§32
Принципы устройства компьютера

В предыдущем параграфе вы увидели, что вычислительная техника в своем развитии прошла целый ряд характерных этапов. Несмотря на это некоторые фундаментальные (базовые, основные) принципы устройства ЭВМ почти не изменились. Поэтому логично начать знакомство с устройством компьютера именно с них.

Классические принципы построения ЭВМ были предложены в работе *А. Беркса, Г. Голдстайна и Дж. фон Неймана «Предварительное рассмотрение логической конструкции электронного вычислительного устройства»*. Обычно выделяют следующие наиболее важные идеи этой работы:

- состав основных компонентов вычислительной машины;
- принцип двоичного кодирования;
- принцип адресности памяти;
- принцип иерархической организации памяти;
- принцип хранимой программы;
- принцип программного управления.

Рассмотрим их подробнее.

Общие принципы

Основные компоненты машины. В самом первом разделе с таким названием фон Нейман с соавторами определили и обосновали состав ЭВМ:

«Так как законченное устройство будет универсальной вычислительной машиной, оно должно содержать несколько основных органов, таких как орган арифметики, памяти, управления и связи с оператором. Мы хотим, чтобы машина была полностью автоматической, т. е. после начала вычислений работа машины не зависела от оператора».

Таким образом, ЭВМ должна состоять из нескольких блоков, каждый из которых выполняет вполне определённую функцию. Эти блоки есть и в сегодняшних компьютерах:

- **арифметико-логическое устройство** (АЛУ), в котором выполняется обработка данных;
- **устройство управления** (УУ), обеспечивающее выполнение программы и организующее согласованное взаимодействие всех узлов машины; сейчас АЛУ и УУ изготавливают в виде единой интегральной схемы — микропроцессора;
-

память — устройство для хранения программ и данных; память обычно делится на **внутреннюю** (для временного хранения данных во время обработки) и **внешнюю** (для длительного хранения между сеансами обработки);

- **устройства ввода,** преобразующие входные данные в форму, доступную компьютеру;
- **устройства вывода,** преобразующие результаты работы ЭВМ в форму, удобную для восприятия человеком.

В классическом варианте все эти устройства взаимодействовали через процессор (рис. 5.7).

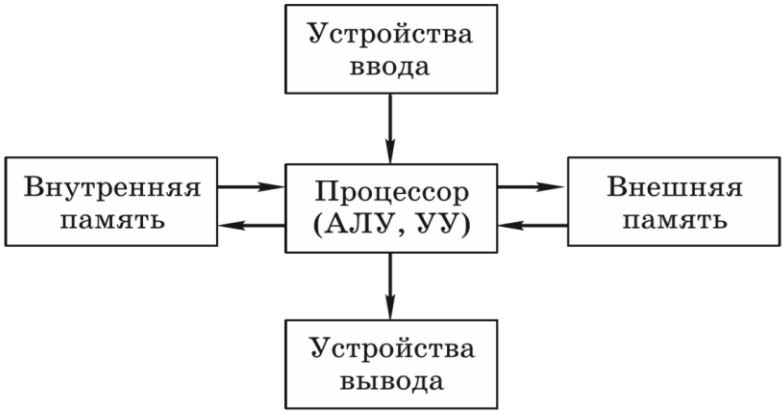


Рис. 5.7

Принцип двоичного кодирования. Устройства для хранения двоичной информации и методы её обработки наиболее просты и дешевы. Поскольку в ЭВМ используется двоичная система счисления, необходимо переводить данные из десятичной формы в двоичную (при вводе) и наоборот (при выводе результатов). Однако такой перевод легко автоматизируется, и многие пользователи даже не знают об этих внутренних преобразованиях.

В первых машинах использовались только числовые данные. В дальнейшем ЭВМ стали обрабатывать и другие виды информации (текст, графика, звук, видео), но это не привело к отмене принципа двоичного кодирования. Даже цифровые сигнальные процессоры, предназначенные для обработки цифровых сигналов в реальном времени, используют двоичное представление данных.

В истории известен пример успешной реализации *троичной ЭВМ «Сетунь»* (1959 г., руководитель проекта *Н. П. Брусенцов*), но он так и остался оригинальным эпизодом и не оказал влияния на эволюцию вычислительной техники. В первую очередь, это связано с серьёзными проблемами, которые возникают при изготовлении элементов троичного компьютера на основе полупроводниковых технологий. Эти проблемы так и не были решены, тогда как наладить массовое производство аналогичных устройств для двоичных компьютеров оказалось значительно проще.

