Министерство образования Республики Беларусь

Учреждение образования
«Гомельский государственный технический
университет имени П. О. Сухого»

Кафедра «Информационные технологии»

А. В. Ковалев, Д. А. Литвинов

ОРГАНИЗАЦИЯ И ФУНКЦИОНИРОВАНИЕ ЭВМ

КУРС ЛЕКЦИЙ
для студентов специальности 1-40 01 02 «Информационные системы и технологии (по направлениям)» заочной формы обучения

Электронный аналог печатного издания

Гомель 2010

УДК 004.3(075.8)
ББК 32.973я73
К56

Рекомендовано к изданию научно-методическим советом факультета автоматизированных и информационных систем ГГТУ им. П. О. Сухого
(*протокол № 7 от 09.03.2009 г.*)

Рецензент: канд. техн. наук, доц. каф. «Промышленная электроника» ГГТУ им. П. О. Сухого *Э. М. Виноградов*

Ковалев, А. В.

К56 Организация и функционирование ЭВМ : курс лекций для студентов специальности 1-40 01 02 «Информационные системы и технологии (по направлениям)» / А. В. Ковалев, Д. А. Литвинов. – Гомель : ГГТУ им. П. О. Сухого, 2010. – 56 с. – Систем. требования: PC не ниже Intel Celeron 300 МГц ; 32 Mb RAM ; свободное место на HDD 16 Mb ; Windows 98 и выше ; Adobe Acrobat Reader. – Режим доступа: http://lib.gstu.local. – Загл. с титул. экрана.

ISBN 978-985-420-932-6.

Изложены краткие теоретические сведения о развитии ЭВМ, об организации на аппаратном и программном уровне, о системах счисления, применяемых в ЭВМ, и о тенденциях развития.

Для студентов специальности 1-40 01 02 «Информационные системы и технологии (по направлениям)».

УДК 004.3(075.8)
ББК 32.973я73

ISBN 978-985-420-932-6

Предисловие

Настоящий курс лекций предназначен для студентов вузов, изучающих дисциплину «Организация и функционирование ЭВМ» и смежные ей. В издании обзорно рассмотрены: история развития и классификация ЭВМ, разделение их на типы; основные виды представления информации ЭВМ и основные приемы работы с данными; суть и взаимодействие основных модулей IBM-совместимых компьютеров в основном на аппаратном уровне с точки зрения прикладного программиста и пользователя; изложены вопросы организации взаимодействия между этими модулями. При этом основное внимание уделяется не столько техническим особенностям изготовления модулей, сколько идеологии их работы в составе ЭВМ. Именно поэтому многие разделы курса лекций изложены достаточно подробно, а некоторые носят исключительно обзорный характер.

Изложение материала построено исходя из того, что стремительные изменения в элементной базе и схемотехнике ЭВМ не дают возможности рассмотрения конкретных технических решений. Здесь важно понять принципы построения тех или иных устройств и механизм взаимодействия между ними.

1. Определение термина ЭВМ и уровни детализации структуры ЭВМ

Существует достаточное количество различных определений, что подразумевает под собой термин **электронно-вычислительная машина** (**ЭВМ**). В то же время нельзя отдать предпочтение какому-то из них:

1) устройство, которое принимает данные, обрабатывает их в соответствии с хранимой программой, генерирует результаты и обычно состоит из блоков ввода, вывода, памяти, арифметики и логики управления;

2) функциональный блок, способный выполнять реальные вычисления, включающие множественные арифметические и логические операции, без участия человека в этом процессе вычисления;

3) устройство, способное:

– хранить программу или программы обработки и по меньшей мере информацию, необходимую для выполнения программы;

– быть свободно перепрограммируемым в соответствии с требованиями пользователя;

– выполнять арифметические вычисления, определяемые пользователем;

– выполнять без вмешательства человека программу обработки, требующую изменения действий путем принятия логических решений в процессе обработки.

Наиболее корректным следует считать следующее определение – комплекс технических и программным средств, предназначенных для автоматизации подготовки и решений задач пользователя. Исходя из вышесказанного, вычислительную систему следует определить как совокупность взаимосвязанных и взаимодействующих процессоров или вычислительных машин, периферийных устройств и программного обеспечения.

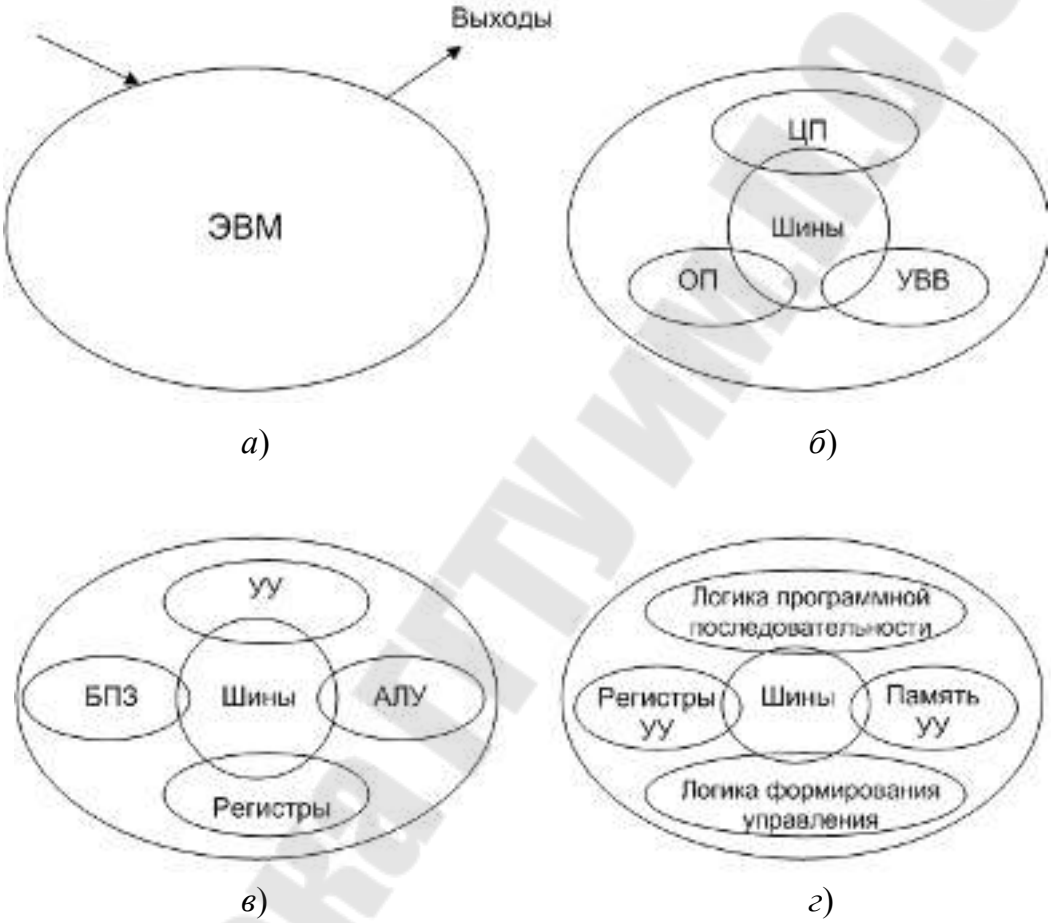
Цифровой компьютер – это машина, которая может решать задачи, выполняя данные ей команды. Последовательность команд, описывающих решение определенной задачи, называется **программой**. Электронные схемы каждого компьютера могут распознавать и выполнять ограниченный набор простых команд. Все программы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

1) сложить два числа;

2) проверить, не является ли число нулем;

3) скопировать блок данных из одной части памяти компьютера в другую.

Под **архитектурой** вычислительной машины обычно понимается логическое построения ЭВМ, т. е. то, какой машина представляется программисту. В зависимости от интересов специалиста, решающих ту или иную задачу, архитектуру ЭВМ можно детализировать на четыре уровня (рис. 1).



а) *б*) *в*) *г*)

Рис. 1. Уровни детализации ЭВМ:
а – уровень «черного ящика»; *б* – уровень общей архитектуры;
в – уровень архитектуры центрального процессора;
г – уровень архитектуры устройства управления

На первом уровне (рис. 1, *а*) ЭВМ рассматривается как устройство, способное хранить, принимать, обрабатывать и выдавать информацию – это уровень пользователя ЭВМ.

На втором уровне (рис. 1, *б*) ЭВМ представляется в виде четырех составляющих: центральный процессор (ЦП), основная память (ОП), устройство ввода/вывода (УВВ) и система шин. Это уровень систем-

ного архитектора ЭВМ, системного программиста прикладных задач и приложений ЭВМ.

На третьем уровне детализируется каждое из устройств второго уровня. Пример – ЦП третьего уровня детализации (рис. 1, *в*), где присутствуют: устройство управления (УУ), арифметико-логическое устройство (АЛУ), блок обработки чисел с плавающей запятой (БПЗ) и ячейки оперативной памяти процессора (регистры). Это уровень разработчика-схемотехника и разработчика-теоретика.

На четвертом уровне рассматриваются все элементы третьего. Пример – УУ третьего уровня детализации (рис. 1, *г*). Это уровень схемотехники элементов.

Все вышеописанные блоки внутри ЭВМ на всех уровнях взаимодействуют между собой посредством электрических сигналов, которые преобразуются из машинных кодов. Разработчик при создании нового компьютера должен решить, какие команды включить в машинный язык этого компьютера. Это зависит от назначения компьютера и от задач, которые он должен решать. Обычно стараются сделать машинные команды как можно проще, чтобы избежать сложностей при разработке компьютера и снизить затраты на необходимую электронику. Большинство машинных языков крайне примитивны, из-за чего писать на них трудно и утомительно.

Это простое наблюдение с течением времени привело к построению ряда уровней абстракций, каждая из которых надстраивается над абстракцией более низкого уровня. Именно таким образом можно преодолеть сложности при общении с компьютером. Мы будем называть этот подход многоуровневой компьютерной организацией и более подробно рассмотрим это далее.

2. История развития и классификация ЭВМ

В ходе эволюции компьютерных технологий были разработаны сотни разных компьютеров. Многие из них давно забыты, в то время как влияние других на современные идеи оказалось весьма значительным. Мы рассмотрим только основные моменты развития.

2.1. Нулевое поколение (1492–1945 гг.)

Нулевое поколение иногда еще называют «механической эрой». Первое упоминание о начале механической эры в виде описания суммирующего устройства на зубчатых колеса можно встретить в 1492 г.

в дневниках Леонардо да Винчи. Первым достоверным доказательством создания первого приближения к ЭВМ нулевого поколения является упоминание о французском ученом Блез Паскале (1623–1662), в честь которого назван один из языков программирования. Паскаль сконструировал эту машину в 1642 г., когда ему было всего 19 лет, для своего отца, сборщика налогов. Это была механическая конструкция с шестеренками и ручным приводом. Счетная машина Паскаля могла выполнять только операции сложения и вычитания.

Тридцать лет спустя великий немецкий математик Готфрид Вильгельм Лейбниц (1646–1716) построил другую механическую машину, которая помимо сложения и вычитания могла выполнять операции умножения и деления над двенадцатиразрядными числами. В сущности Лейбниц три века назад создал подобие карманного калькулятора с четырьмя функциями.

Через 150 лет профессор математики Кембриджского университета Чарльз Бэббидж (1792–1871), изобретатель спидометра, разработал и сконструировал разностную машину. Эта механическая машина, которая, как и машина Паскаля, могла лишь складывать и вычитать, подсчитывала таблицы чисел для морской навигации. В машину был заложен только один алгоритм – метод конечных разностей с использованием полиномов. У этой машины был довольно интересный способ вывода информации: результаты выдавливались стальным штампом на медной дощечке, что предвосхитило более поздние средства ввода/вывода – перфокарты и компакт-диски. Чтобы создать это программное обеспечение, Бэббидж нанял молодую женщину – Аду Августу Ловлейс (Ada Augusta Lovelace), дочь знаменитого британского поэта Байрона. Ада Ловлейс была первым в мире программистом. В ее честь назван современный язык программирования – Ada.

В 1892 г. стартует индустрия офисных калькуляторов благодаря их разработчику Вильяму Барроуз (1857–1898).

В 1937 г. Алан Тьюринг (1912–1954) из Кембриджского университета публикует статью, в которой излагает концепцию теоретической упрощенной вычислительной машины (машина Тьюринга).

Заканчивают механическую эру вычислители ASCC Mark I (руководитель Говард Айкен, 1943 г.) и Z4 с программой на перфоленте и возможностью вычислять квадратный корень (1945 г.), усовершенствованая версия первого в мире компьютера Z1 (Конрад Цузе, 1938 г).

2.2. Первое поколение – электронные лампы (1945–1955 гг.)

Стимулом к созданию электронной вычислительной машины – компьютера – стала Вторая мировая война. В начале войны германские подводные лодки разрушали британские корабли. Германские адмиралы посылали на подводные лодки по радио команды, и хотя англичане могли перехватывать эти команды, проблема была в том, что радиограммы были закодированы с помощью прибора под названием ENIGMA, предшественник которого был спроектирован изобретателем-дилетантом и бывшим президентом США Томасом Джефферсоном.

В начале войны англичанам удалось приобрести ENIGMA у поляков, которые, в свою очередь, украли ее у немцев. Однако чтобы расшифровать закодированное послание, требовалось огромное количество вычислений, и их нужно было произвести сразу после перехвата радиограммы. Поэтому британское правительство основало секретную лабораторию для создания электронного компьютера под названием COLOSSUS. COLOSSUS работал уже в 1943 г., но т. к. британское правительство полностью контролировало этот проект и рассматривало его как военную тайну на протяжении 30 лет, COLOSSUS не стал базой для дальнейшего развития компьютеров. Мы упомянули о нем только потому, что это был первый в мире электронный цифровой компьютер.

Джон Мочли (1907–1980), который был знаком с работами Атанасова и Стибблитса, понимал, что армия заинтересована в счетных машинах. Он потребовал от армии финансирования работ по созданию электронного компьютера. Требование было удовлетворено в 1943 г., и Мочли со своим студентом Дж. Преспером Эккертом начали конструировать электронный компьютер, который они назвали ENIAC (Electronic Numerical Integrator and Computer – электронный цифровой интегратор и калькулятор). ENIAC состоял из 18 000 электровакуумных ламп и 1500 реле, весил 30 тонн и потреблял 140 киловатт электроэнергии.

У машины было 20 регистров, каждый из которых мог содержать десятичное число. (Десятичный регистр – это память очень маленького объема, которая может вмещать число до какого-либо определенного максимального количества разрядов, что-то вроде одометра, запоминающего километраж пройденного автомобилем пути.) В ENIAC было установлено 6000 многоканальных переключателей и имелось множество кабелей, протянутых к разъемам.

Однако основной вклад в этом периоде внес американский математик Джон фон Нейман (1903–1957), который принял участие в роли консультанта при создании ENIAC, и еще до окончания работ по ней вышеупомянутый коллектив приступил к разработке и созданию EDSAC, отличавшуюся идеей хранимой в памяти программы.

