памяти до 64 Гб, внутренней кэш-памяти – 2 Мб. Система обработки мультимедийной информации была усовершенствована: в микропроцессор Pentium III были введены 70 новых так называемых SIMD-команд.

Помимо основной линейки процессоров Intel разрабатывает специальные микросхемы. Например,

1. 1998 г. – Celeron, уступал по производительности Pentium, но и стоил дешевле.

2. 1988 г., Xeon – имеет кэш-память большего объема, ускоренная внутренняя шина, усовершенствованные средства поддержки мультипроцессорного режима.

3. Pentium 4 (2000 г.) – система основана на новом конструктивном решении, по традиции обошел все предыдущие модели по производительности. В версии с тактовой частотой в 3,06 ГГц была введена многопоточность. Для повышения скорости обработки звуковых и видеоданных был внедрен дополнительный набор SSE-команд.

4. В 2003 г. Появилась микросхема Pentium M – мобильный.

Все микросхемы Intel обратно совместимы. Закон Мура применим и к процессорам. Однако возникает проблема теплоотдачи, что, вероятно, внесет коррективы в этот закон.

В ноябре 2004 г. Intel прекратила выпуск Pentium 4 с тактовой частотой 4 ГГц из-за проблем с теплоотдачей. Проблему повышения быстродействия решают за счет размещения на одной микросхеме двух процессоров и увеличения объема кэш-памяти – память потребляет меньше энергии.

ULTRA SPARC III

В 70-х годах в университетах очень популярной была ОС UNIX, но персональные компьютеры не подходили для этой системы. Энди Бехтольсхайм (аспирант из Стенфорда) разрешил эту проблему, самостоятельно построив рабочую станцию UNIX из стандартных компонентов, имеющихся в продаже и назвал ее SUN (сеть Стенфордского университета).

27-летний индиец Винод Косла предложил Энди Бехтольсхайму организовать компанию по производству рабочих станций SUN. Он нанял также аспиранта Скотта Мак-Нили и программиста Билла Джоя, главного создателя системы UNIX. Они организовали компанию Sun Microsystems. Первый компьютер был оснащен процессором Motorola 68020 и имел большой успех, так же, как и последующие модели. Они были мощнее персональных компьютеров и имели аппаратные и программные средства работы в сети ARPANET (предшественник Internet). В 1987 году компания решила разработать свой собственный процессор, основанном на революционном проекте Калифорнийского университета (RISC II). Этот процессор назывался SPARC (Scalable Processor ARCitecture – наращиваемая архитектура процессора).

В отличие от других компаний, Sun решила не производить процессоры, а предоставила патент нескольким компаниям. Так появились MicroSPARC, HyperSPARC, SuperSPARC, TurboSPARC.

Первый SPARC был 32-разрядным, а в 1995 году была разработана 64-разрядная версия. UltraSPARC с самого начала был предназначен для работы с изображениями, аудио, видео и мультимедиа вообще.

UltraSPARC IV представлят собой двухпроцессорный комплекс, где два процессора UltraSPARC III размещены на одной микросхеме с одним КЭШем.

МИКРОСХЕМА 8051

Микросхема применяется во встроенных системах, ее история началась в 1976 году. В это время уже сформировался спрос на встраиваемые вычислительные устройства, а использовать процессор 8080 с дополнительными модулями памяти оказывалось дорого.

Модель 8748 (Intel) – микроконтроллер на основе 17 000 транзисторов, состоит из процессора типа 8080, постоянной памяти в 1 Кбайт для размещения программ, оперативной памяти 64 байт, 8-разрядный таймер, 27 шин ввода/вывода. Микросхема имела коммерческий успех.

Следующая модель 8051 на 60 000 транзисторах появилась в 1980 г., где процессор был более быстрый, 4К постоянной и 128 байт оперативной памяти, 32 шины ввода/вывода, последовательный порт и два 16-разрядных таймера. Вскоре вышли другие модификации и сформировалось семейство микроконтроллеров MSC-51.

Функциональность микросхем определялась программным обеспечением, которое прошивалось в постоянной памяти. Для создания ПО использовались модели 8751 и 8752, предусматривающие возможность тестирования ПО и перезаписи памяти, т.о. можно было корректировать и отлаживать ПО. Разработанное и отлаженное ПО передавалось производителю, который прошивал память программ уже без возможности перезаписи.

В настоящее время объем продаж микроконтроллеров на несколько порядков выше объема продаж процессоров Pentium, и только в 2001 году 8-разрядные стали продаваться лучше 4-х разрядных. Микросхемы серии 8051 одна из более распространенных.

Популярность объясняется ценой (10-15 центов в партии), производством микросхем по лицензии Intel занимаются множество фирм, поскольку микросхемы выпускаются давно, то для них имеется программное обеспечение и средства его разработки.

*Актуальность архитектурных исследований. На одном из первых компьютеров мира EDSAC (1949 г., Кембридж) время такта 2 микросекунды (2\*10-6), можно было 2\*n операций выполнить за 18\*n миллисекунд, т.е. 100 операций в секунду. Вычислительный узел суперкомпьютера HP V2600 время такта составляет 1.8 наносекунды (1.8\*10-9 секунд), а пиковая производительность около 77 миллиардов арифметических операций в секунду. Таким образом производительность возросла в семьсот миллионов раз, а тактовая частота около 1000 раз.*

1. Понятие микроархитектуры, тракт данных, синхронизация тракта данных
2. Понятие микроархитектуры, конвейерная архитектура, кэш-память, ветвления
3. Модели памяти, регистры
4. Типы данных процессоров (с примерами)
5. Форматы команд
6. Адресация в компьютерных системах (с примерами)
7. Компьютеры параллельного действия, принципы организации, функционирование (примеры)
8. Внутрепроцессорный параллелизм, скалярные и VLIM процессоры
9. Внутрипроцессорная многопоточность
10. Гомогенные и гетерогенные однокристальные мультипроцессоры (с примерами)
11. Семейства сопроцессорных систем, мультимедиа процессоры (с примерами)
12. Сетевые сопроцессоры (с примерами)
13. Криптопроцессоры
14. Мультипроцессоры и мультикомпьютеры, схемы, особенности работы (с примерами)
15. Гибридные системы построения вычислительных систем (с примерами на различных уровнях)
16. Организация виртуальной памяти
17. Политика замещения страниц, фрагментация, сегментация
18. Уровни привилегий в защитах по привилегиям
19. Повышение эффективности оперативной памяти, организация статической и динамической памяти
20. Различные реализации микросхем памяти, примеры многоуровневой организации памяти
21. Организация кэш-памяти
22. Шинная архитектура, циклы синхронных и асинхронных шин, многошинная архитектура (с примерами)
23. Чипсет, параметры шин, шинный арбитрах (с примерами реализации)
24. Повышение эффективности процессора, конвейерные конфликты
25. Комплексные оценки производительности вычислительных систем (тактовой частотой, MIPS, FLOPS, LINPACK и др.) (с примерами применения систем тестирования)
26. Классификация архитектур
27. Архитектуры компьютеров параллельного действия, классификация параллельных компьютерных систем
28. Семейства мультипроцессоров
29. Мультикомпьютеры, коммуникационные взаимодействия (с примерами схем построения)
30. Кластерные вычисления, принципы организации (с примерами)
31. Распределенные вычисления, принципы работы (с примерами)