МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ АГЕНТСТВО ЖЕЛЕЗНОДОРОЖНОГО ТРАНСПОРТА
Филиал федерального государственного бюджетного образовательного учреждения высшего образования
«Самарский государственный университет путей сообщения» в г. Саратове
филиал СамГУПС в г.Саратове

КУРС ЛЕКЦИЙ

ОП.09 ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА

для студентов специальности

11.02.06 Техническая эксплуатация транспортного радиоэлектронного оборудования (по видам транспорта)

Саратов 2023

РАССМОТРЕНО
на заседании методического совета
Протокол № 1 от «21» 09 2023г.

СОГЛАСОВАНО
Протокол №1 от «31» 08 2023г.
Председатель ЦМК 11.02.06 Техническая эксплуатация транспортного радиоэлектронного оборудования (по видам транспорта)

_____/Ю.А. Солдатенко/

УТВЕРЖДАЮ
Заместитель директора по учебной работе

_____/С.А. Крижановский/

Автор (составитель):

Солдатенко Ю.А – преподаватель первой квалификационной категории филиала СамГУПС в г. Саратове

Рецензент:

Царенков - преподаватель высшей квалификационной категории филиала СамГУПС в г. Саратове

Содержание

Тема 1.1:Арифметические основы ЭВМ

Лекция 1. Общие сведения о системах счисления. Позиционные системы счисления, применяемые в ЭВМ. Перевод чисел из одной позиционной системы в другую.

Известно множество способов представления чисел. Числа записываются с использованием особых знаковых систем, которые называются системами счисления. Алфавит систем счисления состоит из символов, которые называются цифрами.

Система счисления - это знаковая система, в которой числа записываются по определенным правилам с помощью символов некоторого алфавита, называемых цифрами.

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

В непозиционной системе счисленияцифры не изменяют своего значения при изменении их расположения в числе. Примером непозиционной системы может служить римская система, в которой независимо от местоположения одинаковый символ имеет неизменное значение (например, символ X в числе XXV).

В римской непозиционной системе счисления для каждого числа используется некоторый набор базовых символов (I,V,X,L,C,D и M), соответствующих числам 1,5,10,50,100,500 и 1000. Остальные значения чисел получаются из базовых путем их сложения (например, XVII=17) или вычитания (например, IX=9). Примером является римская система. Алфавит этой системы состоит из специальных символов, обозначающих следующее:
I – 1; V - 5; X – 10; L – 50; C-100; D-500; M -1000.

Если мы рассмотрим, например, число XXX (30), оно состоит из трех цифр X каждая из которых не зависимо от места в числе равна 10. Для изображения чисел отличных от выше перечисленных необходимо комбинировать римские цифры. Помня, что меньшая по величине цифра, стоящая справа от большей, складывается с большей (прим. CXXXIII = 133), а стоящая слева –вычитается из большей (прим. IX = 10-1 =9).

Недостатком римской системы является отсутствие формальных правил записи чисел и, соответственно, арифметических действий с многозначными числами. По причине неудобства и большой сложности в настоящее время римская система счисления используется там, где это действительно удобно: в литературе (нумерация глав), в оформлении документов (серия паспорта,

ценных бумаг и др.), в декоративных целях на циферблате часов и в ряде других случаев.

Позиционные системы счисления

В позиционной системе счисления значение каждой цифры числа зависит от того, в каком месте (позиции или разряде) она записана. Например, меняя позицию цифры 2 в десятичной системе счисления, можно записать разные по величине десятичные числа, например: 2; 20; 2000; 0,02 и т. д. Позиция цифры в числе называется разрядом. Разряд числа возрастает справа налево, от младших разрядов к старшим.

Количество (p) различных символов, используемых для изображения числа в позиционной системе счисления, называется ***основанием системы счисления.***

Основание показывает, во сколько раз изменяется количественное значение цифры при перемещении ее в младший или старший разряд.

Набор символов, используемый для обозначения цифр, называется алфавитом.

Так, например, алфавит двоичной системы счисления содержит всего два символа: 0 и 1, а алфавит шестнадцатеричной системы - 16 символов: десять арабских цифр и шесть латинских букв (0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F).

Любое число N в позиционной системе счисления можно представить в следующем виде:

$$N_p = \pm(a_{k-1} \cdot p^{k-1} + a_{k-2} \cdot p^{k-2} + ... + a_0 \cdot p^0 + a_{-1} \cdot p^{-1} + ... + a_{-m} \cdot p^{-m})$$

Такой вид записи числа называют **развернутой формой записи числа**,

где *p* - основание системы счисления;

ai - цифры, принадлежащие алфавиту данной системы счисления;

k - количество разрядов в целой части числа;

m - количество разрядов в дробной части числа.

Нижние индексы определяют местоположение цифры в числе (разряд):

- положительные значения индексов
- для целой части числа;
- отрицательные значения индексов
- для дробной части числа.

Свернутой формой записи числа называется запись в виде:

$$N = (a_{k-1}a_{k-2} ... a_1a_0, a_{-1}a_{-2} ... a_{-m})_p$$

Например:

- при *p*=10 в записи числа $2466{,}675_{10}$ в десятичной системе счисления k=3, m=3;

- при p=2 в записи числа $1011{,}11_2$ в двоичной системе k=3, m=2.

Свернутой формой записи чисел мы и пользуемся в повседневной жизни, ее называют естественной или цифровой. Основанием позиционной системы счисления может быть любое натуральное число (например, 5, 21, 37). Во избежание путаницы справа от числа нижним индексом приписывают основание: 101101_2, 367_8, $3B8A_{16}$, $3AO_{37}$.

Десятичная система счисления

Основание: p=10.

Алфавит: 0,1,2,3,4,5,6,7,8,9.

Десятичная система счисления наиболее распространенная система счисления в мире. Используется при повседневном счете. Для записи чисел используются арабские цифры (0, 1, 2, 3, 4, 5, 6, 7, 8, 9).

Число в десятичной системе счисления записывается в виде суммы числового ряда степеней основания (в данном случае 10), в качестве коэффициентов которых выступают цифры данного числа.

Пример: $765{,}345_{10}=7\cdot10^2+6\cdot10^1+5\cdot10^0+3\cdot10^{-1}+4\cdot10^{-2}+5\cdot10^{-3}$

Двоичная система счисления

Основание: p=2.

Алфавит: 0,1.

Двоичную систему счисления широко применяют в вычислительной технике. К ее достоинствам относятся:

- возможность использования наиболее простой элементной базы микроэлектроники

- всего с двумя устойчивыми состояниями;

- возможность использования аппарата булевой алгебры для выполнения логических преобразований информации;

- возможность использования простейших правил арифметики.

Основной недостаток двоичной системы - быстрый рост количества разрядов, необходимых для записи чисел. По этой, а также по некоторым другим причинам в вычислительной технике, кроме двоичной, применяются также восьмеричная и шестнадцатеричная системы счисления. Число в двоичной системе счисления записывается в виде суммы числового ряда степеней основания (в данном случае 2), в качестве коэффициентов которых выступают цифры данного числа.

Пример: $1011{,}01_2=1\cdot2^3+0\cdot2^2+1\cdot2^1+1\cdot2^0+0\cdot2^{-1}+1\cdot2^{-2}$

Восьмеричная система счисления

Основание: p=8.

Алфавит: 0,1,2,3,4,5,6,7.

Восьмеричная система чаще всего используется в областях, связанных с цифровыми устройствами. Характеризуется лёгким переводом восьмеричных чисел в двоичные и обратно, путём замены восьмеричных чисел на триады (группы по 3 разряда) двоичных. Ранее широко использовалась в программировании и вообще компьютерной документации, однако в настоящее время почти полностью вытеснена шестнадцатеричной. Число в восьмеричной системе счисления записывается в виде суммы числового ряда степеней основания (в данном случае 8), в качестве коэффициентов которых выступают цифры данного числа.

Пример: $567{,}12_8=5\cdot8^2+6\cdot8^1+7\cdot8^0+1\cdot8^{-1}+2\cdot8^{-2}$

Шестнадцатеричная система счисления

Основание: p=16.