В то время как Эккерт и Мочли работали над машиной EDVAC (1949 г.), Джон фон Нейман поехал в Институт специальных исследований в Принстоне, чтобы сконструировать собственную версию EDVAC под названием IAS (Immediate Address Storage – память с прямой адресацией).

Фон Нейман был гением в тех же областях, что и Леонардо да Винчи. Он знал много языков, был специалистом в физике и математике, обладал феноменальной памятью: он помнил все, что когда-либо слышал, видел или читал. Он мог дословно процитировать по памяти текст книг, которые читал несколько лет назад. Фон Нейман отметил, что десятичная арифметика, используемая в машине ENIAC, где каждый разряд представлялся десятью электронными лампами, должна быть заменена параллельной бинарной арифметикой. Между прочим, Атанасов пришел к аналогичному выводу лишь спустя несколько лет. Основной проект, который фон Нейман описал вначале, известен сейчас как фон-неймановская вычислительная машина. Сам замысел и машина IAS оказали очень большое влияние на дальнейшее развитие компьютерной техники, поэтому стоит кратко описать проект фон Неймана. Стоит иметь в виду, что хоть проект и связан с именем фон Неймана, в его разработке приняли деятельное участие и другие ученые. Архитектура этой машины представлена на рис. 2.

Рис. 2. Схема вычислительной машины фон Неймана

Машина фон Неймана состояла из пяти основных частей: памяти, арифметико-логического устройства, устройства управления, а также устройств ввода/вывода. Память включала 4096 слов размером по 40 бит (бит – это 0 или 1). Каждое слово содержало или 2 команды по 20 бит, или целое число со знаком на 40 бит. 8 бит указывали на тип команды, а остальные 12 бит определяли одно из 4096 слов. Арифметический блок и блок управления составляли «мозговой центр» компьютера. В современных машинах эти блоки сочетаются в одной микросхеме, называемой центральным процессором.

Внутри арифметико-логического устройства находился особый внутренний регистр на 40 бит, так называемый аккумулятор. Типичная команда добавляла слово из памяти в аккумулятор или сохраняла содержимое аккумулятора в памяти. Эта машина не выполняла арифметические операции с плавающей точкой, поскольку фон Нейман считал, что любой сведущий математик способен держать плавающую точку в голове.

В это же время в 1947 г. под руководством С. А. Лебедева начаты работы по созданию малой электронной счетной машины (МЭСМ). Она была запущена в 1951 г. и стала первой ЭВМ в СССР и континентальной Европе. В 1952 г. вступила в эксплуатацию ЭВМ М-1 (И. С. Брук и др.) с впервые предложенной двухадресной системой команд. Чуть позже командой выпускников МЭИ под руководством И. С. Брука была создана ЭВМ М-2 (быстродействие – 2000 операций/с).

В 1953 г. на рынок США выходит начинающая корпорация IBM (модели 701, 704) с памятью на магнитных сердечниках, а в СССР поступила в эксплуатацию самая быстрая ЭВМ в Европе БЭСМ (С. А. Лебедев) с быстродействием до 10 000 операций/с.

2.3. Второе поколение – транзисторы (1955–1965 гг.)

Благодаря изобретению транзистора сотрудниками лаборатории Bell Laboratories, за что в 1956 г. они получили Нобелевскую премию в области физики, в течение десяти лет была совершена революция в производстве компьютеров, и к концу 50-х гг. компьютеры на вакуумных лампах уже безнадежно устаревают. Первый компьютер для гражданских целей на транзисторах был построен в лаборатории Массачусетского технологического института (МТИ). Компьютер назывался TX-0 (Transistorized experimental computer 0 – экспериментальная транзисторная вычислительная машина 0) и предназначался только для тестирования будущей машины TX-2.

Машина TX-2 не имела большого значения, но один из инженеров этой лаборатории Кеннет Ольсен в 1957 г. основал компанию DEC (Digital Equipment Corporation – корпорация по производству цифровой аппаратуры), чтобы производить серийную машину, сходную с TX-0. Эта машина, PDP-1, появилась только через четыре года главным образом потому, что те, кто финансировал DEC, считали производство компьютеров невыгодным. Поэтому компания DEC продавала в основном небольшие электронные платы.

PDP-1 стоил 120 000 долларов, в то время как 7090 (IBM) стоил миллионы. Компания DEC продала десятки компьютеров PDP-1, и так началась зарождаться компьютерная промышленность. Одну из первых машин модели PDP-1 отдали в МТИ, где она сразу привлекла внимание некоторых молодых исследователей, подающих большие надежды. Одним из нововведений PDP-1 был дисплей размером 512 × 512 пикселей, на котором можно было рисовать точки. Вскоре студенты МТИ составили специальную программу для PDP-1, чтобы играть в «Войну миров» – первую в мире компьютерную игру.

Через несколько лет компания DEC разработала модель PDP-8 – 12-разрядный компьютер (PDP-8 стоил гораздо дешевле, чем PDP-1). Главным нововведением стало использование единственной шины (omnibus).

Шина – это набор параллельно соединенных проводов для связи компонентов компьютера. Это нововведение радикально отличало PDP-8 от IAS. Такая структура с тех пор стала использоваться во всех компьютерах. Компания DEC продала 50 000 компьютеров модели PDP-8 и стала лидером на рынке мини-компьютеров.

Следует упомянуть еще один компьютер – Burroughs B5000. Разработчики машин PDP-1, 7094 и 6600 занимались только аппаратным обеспечением, стараясь снизить его стоимость (DEC) или заставить работать быстрее (IBM и CDC). Программное обеспечение не менялось. Производители B5000 пошли другим путем. Они разработали машину с намерением программировать ее на языке Algol 60 (предшественнике языков C и Java), сконструировав аппаратное обеспечение так, чтобы упростить задачу компилятора. Так появилась идея, что при разработке компьютера нужно также учитывать и программное обеспечение. Но вскоре эта идея была забыта.

В этот период в СССР также ведутся разработки и освоение ЭВМ нового поколения, но в розничную торговлю они не идут, ос-

новное предназначение – НИИ и оборонно-промышленный комплекс СССР и стран Варшавского договора. Это малые ЭВМ серии «Наири», «Мир», средние «Минск-22», «Минск-32», БЭСМ-46 (скорость – 1 млн операций/с)

2.4. Третье поколение – интегральные схемы (1965–1980 гг.)

Третье поколение ознаменовалось резким увеличением вычислительной мощности ЭВМ, ставшие следствием больших успехов в области архитектуры, программирования и технологии, основанной на изобретении в 1958 г. Робертом Нойсом кремниевой интегральной схемы, позволившей размещать на одной небольшой микросхеме несколько десятков транзисторов. Компьютеры на интегральных схемах были меньшего размера, работали быстрее и стоили дешевле, чем их предшественники на транзисторах.

К 1964 г. компания IBM лидировала на компьютерном рынке, но существовала одна большая проблема: компьютеры 7094 и 1401, которые она выпускала, были несовместимы друг с другом. Один из них предназначался для сложных расчетов, в нем использовалась двоичная арифметика на регистрах по 36 бит, во втором применялась десятичная система счисления и слова разной длины. У многих покупателей были оба этих компьютера, и им не нравилось, что они совершенно несовместимы. Когда пришло время заменить эти две серии компьютеров, компания IBM сделала решительный шаг. Она выпустила линейку транзисторных компьютеров System 360, которые были предназначены как для научных, так и для коммерческих расчетов. Линейка System 360 имела много нововведений. Это было целое семейство компьютеров для работы с одним языком (ассемблером). Каждая новая модель была больше по возможностям, чем предыдущая. Компания смогла заменить 1401 на 360 (модель 30), а 7094 – на 360 (модель 75). Модель 75 была больше по размеру, работала быстрее и стоила дороже, но программы, написанные для одной из них, могли использоваться в другой. На практике программы, написанные для маленькой модели, выполнялись большой моделью без особых затруднений.

СССР в сотрудничестве со странами Европы начала осваивать в производстве разрабатываемые ЭВМ серии ЕС.

2.5. Четвертое поколение – сверхбольшие интегральные схемы (1980–2008 гг.)

Появление сверхбольших интегральных схем (СБИС) в 80-х гг. позволило помещать на одну плату сначала десятки тысяч, затем сотни тысяч и, наконец, миллионы транзисторов. Это привело к созданию компьютеров меньшего размера и более быстродействующих. До появления PDP-1 компьютеры были настолько велики и дороги, что компаниям и университетам приходилось иметь специальные отделы (вычислительные центры). К 80-м гг. цены упали так сильно, что возможность приобретать компьютеры появилась не только у организаций, но и у отдельных людей. Началась эра персональных компьютеров.

Первые персональные компьютеры продавались в виде комплектов. Каждый комплект содержал печатную плату, набор интегральных схем, обычно включающий схему Intel 8080, несколько кабелей, источник питания и иногда 8-дюймовый дисковод. Собрать из этих частей компьютер покупатель должен сам. Программное обеспечение к компьютеру не прилагалось. Покупателю приходилось писать программное обеспечение самому. Позднее появилась операционная система CP/M, написанная Гари Килдаллом для Intel 8080. Эта действующая операционная система помещалась на дискету, она включала в себя систему управления файлами и интерпретатор для выполнения пользовательских команд, которые набирались с клавиатуры.

Еще один персональный компьютер, Apple (а позднее и Apple II), был разработан Стивом Джобсом и Стивом Возняком. Этот компьютер стал чрезвычайно популярным среди домашних пользователей и школ, что очень быстро сделало компанию Apple серьезным игроком на рынке.

Наблюдая за тем, чем занимаются другие компании, компания IBM, лидирующая тогда на компьютерном рынке, тоже решила заняться производством персональных компьютеров.

Одним из значимых событий архитектуры стала идея создания ЭВМ с сокращенным набором команд (RISC) в1980 г. С этого момента выделяются крупные компании (брэнды) в области разработки и производства комплектующих и основных элементов ЭВМ процессоров. В этом поколении начинают создаваться супер- и мини-суперкомпьютеры.

В 1981 г. правительство Японии объявило о намерениях выделить национальным компаниям 500 млн долларов на разработку компьютеров пятого поколения на основе технологий искусственного ин-

теллекта, которые должны были потеснить «тугие на голову» машины четвертого поколения. Наблюдая за тем, как японские компании оперативно захватывают рыночные позиции в самых разных областях промышленности – от фотоаппаратов до стереосистем и телевизоров, американские и европейские производители в панике бросились требовать у своих правительств аналогичных субсидий и прочей поддержки. Однако, несмотря на большой шум, японский проект разработки компьютеров пятого поколения в конечном итоге показал свою несостоятельность и был аккуратно «задвинут в дальний ящик». В каком-то смысле эта ситуация оказалась близка к той, с которой столкнулся Беббидж: идея настолько опередила свое время, что для ее реализации не нашлось адекватной технологической базы.

До настоящего времени идет процесс наращивания частоты процессоров, плотности упаковки элементов в корпусе СБИС, улучшения их схемотехники и потребительских качеств. Ведутся подготовительные работы для перехода на нанотехнологии с использованием нанотрубок в производстве элементов ЭВМ.

3. Типы компьютеров

Хотя персональные компьютеры – наиболее известные типы «умных» машин, в наши дни существуют и другие типы машин, поэтому стоит кратко рассказать о них.

Компьютерная промышленность двигается вперед как никакая другая. Главная движущая сила – способность производителей помещать с каждым годом все больше и больше транзисторов на микросхему. Чем больше транзисторов (крошечных электронных переключателей), тем больше объем памяти и мощнее процессоры. Гордон Мур, один из основателей и бывший председатель совета директоров Intel, однажды сострил по поводу того, что, если бы авиационные технологии развивались с такой же скоростью, как компьютерные, самолеты стоили бы 500 долларов и облетали землю за 20 минут на 20 литрах топлива. Правда, для этого они должны стать размером с обувную коробку. Он же сформулировал закон технологического прогресса, известный теперь под названием «закон Мура». Когда Гордон готовил доклад для одной из промышленных групп, он заметил, что каждое новое поколение микросхем появляется через три года после предыдущего. Поскольку у каждого нового поколения компьютеров было в 4 раза больше памяти, чем у предыдущего, стало понятно, что число транзисторов на микросхеме возрастает на постоянную величину и таким образом этот рост можно предсказать на го-

ды вперед. Закон Мура гласит, что количество транзисторов на одной микросхеме удваивается каждые 18 месяцев, т. е. увеличивается на 60 % каждый год. Размеры микросхем и даты их производства подтверждают, что закон Мура действует до сих пор. Многие специалисты считают, что закон Мура будет действовать еще лет десять, а возможно и дольше. Вероятно в конечном итоге количество атомов, из которых состоят транзисторы, уменьшится до критической величины, хотя последние достижения квантовой компьютерной техники может быть изменят ситуацию.

Типы ЭВМ можно разделить по различным критериям: быстродействие, уровень архитектуры, производители, стоимость и т. д. Однако, на наш взгляд, в связи с моральным устареванием технологий из года в год ЭВМ более высокого класса постепенно будут замещать более низкопроизводительные ниши и соответственно с каждым годом будет падать их стоимость.

Таблица 1

Типы современных компьютеров (указанные цены приблизительны)

Тип	Цена, дол. США	Сфера применения
«Одноразовые» компьютеры	0,5	Поздравительные открытки
Встроенные компьютеры	5	Часы, машины, различные приборы (микроконтроллеры)
Игровые компьютеры	50	Домашние компьютерные игры
Персональные компьютеры	500–2000	Настольные и портативные компьютеры
Серверы	5000	Сетевые серверы
Комплексы рабочих станций	50 000–500 000	Мини-суперкомпьютеры
Мэйнфреймы	5 000 000	Пакетная обработка данных в банке

4. Многоуровневая организация и архитектура ЭВМ

4.1. Многоуровневая организация ЭВМ

Как мы уже отметили, существует огромная разница между тем, что удобно людям, и тем, что могут компьютеры. Люди хотят сделать *X*, но компьютеры могут сделать только *Y*. Из-за этого возникает проблема, которую можно решить двумя способами. Оба способа

подразумевают разработку новых команд, более удобных для человека, чем встроенные машинные команды. Эти новые команды в совокупности формируют язык, который мы будем называть Я1. Встроенные машинные команды тоже формируют язык, и мы будем называть его Я0. Компьютер может выполнять только программы, написанные на его машинном языке Я0. Два способа решения проблемы различаются тем, каким образом компьютер будет выполнять программы, написанные на языке Я1 – ведь в конечном итоге компьютеру доступен только машинный язык Я0.

Первый способ выполнения программы, написанной на языке Я1, подразумевает замену каждой команды эквивалентным набором команд на языке Я0. В этом случае компьютер выполняет новую программу, написанную на языке Я0, вместо старой программы, написанной на Я1. Эта технология называется *трансляцией*.