Принципы организации памяти

Принцип адресности памяти. Оперативная память машины состоит из отдельных **битов.** Для записи или считывания группы соседних битов объединяются в **ячейки памяти,** каждая из которых имеет свой адрес (номер). Нумерацию ячеек принято начинать с нуля.

Адрес ячейки памяти — это её номер.

Ячейка содержит минимально возможный считываемый из памяти объём данных: невозможно прочитать меньшее количество битов, а тем более отдельный бит.

Использование чисел для нахождения в памяти требуемых ячеек выглядит абсолютно естественно: в компьютерах любая информация кодируется числами, так что адреса ячеек — не исключение из этого фундаментального правила. Если номера соседних ячеек отличаются на единицу, удобно организовывать их последовательную обработку.

Разрядность ячеек памяти (количество битов в ячейке) в разных поколениях была различной. Первоначально ЭВМ были построены исключительно для математических расчётов. Числа желательно было представлять как можно точнее, поэтому ячейки ОЗУ в первых машинах были длинными. Кроме чисел, машина должна была хранить в памяти еще и команды программы; как правило, в то время размер числовой ячейки совпадал с размером команды, что существенно упрощало устройство памяти.

Примерно на стыке второго и третьего поколений ЭВМ стали использовать для обработки символьной информации, что привело к серьёзному неудобству: в существующую числовую ячейку памяти помещалось 4–5 символов. Инженеры выбрали наиболее простое решение проблемы — уменьшить размер ячейки так, чтобы можно было обращаться к каждому символу отдельно. **Байтовая память,** основой которой стала **восьмибитная ячейка,** прекрасно зарекомендовала себя и используется в компьютерной технике до настоящего времени.

В результате перехода к «коротким» ячейкам памяти числа стали занимать несколько ячеек (байтов), каждая из которых имеет собственный адрес. На рисунке 5.8, *а* показана организация ячеек памяти первых ЭВМ, а на рис. 5.8, *б* — современная (байтовая) структура памяти.

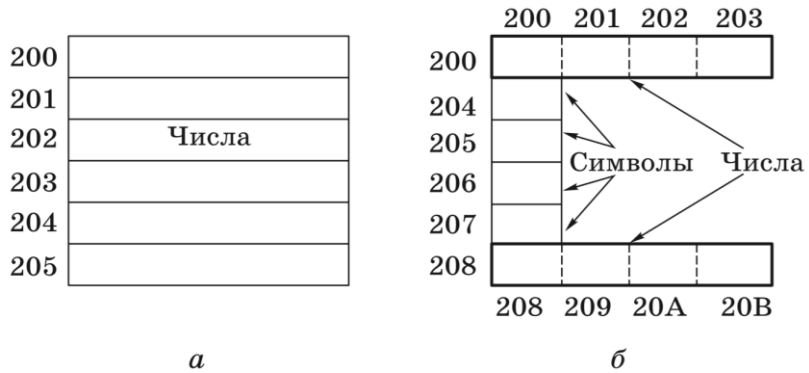


Рис. 5.8

На рисунке 5.8, *а* числа занимают по одной ячейке, причём номера этих ячеек отличаются на единицу. Справа показаны два 32-битных числа, которые хранятся в

байтах 200–203 и 208–20B (адреса указаны в шестнадцатеричной системе). По принятому правилу за адрес числа принимается наименьший из адресов, так что в данном случае адреса чисел — 200 и 208. Кроме того, на рис. 5.8, *б* между числами (в байтах с 204 по 207) размещены четыре однобайтных символа. Заметим, что современные компьютеры могут извлекать из памяти до восьми соседних байтовых ячеек за одно обращение к памяти.

287

Очень важно, что информация может считываться из ячеек и записываться в них в произвольном порядке, поэтому организованную таким образом память принято называть **памятью с произвольным доступом** (англ. **RAM** — *random access memory*). Чтобы лучше понять смысл этого термина, сравните такую память с магнитной лентой, данные с которой можно получить только путем последовательного чтения.