Алфавит: 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F.

Здесь только десять цифр из шестнадцати имеют общепринятое обозначение 0,1,...,9. Для записи остальных цифр (10,11,12,13,14 и 15) обычно используются первые шесть букв латинского алфавита.

Шестнадцатеричная система счисления, на сегодняшний день является наиболее популярным средством компактной записи двоичных чисел. Очень широко используется при разработке и проектировании цифровой техники. Число в шестнадцатеричной системе счисления записывается в виде суммы числового ряда степеней основания (в данном случае 16), в качестве коэффициентов которых выступают цифры данного числа.

Пример: $10FC_{16}=1\cdot16^3+0\cdot16^2+F\cdot16^1+C\cdot16^0$

Помимо рассмотренных выше позиционных систем счисления, существуют и другие, например: - троичная (0,1,2); - пятеричная (0,1,2,3,4) - двенадцатеричная (0,1,2,3,4,5,6,7,8,9,A,B) - тринадцатеричная (0,1,2,3,4,5,6,7,8,9,A,B,C).

В системах счисления с основанием больше 10 для представления чисел после цифр 0,1,2,…,9 используют латинские буквы в алфавитном порядке: A (10), B (11), C (12) и т. д. Если все слагаемые в развернутой форме недесятичного числа представить в десятичной системе и вычислить полученное выражение по правилам десятичной арифметики, то получится число в десятичной системе, равное данному. По этому принципу производится перевод чисел из недесятичной системы в десятичную систему счисления.

Познакомимся с общими правилами перевода чисел из одной позиционной системы счисления в другую на нескольких примерах.

Пример:

1) Представим двоичное число $10110{,}101_2$ *в виде суммы слагаемых, а затем произведем их сложение:*

$10110{,}101_2=1\cdot2^4+0\cdot2^3+1\cdot2^2+1\cdot2^1+0\cdot2^0+1\cdot2^{-1}+0\cdot2^{-2}+1\cdot2^{-3}==16+0+4+2+0+0{,}5+0+0{,}125=22{,}625_{10}$

Таким образом, $10110{,}101_2=22{,}625_{10}$

2) *Представим шестнадцатеричное число 5D8,AC$_{16}$ в виде суммы слагаемых, а затем произведем их сложение:*

$5D8{,}AC_{16}=5\cdot16^2+13\cdot16^1+8\cdot16^0+10\cdot16^{-1}+12\cdot16^{-2}=1280+208+8+0{,}625+0{,}046875=1496{,}67187510$

Таким образом, $5D8{,}AC_{16}=1496{,}671875_{10}$

3*) Вычислим сумму чисел 2F$_{16}$, 232$_4$ и 53$_8$, представив результат в десятичной системе счисления.*

Переведем все числа в десятичную систему счисления, и сложим их:

$2F_{16}=2\cdot16^1+15\cdot16^0=32+15=47_{10}$

$232_4=2\cdot4^2+3\cdot4^1+2\cdot4^0=32+12+2=46_{10}$

$53_8=5\cdot8^1+3\cdot8^0=40+3=43_{10}$

$47_{10}+46_{10}+43_{10}=136_{10}$

Таким образом, $2F_{16}+232_4+53_8=136_{10}$

Представим десятичное число в общем виде N,M, где N - целая часть числа, а M - его дробная часть. Для перевода десятичного числа в позиционную систему счисления с основанием p необходимо воспользоваться двумя правилами: одно определяет технологию перевода целой части числа, а другое - дробной части.

Правило перевода целой части числа состоит из следующих этапов:

-число N делится на новое основание *p*;

-полученный остаток запоминается или записывается (это будет цифра младшего разряда);

-целая часть полученного частного снова делится на *p*;

- пять запоминаем полученный остаток (это будет цифра следующего разряда) и т. д.

Такое последовательное деление продолжается до тех пор, пока целая часть частного не окажется меньше, чем основание системы счисления *p*. Эта последняя целая часть частного будет цифрой старшего разряда. Результат формируется путем последовательной записи слева направо цифры старшего разряда и всех записанных остатков в порядке, обратном их получению

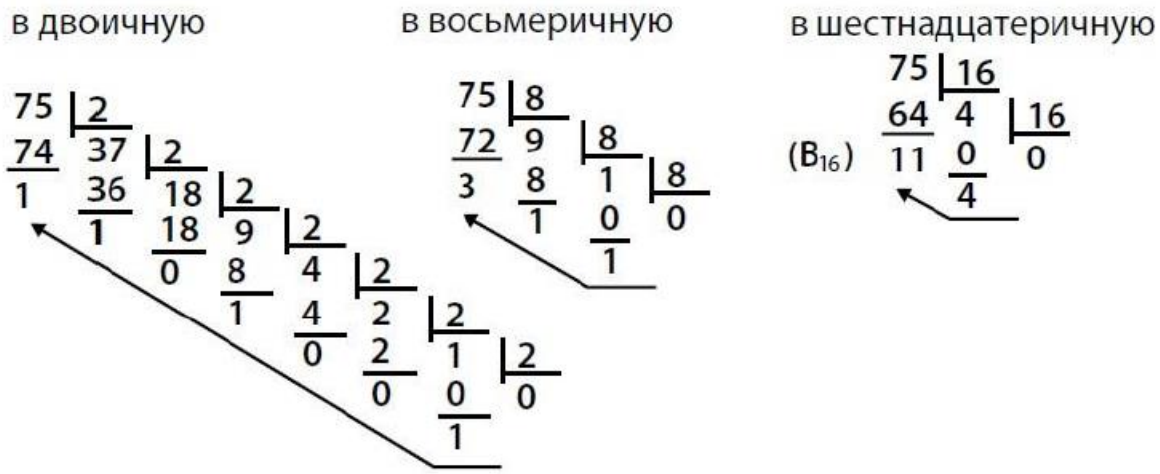
Правило перевода дробной части числа состоит из следующих этапов:

- дробная часть числа умножается на основание p;

- запоминается или записывается цифра результата, переносимая в целую часть;

- оставшаяся дробная часть числа умножается на основание p;

- снова фиксируется цифра результата, переносимая в целую часть, и т. д. Такое последовательное умножение продолжается до тех пор, пока в дробной части не будет получен ноль или достигнута требуемая точность, например 5 знаков после запятой**.**



Замечание: остаток 11_{10} записывается шестнадцатеричной цифрой $B_{16.}$ Ответ: $75_{10}=1001011_2=113_8=4B_{16}$

Задачи

1**.** Среди приведённых ниже трёх чисел, записанных в различных системах счисления, найдите максимальное и запишите его в ответе в десятичной системе счисления

23_{16}, 32_8, 11110_2.

38_{16}, 75_8, 110100_2.

14_{16}, 26_8, 11000_2.

24_{16}, 50_8, 101100_2.

50_{16}, 106_8, 1001010_2.

2. Переведите число 97,147,136,128,1201 из десятичной системы счисления в двоичную систему счисления. Сколько единиц содержит полученное число.

3. Переведите число 125,134,204,506 из десятичной системы счисления в восьмеричную систему счисления. В ответе укажите полученное число.

4. Переведите число 156,147,68,803 из десятичной системы счисления в шестнадцатеричную систему счисления. В ответе укажите полученное число.