Второй способ означает создание программы на языке Я0, получающей в качестве входных данных программы, написанные на языке Я1. При этом каждая команда языка Я1 обрабатывается поочередно, после чего сразу выполняется эквивалентный ей набор команд языка Я0. Эта технология не требует составления новой программы на Я0. Она называется *интерпретацией*, а программа, которая осуществляет интерпретацию, называется *интерпретатором*.

Между трансляцией и интерпретацией много общего. В обоих подходах компьютер в конечном итоге выполняет набор команд на языке Я0, эквивалентных командам Я1. Различие лишь в том, что при трансляции вся программа Я1 переделывается в программу Я0, программа Я1 отбрасывается, а новая программа на Я0 загружается в память компьютера и затем выполняется.

При интерпретации каждая команда программы на Я1 перекодируется в Я0 и сразу же выполняется. В отличие от трансляции здесь не создается новая программа на Я0, а происходит последовательная перекодировка и выполнение команд. С точки зрения интерпретатора, программа на Я1 есть не что иное, как «сырые» входные данные. Оба подхода широко используются как вместе, так и по отдельности. Впрочем, чем мыслить категориями трансляции и интерпретации, гораздо проще представить себе существование гипотетического компьютера или виртуальной машины, для которой машинным языком является язык Я1.

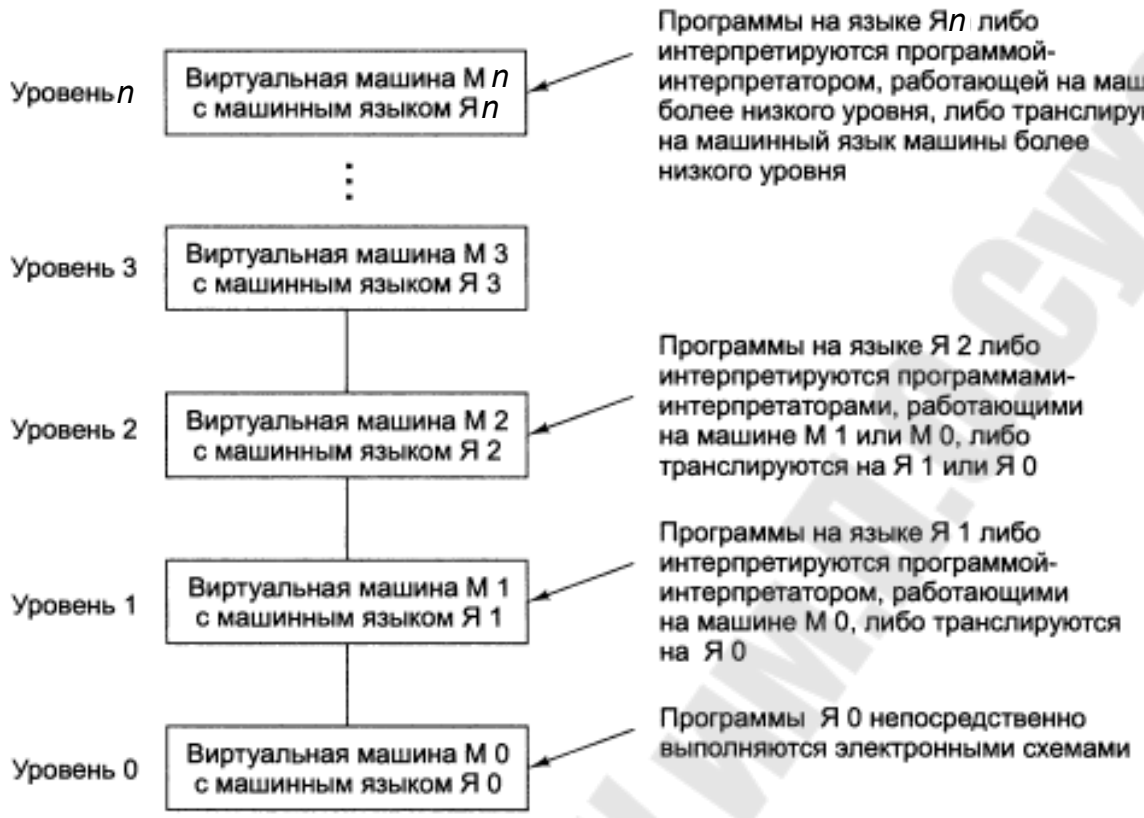


Рис. 3. Многоуровневая машина

Между языком и виртуальной машиной существует важная зависимость. Каждая машина поддерживает какой-то определенный машинный язык, состоящий из всех команд, которые эта машина может выполнять. В сущности машина определяет язык. Сходным образом язык определяет машину, которая может выполнять все программы, написанные на этом языке. Машину, определяемую тем или иным языком, очень сложно и дорого конструировать из электронных схем, однако представить себе такую машину мы можем. Компьютер для работы с машинным языком С или С++ был бы слишком сложным, но в принципе его можно разработать, учитывая высокий уровень современных технологий. Однако существуют веские причины не создавать такой компьютер – это крайне неэффективное по сравнению с другими решение. Действительно, технология должна быть не только осуществимой, но и рациональной. Те, кто хочет понять, как в действительности работает компьютер, должны изучить все уровни. Также должны быть знакомы со всеми уровнями разработчики новых компьютеров или новых уровней (т. е. новых виртуальных машин).

4.2. Современные многоуровневые машины

Большинство современных компьютеров состоит из двух и более уровней. Существуют машины даже с шестью уровнями (рис. 4). Уровень 0 – это аппаратное обеспечение машины. Электронные схемы на уровне 1 выполняют машиннозависимые программы. Ради полноты нужно упомянуть о существовании еще одного уровня, который расположен ниже нулевого. Этот уровень не показан на рис. 4, т. к. он попадает в сферу электронной техники и, следовательно, не рассматривается в этом курсе. Он называется уровнем физических устройств. На этом уровне находятся транзисторы, которые для разработчиков компьютеров являются примитивами. Объяснить, как работают транзисторы – задача физики.

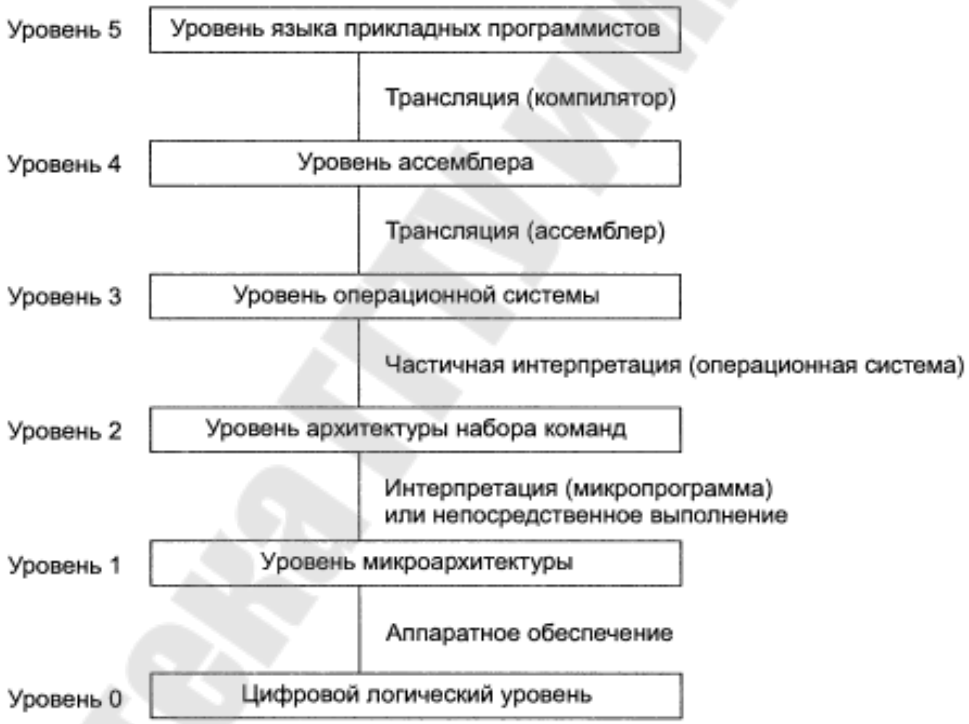


Рис. 4. Структура шестиуровневого компьютера

На самом нижнем уровне из тех, что мы можем изучить в рамках курса, а именно на *цифровом логическом уровне*, объекты называются вентилями. Хотя вентили состоят из аналоговых компонентов, таких как транзисторы, они могут быть точно смоделированы как цифровые устройства. У каждого вентиля есть один или несколько цифровых входов (сигналов, представляющих 0 или 1). Вентиль вычисляет простые функции этих сигналов, такие как И или ИЛИ. Каждый вентиль

формируется из нескольких транзисторов. Несколько вентилей формируют 1 бит памяти, который может содержать 0 или 1. Биты памяти, объединенные в группы, например, по 16, 32 или 64, формируют регистры. Каждый регистр может содержать одно двоичное число до определенного предела.

Следующий уровень называется *уровнем микроархитектуры*. На этом уровне находятся совокупности 8 или 32 регистров, которые формируют локальную память и схему, называемую АЛУ (арифметико-логическое устройство). АЛУ выполняет простые арифметические операции. Регистры вместе с АЛУ формируют тракт данных, по которому поступают данные. Тракт данных работает следующим образом: выбирается один или два регистра, АЛУ производит над ними какую-либо операцию, например, сложения, после чего результат вновь помещается в один из этих регистров. В следующих лекциях это будет рассмотрено подробнее.

На некоторых машинах работа тракта данных контролируется особой программой, которая называется микропрограммой. На других машинах тракт данных контролируется аппаратными средствами. Поскольку сейчас тракт данных обычно контролируется аппаратным обеспечением, мы изменили название, чтобы точнее отразить смысл.

На машинах, где тракт данных контролируется программным обеспечением, микропрограмма – это интерпретатор для команд на уровне 2. Микропрограмма вызывает команды из памяти и выполняет их одну за другой, используя при этом тракт данных. Например, при выполнении команды ADD она вызывается из памяти, ее операнды помещаются в регистры, АЛУ вычисляет сумму, а затем результат переправляется обратно. На компьютере с аппаратным контролем тракта данных происходит такая же процедура, но при этом нет программы, интерпретирующей команды уровня 2.

Уровень 2 мы будем называть *уровнем архитектуры набора команд*. Каждый производитель публикует руководство для компьютеров, которые он продает, под названием «Руководство по машинному языку *X*», «Принципы работы компьютера *Y*» и т. п. Подобное руководство содержит информацию именно об этом уровне. Описываемый в нем набор машинных команд в действительности выполняется микропрограммой-интерпретатором или аппаратным обеспечением. Если производитель поставляет два интерпретатора для одной машины, он должен издать два руководства по машинному языку, отдельно для каждого интерпретатора.

Следующий уровень обычно является гибридным. Большинство команд в его языке есть также и на уровне архитектуры набора команд (команды, имеющиеся на одном из уровней, вполне могут быть представлены и на других уровнях). У этого уровня есть некоторые дополнительные особенности: новый набор команд, другая организация памяти, способность выполнять две и более программы одновременно и некоторые другие. При построении уровня 3 возможно больше вариантов, чем при построении уровней 1 и 2.

Новые средства, появившиеся на уровне 3, выполняются интерпретатором, который работает на уровне 2. Этот интерпретатор был когда-то назван операционной системой. Команды уровня 3, идентичные командам уровня 2, выполняются микропрограммой или аппаратным обеспечением, но не операционной системой. Другими словами, одна часть команд уровня 3 интерпретируется операционной системой, а другая часть – микропрограммой. Вот почему этот уровень считается гибридным. Мы будем называть этот уровень *уровнем операционной системы*.

Между уровнями 3 и 4 есть существенная разница. Нижние три уровня задуманы не для того, чтобы с ними работал обычный программист. Они изначально ориентированы на интерпретаторы и трансляторы, поддерживающие более высокие уровни. Эти трансляторы и интерпретаторы составляются так называемыми системными программистами, которые специализируются на разработке новых виртуальных машин.

Уровни с четвертого и выше предназначены для прикладных программистов, решающих конкретные задачи. Еще одно изменение, появившееся на уровне 4, – механизм поддержки более высоких уровней. Уровни 2 и 3 обычно интерпретируются, а уровни 4, 5 и выше обычно, хотя и не всегда, транслируются.

Другое отличие между уровнями 1, 2, 3 и уровнями 4, 5 и выше – особенность языка. Машинные языки уровней 1, 2 и 3 – цифровые. Программы, написанные на этих языках, состоят из длинных рядов цифр, которые воспринимаются компьютерами, но малопонятны для людей. Начиная с уровня 4, языки содержат слова и сокращения, понятные человеку.

Уровень 5 обычно состоит из языков, разработанных для прикладных программистов. Такие языки называются языками высокого уровня. Существуют сотни языков высокого уровня. Наиболее известные среди них – C, C++, Java, LISP и Prolog. Программы, написанные на этих языках, обычно транслируются на уровень 3 или 4.

Трансляторы, которые обрабатывают эти программы, называются компиляторами. Отметим, что иногда также имеет место интерпретация. Например, программы на языке Java сначала транслируются на язык, напоминающий ISA и называемый байт-кодом Java, который затем интерпретируется.

В некоторых случаях уровень 5 состоит из интерпретатора для конкретной прикладной области, например, символической логики. Он предусматривает данные и операции для решения задач в этой области, выраженные при помощи специальной терминологии.

Таким образом, компьютер проектируется как иерархическая структура уровней, которые надстраиваются друг над другом. Каждый уровень представляет собой определенную абстракцию различных объектов и операций. Рассматривая компьютер подобным образом, мы можем не принимать во внимание ненужные нам детали и таким образом сделать сложный предмет более простым для понимания.

Набор типов данных, операций и характеристик каждого отдельно взятого уровня называется *архитектурой*.

Архитектура связана с программными аспектами. Например, сведения о том, сколько памяти можно использовать при написании программы – часть архитектуры. Аспекты реализации (например, технология, применяемая при реализации памяти) не являются частью архитектуры. Изучая методы проектирования программных элементов компьютерной системы, мы изучаем компьютерную архитектуру. На практике термины «компьютерная архитектура» и «компьютерная организация» употребляются как синонимы.

4.3. Развитие многоуровневых машин

В самых первых компьютерах граница между аппаратным и программным обеспечением была очевидна. Со временем, однако, произошло значительное размывание этой границы, в первую очередь благодаря тому, что в процессе развития компьютеров уровни добавлялись, убирались и сливались друг с другом. Аппаратное и программное обеспечение логически эквивалентны, т. е. аппаратное обеспечение – это всего лишь окаменевшее программное обеспечение.

У первых цифровых компьютеров в 40-х гг. было только два уровня: уровень архитектуры набора команд, на котором осуществлялось программирование, и цифровой логический уровень, выполнявший программы. Схемы цифрового логического уровня были ненадежны, сложны для производства и понимания.

4.4. Изобретение операционной системы

В те времена, когда компьютеры только появились, принципы работы с ними сильно отличались от современных. Одним компьютером пользовалось большое количество людей. Рядом с машиной лежал листок бумаги, и если программист хотел запустить свою программу, он записывался на какое-то определенное время, скажем, на среду с трех часов ночи до пяти утра (многие программисты любили работать в тишине).