Часто термин RAM отождествляют с русским термином ОЗУ — **оперативное запоминающее устройство.** Это не совсем точно. Дело в том, что кроме ОЗУ существует еще одна разновидность памяти с произвольным доступом — **постоянное запоминающее устройство** (**ПЗУ**, англ. **ROM** — *Read Only Memory* — память только для чтения). Главное отличие ПЗУ от ОЗУ заключается в том, что при решении задач пользователя содержимое ПЗУ не может быть изменено. ПЗУ гораздо меньше ОЗУ по объёму, но это очень важная часть компьютера, поскольку в нём хранится доступное в любой момент программное обеспечение. Благодаря этому ПО компьютер сохраняет работоспособность даже тогда, когда в ОЗУ нет никакой программы.

Таким образом, ОЗУ и ПЗУ — это два вида памяти с произвольным доступом, обращение к данным в которых построено на основе принципа адресности.

Принцип иерархической организации памяти. К памяти компьютера предъявляются два противоречивых требования: её объём должен быть как можно больше, а скорость работы — как можно выше. Ни одно реальное устройство не может удовлетворить им одновременно. Любое существенное увеличение объёма памяти неизбежно приводит к уменьшению скорости её работы. Действительно, если память большая, то обязательно усложняется поиск в ней требуемых данных, а это сразу замедляет чтение из памяти. Кроме того, чем быстрее работает память, тем она дороже, и, следовательно, меньше памяти можно установить за приемлемую для потребителей стоимость.

288

Чтобы преодолеть противоречие между объёмом памяти и её быстродействием, используют несколько различных видов памяти, связанных друг с другом. Когда в 1946 г. впервые формулировался этот принцип, в состав ЭВМ предполагалось включить всего два вида памяти: оперативную память и память на магнитной проволоке (предшественник устройств хранения данных на магнитной ленте). Дальнейшее развитие вычислительной техники подтвердило необходимость построения иерархической памяти: в современном компьютере уровней иерархии гораздо больше.

Выполнение программы

Принцип хранимой программы. Первые ЭВМ программировались путём установки перемычек на специальных панелях (рис. 5.9), так что процесс подготовки к решению задачи мог растянуться на несколько дней. Такое положение дел никого не

устраивало, и в фон-неймановской архитектуре было предложено представлять команды в виде двоичного кода. Код программы, записанный заранее на перфокарты или магнитную ленту, можно было ввести в машину достаточно быстро.

Рис. 5.9. *Фрагмент коммутационной панели устройства IBM-557; требуемая операция получается соединением (коммутацией) соответствующих отверстий*

289

Поскольку команды программы и данные по форме представления стали одинаковыми, их можно хранить в *единой памяти* вместе с данными. Не существует принципиальной разницы между двоичными кодами машинной команды, числа, символа и т. д. Это утверждение иногда называют **принципом однородности памяти.** Из него следует, что команды одной программы могут быть получены как результат работы другой программы. Именно так текст программы на языке высокого уровня переводится (транслируется) в машинные коды конкретной машины.

Код программы может сохраняться во внешней памяти (например, на дисках) и затем загружаться в оперативную память для повторных вычислений. Благодаря простоте замены программ, ЭВМ стали **универсальными устройствами,** способными решать самые разнообразные задачи в произвольном порядке и даже одновременно.

Принцип программного управления. Любая обработка данных в вычислительной машине происходит по программе. Принцип программного управления определяет наиболее общий механизм автоматического выполнения программы.

Важным элементом устройства управления в машине фон-неймановской архитектуры является специальный регистр — **счётчик адреса команд**
*

. В нём в любой момент хранится адрес команды программы, которая будет выполнена следующей.

Используя значение из счётчика, процессор считывает из памяти очередную команду программы, расшифровывает её и выполняет. Затем те же действия повторяются для следующей команды и т. д. Процессор выполняет команды по следующему алгоритму (его часто называют **основным алгоритмом работы процессора**):

1) из ячейки памяти, адрес которой записан в счётчике адреса команд, выбирается очередная команда программы; на время выполнения она сохраняется в специальном **регистре команд;**

290

2) значение счётчика адреса команд увеличивается так, чтобы он указывал на следующую команду;
3) выбранная команда выполняется (например, при сложении двух чисел оба слагаемых считываются в АЛУ, складываются и результат операции сохраняется в регистре или ячейке памяти);
4) далее весь цикл повторяется сначала.