Таблица 2

$2^0 = 1$	$8^0 = 1$	$10^0 = 1$	$16^0 = 1$
$2^1 = 2$	$8^1 = 8$	$10^1 = 10$	$16^1 = 16$
$2^2 = 4$	$8^2 = 64$	$10^2 = 100$	$16^2 = 256$
$2^3 = 8$	$8^3 = 512$	$10^3 = 1\,000$	$16^3 = 4\,096$
$2^4 = 16$	$8^4 = 4\,096$	$10^4 = 10\,000$	$16^4 = 65\,536$
$2^5 = 32$	$8^5 = 32\,768$	$10^5 = 100\,000$	$16^5 = 1\,048\,576$
$2^6 = 64$	$8^6 = 262\,144$	$10^6 = 1\,000\,000$	$16^6 = 16\,777\,216$
$2^7 = 128$	$8^7 = 2\,097\,152$	$10^7 = 10\,000\,000$	
$2^8 = 256$	$8^8 = 16\,777\,216$		
$2^9 = 512$			
$2^{10} = 1024$			
$2^{11} = 2048$			
$2^{12} = 4096$			

Шестнадцатиричная	Десятичная	Восьмеричная	Двоичная
0	0	0	0
1	1	1	1
2	2	2	10
3	3	3	11
4	4	4	100
5	5	5	101
6	6	6	110
7	7	7	111
8	8	10	1 000
9	9	11	1 001
A	10	12	1 010
B	11	13	1 011
C	12	14	1 100
D	13	15	1 101
E	14	16	1 110
F	15	17	1 111
10	16	20	10 000
11	17	21	10 001
12	18	22	10 010

Лекция 2. Представление чисел с фиксированной и плавающей запятой. Представление положительных и отрицательных двоичных чисел в прямом, обратном и дополнительных кодах. Выполнение арифметических операций над двоичными числами со знаком.

При проектировании ЭВМ, создании инструментального и прикладного программного обеспечения разработчикам приходится решать вопрос о представлении в ЭВМ числовых данных. Для решения большинства прикладных задач обычно достаточно использовать целые и вещественные числа. Запись целочисленных данных в запоминающем устройстве ЭВМ не представляет затруднений: число переводится в двоичную систему и записывается в прямом коде. Диапазон представляемых чисел в этом случае ограничивается количеством выделенных для записи разрядов. Для вещественных данных обычно используются две формы записи: число с фиксированной точкой (ЧФТ) и число с плавающей точкой (ЧПТ).

Память ЭВМ построена из запоминающих элементов, обладающих двумя устойчивыми состояниями, одно из которых соответствует нулю, а другое - единице. Таким физическим элементом представляется в памяти ЭВМ каждый разряд двоичного числа (бит). Совокупность определенного количества эти элементов служит для представление многоразрядных двоичных чисел и составляет разрядную сетку ЭВМ.

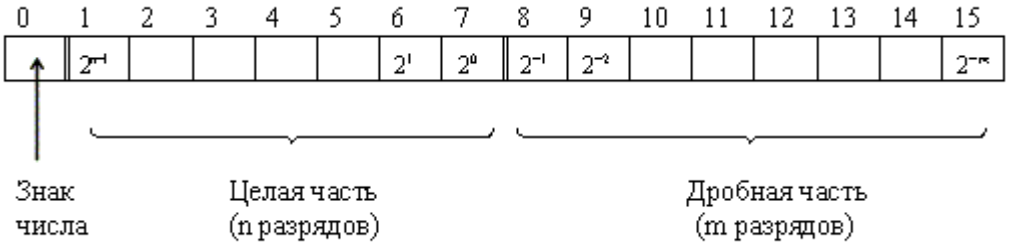
Каждая группа из 8-ми запоминающих элементов (байт) пронумерована. Номер байта называется его адресом. Определенное число последовательно расположенных байт называется словом. Для разных ЭВМ длина слова различна - два, четыре или восемь байт. (Мне думается, что это зависит от разрядности процессора).

Числа с фиксированной точкой

Форма записи числа с фиксированной точкой использовалась в основном на ранних этапах развития вычислительной техники. Запись числа с фиксированной точкой обычно имеет знаковый и цифровой разряды. Фиксированная точка означает, что на этапе конструирования ЭВМ было определено, сколько и какие разряды машинного слова отведены под изображение целой и дробной частей числа. Запятая в разрядной сетке может быть зафиксирована, в принципе, после любого разряда.

Пример.

Ячейка с целой и дробной частью.



Как частный случай числа с фиксированной точкой может быть рассмотрена запись целого числа (в этом случае все разряды, кроме знакового, используются для записи целой части).

Пример.

Ячейка с записью целого числа.

К достоинствам использования чисел с фиксированной точкой относятся простота выполнения арифметических операций и высокая точность изображения чисел. К недостаткам - небольшой диапазон представления чисел.

При представлении в ЭВМ чисел в естественной форме устанавливается фиксированная длина разрядной сетки. При этом распределение разрядов между целой и дробной частями остается неизменным для любых чисел. В связи с эти в информатике существует другое название естественной формы представления чисел - с фиксированной точкой (запятой).

Работая на компьютере, мы можем вводить числа с фиксированной запятой в любом виде. Так же они будут высвечиваться на экране компьютера, но перед занесением в память компьютера они преобразуются в соответствии с разрядной сеткой и хранятся либо с запятой, фиксированной после последнего разряда (целые числа), либо с запятой перед старшим разрядом дроби.

Современные ЭВМ работают в режиме с плавающей точкой, но сохранен и режим работы с фиксированной точкой, который используется преимущемтвенно для представления целых чисел.

Обычно целые числа в ЭВМ занимают один, два или четыре байта. Один, как правило, старший бит отводится под знак числа. Знак положительного числа "+" кодируется нулем, а знак отрицательного числа "-" - единицей. Целые числа без знака в двух байтовом формате могут принимать значения от 0 до 2^{16-1} (до 65535), а со знаком "-" от -2^{15} до 2^{15-1}, то есть от -32768 до 32767.

Во всех разрядах всегда должно быть что-то записано, даже если это "незначащий" ноль. Число располагается так, что его самый младший двоичный разряд записывается в крайний првый бит разрядной сетки. Например, десятичное число 19 (10011_2) в 16-разрядной сетке записывается так:

Достоинством естественной формы являются простота и наглядность представления чисел, простота алгоритмов реализации операций, а следовательно, простота устройств и высокая скорость выполнения операций.

Существенным недостатком машин с фиксированной точкой является конечный диапазон представления величин. Может показаться, что это ограничивает вычислительные возможности ЭВМ. Но на самом деле короткая длина слова приводит только к снижению быстродействия машин: обработка больших чисел ведется последовательно-параллельным способом, сами числа представляются несколькими машинными словами, и для выполнения операций над ними необходимо составлять специальные программы. Поэтому если результат вычислений в естественной форме выходит за допустимые пределы, то в современных компьютерах производиться автоматический переход к представлению данных в экспотанциальной форме (но только если это оговорено программой).

Применение

Для ускорения вычислений в местах, где не требуется высокая точность. В большинстве современных процессоров ФЗ аппаратно не реализована, но даже программная ФЗ очень быстра — поэтому она применяется в разного рода игровых движках, растеризаторах[1] и т. д. Например, движок Doom для измерения расстояний использует фиксированную запятую 16,16, для измерения углов — 360°=65536.

Чтобы обеспечить минимальную поддержку дробных чисел на целочисленном процессоре — микроконтроллера, мобильного телефона, приставок вплоть до Playstation и т. д. Если не решаются некорректные задачи и СЛАУ высокого порядка, фиксированной запятой зачастую достаточно — важно только подобрать подходящую цену (вес) младшего разряда для каждой из величин.

Для записи чисел, которые по своей природе имеют постоянную абсолютную погрешность: координаты в программах вёрстки, денежные суммы. Например, файлы метрики шрифтов TeX используют 32-битный знаковый тип с фиксированной запятой (12,20).

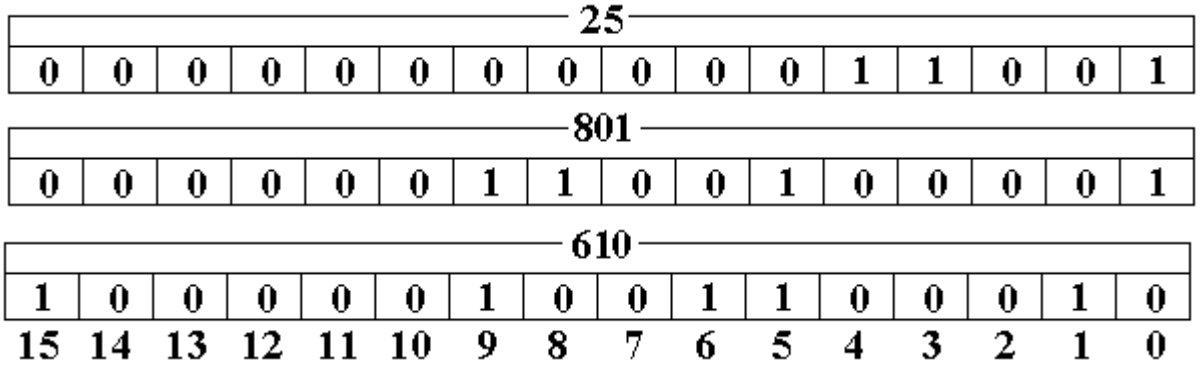
Кроме того, фиксированная запятая ведёт себя абсолютно предсказуемо — при подсчёте денег это позволяет наладить разные виды округления, а в играх — наиболее простой способ реализовать мультиплеер и запись повторов.