В 60-е гг. человек попытался ускорить дело, автоматизировав работу оператора. Программа под названием «операционная система» загружалась в компьютер на все время его работы. Программист приносил пачку перфокарт со специализированной программой, которая выполнялась операционной системой. В последующие годы операционные системы все больше и больше усложнялись. К уровню архитектуры набора команд добавлялись новые команды, приспособления и особенности, и в конечном итоге сформировался новый уровень. Некоторые команды нового уровня были идентичны командам предыдущего, но некоторые (в частности, команды ввода/вывода) полностью отличались. Эти новые команды тогда назывались макросами операционной системы, или вызовами супервизора. Сейчас обычно используется термин «системный вызов».

Программное обеспечение сегодня может быть аппаратным обеспечением завтра и наоборот. Также обстоит дело и с уровнями – между ними нет четких границ.

Для программиста не важно, как на самом деле выполняется команда (за исключением, может быть, скорости выполнения). Программист, работающий на уровне архитектуры системы, может использовать команду умножения, как будто это команда аппаратного обеспечения, и даже не задумываться об этом. То, что для одного человека – программное обеспечение, для другого – аппаратное.

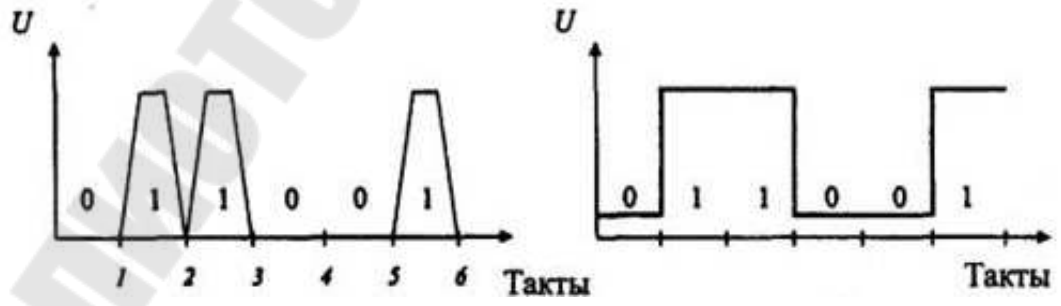
5. Основные виды предоставления информации в ЭВМ

Все современные ЭВМ строятся на комплексах системах интегральных микросхем (ИС). Электронная микросхема называется интегральной, если ее компоненты и соединения между ними выполнены в едином технологическом цикле, на едином основании и имеют общую герметизацию и защиту от механических воздействий. Каждая микросхема представляет собой миниатюрную электронную схему, сформированную послойно в кристалле полупроводника: кремния,

германия и т. д. В состав микропроцессорных наборов включаются различные типы микросхем, но все они должны иметь единый тип межмодульных связей, основанный на стандартизации параметров сигналов взаимодействия (амплитуда, полярность, длительность импульсов и т. п.). Основу набора обычно составляют БИС, СБИС и ультра СБИС. Кроме них обычно используются микросхемы с малой и средней степенью интеграции (СИС). Функционально микросхемы могут соответствовать устройству, узлу или блоку, но каждая из них состоит из комбинации простейших логических элементов, реализующих функции формирования, преобразования, запоминания сигналов и т. д. Принципы функционирования этих узлов можно понять, зная курсы «Дискретная математика», «Логические и арифметические основы функционирования ЭВМ» или им подобные.

Элементы ЭВМ можно классифицировать по различным признакам. Наиболее часто такими признаками являются: тип сигналов, назначение элементов, технология их изготовления и т. д.

В соответствии с используемой формой представления информации вычислительные машины в первую очередь делятся на два класса: непрерывного действия – *аналоговые* и дискретного действия – *цифровые*. Соответственно будем считать, что в нашем случае ЭВМ – это второй класс. В ЭВМ широко применяют два способа физического представления сигналов: импульсный и потенциальный. При импульсном способе представления сигналов единичному значению некоторой двоичной переменной ставится в соответствие наличие импульса (тока или напряжения), нулевому значению – отсутствие импульса (рис. 5, *а*). Длительность импульсного сигнала не превышает один такт синхроимпульсов. При потенциальном или статическом представлении сигналов единично значение двоичной переменной отображается высоким уровнем напряжения, а нулевое значение – низким уровнем (рис. 5, *б*).



а) *б*)

Рис. 5. Представление информации в ЭВМ:
а – импульсные сигналы; *б* – потенциальные сигналы

Независимо от вида сигналов различают *последовательный* и *параллельный* коды передачи и представления информации в ЭВМ.

При последовательном коде представления данных используются одиночные шины или линии передачи, в которых сигналы, соответствующие отдельным разрядам данных, разнесены во времени. Обработка такой информации производится последовательно разряд за разрядом. Такой вид представления и передачи данных требует весьма экономичных по аппаратурным затратам схем обработки данных. Время же обработки определяется числом обрабатываемых сигналов (разрядов).

Параллельный код отображения и передачи информации предполагает параллельную и одновременную фиксацию всех разрядов данных на различных шинах, т. е. параллельный код данных развернут в пространстве. Это дает возможность ускорить обработку во времени, но затраты на аппаратурные средства при этом возрастают пропорционально числу обрабатываемых разрядов.

Во всех ЭВМ используются и параллельно-последовательные коды представления информации. При этом информация отображается частями. Части поступают на обработку последовательно, а каждая часть данных представляется параллельным кодом.

6. Системы счисления

Система счисления – это совокупность приемов и правил, по которым числа записываются и читаются.

В повседневной практике мы пользуемся, как правило, десятичной *системой счисления*. Ответ на вопрос «Почему именно эта система счета получила наибольшее распространение?» сейчас дать затруднительно. В литературе, как правило, в качестве обоснования приводится тот факт, что на руках человека в сумме 10 пальцев. Вряд ли это обоснование можно принимать всерьез. На практике мы сталкиваемся и с более сложными, в частности, со смешанными *системами*. Например, система счета времени, где за единицу принята секунда, минута, час, сутки, неделя, месяц, год. Или система счета денег, до недавнего времени применявшаяся в Англии (пенс, шиллинг, фунт):

12 п = 1 ш, 20 ш = 1 ф.

Или еще более интересная – римская система счета, которая использует символы: I – 1, V – 5, X – 10, L – 50, C – 100, D – 500, M – 1000.

Эта система является особой и применяется редко (циферблат, архитектура, история и т. д.). Системы счисления принято делить на:

1) позиционные;

2) непозиционные;

3) символические.

В символических системах каждому числу ставится в соответствие свой символ. Эти системы не находят широкого применения в силу естественной их ограниченности (алхимия, кодированные сообщения) – бесчисленного множества символов, которое требуется для изображения всех возможных чисел. Поэтому эти системы из рассмотрения опустим.

6.1. Позиционные системы счисления

Само название этих систем указывает на связь значимости числа и его изображения от позиции.

Позиция – некоторое место, в котором может быть представлен лишь один символ. Примером позиционной системы счисления является десятичная система. В этой системе число представляется в виде полинома *n*-й степени, а изображается совокупностью некоторых символов, каждый из которых имеет различный вес в зависимости от позиции, которую он занимает:

$a_4a_3a_2a_1$ – число; a_1, a_2, a_3, a_4 – символы.

Всем позициям приписывается различный вес, который чаще всего выбирается как целая степень основания системы. **Основание системы счисления** – число, которое является мощностью множества различных символов, допустимых в каждой позиции числа.

Так, для десятичной системы допускаемыми являются символы 0, 1, 2, 3, ..., 9.

Обозначим основание системы через p. Тогда веса позиций числа могут быть представлены так: $\ldots p^3\ p^2\ p^1\ p^0$.

Само число, изображение которого имеет вид, например, $a_4a_3a_2a_1$, может быть представлено так: $a_0p^0 + a_1p^1 + a_2p^2 + a_3p^3$, что является развернутой записью числа в позиционной системе. Например:

$$973_{10} = 3 \cdot 10^0 + 7 \cdot 10^1 + 9 \cdot 10^2 = 3 + 70 + 900.$$

В отличие от системы счета времени, десятичная система является однородной, т. е. одних и тех же десятичных символов достаточно, чтобы изобразить любое число. В то время как в смешанных системах нужно придумывать все новые и новые символы для того, чтобы изобразить следующее по величине число. Таким образом, *однородность* – одно из важных свойств позиционных систем.

Любая позиционная система счисления определяется основанием системы, алфавитом и правилами выполнения арифметических операций. Правила выполнения арифметических операций в десятичной системе счисления применимы и к другим позиционным системам счисления. Однако при выполнении операций сложения и умножения необходимо пользоваться таблицами сложения и умножения для конкретной системы счисления.

Для записи чисел в позиционной системе с основанием «*p*» необходимо иметь алфавит из *n* цифр. В качестве цифр используются обозначение соответствующих цифр десятичной системы счисления 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, а в случае, когда десятичных цифр «не хватает» (для систем счисления с основанием *p*, большим, чем 10), для цифр, превышающих 9, вводятся дополнительные обозначения, например, для $q = 16$ добавляют буквы *A*, *B*, *C*, *D*, *E*, *F*, которые соответствуют шестнадцатеричным цифрам, десятичные эквиваленты которых равны соответственно 10, 11, 12, 13, 14, 15. В табл. 2 приведены примеры нескольких систем.

Таблица 2

Алфавиты для разных систем счисления

Основание	**Название**	**Алфавит**
$n = 2$	Двоичная (*B*)	0 1
$n = 3$	Троичная	0 1 2
$n = 8$	Восьмеричная (*O*)	0 1 2 3 4 5 6 7
$n = 16$	Шестнадцатеричная (*H*)	0 1 2 3 4 5 6 7 8 9 *A B C D E F*

Чтобы указать основание системы, к которой относится число, необходимо приписать либо условный символ системы счисления, либо нижний индекс к числу в виде цифры. Например:

10101101_2, 1221_3, 747_8, $3BE5_{16}$ либо **10101101 B, 747 O, 3BE5 H**.

Любое число *X* в позиционной системе счисления можно представить в виде:

$$X = \pm p^m \sum_{i=1}^{n} a_i p^{-i},$$

где *m* – число позиций или разрядов, отведенное для изображения целой части числа; *n* – общее число разрядов в числе; a_i – любой допустимый символ в разряде, т. е. $a_i = \{0, 1, 2, ..., p - 1\}$.

Например:

$$-961{,}13 = -(9 \cdot 10^2 + 6 \cdot 10^1 + 1 \cdot 10^0 + 1 \cdot 10^{-1} + 3 \cdot 10^{-2}).$$

Следует заметить, что число, равное основанию системы счисления, т. е. p, в самой системе с основанием p записывается только в двух позициях (разрядах), а именно так: $p_p = 10_p$. Заметим также, что разделение числа на две части – дробную и целую – имеет смысл лишь в позиционных системах.

Возникает естественный вопрос, является ли общепринятая система счисления с основанием 10 оптимальной? Если да, то с каких позиций? Вопрос заслуживает внимания, т. к. одна из первых ЭВМ (ENIAC) использовала именно десятеричную систему.

Прямой и однозначный ответ на этот вопрос невозможен. Можно дать много различных ответов, и все они будут справедливы лишь для каких-то определенных условий.

Введя общее представление числа в позиционной системе, мы усомнились в достоинствах десятеричной не потому, что она вдруг проявила свои отрицательные качества, а потому, что ее преимущества явны лишь при ручных методах счета. Нас же интересуют, прежде всего, такие системы счисления, которые будут удобны и экономичны при автоматических вычислениях с помощью ЭВМ. Мы должны также помнить, что необходимо для этого иметь саму ЭВМ.

Покажем, что десятеричная система не устарела. Например, для производства экономичных расчетов обычно приходится иметь дело с очень большими объемами числовой информации. Тогда с введением новой системы пришлось бы воспользоваться цепочкой действий (рис. 6).

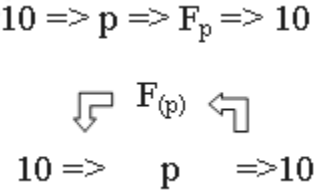


Рис. 6. Схема преобразования чисел
из одной системы счисления в другую

То есть нужно было бы из десятеричной системы перевести информацию в p-систему, произвести над ней необходимые операции в системе p, затем снова сделать, но обратный перевод из p-системы в десятеричную, т. к. отказ от десятеричной системы потребовал бы

и устранения первого этапа. Если преобразование из десятеричной системы в p-систему требует не слишком много времени, в то же время, если выполнение функции F будет в системе p сделано много быстрее, то тогда эта цепочка действий будет оправданной.

Но для экономической информации характерно то, что очень несложные операции нужно производить всякий раз над большим объемом исходных данных. Так что в данном случае вряд ли целесообразно переходить к новой системе. Это и является объяснением того факта, что в настоящее время значительное число ЭВМ строится именно в десятеричной системе счисления. Однако ЭВМ предназначены не только для выполнения экономических расчетов. В большинстве случаев неэкономических применений ЭВМ имеют дело с задачами, в которых общий объем исходных данных невелик, но общее число необходимых операций огромно. Именно для такого рода применений рассмотренная последовательность действий может оказаться выгодной.

Очевидно, что можно, не сужая области применения ЭВМ, задаться величиной некоторого самого большого числа. Пусть это будет число M. Воспользуемся позиционной системой счисления с основанием p, и тогда потребуется n разрядов, чтобы представить все M чисел:

$$M = p^n - 1 \text{ (от 0 до } p^n - 1 \text{) или в первом приближении } M \approx p^n,$$

$$\log_p M = n \cdot \log_p p,$$

где $\log_p p = 1$, тогда $n = \log_p M$.

Оборудование, которое нужно для хранения любого числа от 0 до M, пропорционально произведению основания системы счисления на количество разрядов. Таким образом при заданном числе M количество цифроразрядов при основании p можно будет найти из следующего соотношения:

$$p \cdot n = p \cdot \log_p M, \qquad (1)$$

где $\log_p M$ – цифроразряд (эквивалент оборудования); $p \cdot n$ – число устойчивых состояний элемента памяти; n – число разрядов в числе.

Рассмотрим пример.

Пусть есть 24 цифроразряда (табл. 3).

Таблица 3

Цифроразряды

Основание *p*	**Возможное число цифроразрядов**	**Наибольшее число *M***
2	$2 \cdot 12$	$\underbrace{1 \cdot 1 \cdot \ldots \cdot 1_2}_{12} = 4095_{10}$
3	$3 \cdot 8$	$\underbrace{2 \cdot 2 \cdot \ldots \cdot 2_3}_{8} = 6560_{10}$
4	$4 \cdot 6$	$\underbrace{3 \cdot 3 \cdot \ldots \cdot 3_4}_{6} = 4095_{10}$
6	$6 \cdot 4$	$\underbrace{5 \cdot 5 \cdot 5 \cdot 5_6}_{4} = 1295_{10}$
8	$8 \cdot 3$	$\underbrace{7 \cdot 7 \cdot 7_8}_{3} = 511_{10}$

Количество цифроразрядов говорит как о величине оборудования, так и является характеристикой быстродействия. Как увидим позже, в позиционной системе счисления время выполнения операций может быть выражено через количество разрядов в числе.