Таким образом, автоматически выполняя одну команду программы за другой, компьютер может исполнить любой линейный алгоритм. Для того чтобы в программе

можно было использовать ветвления и циклы, необходимо нарушить естественную последовательность выполнения команд. Для этого существуют специальные **команды перехода,** которые на этапе 3 заносят в счётчик адреса новое значение — адрес перехода. Чаще всего в программах используется **условный переход,** т. е. переход происходит только при выполнении определенного условия.

Легко понять, что для запуска основного алгоритма работы процессора в счётчик адреса команд должно быть предварительно занесено начальное значение — адрес первой выполняемой команды. В первых ЭВМ оператор вводил этот адрес вручную. В современных компьютерах при включении питания в счётчик аппаратно заносится некоторое значение, которое указывает на начало программы, хранящейся в ПЗУ. Эта программа тестирует устройства компьютера и приводит их в рабочее состояние, а затем загружает в ОЗУ **начальный загрузчик операционной системы** (как правило, с диска). Ему и передаётся дальнейшее управление, а стартовая программа из ПЗУ завершает свою работу. Начиная с этого момента, поведение компьютера уже определяется установленным на нём программным обеспечением (см. главу 6).

Чтобы ускорить выполнение программы, основной алгоритм работы процессора был значительно усовершенствован. Идея была заимствована из **конвейерного** производства, где несколько рабочих одновременно выполняют различные операции (каждый над своим экземпляром изделия). Аналогично в современных микропроцессорах для каждого этапа выполнения команды создан отдельный аппаратный блок. Выполнив свою операцию, он передает результаты следующему блоку, а сам начинает выполнять очередную команду.

Проще всего понять этот механизм на примере первого этапа — выборки команды из ОЗУ. Специализированный блок выборки извлекает из памяти последовательно расположенные команды, не дожидаясь окончания их обработки. Прочитанные команды размещаются в специальной рабочей памяти внутри микропроцессора. В итоге, когда первая из выбранных команд будет завершена, за следующей не придется обращаться к ОЗУ, так как она уже находится во внутренней памяти микропроцессора. Учитывая, что обращение к ОЗУ занимает значительно большее времени, чем пересылка данных внутри процессора, такая *опережающая выборка* значительно ускоряет выполнение программы.

На практике применение конвейерного метода не так просто. Например, следующую команду часто не удается выполнить, поскольку она использует результат предыдущей, или сразу нескольким командам потребуется одновременно обратиться к ОЗУ. Тем не менее этот метод широко применяется в микропроцессорах. В некоторых моделях используются **параллельные конвейеры,** так что во многих случаях к моменту завершения выполнения одной команды уже готов результат следующей.

Что называют архитектурой

Описанные фон Нейманом и его соавторами классические принципы построения вычислительных устройств применялись во всех поколениях ЭВМ. В дополнение к ним в каждом конкретном семействе (-D-, ЕС ЭВМ, Apple, IBM PC и др.) формулируются свои собственные принципы устройства, благодаря которым обеспечивается аппаратная и программная совместимость моделей. Для пользователей это означает, что все существующие программы будут работать и на новых моделях того же семейства компьютеров. В литературе общие принципы построения

конкретного семейства компьютеров называют **архитектурой**. К архитектуре обычно относят:

- принципы построения системы команд и их кодирования;
- форматы данных и особенности их машинного представления;
- алгоритм выполнения команд программы;
- способы доступа к памяти и внешним устройствам;
- возможности изменения конфигурации оборудования.

Стоит обратить внимание на то, что архитектура описывает именно общее устройство вычислительной машины, а не особенности изготовления конкретного компьютера (набор микросхем, тип жёсткого диска, ёмкость памяти, тактовая частота). Например, наличие видеокарты как устройства для организации вывода информации на монитор входит в круг вопросов архитектуры. А вот является ли видеокарта частью основной платы компьютера или устанавливается на неё в виде отдельной платы, с точки зрения архитектуры значения не имеет. Иначе могло бы получиться, что для интегрированной в плату видеокарты потребовалась бы отдельная версия графического редактора!