Задания

1. Определите максимальные значения целых чисел со знаком и без знака при их 8- и 32-разрядном представлении.

	8 разрядов	32 разряда
со знаком	$2^7-1=127$	$2^{31}-1=2{,}15\times10^9$
без знака	$2^8-1=255$	$2^{32}-1=4{,}29\times10^9$

2. Представьте следующие числа в 16-разрядной сетке в формате с фиксированой точкой: 25, 801, -610.



Числа с плавающей точкой (запятой)

Неудобство представления чисел в форме с фиксированной точкой проявляется при решении задач, в которых фигурируют как очень малые так и очень большие числа. В конкретных физических, математических и других задачах диапазон изменения величин может составлять, например от 10^{-30} до 10^{30}. Можно убедиться, что в представлении с фиксированной запятой понадобились бы двоичные слова длинной около 256 бит (32 байт), по 128 бит на целую и дробную части. Однако работа ЭВМ с операндами такой длины была бы крайне неэффективной.

Точность числа определяется не его длиной, а количеством верных значащих цифр.

Например, мы хотим измерить длину отрезка линейкой с сантиметровыми делениями. Отрезок не совпадает с делениями точно - его

длина между 47 и 47,5 см. На глазок прикидываем, что это 47,2 см (472 мм). Ясно, что в каких единицах ни записать длину отрезка: 472000микрон 472мм. 0,000472км - точных цифр только две: 47 при точности измерения до 1 см (ведь линейка-то сантиметровая).

Точность результата вычисления выражений, содержащих несколько чисел, определяется, как правило, точностью числа имеющего наименьшее количество верных значащих цифр. Поэтому в практических расчетах редко используют более трех значащих цифр, соответствующим образом округляя промежуточные результаты. Ясно, что для хранения в памяти ЭВМ чисел с небольшим числом значащих цифр целесообразно представлять их в экспоненциальной форме. В приведенном примере это представление может иметь вид:

$4{,}72 \times 10^5$; 472×10^3; 4720×10^2микрон

или

$4{,}72 \times 10^{-4}$; $47{,}2 \times 10^{-5}$;472×10^{-6}км.

Из этого примера также видно, что положение запятой может изменяться. Поэтому в информатике представление в ЭВМ числа в экспотенциальной форме называются представлением с плавающей точкой (запятой).

Представление чисел в форме с плавающей точкой очень удобно для решения научных и инженерных задач. Нормализованное представление чисел не только позволяет сохранить в разрядной сетке большое количество значащих цифр, но также упрощает действие над порядками и мантисами.

Для представления чисел с плавающей точкой (ЧПТ) используется полулогарифмическая форма записи числа:

$$N = \pm mq^{\pm p}$$

где q- основание системы счисления, p - порядок числа, m - мантисса числа N.

Положение точки определяется значением порядка p. С изменением порядка точка перемещается (плавает) влево или вправо.

Пример.

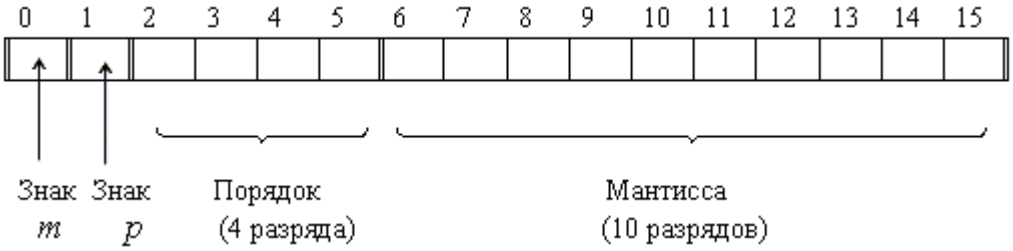
$125_{10}=12.5*10^1=1.25*10^2=0.125*10^3=0.0125*10^4=...$

Для установления однозначности при записи чисел принята нормализованная форма записи числа. Мантисса нормализованного числа может изменяться в диапазоне: $1/q \le |m| < 1$. Таким образом в нормализованных числах цифра после точки должна быть значащей.

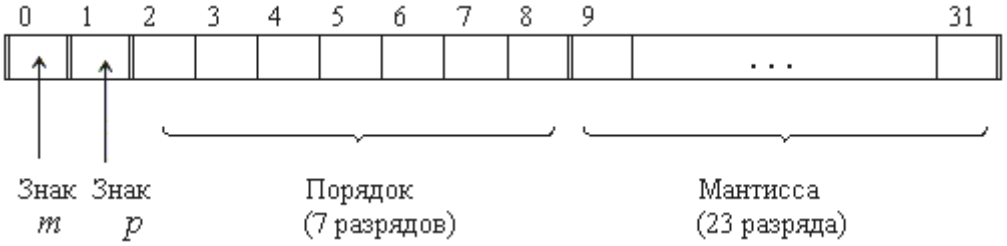
Пример.

$$\underbrace{0.0832 \cdot 10^3}_{\text{ненормализованное число}} = \underbrace{0832 \cdot 10^2}_{\text{нормализованное число}}$$

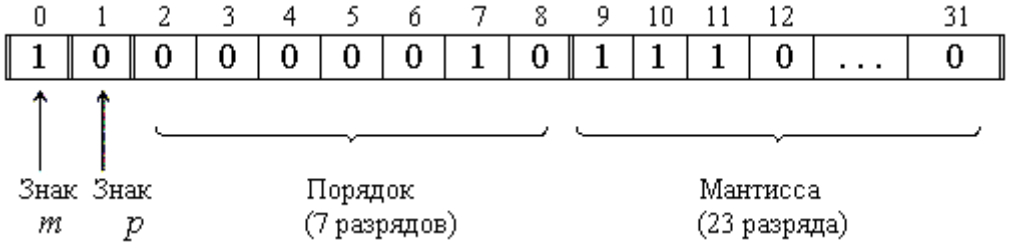
Для представления чисел в машинном слове выделяют группы разрядов для изображения мантиссы, порядка, знака числа и знака порядка:
а) представление чисел в формате полуслова



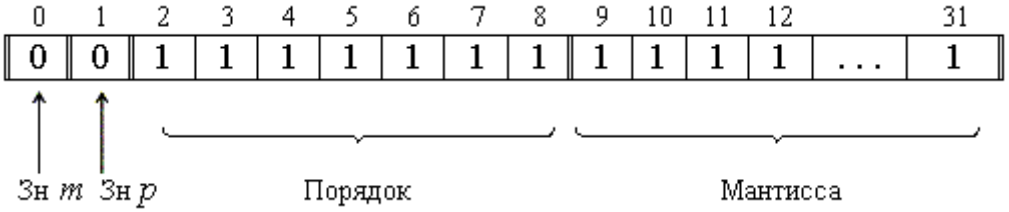
б) представление чисел в формате слова



Наиболее типично представление ЧПТ в формате слова (32 разряда).
Пример.
Число $A=-3.5_{10}=-11.1_2=-0.111\cdot10^{10}$



Максимальным числом представимым в формате слова будет $A=(0.1111...1\cdot10^{1111111})_2\approx(1\cdot2^{127})_{10}$.