Считаем p величиной непрерывной. Находим производную от (1) по величине p. Берем вторую производную по p. Можно доказать, что первая производная обращается в нуль, а вторая – больше нуля при $p = e$. То есть получаем минимум при $p = e$. Таким образом, оптимальной по оборудованию и быстродействию является система с основанием e. Но $e = 2{,}718...$

Поэтому оптимальной, определившись, что основание может быть только целым числом, является система с основанием $p = 3$.

Построим функцию, характеризующую отношение оборудования в системе с основанием p относительно системы с основанием 2 (табл. 4).

Таблица 4

Значения функции, характеризующей отношение оборудования в системе с основанием *p* относительно системы с основанием 2

p	2	3	4	5	6	7	8	9	10
***f*(*p*)**	1,000	0,946	1,000	1,078	1,148	1,247	1,333	1,420	1,595

То есть десятеричная система является более чем в 1,5 раза неэкономичной по отношению к двоичной системе, а троичная система оказывается лишь на 5 % экономичнее двоичной. Действительное обоснование экономичности той или иной системы выглядит несколько сложнее.

Когда говорим об экономичности, то прежде всего имеем в виду объем оборудования, сосредоточенный в АЛУ и ЗУ. Объем оборудования УУ не находится в столь простой зависимости от p, да и в АЛУ учитывается лишь оборудование, связанное с элементами хранения информации, но не логическое оборудование.

Более детальный анализ показывает, что наиболее эффективными являются системы с основанием, кратным 2, т. е. 2, 4, 8, 16. Специфика построения схем ЭВМ показывает, что наиболее эффективной является шестнадцатеричная система. Именно она и применяется в современных машинах.

Мы же будем считать эффективной систему с основанием 2 по причине ее наибольшего распространения.

Основные соображения в пользу этой системы:

1. Высокая информационная эффективность.

2. Простота и надежность работы двоичного элемента хранения информации (т. е. имеющего 2 устойчивых состояния, например, есть ток – нет тока, намагничен – не намагничен и т. п.).

3. Совпадение максимального числа состояний элемента с максимальным числом значений двоичной переменной, дающее возможность не строить специальные устройства для выполнения логических операций, т. е. возможно применение аппарата булевой алгебры для выполнения логических преобразований информации.

4. Простота построения схем для выполнения простых операций.

5. Более высокая скорость выполнения основных арифметических операций.

Недостаток двоичной системы – быстрый рост числа разрядов, необходимых для записи чисел.

6.2. Способы преобразования данных одной системы счисления в другую

Всякий раз, когда используется для вычислений система счисления, отличная от фактической, необходимо выполнить перевод $10 \Rightarrow p$, $p \Rightarrow 10$. Есть системы, дающие значительно более высокие скорости, но и требующие большего количества оборудования. Этот перевод может быть выполнен вручную или на ЭВМ (с помощью специальных программ).

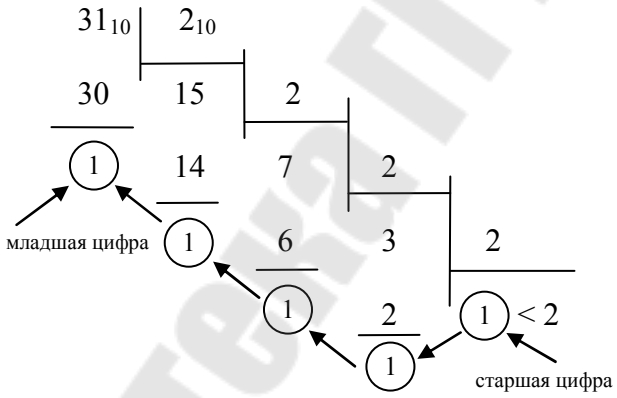
Во всех этих случаях принципиально используется различные подходы и методы. В связи с тем, что нам придется готовить информацию для программы вручную, мы рассмотрим, прежде всего, методы, направленные на ручной перевод. Итак, имеем дело с позиционной системой счисления с основанием p, с естественными весами разрядов. В качестве промежуточной используется десятичная система. Вначале число переводится из системы p в десятеричную, затем из десятеричной в систему с нужным основанием. Мы отступим от этого правила и воспользуемся алгоритмом непосредственного перевода из системы с основанием p в систему с основанием q. Обычно произвольное число, содержащее целую и дробную части, переводят по частям: вначале целую, затем дробную часть.

Рассмотрим перевод целых чисел.

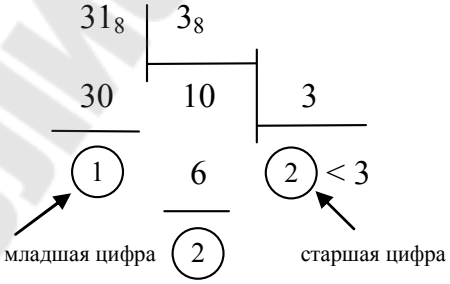
Перевод осуществляется по следующему правилу: исходное число, записанное в системе с основанием p, и его частные последовательно делятся на число q, представленное в системе p. Деление производится в системе с основанием p и продолжается до получения результата, меньшего q. Первый остаток, меньший q, дает старшую цифру числа N_q. Остатки от деления дают остальные цифры числа N_q.

Примеры

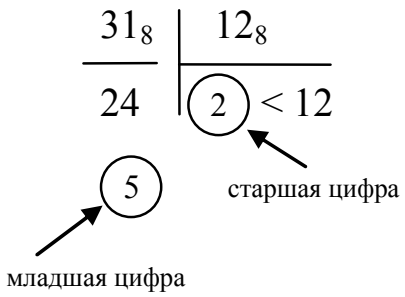
1. $31_{10} \Rightarrow 2$; $31_{10} = 11111_2$.



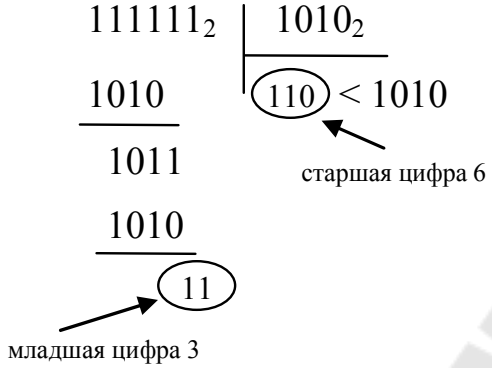
2. $31_8 \Rightarrow 3$; $31_8 = 221_3 = 2 \cdot 3^2 + 2 \cdot 3^1 + 1 \cdot 3^0 = 18 + 6 + 1 = 25_{10}$.



3. $31_8 \Rightarrow 10$; $31_8 = 25_{10}$.



4. $111111_2 \Rightarrow 10$; $111111_2 = 63_{10}$.



Для перевода правильной десятичной дроби F в систему счисления с основанием q необходимо F умножить на q, записанное в той же десятичной системе, затем дробную часть полученного произведения снова умножить на q и т. д. до тех пор, пока дробная часть очередного произведения не станет равной нулю, либо не будет достигнута требуемая точность изображения числа F в q системе. Представлением дробной части числа F в новой системе счисления будет последовательность целых частей полученных произведений, записанных в порядке их получения и изображенных одной q цифрой. Если требуемая точность перевода числа F составляет k знаков после запятой, то предельная абсолютная погрешность при этом равняется $q^{-(k+1)} / 2$.

Примеры

1. $0{,}5314_8 \Rightarrow 5$; $0{,}5314_8 = 0{,}3141..._5$.

0,	$5314_8 \cdot 5_8$
3	2774 · 5
1	6754 · 5
4	2634 · 5
1	6014

2. $0{,}318_{10} => 2$; $0{,}318_{10} = 0{,}01010..._2$.

0,	$318_{10} \cdot 2_{10}$
0	636 · 2
1	272 · 2
0	544 · 2
1	088 · 2
0	176

Для чисел, имеющих как целую, так и дробную части, перевод из десятичной системы счисления в другую осуществляется отдельно для целой и дробной частей по правилам, указанным выше.

Для перевода чисел из одной системы счисления в другую, когда одно основание является целой степенью другого, следует придерживаться следующей методики.

Как мы уже знаем, в ЭВМ наибольшее применение находит система с основаниями 2, 4, 8, 16, т. е. системы, которые кратны степени 2. Поэтому целесообразно рассмотреть лишь правила перевода чисел в этих системах. Аналогичные правила будут справедливы и для других систем. Допустим, что имеется некоторое целое число N_8 в восьмеричной системе. Оно может быть представлено в виде:

$$N_8 = a_1 \cdot 8^{n-1} + a_2 \cdot 8^{n-2} + a_3 \cdot 8^{n-3} + \ldots + a_{n-2} \cdot 8^2 + a_{n-1} \cdot 8^1 + a_n \cdot 8^0.$$

Пусть каким-либо образом мы получили запись этого числа в виде двоичного:

$$N_2 = b_1 \cdot 2^{k-1} + b_2 \cdot 2^{k-2} + \ldots + b_{k-2} \cdot 2^2 + b_{k-1} \cdot 2^1 + b_k \cdot 2^0.$$

Разделим эти выражения на $2^3 = 8$:

$$a_1 \cdot 8^{n-2} + a_2 \cdot 8^{n-3} + a_3 \cdot 8^{n-4} + \ldots + a_{n-1} \cdot 8^0 + a_n \cdot 8^{-1}.$$

дробная часть

$$b_1 \cdot 2^{k-4} + b_2 \cdot 2^{k-5} + \ldots + b_{k-3} \cdot 2^0 + b_{k-2} \cdot 2^{-1} + b_{k-1} \cdot 2^{-2} + b_k \cdot 2^{-3}.$$

дробная часть

Так как числа были равны, то получается одинаковые частные и одинаковые остатки:

$$a_n \cdot 8^{-1} = b_{k-2} \cdot 2^{-1} + b_{k-1} \cdot 2^{-2} + b_k \cdot 2^{-3}. \qquad (2)$$

Если снова разделим целые части на $2^3 = 8$, то опять получим равные частные и равные остатки. При этом видим, что каждой вось-

меричной цифре соответствует ее двоичный эквивалент. Поэтому перевод выполняется простой заменой цифры восьмеричной системы ее двоичным эквивалентом и обратно.

Пример

$$62{,}753_8 = 110010{,}111101011_2.$$

Аналогично для четверичной системы:

$$321{,}2233_4 = 111001{,}10101111_2.$$

Аналогично для шестнадцатеричной системы:

$$1D876{,}72 = 00011101100001110110{,}01110010_2.$$

Из данных примеров видно, что чем выше основание системы счисления, тем компактнее запись:

$$\left.\begin{matrix} b_{k-2} \\ b_{k-1} \\ b_k \end{matrix}\right\} = \begin{cases} 0 \\ 1 \end{cases}$$

Если умножить последние соотношения (2) на 8, то

$$a_n \cdot 8^{-1} \cdot 8 = (b_{k-2} \cdot 2^{-1} + b_{k-1} \cdot 2^{-2} + b_k \cdot 2^{-3}) \cdot 2^3;$$

$$a_n = b_{k-2} \cdot 2^2 + b_{k-1} \cdot 2^1 + b_k \cdot 2^0.$$

Исходя из вышеизложенного, можно сделать вывод, что возможно составить универсальный алгоритм перевода чисел из любой системы счисления в любую по желанию пользователя.

7. Представление чисел в ЭВМ

Числа в ЭВМ по своей сути подразделяются на целые числа и вещественные. Целые числа со знаком обычно занимают в памяти компьютера один, два или четыре байта, при этом самый левый (старший) разряд содержит информацию о знаке числа (табл. 5).

Таблица 5

Диапазоны значений целых чисел со знаком

Формат числа в байтах	**Диапазон**	
	Запись с порядком	**Обычная запись**
1	$-2^7 \ldots 2^7 - 1$	–128...127
2	$-2^{15} \ldots 2^{15} - 1$	–32768...32767
4	$-2^{31} \ldots 2^{31} - 1$	–2147483648...2147483647

Рассмотрим особенности записи целых чисел со знаком на примере однобайтового формата, при котором для знака отводится один разряд, а для цифр абсолютной величины – семь разрядов. В компьютерной технике применяются три формы записи (кодирования) целых чисел со знаком: прямой код, обратный код, дополнительный код. Последние две формы применяются особенно широко, т. к. позволяют упростить конструкцию арифметико-логического устройства компьютера путем замены разнообразных арифметических операций операцией сложения.

Положительные числа в прямом, обратном и дополнительном кодах изображаются одинаково – двоичными кодами с цифрой 0 в знаковом разряде (старшем).

Отрицательные числа в прямом, обратном и дополнительном кодах имеют разное изображение.

Прямой код. В знаковый разряд помещается цифра 1, а в разряды цифровой части числа – двоичный код его абсолютной величины (рис. 7).

Рис. 7. Прямой код в двоичной системе счисления

Обратный код. Получается инвертированием всех цифр двоичного кода абсолютной величины числа, включая разряд знака: нули заменяются единицами, а единицы – нулями (рис. 8).

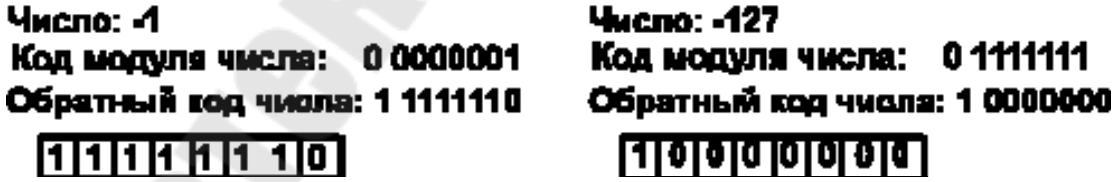


Рис. 8. Обратный код в двоичной системе счисления

Дополнительный код. Получается образованием обратного кода с последующим прибавлением единицы к его младшему разряду (рис. 9).

Рис. 9. Дополнительный код в двоичной системе счисления

Обычно отрицательные десятичные числа при вводе в машину автоматически преобразуются в обратный или дополнительный двоичный код и в таком виде хранятся, перемещаются и участвуют в операциях. При выводе таких чисел из машины происходит обратное преобразование в отрицательные десятичные числа.