1. Найдите материалы, подтверждающие, что Джон фон Нейман не был единоличным автором «фон-неймановской» архитектуры ЭВМ.
2. Перечислите принципы фон-неймановской архитектуры и кратко объясните каждый из них.
3. Назовите основные компоненты вычислительного устройства. Каково их назначение? Согласны ли вы с тем, что полученный набор узлов логичен и обоснован?
4. В чём состоит принцип двоичного кодирования?
5. Вспомните, как кодируются в компьютере числа, тексты, графика. Соблюдается ли принцип двоичного кодирования?
6. По какому алгоритму вводимые в компьютер десятичные числа можно перевести во внутреннее двоичное представление? Как перевести обратно результаты расчёта?
7. Что такое ячейка памяти? Что такое адрес ячейки?
8. Что вы знаете о разрядности ячеек ОЗУ разных поколений?
9. Почему появилась байтовая память?
10. Можно ли заменить в ячейке памяти содержимое одного бита, не затрагивая

значений соседних? Почему?

11. Приведите примеры различных типов данных и назовите их разрядность. Сколько байтов памяти потребуется для хранения данных каждого из этих типов?

12. Что такое иерархическая организация памяти?

13. Почему большая по объёму память обычно работает медленнее, чем маленькая?

14. В чём состоит принцип хранимой программы?

15. Где может храниться программа?

16. Можно ли к нечисловым данным (символам, графическим и звуковым данным) применять арифметические операции?

17. Как вы понимаете фразу «любая обработка данных в вычислительной машине происходит по программе»? Чем компьютер в этом отношении отличается от простого калькулятора?

18. Сформулируйте основной алгоритм выполнения команды в компьютере.

19. Что такое счётчик адреса команд и какова его роль в основном алгоритме?

20. Опишите, что происходит в момент включения компьютера с точки зрения принципа программного управления.

21. Можно ли нарушить последовательность выполнения команд в программе? Для чего это может потребоваться?

22. Всегда ли в новом компьютере есть какая-либо программа?

23. Что такое конвейер и как он работает при выполнении программы?

***24.** Почему команды перехода нарушают работу конвейера?

25. Какие из принципов, предложенных в работе «Предварительное рассмотрение логической конструкции электронного вычислительного устройства», продолжают применяться в современных компьютерах безо всяких изменений, а какие сохранились, но в несколько изменённом виде? Объясните, почему потребовались эти изменения.

26. Что такое архитектура? Какие детали устройства компьютера к ней не относятся?

27. В чем преимущества единой архитектуры семейств ЭВМ для пользователей и для производителей?

28. Какие семейства вычислительных машин вы знаете?

Вопросы и задания

Подготовьте сообщение

а) «Джон фон Нейман и его вклад в науку»
б) «Троичная ЭВМ "Сетунь"»
в) «Гарвардская архитектура»
г) «Архитектуры современных компьютеров»

Задачи

***1.** Используя условные команды «считать байт из памяти», «записать байт в память», а также битовые логические операции «И», «ИЛИ» и «НЕ», составьте алгоритм который в байте, хранящемся в памяти:

а) установит младший бит данных в единицу, не затрагивая содержимого остальных двоичных разрядов;

б) сбросит его в ноль.

2. Описанная в «Предварительном рассмотрении…» конструкция ЭВМ имела ячейки памяти, состоящие из 40 двоичных разрядов. Сколько современных байтовых ячеек потребуется для хранения числа такой же разрядности?

3. В языке Паскаль есть тип данных word, значением которого являются целые положительные 16-разрядные числа. Сколько байтов занимает такое число в памяти? Какое максимальное значение может иметь этот тип чисел?

4. В микропроцессорах семейства Intel для увеличения на единицу одного из регистров процессора используется команда, имеющая код 41_{16}. Пользуясь таблицей символов, определите, какая буква соответствует этому же самому коду. Если рассматривать этот код как целое число, чему оно будет равно в десятичной системе счисления?

***5.** Изучите содержимое текстового файла, используя программу, которая способна отображать данные в виде шестнадцатеричных кодов. Попробуйте найти коды известных вам символов. Сделайте то же самое с графическим файлом: удастся ли вам найти там те же самые коды («символы»)?

***6.** Найдите таблицу кодов команд процессора Intel. Сравните эти коды с кодами, которые вы нашли в предыдущем задании. Удалось ли найти совпадения?