Прямой код (ПК) - способ представления двоичных чисел с фиксированной запятой в компьютерной арифметике. Главным образом используется для записи неотрицательных чисел. В случае использования прямого кода для чисел как положительных, так и отрицательных, то есть числа, запись которых подразумевает возможность использования знака минус (знаковых чисел), хранимые цифровые разряды числа дополняются *знаковым разрядом*.

При записи числа в прямом коде старший разряд (старший бит) объявляется *знаковым разрядом*. Если его значение равно 0 — то число положительное, если 1 — то отрицательное. В остальных разрядах (которые называются **цифровыми разрядами**) записывается двоичное представление модуля числа.

десятичный	двоичный	8-разрядный прямой	
0	0	00000000	положительный ноль
-0	-0	10000000	отрицательный ноль
5	101	00000101	
10	1010	00001010	
-5	-101	10000101	
-16	-10000	10010000	

Обратный код (ОК)— метод вычислительной математики, позволяющий вычесть одно число из другого, используя только операцию сложения над натуральными числами. В настоящее время используется в основном в современных компьютерах.

Обратный код положительного числа совпадает с прямым кодом.

Пример. Двоичное представление числа 5 есть 101. Обратный 10-разрядный двоичный код числа +5 записывается как 0000000101.

Следует отметить, что для изменения знака числа достаточно проинвертировать все его разряды не обращая внимания знаковый ли это разряд или информационные.

Пример. Двоичное представление числа 5 есть 101, его 10-разрядное двоичное представление — 0000000101. Обратный 10-разрядный двоичный код числа −5 есть 1111111010 (проинвертировали все знаки).

Дополнительный код (ДК) — наиболее распространённый способ представления отрицательных целых чисел в компьютерах. Он позволяет заменить операцию вычитания на операцию сложения и сделать операции сложения и вычитания одинаковыми для знаковых и беззнаковых (неотрицательных) чисел, чем упрощает архитектуру ЭВМ.

Для получения дополнительного k-разрядного кода отрицательного числа необходимо:

1. модуль отрицательного числа представить прямым кодом в k двоичных разрядах;
2. значение всех бит инвертировать: все нули заменить на единицы, а единицы на нули(таким образом, получается k-разрядный обратный код исходного числа);
3. к полученному обратному коду прибавить единицу.
   Пример:
   Получим 8-разрядный дополнительный код числа -52:
   00110100 - число |-52|=52 в прямом коде

   11001011 - число -52 в обратном коде

   11001100 - число -52 в дополнительном коде

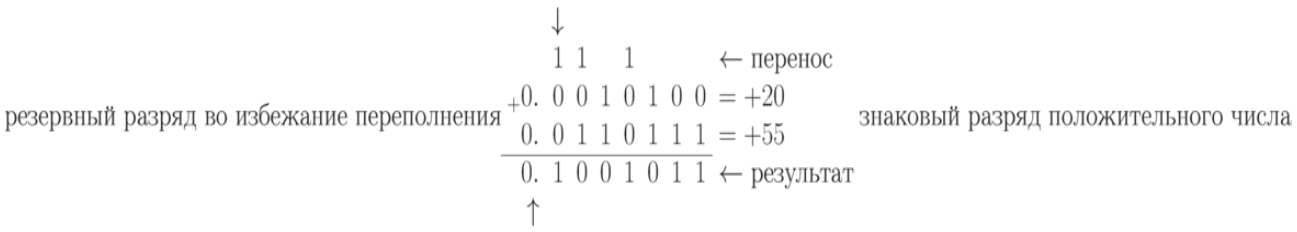
Выполнение арифметических операций над двоичными числами со знаком.

Двоичные числа со знаком (целые) могут занимать также 8 или 16 бит. Значение старшего бита (самого левого) задает знак числа : 0 - положительное, 1 - отрицательное.

1. перевести десятичные числа в двоичный код;
2. уравнять форматы полученных двоичных чисел;
3. если знаки чисел одинаковые, добавить по одному резервному нулю слева от каждого числа во избежание переполнения;
4. получить дополнительные коды чисел;
5. приписать знаковые разряды;
6. сложить полученные коды по правилам двоичного сложения;
7. перенос из знакового разряда (если он есть) отбросить;
8. результат получен в дополнительном коде, поэтому для проверки значащих разрядов отрицательного числа необходимо сделать вычисления, противоположные формуле (11.3): сначала вычислить **обратный код** по формуле $\text{ОК} = \text{ДК} - 1$, после чего произвести инверсию полученного числа.

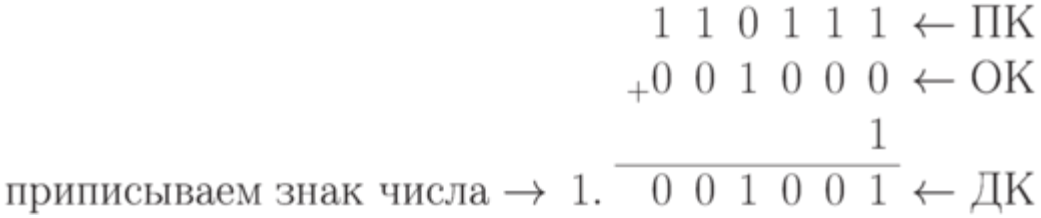
Для наглядности возьмем два десятичных числа, например, 20 и 55, и сделаем все возможные варианты вычислений:

- $20 + 55 = (+20) + (+55) = +75$.



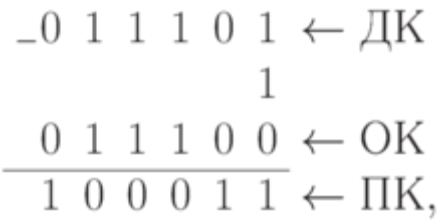
Число положительное, поэтому **ОК=ПК**, для проверки числа нужно перевести его значащие разряды в десятичный код по (П3-2): $1001011_2 = 1 + 2 + 8 + 64 = 75$.

- $20 - 55 = (+20) + (-55) = -35$. Сначала получим дополнительный код отрицательного числа (-55):



Здесь важно уяснить, что крайние левые нули в значащих разрядах сокращать нельзя, поскольку они являются значимыми. Иными словами, все вычисления для каждого примера производятся в неизменном формате, в данном случае в примере (б) - это шесть значащих разрядов, т.е. столько, сколько содержится в большем числе.

Вновь получили знак числа и его значащие разряды, занимающие жестко заданные позиции в выбранном формате числа. Поскольку получено отрицательное число, то **ДК ПК**, для проверки его значащих разрядов нужно сначала вычислить *обратный код*, затем перевести его в прямой код инверсией -



а затем уже перевести его в десятичный код по (П3-2): $100011_2 = 1 + 2 + 32 = 35$.

- $-20 + 55 = (-20) + (+55) = +35$ - Сначала получим **ДК** отрицательного числа (-20).

$$
\begin{array}{r}
0\ 1\ 0\ 1\ 0\ 0 \leftarrow \text{ПК} \\
\quad 1\ 1 \quad \leftarrow \text{перенос при вычислении ДК} \\
+1\ 0\ 1\ 0\ 1\ 1 \leftarrow \text{ОК} \\
\hline
1\ 0\ 1\ 1\ 0\ 0 \leftarrow \text{ДК}
\end{array}
$$

приписываем знак числа → 1.

После этого произведем вычисления:

$$
\begin{array}{l}
\qquad\qquad\quad \downarrow \\
\boxed{1}\ 1\ 1\ 1\ 1 \qquad \leftarrow \text{перенос} \\
1.\ 1\ 0\ 1\ 1\ 0\ 0 = -20 \\
0.\ 1\ 1\ 0\ 1\ 1\ 1 = +55 \\
\hline
0.\ 1\ 0\ 0\ 0\ 1\ 1 \leftarrow \text{результат} \\
\qquad\qquad\quad \uparrow
\end{array}
$$

перенос из знакового разряда (отбрасывается) +

знаковый разряд положительного числа

Получено положительное число, поэтому $\text{ДК} = \text{ПК}$, для проверки результата нужно только перевести значащие разряды в десятичный код: $100011_2 = 1 + 2 + 32 = 35$.