Система вещественных чисел в математических вычислениях предполагается непрерывной и бесконечной, т. е. не имеющей ограничений на диапазон и точность представления чисел. Однако в ЭВМ числа хранятся в регистрах и ячейках памяти с ограниченным количеством разрядов. Вследствие этого система вещественных чисел, представляемых в машине, является дискретной (прерывной) и конечной. При написании вещественных чисел в программах вместо привычной запятой принято ставить точку. Для отображения вещественных чисел, которые могут быть как очень маленькими, так и очень большими, используется форма записи чисел с порядком основания системы счисления. Например, десятичное число 1,25 в этой форме можно представить следующим образом:

$$1{,}25 \cdot 10^0 = 0{,}125 \cdot 10^1 = 0{,}0125 \cdot 10^2 = \ldots$$

или

$$12{,}5 \cdot 10^{-1} = 125{,}0 \cdot 10^{-2} = 1250{,}0 \cdot 10^{-3} = \ldots .$$

Любое число N в системе счисления с основанием q можно записать в виде $N = M \cdot q^p$, где M – множитель, содержащий все цифры числа (мантисса), а p – целое число, называемое порядком. Такой способ записи чисел называется представлением числа с «плавающей» точкой. Если «плавающая» точка расположена в мантиссе перед первой значащей цифрой, то при фиксированном количестве разрядов, отведенных под мантиссу, обеспечивается запись максимального количества значащих цифр числа, т. е. максимальная точность представления числа в машине.

Мантисса должна быть правильной дробью, у которой первая цифра после точки (запятой в обычной записи) отлична от нуля: $0{,}1_2 <= |M| < 1$. Если это требование выполнено, то число называется нормализованным. Мантиссу и порядок q-го числа принято записывать в системе с основанием q, а само основание – в десятичной системе. Примеры нормализованного представления приведены в табл. 6.

Таблица 6

Десятичная система	Двоичная система
$753{,}15 = 0{,}75315 \cdot 10^{3}$	$-101{,}01 = -0{,}10101 \cdot 2^{11}$ (порядок $11_2 = 3_{10}$)
$-0{,}000034 = -0{,}34 \cdot 10^{-4}$	$0{,}000011 = 0{,}11 \cdot 2^{-100}$ (порядок $-100_2 = -4_{10}$)

Вещественные числа в ЭВМ различных типов записываются поразному, тем не менее все компьютеры поддерживают несколько международных стандартных форматов, различающихся по точности, но имеющих одинаковую структуру (рис. 10).

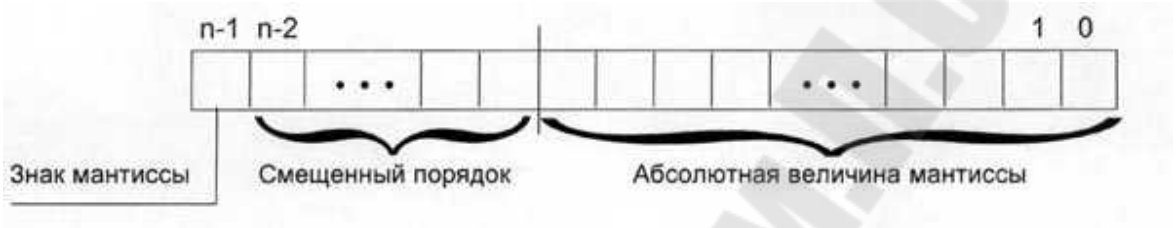


Рис. 10. Структура числа в ЭВМ

Здесь порядок *n*-разрядного нормализованного числа задается в так называемой смещенной форме: если для задания порядка выделено *k* разрядов, то к истинному значению порядка, представленного в дополнительном коде, прибавляют смещение, равное $2^{k-1} - 1$. Например, порядок, принимающий значения в диапазоне от –128 до +127, представляется смещенным порядком, значения которого меняются от 0 до 255. Использование смещенной формы позволяет производить операции над порядками как над беззнаковыми числами, что упрощает операции сравнения, сложения и вычитания порядков, а также упрощает операцию сравнения самих нормализованных чисел.

Чем больше разрядов отводится под запись мантиссы, тем выше точность представления числа. Чем больше разрядов занимает порядок, тем шире диапазон от наименьшего отличного от нуля числа до наибольшего числа, представимого в машине при заданном формате.

Стандартные форматы представления вещественных чисел:

1) одинарный – 32-разрядное нормализованное число со знаком, 8-разрядным смещенным порядком и 24-разрядной мантиссой (старший бит мантиссы, всегда равный 1, не хранится в памяти, и размер поля, выделенного для хранения мантиссы, составляет только 23 разряда);

2) двойной – 64-разрядное нормализованное число со знаком, 11-разрядным смещенным порядком и 53-разрядной мантиссой (старший

бит мантиссы не хранится, размер поля, выделенного для хранения мантиссы, составляет 52 разряда);

3) расширенный – 80-разрядное число со знаком, 15-разрядным смещенным порядком и 64-разрядной мантиссой. Позволяет хранить ненормализованные числа.

8. Режимы работы ЭВМ

ЭВМ могут выполнять обработку информации в разных режимах (рис. 11):

1) однопрограммный (монопольный) режим;

2) многопрограммный режим, который можно подразделить:

– на режим пакетной обработки информации;

– на режим разделения машинного времени.

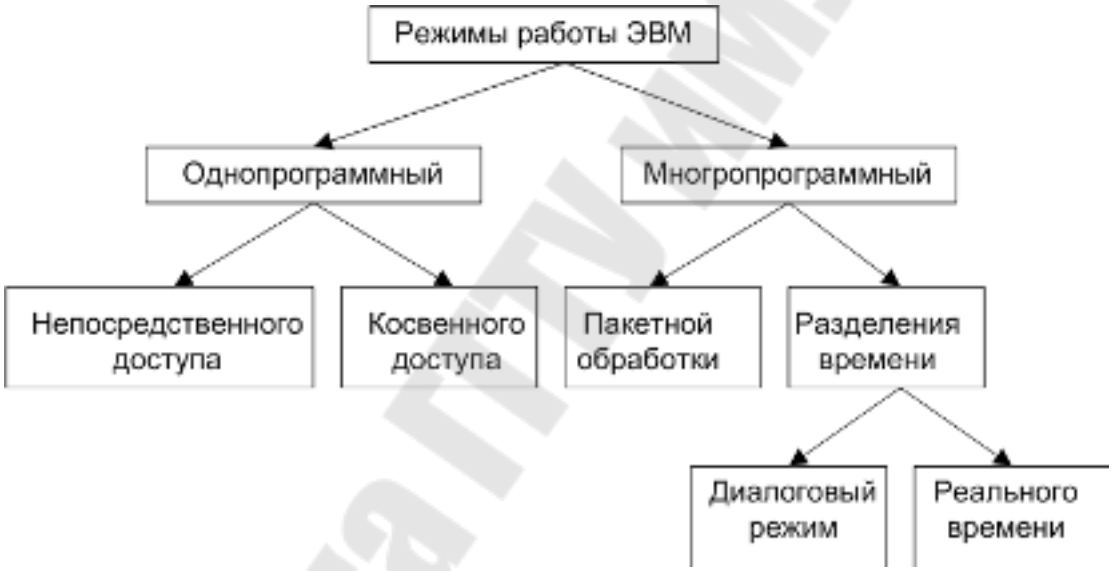


Рис. 11. Классификация режимов работы компьютеров

Режим разделения времени имеет разновидности: диалоговый режим и режим реального времени.

Однопрограммный режим использования ЭВМ – самый простой и применяется во всех поколениях компьютеров. Из современных машин этот режим чаще всего используется в персональных компьютерах, где он называется реальным режимом работы микропроцессора. В этом режиме все ресурсы ПК передаются одному пользователю. Пользователь сам готовит и машину и всю необходимую для решения задач информации, загружает программу и данные, непосредственно наблюдает за ходом решения задачи и выводом результатов. Такой вариант называют режимом непосредственного доступа.

Однопрограммный режим имеет и второй вариант – косвенного доступа, при котором пользователь не имеет непосредственного контакта с компьютером. В этом варианте пользователь готовит свое задание и отдает его на обработку. Задача запускается в порядке очередности, и по мере готовности результаты решения задачи выдаются пользователю. Этот вариант, бывший когда-то самым распространенным, сейчас практически, по крайней мере на персональных компьютерах, не используется. Однопрограммный режим непосредственного доступа весьма удобен для пользователя, но для него характерен чрезвычайно низкий коэффициент загрузки оборудования – временные простои многих устройств машины и в период подготовки задачи для решения и непосредственно при решении задачи (при вычислениях в процессоре простаивают внешние устройства, при печати простаивает процессор, основная и внешняя память и т. д.). Поэтому даже в современных ПК, для которых характерен именно однопрограммный режим (в силу их «персональности»), последний в микропроцессорах обогащается многоступенчатой суперконвейерной обработкой данных, использующей некоторые элементы многопрограммности.

Многопрограммный (его также называют мультипрограммным, многозадачным, а в ПК и многопользовательским) режим обеспечивает лучшее расходование ресурсов компьютера, но несколько ущемляет интересы пользователя. Для реализации этого режима необходимо, прежде всего, разделение ресурсов машины в пространстве (на множество устройств компьютера) и во времени. Естественно, такое разделение ресурсов эффективно может выполнятся только автоматически, следовательно, требуется автоматическое управление вычислениями. Автоматическое управление особо необходимо для распределения памяти между несколькими одновременно запускаемыми программами, поскольку программы готовятся пользователями независимо друг от друга, в них не выполняется предварительно статическое распределение памяти (как и других программных и технических ресурсов машины). В процессе решения задач недопустимо одновременное обращение двух программ к одному и тому же файлу или устройству.

Все названные проблемы решают операционные системы, обеспечивающие многопрограммную работу компьютера, помогают им в этом драйверы устройств машины и автозагрузчики (загрузчики) программ.

Важнейшая проблема – *защита памяти*. Недопустимо несанкционированное, пусть и неодновременное, обращение двух программ

к одним и тем же областям памяти для изменения информации. Для предотвращения такого несанкционированного случайного доступа к памяти, выделенной для другой задачи, служит специальная операционная система защиты памяти. Важность проблемы защиты памяти подчеркивается тем фактом, что многопрограммный режим работы микропроцессоров ПК обычно называют *защищенным режимом*.

Простейшим вариантом многопрограммного режима является *режим пакетной обработки*. Он в максимальной степени обеспечивает загрузку всех ресурсов машины, но наименее удобен пользователю. В классических системах пакетной обработки информации все подлежащие решению задачи анализировались и объединялись в различные группы (пакеты) с тем, чтобы в пределах пакета обеспечивалось равномерная загрузка всех устройств машины. Например, задача, связанная с длительным выводом информации на печать, объединялась с задачей, интенсивно использующей внешнюю память, и с задачей, требующей сложных вычислений в процессоре и т. п. После формирования всех пакетов они по очереди запускались на обработку. Пользователь в этом режиме обращался к машине два раза: первый раз для ввода задания, второй раз для получения результатов – по современной терминологии такой режим относится к режимам группы «offline».

В ПЭВМ из-за небольшого количества одновременно решаемых задач режим пакетной обработки претерпел существенные изменения и сводится, по существу, к последовательному решению одновременно поступающих задач (пакета задач) в соответствии с их важностью (приоритетностью) и времени поступления. Переход к решению следующей задачи выполняется только после окончательного завершения текущей. Правда, в развитых системах такой пакетной обработки при внезапном поступлении информации по более приоритетной задаче выполняемая на компьютере менее приоритетная задача уступает место (прерывается).

Второй частный случай многопрограммного режима – *режим разделения времени* – характерен тем, что на машине действительно одновременно решается несколько задач, каждой из которых по очереди выделяются кванты времени, обычно недостаточные для полного решения задачи. Условием прерывания решения текущей задачи служит либо истечение кванта выделенного времени, либо обращение к процессору какого-либо приоритетного внешнего устройства, например, клавиатуры для ввода информации.

Прерывание задачи от клавиатуры является типичным для диалогового режима работы ПК, представляющего собой частный случай режима разделения времени. Диалоговые режимы характерны для многопользовательских систем: они обеспечивают одновременную работу нескольких пользователей при решении задач в интерактивном режиме. В процессе решения задачи пользователь имеет возможность корректировать ход выполнения своего задания. Диалоговые системы активно используются при совместной работе нескольких пользователей даже с одной программой: формирование и корректировка баз данных, программ, чертежей, схем и документов.

Режим реального времени – еще один вариант режима с разделением машинного времени. Этот режим применяется в основном в динамических системах управления и диагностики, когда строго регламентируется время ответа системы (выполнения задания) на случайно поступающие запросы. Все режимы разделения машинного времени обеспечивают пользователю работу в режиме «online».

Основная нагрузка при реализации многопрограммных режимов, как уже говорилось, ложится на операционную систему. Все операционные системы обеспечивают выполнения этих режимов. Все современные операционные системы обладают эффективными возможностями, поддерживающими не только многозадачные и многопользовательские режимы с развитой системой приоритетного прерывания, но и многопроцессорность их исполнения, т. е. распределение заданий между несколькими микропроцессорами, имеющимися в системе.

9. Архитектура персональной ЭВМ

Конфигурацию ПЭВМ можно изменять по мере необходимости. Но существует понятие базовой конфигурации, которую можно считать типичной: системный блок, монитор, клавиатура, мышка и набор периферийных устройств.

ПЭВМ выпускаются и в портативном варианте (laptop- или notebook-исполнении). В этом случае системный блок, монитор и клавиатура размещены в одном корпусе: системный блок находится под клавиатурой, а монитор встроен в крышку.

Системный блок – основная составляющая ПК, в середине которой находятся важнейшие компоненты. Устройства, находящиеся в середине системного блока называют внутренними, а устройства, подсоединенные извне, называют внешними. Внешние дополнительные устройства, предназначенные для ввода и вывода информации, называются также *периферийными*.

По внешнему виду системные блоки отличаются формой корпуса, который может быть горизонтального (desktop) или вертикального (tower) выполнения. Корпусы вертикального выполнения могут быть полноразмерными (BigTower), среднеразмерными (MidiTower), малоразмерными (MiniTower). Корпусы горизонтального выполнения бывают двух форматов: узкие (Full AT) и очень узкие (Baby AT). Корпусы персональных компьютеров имеют разные конструкторские особенности и дополнительные элементы (элементы блокировки несанкционированного доступа, средства контроля внутренней температуры, шторки от пыли).

Корпусы поставляются вместе с блоком питания, мощность которого является одним из важнейших параметров корпуса. Для массовых моделей достаточной является мощность 300–500 Вт.

Персональный компьютер типа IBM PC имеет довольно традиционную архитектуру микропроцессорной системы и содержит все обычные функциональные узлы: процессор, постоянную и оперативную память, устройства ввода/вывода, системную шину, источник питания и т. п. (рис. 12). Основные особенности архитектуры ПЭВМ сводятся к принципам компоновки аппаратуры, а также к выбранному набору системных аппаратных средств.

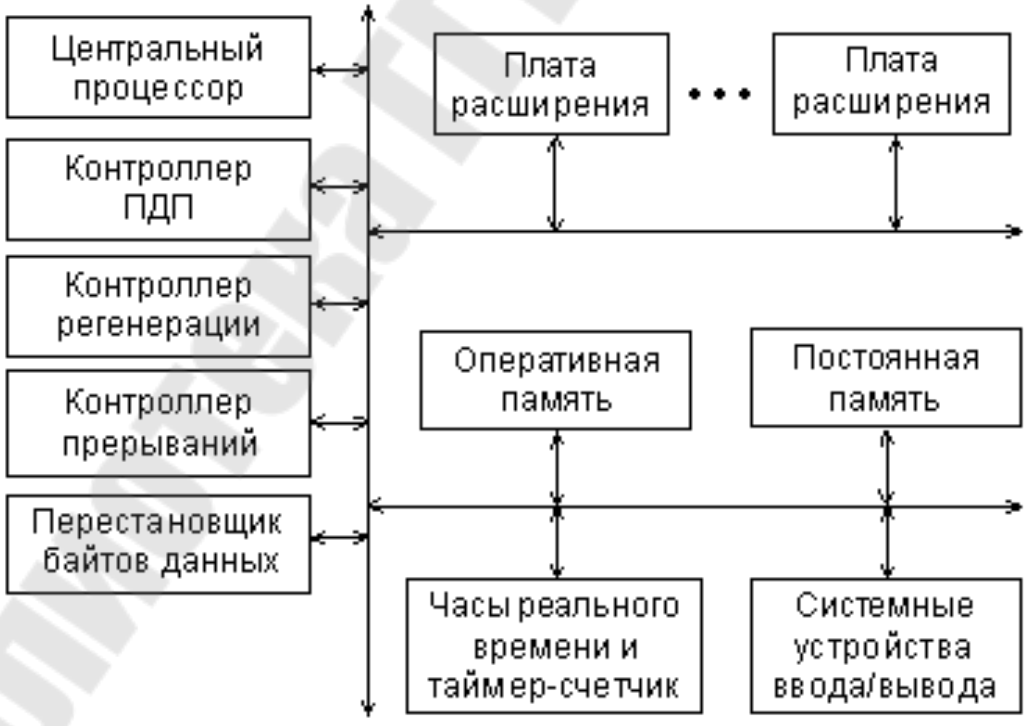


Рис. 12. Архитектурная реализации ПЭВМ типа IBM PC

Функции основных узлов ПЭВМ:

1. **Центральный процессор** – это микропроцессор со всеми необходимыми вспомогательными микросхемами, включая внешнюю кэш-память и контроллер системной шины. В большинстве случаев именно центральный процессор осуществляет обмен по системной шине.

2. **Оперативная память** (ОЗУ) может занимать почти все адресуемое пространство памяти процессора. Однако чаще всего ее объем гораздо меньше. В современных персональных компьютерах стандартный объем системной памяти составляет, как правило, от 256 до 4098 Мбайт. Оперативная память компьютера выполняется на микросхемах динамической памяти и поэтому требует регенерации.

3. **Постоянная память** (ROM BIOS – Base Input/Output System) имеет небольшой объем (до 128 Кбайт), содержит программу начального запуска, описание конфигурации системы, а также драйверы (программы нижнего уровня) для взаимодействия с системными устройствами.

4. **Контроллер прерываний** преобразует аппаратные прерывания системной магистрали в аппаратные прерывания процессора и задает адреса векторов прерывания. Все режимы функционирования контроллера прерываний задаются программно процессором перед началом работы.

5. **Контроллер прямого доступа к памяти** (контроллер ПДП) принимает запрос на ПДП из системной магистрали, передает его процессору, а после предоставления процессором магистрали производит пересылку данных между памятью и устройством ввода/вывода. Все режимы функционирования контроллера ПДП задаются программно процессором перед началом работы. Использование встроенных в компьютер контроллеров прерываний и ПДП позволяет существенно упростить аппаратуру применяемых плат расширения.

6. **Контроллер регенерации** осуществляет периодическое обновление информации в динамической оперативной памяти путем проведения по шине специальных циклов регенерации. На время циклов регенерации он становится хозяином (задатчиком) шины.

7. **Перестановщик байтов данных** помогает производить обмен данными между 16-разрядным и 8-разрядным устройствами, пересылать целые слова или отдельные байты.

8. **Часы реального времени** и **таймер-счетчик** – это устройства для внутреннего контроля времени и даты, а также для программной выдержки временных интервалов, программного задания частоты и т. д.

9. **Системные устройства ввода/вывода** – это те устройства, которые необходимы для работы компьютера и взаимодействия со стандартными внешними устройствами по параллельному и последовательному интерфейсам. Они могут быть выполнены на материнской плате, а могут располагаться на платах расширения.

10. **Платы расширения** устанавливаются в слоты (разъемы) системной магистрали и могут содержать ОЗУ и устройства ввода/вывода. Они могут обмениваться данными с другими устройствами на шине в режиме программного обмена, в режиме прерываний и в режиме ПДП. Предусмотрена также возможность захвата шины, т. е. полного отключения от шины всех системных устройств на некоторое время.

Важная особенность подобной архитектуры – ее открытость, т. е. возможность включения в ЭВМ дополнительных устройств, причем как системных устройств, так и разнообразных плат расширения. Открытость предполагает также возможность простого встраивания программ пользователя на любом уровне программного обеспечения компьютера.

Первый компьютер семейства IBM PC XT, получивший широкое распространение, был выполнен на базе оригинальной системной магистрали PC XT-Bus. В дальнейшем (начиная с IBM PC AT) она была доработана до магистрали, ставшей стандартной и получившей название ISA (Industry Standard Architecture). До недавнего времени ISA оставалась основой компьютера. Однако, начиная с появления процессоров i486 (в 1989 г.), она перестала удовлетворять требованиям производительности, и ее стали дублировать более быстрыми шинами VLB (VESA Local Bus) и PCI (Peripheral Component Interconnect bus) или заменять совместимой с ISA магистралью EISA (Enhanced ISA). Постепенно шина PCI вытеснила конкурентов и стала фактическим стандартом, а начиная с 1999 г. в новых ПЭВМ решили отказываться от магистрали ISA, оставляя только PCI. Правда, при этом приходится отказываться от применения плат расширения, разработанных за долгие годы для подключения к магистрали ISA.

Другое направление совершенствования архитектуры ПЭВМ связано с максимальным ускорением обмена информацией с системной памятью. Именно из системной памяти компьютер читает все исполняемые команды и в системной же памяти он хранит данные. То есть больше всего обращений процессор совершает именно к памяти. Ускорение обмена с памятью приводит к существенному ускорению работы всей системы в целом. Но при использовании для обмена с

памятью системной магистрали приходится учитывать скоростные ограничения магистрали.

Разработчиками был предложен следующий подход. Системная память подключается не к системной магистрали, а к специальной высокоскоростной шине, находящейся «ближе» к процессору, не требующей сложных буферов и больших расстояний. В таком случае обмен с памятью идет с максимально возможной для данного процессора скоростью, и системная магистраль не замедляет его. Особенно актуальным это становится с ростом быстродействия процессора (сейчас тактовые частоты процессоров персональных компьютеров достигают 2–5 ГГц).

Таким образом структура персонального компьютера из одношинной, применявшейся только в первых компьютерах, становится трехшинной (рис. 13).

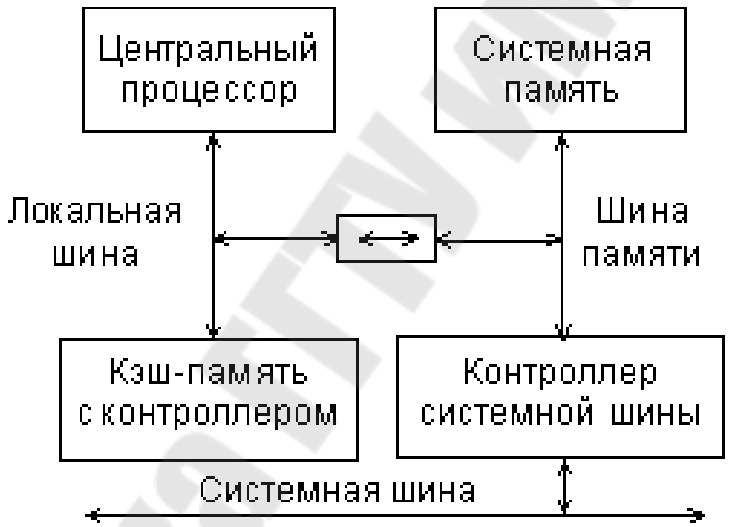


Рис. 13. Организация связей в случае трехшинной архитектуры

Все три шины имеют адресные линии, линии данных и управляющие сигналы. Но состав и назначение линий этих шин не совпадают между собой, хотя они и выполняют одинаковые функции. С точки зрения процессора, системная шина (магистраль) в системе всего одна, по ней он получает данные и команды и передает данные как в память, так и в устройства ввода/вывода.

Временные задержки между системной памятью и процессором в данном случае минимальны, т. к. локальная шина и шина памяти соединены только простейшими быстродействующими буферами. Еще меньше задержки между процессором и кэш-памятью, подключаемой

непосредственно к локальной шине процессора и служащей для ускорения обмена процессора с системной памятью.

Если в компьютере применяются две системные шины и более, например, ISA и PCI, то каждая из них имеет свой собственный контроллер шины, и работают они параллельно, не влияя друг на друга. Тогда получается уже четырехшинная, а в современных ПЭВМ – многошинная структура. Пример такой структуры компьютера приведен на рис. 14.

Рис. 14. Пример многошинной структуры организации ПЭВМ

В наиболее распространенных настольных компьютерах класса Desk-top в качестве конструктивной основы используется системная или материнская плата (motherboard), на которой располагаются все основные системные узлы компьютера, а также несколько разъемов (слотов) системной шины для подключения дочерних плат – плат расширения (интерфейсных модулей, контроллеров, адаптеров). Как правило, современные системные платы допускают замену процессора, выбор его тактовой частоты, замену и наращивание оперативной памяти, выбор режимов работы других узлов.

9.1. Системные платы

Системная плата (System board, SB) или **объединительная**, **материнская плата** (Mother board, MB) – это важнейшая часть ПЭВМ, содержащая его основные электронные компоненты. С помо-

щью МВ осуществляется взаимодействие между большинством устройств
машины.

Конструктивно МВ настольного ПЭВМ представляет собой печатную плату площадью 100–150 см^2, на которой размещается большое число различных микросхем, разъемов и других элементов. Существуют две основные разновидности конструкций системной платы:

1. На плате жестко закреплены все необходимые для работы микросхемы – сейчас такие платы используются лишь в простейших домашних компьютерах, называемых одноплатными.

2. Непосредственно на системной плате размещается минимальное количество микросхем, а все остальные компоненты объединяются при помощи системной шины и конструктивно устанавливаются на дополнительных платах (платах расширения*)*, устанавливаемых в специальные разъемы (слоты), имеющиеся на материнской плате. Компьютеры, использующие такую технологию, относятся к вычислительным системам с шинной архитектурой.

На системной плате непосредственно расположены:

1) разъем для подключения микропроцессора;

2) набор системных микросхем (чипсет, chipset), обеспечивающих работу микропроцессора и других узлов машины;

3) микросхема постоянного запоминающего устройства, содержащего программы базовой системы ввода/вывода (Basic Input-Output System – BIOS);

4) микросхема энергозависимой памяти (питается от автономного аккумулятора, расположенного на МВ) по типу используемых в ней полевых транзисторов называемая CMOS;

5) микросхемы кэш-памяти 2-го уровня (если они отсутствуют на плате микропроцессора) или 3-го уровня;

6) разъемы для подключения моделей оперативной памяти;

7) наборы микросхем и разъемы для системных, локальных и периферийных интерфейсов;

8) микросхемы мультимедийных устройств и т. д.

Существуют различные базовые типоразмеры (форм-факторы) плат. Форм-фактор определяет не только внешние размеры системных плат, но и ряд специфических параметров, характеризующих функциональные и эксплуатационные свойства МВ:

1. **Full-size AT** размером 12 × 13,8 дюйма (использовались в первых моделях IBM PC, сейчас не выпускаются).

2. **Baby AT** – имеют размеры 8,57 × 13,04 дюйма. Их разновидность Mini AT – 8,57 × 9,85 дюйма; они могут устанавливаться во все корпуса, кроме Slim line (сейчас не выпускаются).

3. **Полномасштабные платы AT** – отличаются от Baby AT только размером (ширина составляет 12 дюймов, что затрудняет установку в корпус).

4. **LPX** и **Mini LPX** размерами 9 × 13 и 8,2 × 10,4 дюйма соответственно – устанавливаются в сверхнизкие корпуса Slim Line.

5. **ATX** – самый популярный формат системной платы размером 9,6 × 12 дюймов, отличающийся от Baby AT более удобным расположением элементов на плате (позволяет легко без снятия платы менять ее элементы), лучшей вентиляцией (не требуется устанавливать отдельный вентилятор на микропроцессор), наличием разъема универсальной шины USB и возможностью дистанционного выключения питания компьютера с модема или по локальной сети. На плате установлены разъемы только под модули оперативной памяти DRAM.

6. В 1997 г. была представлена модификация **MiniATX** – СП размером 9,6 × 9,6 дюймов.

7. В 2002 г. Intel анонсировала плату формата **BTX** (Balanced Technology eXtended – расширенная сбалансированная технология), призванную начиная с 2005 г. активно заменять плату ATX, поскольку BTX имеет обособленные области для разного типа комплектующих и линейное расположение компонентов, что обеспечивает меньшие взаимные тепловые и электромагнитные влияния компонентов друг на друга, лучшее охлаждение, низкий уровень шума системы охлаждения и более удобную раскладку этих компонентов для построения малогабаритных систем. Спецификация BTX 1.0 оговаривает три размера материнских плат: BTX, Micro BTX и Pico BTX. Ширина плат BTX составляет 325 мм, Micro BTX – 264 мм и Pico BTX – 203 мм. На платах BTX используется термальный модуль, который позволит сохранять на низком уровне температуры ключевых компонентов, включая процессор Pentium 4 Prescott, выделяющий большое количество тепла, системный и графический чипсеты.

Системные платы поддерживающие разные виды интерфейсных систем, локальных и периферийных шин, количества слотов для этих шин, имеющихся на СП, существенно зависит эффективность работы ПК в целом.

В современных МВ используются, как правило, микросхемы Flash BIOS, программы в которых могут перезаписываться при по-

мощи специальных средств, что облегчает модернизацию этих программ при появлении новых устройств, которым нужно обеспечить поддержку (например, новых типов микросхем оперативной памяти). У Flash BIOS есть один существенный недостаток: существует много вирусов, которые, попав в систему, просто стирают все содержимое Flash BIOS, после чего системная плата выходит из строя. От вирусов можно защититься только одним способом – в System Setup запретить перезапись содержимого BIOS. Если эта установка активизирована, то ни один вирус ничего сделать не сумеет.

Важным компонентом, размещенным на системной плате (часто в системном чипсете), является микросхема CMOS-памяти. Она питается от своего локального аккумулятора (батарейки) и поэтому является энергозависимой (сохраняет информацию при отключении компьютера от сети). Название CMOS произошло от названия технологии исполнения ее элементов. Память хранит информацию о параметрах многих устройств, входящих в ПК.