- $-20 - 55 = (-20) + (-55) = -75$ - Сначала получим дополнительный код отрицательных чисел. Для числа (-20) он получается следующим образом:

$$
\begin{array}{r}
\qquad\quad \downarrow \\
\boxed{0}\ 0\ 1\ 0\ 1\ 0\ 0 \leftarrow \text{ПК} \\
\boxed{\ }\quad\ 1\ 1 \quad \leftarrow \text{перенос при вычислении ДК} \\
+\ \boxed{1}\ 1\ 0\ 1\ 0\ 1\ 1 \leftarrow \text{ОК} \\
\hline
\boxed{1}\ 1\ 0\ 1\ 1\ 0\ 0 \leftarrow \text{ДК}
\end{array}
$$

резервный разряд во избежание переполнения

приписываем знак числа → 1.

А для числа (-55) -

$$
\begin{array}{rl}
 & \downarrow \\
 & \boxed{0}\,1\,1\,0\,1\,1 \leftarrow \text{ПК} \\
\text{резервный разряд во избежание переполнения} & +\ \boxed{1}\,0\,0\,1\,0\,0 \leftarrow \text{ОК} \\
 & \boxed{}\qquad\quad 1 \\
\hline
\text{приписываем знак числа} \rightarrow & 1.\ \boxed{1}\,0\,0\,1\,0\,1 \leftarrow \text{ДК}
\end{array}
$$

Сложим полученные числа в том же формате:

$$
\begin{array}{rl}
\text{перенос из знака (игнорируется)} \rightarrow & \boxed{1}\ 1\qquad 1\qquad\quad \leftarrow \text{перенос} \\
 & +\ 1.\ 1\ 1\ 0\ 1\ 1\ 0\ 0 = -20 \\
 & \quad 1.\ 1\ 0\ 0\ 1\ 0\ 0\ 1 = -55 \\
\hline
 & \quad 1.\ 0\ 1\ 1\ 0\ 1\ 0\ 1 \leftarrow \text{результат} \\
 & \quad\ \uparrow \\
 & \quad \text{знаковый разряд отрицательного числа}
\end{array}
$$

Поскольку число отрицательное, $\text{ДК} \neq \text{ПК}$. Для проверки значащих разрядов числа нужно сначала вычислить *обратный код*, после чего перевести его в прямой код инверсией -

$$
\begin{array}{r}
\_0\ 1\ 1\ 0\ 1\ 0\ 1 \leftarrow \text{ДК} \\
\qquad\qquad\ 1 \\
\hline
0\ 1\ 1\ 0\ 1\ 0\ 0 \leftarrow \text{ОК} \\
\hline
1\ 0\ 0\ 1\ 0\ 1\ 1 \leftarrow \text{ПК.}
\end{array}
$$

И только после этого полученное число проверяется переводом в десятичный код по (11.2): $1001011_2 = 1 + 2 + 8 + 64 = 75$.

Тема 1.2:Представление информации в ЭВМ.

Лекция 3. Виды информации и способы ее представления в ЭВМ. Классификация информационных единиц, обрабатываемых ЭВМ. Числовые и нечисловые типы данных и их виды. Структуры данных и их разновидности. Форматы файлов.

Для автоматизации работы с данными, относящимися к различным типам, очень важно унифицировать их форму представления — для этого обычно используется прием *кодирования,* то есть выражение данных одного типа через данные другого типа. Естественные человеческие *языки* — это не что иное, как системы кодирования понятий для выражения мыслей посредством речи. К языкам близко примыкают *азбуки* (системы кодирования компонентов языка с помощью графических символов).

История знает интересные, хотя и безуспешные попытки создания «универсальных» языков и азбук. Та же проблема универсального средства кодирования достаточно успешно реализуется в отдельных отраслях техники, науки и культуры. В качестве примеров можно привести систему записи математических выражений, телеграфную азбуку, морскую флажковую азбуку, систему Брайля для слепых и многое другое. Своя система существует и в вычислительной технике — она называется *двоичным кодированием* и основана на представлении данных последовательностью всего двух знаков: 0 и 1. Эти знаки называются *двоичными цифрами,* по-английски — *binarydigit* или, сокращенно, *bit (бит).*

Одним битом могут быть выражены два значения: 0 или 1 *(да* или *нет, черное* или *белое, истина* или *ложь* и т. п.). Если количество битов увеличить до двух, то уже можно выразить четыре различных значения:
00 01 10 11

Тремя битами можно закодировать восемь различных значений:
000 001 010 011 100 101 110 111

Увеличивая на единицу количество разрядов в системе двоичного кодирования, мы увеличиваем в два раза количество значений, которое может быть выражено в данной системе, то есть общая формула имеет вид:
$N=2^m$, где:
N— количество независимых кодируемых значений;
m— разрядность двоичного кодирования, принятая в данной системе.

Типы и структуры данных

Символьные, числовые и логические данные (переменные) представляют собой простейшие и основные типы данных, хранимых и обрабатываемых в ЭВМ.

Типы данных.

Ранние языки программирования (ЯП), а точнее, системы программирования (СП) — Фортран, Алгол, будучи ориентированы исключительно на вычисления, не содержали развитых систем типов и структур данных. В ЯП Алгол символьные величины и переменные вообще не предусматривались, в некоторых реализациях строки (символы в апострофах) могли встречаться только в операторах печати данных.

Типы числовых данных Алгола: integer (целое число), real (действительное) — различались диапазонами изменения, внутренними представлениями и применяемыми командами процессора ЭВМ (соответственно арифметика с фиксированной и плавающей точкой). Нечисловые данные были представлены типом boolean — логические, имеющие диапазон значений {true, false}.

Позже появившиеся ЯП (СП) COBOL, PL/1, Pascal вводят новые типы данных:

-символьные (цифры, буквы, знаки препинания и пр.);

-числовые символьные для вывода;

-числовые двоичные для вычислений;

-числовые десятичные (цифры 0—9) для вывода и вычислений.

Разновидности числовых данных здесь соответствуют внутреннему представлению и машинным (или эмулируемым) командам обработки. Кроме того, вводятся числа двойного формата (2 машинных слова), для обработки которых также необходимо наличие в процессоре (или эмуляция) команд обработки чисел двойной длины (точности).

Уместно привести пример представления числовой информации в различных перечисленных формах. Пусть задано число $135_{10} = 207_8 = 87_{16} = 100\ 0\ 00\ 111_2$ тогда:

• внутренняя стандартная форма представления (тип binary для обработки в двоичной арифметике) сохраняется ($100\ 000\ 111_2$). Объем — 1 байт, или 8 двоичных разрядов;

• внутренняя форма двоично-десятичного представления (тип decimal, каждый разряд десятичного числа представляется двоично-десятичной, в 4 бита, комбинацией). Представление 135 есть $001\ 011\ 101_2$. Объем — 2,5 байта, 12 двоичных разрядов;

• символьное представление (тип alphabetic, для вывода) — каждый разряд представляется байтом в соответствии с кодом ASCII (табл, приложения 2). Представление 135 есть - 00110001 00110011 00110101_2. Объем - 3 байта.

Появление систем управления базами данных и систем программирования для разработки ИС приводит к появлению ряда других типов данных:

• дата и время;

• бинарные (BLOB — Binary Large Object) и текстовые объекты без внутренней структуры (интерпретация возлагается на прикладные программы).

Понятие типа данных ассоциируется также с допустимыми значениями переменной и операциями над ними, например, данные типа время (чч:ММ:СС) или дата (гг/мм/дд) предполагают определенные диапазоны значений каждого из разрядов, а также машинные или эмулируемые операции (сложение/вычитание дат и/или моментов времени).

Структуры данных.

В языке Алгол определены два типа структур: элементарные данные и массивы (векторы, матрицы, тензоры, состоящие из арифметических или логических переменных — рис. 1.2, а, б, в). Основным нововведением, появившимся первоначально в Коболе (затем ПЛ/1, Паскаль и пр.), являются агрегаты данных (структуры или записи), представляющие собой именованные комплексы переменных разного типа, описывающих некоторый объект илиобразующих некоторый достаточно сложный документ (рис. 1.2, г).