9.2. Центральный микропроцессор

Центральный процессор – это главная микросхема компьютера, его «мозг». Он разрешает выполнять программный код, находящийся в памяти, и руководит работой всех устройств компьютера. Скорость его работы определяет быстродействие компьютера. Конструктивно процессор – это кристалл кремния очень маленьких размеров. Процессор имеет специальные ячейки, которые называются регистрами. Именно в регистрах помещаются команды, которые выполняются процессором, а также данные, которыми оперируют команды. Работа процессора состоит в выборе из памяти в определенной последовательности команд и данных и их выполнении. На этом и базируется выполнение программ.

В ЭВМ обязательно должен присутствовать центральный процессор (Central Rrocessing Unit – CPU), который выполняет все основные операции. Часто ПЭВМ оснащен дополнительными сопроцесорами, ориентированными на эффективное выполнение специфических функций, такими как математический сопроцесор для обработки числовых данных в формате с «плавающей» точкой, графический сопроцесор для обработки графических изображений, сопроцесор ввода/вывода для выполнения операции взаимодействия с периферийными устройствами.

Основными параметрами процессоров являются: тактовая частота, разрядность, рабочее напряжение, коэффициент внутреннего умножения тактовой частоты, размер кэш-памяти.

Тактовая частота определяет количество элементарных операций (тактов), выполняемых процессором за единицу времени. Тактовая частота современных процессоров измеряется в ГГц (1 Гц соответствует выполнению одной операции за одну секунду, 1 МГц = 106 Гц). Чем больше тактовая частота, тем больше команд может выполнить процессор и тем больше его производительность.

Разрядность процессора показывает, сколько бит данных он может принять и обработать в своих регистрах за один такт. Разрядность процессора определяется разрядностью командной шины, т. е. количеством проводников в шине, по которой передаются команды. Современные распространенные процессоры могут быть как 32-разрядными, так и 64-разрядными.

Рабочее напряжение процессора обеспечивается материнской платой, поэтому разным маркам процессоров отвечают разные материнские платы. Рабочее напряжение современных массовых процессоров не превышает 3 В. Снижение рабочего напряжения разрешает уменьшить размеры процессоров, а также уменьшить тепловыделение в процессоре, что разрешает увеличить его производительность без угрозы перегрева.

Коэффициент внутреннего умножения тактовой частоты – это коэффициент, на который следует умножить тактовую частоту материнской платы для достижения частоты процессора. Тактовые сигналы процессор получает от материнской платы, которая из чисто физических причин не может работать на таких высоких частотах, как процессор. На сегодня тактовая частота материнских плат составляет 100, 133, 150, 200, 266, 400 и 800 МГц.

Кэш-память. Обмен данными внутри процессора происходит намного быстрее, чем обмен данными между процессором и оперативной памятью. Поэтому, для того чтобы уменьшить количество обращений к оперативной памяти, внутри процессора создают так называемую сверхоперативную или кэш-память. Когда процессору нужны данные, он сначала обращается к кэш-памяти, и только тогда, когда там отсутствуют нужные данные, происходит обращение к оперативной памяти. Чем больше размер кэш-памяти, тем больше вероятность, что необходимые данные находятся там. Поэтому высокопроизводительные процессоры имеют повышенные объемы кэш-памяти. Различают кэш-память первого уровня (выполняется на одном кристалле с процессором) и второго уровня (выполняется на отдельном кристалле, но в границах процессора) и третьего уровня (выполняется на

отдельных быстродействующих микросхемах с расположением на материнской плате и имеет объем 1 и более Мб).

В процессе работы процессор обрабатывает данные, находящиеся в его регистрах, оперативной памяти и внешних портах процессора. Часть данных интерпретируется как собственно данные, часть данных – как адресные данные, а часть – как команды. Совокупность разнообразных команд, которые может выполнить процессор над данными, образует систему команд процессора. Чем больше набор команд процессора, тем сложнее его архитектура, тем длиннее запись команд в байтах и тем дольше средняя продолжительность выполнения команд.

Как уже упоминалось выше, по системе команд и архитектуре различаются процессоры RISC и CISC.

RISC – Reduced (Restricted) Instruction Set Computer – процессоры (компьютеры) с сокращенной системой команд. Эти процессоры обычно имеют набор однородных регистров универсального назначения, и их система команд отличается относительной простотой. В результате аппаратная реализация такой архитектуры позволяет с небольшими затратами выполнить за минимальное (в пределе одного) число тактов синхронизации.

CISC – Complete Instruction Set Computer – процессоры (компьютеры) с полным набором инструкций, к которым и относится семейство x86. Состав и назначения их регистров существенно не однородны, широкий набор усложняет декодирование инструкций, на что расходуются аппаратные ресурсы. Возрастает и число тактов, необходимое для выполнения инструкций.

Поэтому CISC-процессоры используются в универсальных компьютерных системах, а RISC-процессоры – в специализированных. Для ПК платформы IBM PC доминирующими являлись CISC-процессоры фирмы Intel, хотя в свое время компания AMD начала изготавливать процессоры семейства AMD-K6, которые имели гибридную архитектуру (внутреннее ядро этих процессоров выполненное по RISC-архитектуре, а внешняя структура – по архитектуре CISC). В настоящее время большинство процессоров имеют такую комбинированную структуру.

9.3. Основные шинные интерфейсы материнских плат

Интерфейс – это аппаратное и программное обеспечение (элементы соединения и вспомогательные схемы управления, их физические, электрические и логические параметры), предназначенное для

сопряжения систем или частей системы (программ или устройств). Под сопряжением подразумеваются следующие функции: выдача и прием информации; управление передачей данных; согласование источника и приемника информации.

Важное значение имеют также следующие технические характеристики интерфейсов:

1) вместимость (максимально возможное количество абонентов, одновременно подключаемых к контроллеру интерфейса без расширителей);

2) пропускная способность или скорость передачи (длительность выполнения операций установления и разъединения связи и степень совмещения процессов передачи данных);

3) максимальная длина линии связи;

4) разрядность;

5) топология соединения.

Различают два класса системных интерфейсов: с общей шиной (сигналы адреса и данных мультиплексируются) и с изолированной шиной (раздельные сигналы данных и адреса). Прародителями современных системных шин являются **Unibus** фирмы DEC (интерфейс с общей шиной) и **Multibus** фирмы Intel (интерфейс с изолированной шиной).

ISA (Industry Standard Architecture). Разрешает связать между собой все устройства системного блока, а также обеспечивает простое подключение новых устройств через стандартные слоты. Пропускная способность составляет до 5,5 Мб/с. В современных компьютерах не используется.

EISA (Extended ISA). Расширение стандарта ISA. Пропускная способность – 32 Мб/с. Как и стандарт ISA, этот стандарт исчерпал свои возможности.

VLB (VESA Local Bus). Интерфейс локальной шины стандарта VESA. Локальная шина соединяет процессор с оперативной памятью в обход основной шины. Она работает на большей частоте, чем основная шина, и позволяет увеличить скорость передачи данных. Позже в локальную шину «врезали» интерфейс для подключения видеоадаптера, который требует повышенной пропускной способности, что и привело к появлению стандарта VLB. Пропускная способность – до 130 Мб/с, рабочая тактовая частота – 50 МГц, но она зависит от количества устройств, подсоединенных к шине, что является главным недостатком интерфейса VLB.

PCI (Peripherial Component Interconnect). Стандарт подключения внешних устройств, введенный в ПК на базе процессора Pentium. По своей сути, это интерфейс локальной шины с разъемами для подсоединения внешних компонентов. Данный интерфейс поддерживает частоту шины до 66 МГц и обеспечивает быстродействие до 264 Мб/с независимо от количества подсоединенных устройств. Важным нововведением этого стандарта является поддержка механизма plug-and-play, суть которого состоит в том, что после физического подключения внешнего устройства к разъему шины PCI происходит автоматическая конфигурация этого устройства. Существуют три варианта плат PCI: с уровнями сигналов 3,3 В, с уровнями сигналов 5 В и универсальные. Ключ в разъеме гарантирует, что платы с одним уровнем сигнала и невзаимозаменяемые не будут по ошибке вставлены в разъем с другим уровнем сигнала. Платы с пониженным напряжением питания в основном используются в мобильных компьютерах.

FSB (Front Side Bus). Начиная с процессора Pentium Pro, для связи с оперативной памятью используется специальная шина FSB. Эта шина работает на частоте 100–133 МГц и имеет пропускную способность до 800 Мб/с. Частота шины FSB является основным параметром, именно она указывается в спецификации материнской платы. За шиной PCI осталась лишь функция подключения новых внешних устройств. С 2005 г. в ПЭВМ класса Pentium 4 вместо PCI используют новый системный интерфейс – **PCI Express *x***, который совершенствуется и развивается до сих пор, где *x* определяет количество полос (ширину канала) канала передачи данных.

AGP (Advanced Graphic Port). Специальный шинный интерфейс для подключения видеоадаптеров. Разработан в связи с тем, что параметры шины PCI не отвечают требованиям видеоадаптеров по быстродействию. Частота этой шины – 33 или 66 МГц, пропускная способность достигнута 2133 Мб/с в режиме 8x.

USB (Universal Serial Bus). Стандарт универсальной последовательной шины определяет новый способ взаимодействия компьютера с периферийным оборудованием. Он разрешает подключать до 256 разных устройств с последовательным интерфейсом, причем устройства могут подсоединяться цепочкой. Производительность шины USB 1.0 относительно небольшая и составляет 1,55 Мбит/с. После усовершенствования до уровня 2.0 пропускная способность канала увеличена до 60 Мб/с. Среди преимуществ этого стандарта следует отметить возможность подключать и отключать устройства в «горячем режиме»

(т. е. без перезагрузки компьютера), а также возможность объединения нескольких компьютеров в простую сеть без использования специального аппаратного и программного обеспечения.

В настоящее время для обеспечения взаимодействия с жесткими дисками распространены интерфейсы системного уровня, использующие сигналы в логике центрального процессора, что предполагает реализацию функций контроллера накопителя в самом накопителе, а устройство, сопрягающее интерфейс накопителя с системной шиной ПК, выполняет лишь роль адаптера интерфейса (моста). В IBM PC таким интерфейсом является EIDE/ATA. Он представляет собой «приставку» к 16-битной шине ISA, иначе называемой AT Bus, поэтому стандарт именуется AT Attachment (ATA). Другое название интерфейса – Enhanced Integrated Drive Electronics (EIDE). Первая спецификация ATA (IDE) определяла возможность подключения двух устройств к одному интерфейсу. Спецификация ATA-2 (EIDE) описывает совместную работу двух интерфейсов, позволяя таким образом подключать до четырех устройств. С внедрением стандарта ATA-4 на поддержку пакетных команд (ATAPI – ATA Packet Interface) стало возможным подключение устройств со сменным накопителем (приводы CD-ROM/DVD-ROM, стримеры, приводы флоппи-дисков большого объема).

В современной вычислительной технике наблюдается тенденция перехода на высокоскоростные последовательные интерфейсы. Так, для накопителей был предложен последовательный интерфейс SerialATA, по своим характеристикам представляющий собой «приставку» к PCI Express. Стандарт SATA/300 обеспечивает пропускную способность до 3 Гбит/с (без учета кодирования 8B/10B). Каждое устройство работает на отдельном кабеле. Стандарт предусматривает горячую замену устройств и функцию очереди команд. Передача данных происходит в дуплексном режиме по двум парам проводником (одна пара – на прием, другая – на передачу) с использованием дифференциального кодирования сигналов.

Кроме рассмотренных интерфейсов в ПЭВМ используются универсальные периферийные интерфейсы SCSI, USB, FireWire и т. п.

В связи с тем, что технический прогресс не стоит на месте, технологии и идеи в организации архитектуры ЭВМ могут меняться достаточно быстро, однако это не говорит о том, что не надо иметь представление о идеях, позволивших эволюционировать архитектуре ЭВМ. Соответственно, чтобы быть хорошим специалистом в области информационных технологий, следует изучать новые разработки в аппаратной и программной части ЭВМ.

Литература

1. Бердышев, Е. М. Технология ММХ. Новые возможности процессоров P5 и P6 / Е. М. Бердышев. – Москва : Диалог МИФИ, 1998. – 234 с.

2. Борзенко, А. Е. IBM PC: устройство, ремонт, модернизация / А. Е. Борзенко. – Москва : Компьютер пресс, 1995. – 297 с.

3. Бройдо, В. Л. Архитектура ЭВМ и систем : учеб. для вузов / В. Л. Бройдо, О. П. Ильина. – Санкт-Петербург : Питер, 2006. – 718 с.

4. Василевский, А. В. Устройство и функционирование ЭВМ / А. В. Василевский. – Минск : ЕГУ, 2002.

5. Григорьев, В. Л. Микропроцессор i486. Архитектура и программирование. В 4 кн. / А. В. Григорьев. – Москва : Пранал, 1993.

6. Гук, М. Аппаратные средства IBM PC : энциклопедия / М. Гук. – Санкт-Петербург : Питер, 2000.

7. Злобин, В. К. Программирование арифметических операций в микропроцессорах : учеб. пособие для техн. вузов. / В. К. Злобин, В. Л. Григорьев. – Москва : Высш. шк., 1991. – 301 с.

8. Пешков, А. Т. Организация и функционирование ЭВМ : метод. пособие для студентов специальности «Программное обеспечение информационных технологий» днев. формы обучения : в 3 ч. Ч. 1. Арифметические основы ЭВМ / А. Т. Пешков. – Минск : БГУИР, 2004. – 61 с.

9. Сван, К. Освоение Turbo Assembler / К. Сван. – Киев : Диалектика, 1996. – 544 с.

10. Таненбаум, Э. Архитектура компьютера / Э. Таненбаум. – Санкт-Петербург : Питер, 2002. – 704 с.

11. Цилькер, Б. Я. Организация ЭВМ и систем : учеб. для вузов / Б. Я. Цилькер, С. А. Орлов. – Санкт-Петербург : Питер, 2006. – 668 с.

Содержание

Учебное электронное издание комбинированного распространения

Учебное издание

Ковалев Алексей Викторович
Литвинов Дмитрий Александрович

ОРГАНИЗАЦИЯ И ФУНКЦИОНИРОВАНИЕ ЭВМ

Курс лекций для студентов специальности 1-40 01 02 «Информационные системы и технологии (по направлениям)»

Электронный аналог печатного издания

Редактор *М. В. Аникеенко*
Компьютерная верстка *Н. Б. Козловская*

Подписано в печать 23.03.10.
Формат 60x84/16. Бумага офсетная. Гарнитура «Таймс».
Ризография. Усл. печ. л. 3,49. Уч.-изд. л. 3,31
Изд. № 204.
E-mail: ic@gstu.by
http://www.gstu.by

Издатель и полиграфическое исполнение:
Издательский центр учреждения образования
«Гомельский государственный технический университет
имени П. О. Сухого».
ЛИ № 02330/0549424 от 08.04.2009 г.
246746, г. Гомель, пр. Октября, 48.