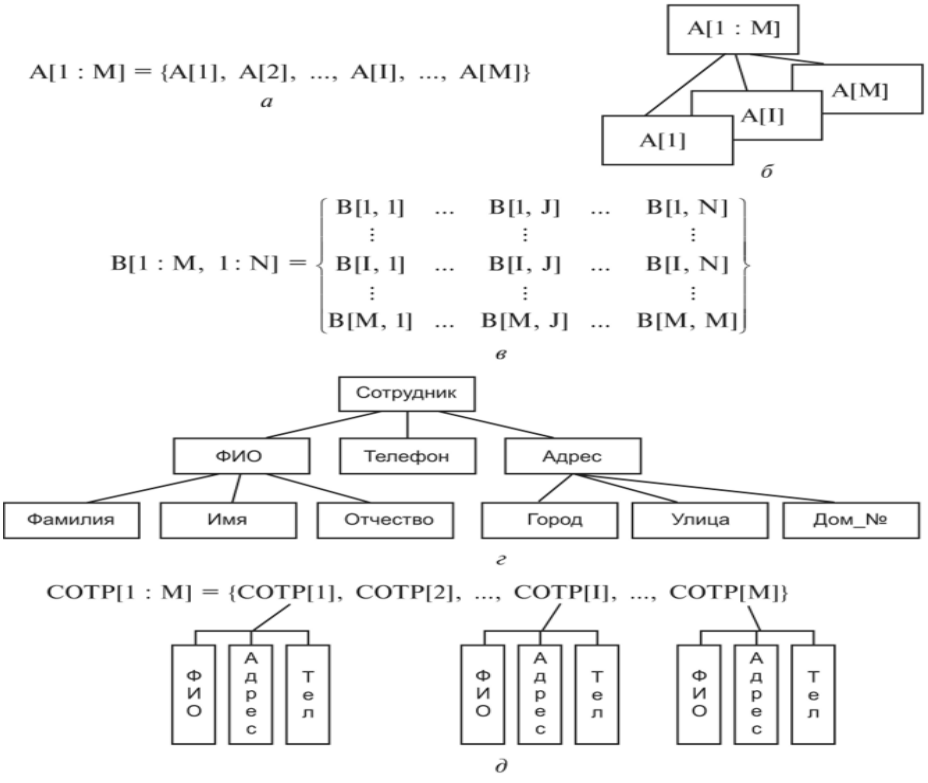


Рис. 1.2. Структуры данных:

а, б — одномерный массив (вектор); в — двумерный массив (матрица); г — запись (структура, агрегат данных); д — массив в записи (множественное поле записи)

Термин запись подразумевает наличие множества аналогичных по структуре агрегатов, образующих файл (картотеку) и содержащих данные по совокупности однородных объектов. Элементы данных образуют поля, среди которых выделяются элементарные и групповые (агрегатные). В языках ПЛ/1, Паскаль появляются массивы агрегатов/записей (рис. 1.2, д).

С появление СУБД и АИПС возникают новые разновидности структур :

• множественные поля данных;

• периодические групповые поля;

• текстовые объекты (документы), имеющие иерархическую структуру (документ, сегмент, предложение, слово).

Файлы и файловые системы

Рассмотренные выше (а также и ряд других) типы и структуры данных связываются с внешними устройствами ЭВМ через концепцию файла (структурированного набора данных). Физическое размещение данных на периферийных устройствах и логическое «восприятие» их прикладными программами осуществляются операционными системами (ОС) и системами управления базами данных (СУБД), которые реализуют функцию управления данными, центральное место в которой принадлежит именно файлам. На уровне ОС осуществляется связь между адресом данных и именем (файла). На уровне СУБД — между содержимым и адресом данных [7].

Понятие файла появляется впервые в операционной системе OS/360 фирмы IBM, причем в ранних версиях системы «настоящим файлом» считался только перфокарточный массив («file» = «картотека»), данные на МД и МЛ обозначались как DS (Data Set — набор данных). В последующих ОС (RSX, UNIX, MS DOS) файлами становятся именованные организованные наборы данных на любых носителях и устройствах, за сохранность и обновление которых (а также передачу в прикладные программы/из прикладных программ) несет ответственность ОС ЭВМ.

В качестве простого примера рассмотрим структуру файлов на МЛ, которая была принята в OS/360 (так называемая лента без меток — nl)

Хотя магнитные ленты для цифровой записи данных размещаются на бобинах или кассетах (подобно лентам для бытовой аудио- или видеозаписи), принципы размещения информации на МЛ в данном случае существенно другие (рис. 1.3):

- информация размещается на носителе в виде блоков (массивов данных фиксированной или переменной длины);

- информационные блоки разделены пустыми промежутками {gap), позволяющими считывающему устройству распознать начало (окончание) блока. Размер промежутка между записями выбирается достаточным для разгона ленты до установленной скорости и остановки ее точно на следующем промежутке. Недостаток промежутков между записями — уменьшение полезного объема МЛ, так как области, отведенные под промежутки, нельзя использовать для хранения данных. Частично указанный недостаток устраняет процесс блокирования, суть которого состоит в объединении нескольких записей в блоки;

- блоки подразделяются на информационные (ИБ — распознаются программами) и служебные (распознаются устройством — конец файла, конец тома и пр.);

- физическое начало и физический конец ленты обычно определяются оптическим или механическим датчиком (независимо от содержания ленты).

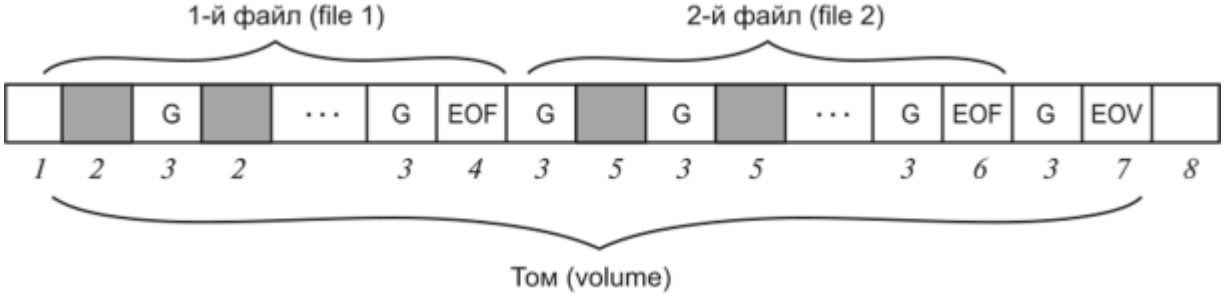


Рис. 1.3. Структура данных на магнитных лентах:

1 — физическое начало ленты (начальный ракорд); 2 — информационные блоки (ИБ) 1-го файла; 3 — GAP, промежуток между блоками; 4 — конец файла (eof, end-of-file), служебный блок, задающий конец 1-го файла; 5 — информационные блоки (ИБ) 2-го файла; 6 — конец 2-го файла; 7 — служебный блок, задающий логический конец ленты (eov, end-of-volume); 8 — физический конец ленты.

Максимальное ограничение на размер блока зависит от размера доступной оперативной памяти (возможность размещения буфера считывания

файла). Блокирование увеличивает полезный объем магнитной ленты за счет сокращения числа промежутков между записями. Кроме того, уменьшается количество операций ввода-вывода, так как за одну операцию производится пересылка не одной записи, а сразу нескольких. Преимущества блокирования, заключающиеся в увеличении полезного объема МЛ и уменьшении общего времени работы программы на ввод-вывод данных, значительно превосходят возникающие при этом недостатки, связанные с увеличением объемов данных в программе пользователя и необходимостью выполнения процедур по формированию блоков и разделению принятых блоков на записи. Рассмотрим данный вопрос подробнее.

Блокирование (физические и логические записи). На логическом уровне («взгляд» прикладной программы) файл обычно представляет собой совокупность логических записей одинаковой структуры, каждая из которых — набор (агрегат) разнородных данных (см. рис. 1.2). Логическая запись обычно размещается в рабочей области памяти прикладной программой отдельно, как единое целое.

Физическая запись, с которой работает файловая система — это совокупность данных, которые размещаются в файле на внешнем носителе, и могут быть считаны или записаны как единое целое одной командой ввода-вывода (информационные блоки на рис. 1.3).

Организация данных в случаях логического и физического представления может не совпадать, в частности, одна физическая запись может включать несколько логических (блокирование записей — рис. 1.4, а). При этом алгоритмы выделения логических записей из физической в значительной степени зависят

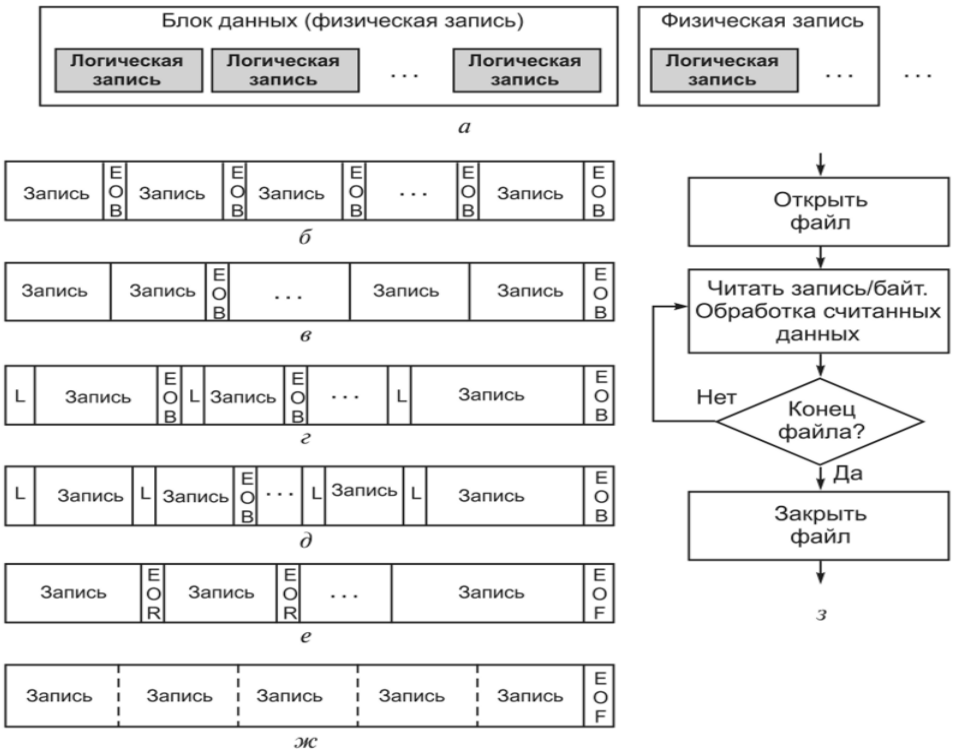


Рис. 1.4. Организация и обработка файлов данных: а — логические и физические записи; б — записи фиксированной длины (ФД) несблокированные; в — записи ФД с блокировкой; г — записи переменной длины (ПД); д — записи ПД с блокировкой; е — записи неопределенной длины; ж — потоко-ориентированный файл; з — типичный цикл обработки файла; EOB — конец блока; EOF — конец файла; EOR — конец записи; L — байты длины записи

от ***типа записи,*** рассматриваемого как характер организации последовательности байтов.

На логическом уровне выделяют следующие ***типы записей*** (рис. 1.4):

• фиксированной длины, для размещения каждой из которых на носителе выделяется память постоянного объема, объявляемого заранее. В этом случае данные, образующие запись, имеют устойчивую природу и представляются жесткими структурами, например ряд числовых полей или символьная последовательность заданной длины;

• переменной длины, когда каждый экземпляр записи может иметь длину, отличную от длины другой в том же наборе. В этом случае запись содержит либо элементы данных переменной длины (например, текстовые строки), либо переменное число элементов фиксированной длины. Структура представления логической записи здесь отличается тем, что байтам содержания (собственно

данным, образующим логическую запись) предшествуют байты значения длины содержания этой логической записи (L на рис. 1.4).

• неопределенной длины — представление записей, имеющих переменную длину, когда данные, образующие логическую запись, завершаются разделителем конец записи (или терминатором записи). Порядок доступа к записям может быть только последовательным, поскольку для определения точки начала следующей записи надо знать значение длины предшествующей.

Конец блока (еов) во всех рассмотренных случаях — обычно уже известная пустая область (gap, см. рис. 1.3), хотя это может быть и особый код, распознаваемый устройством, т. е. тот же разделитель. Однако в записях переменной или неопределенной длины в начале блока могут размещаться байты длины блока (некоторое значение L', используемое аналогично L).

Цикл обработки файла (например, внесение изменений в счета клиентов) включает следующие операции (см. рис. 1.4):

• открытие файла — занятие устройства, на котором файл размещен (например, МД), создание в ОП управляющего блока, в котором записывается справка о состоянии файла и буфера (или набора буферов — буферного пула) для хранения текущей, обрабатываемой записи файла;

• организация цикла, управляемого файлом (заканчивается по исчерпании записей/байтов файла — наступлении состояния EOF — end-of-f ile), после этого выполняется некоторый оператор, обычно завершения обработки. Цикл должен содержать команду типа read, get (ввод записи) или put, write (вывод записи) либо rewrite (обновить за- пись). Команда read может являться функциональным аналогом заголовка цикла;

• закрытие файла — выполнение операций по внесению всех окончательных изменений в файл и его реквизиты, освобождение памяти, отведенной под файл, устройства, на котором он размещался, и других связанных ресурсов.

Файловые системы (ФС).

Операционными системами на каждом томе (дискете, диске, пакете дисков, CD-ROM и пр.) создается совокупность системных данных, которая называется файловой системой (файловой структурой).

Файловая система (пустая) создается при инициализации (разметке) тома, затем корректируется ОС (подсистемой управления данными) при текущей работе, в процессе создания, удаления, модификации (увеличения или уменьшения объема) файлов пользователя, содержащих программы или данные.

Типы файлов в ОС.

В системе OS основную роль играли два типа файлов:

• символьные (исходные программы или данные);

• двоичные (программы в машинных кодах).

В современных системах активно используется значительно большее разнообразие файлов, из которых мы перечислим наиболее типичные (табл. 1.6):

текстовые файлы — обобщенное название для простых и размеченных текстов, ASCII-файлов и других наборов данных символьной информации, которые интерпретируются и обрабатываются текстовыми редакторами, процессорами, анализаторами (Lexicon, Word, TEC, анализаторы SGML, HTML);

• текст без разметки (планарный, «плоский») — файл, содержащий только отображаемые (воспроизводимые на всех печатающих устройствах и терминалах) символы кода ASCII, а также простейшие управляющие символы: возврат каретки (cr); перевод строки (lf); символ табуляции (tab), иногда — новая страница (lf);

• текст с разметкой — планарный файл, содержащий бинарную (обычно ESC-последовательности) или символьную раз- метку, управляющую отображением информации (программно и/или аппаратурной;

• ASCII-файл содержит только отображаемые коды кодовой таблицы ASCII (латиница и служебные символы), обычнопременяется для хранения документов с символьной разметкой (RTF, SGML, HTML);

табличный файл содержит форматированные данные (символьные, численные и др.), образующие строки и столбцы таблиц, создаваемых и обрабатываемых табличными СУБД (FoxPro, Clipper, MS Access) и/или процессорами (Super- Calc, MS Excell и др.);

• графический файл — бинарный файл, содержащий графическую информацию. Форматы: tif (Tagged Image File), bmp (Bit-Mapped Picture), атакжеряддругих — PCX, PIC ит. д.;

• мультимедийные файлы — бинарные, содержащие оцифрованную аудио- (типы wav или MIDI-Sequencer) видео- формат MPEG) или смешанную информацию.

Лекция 4. Кодирование информации. Символьные коды: ASCII и UNICODE и др. Кодирование графической информации. Двоичное кодирование звуковой информации. Сжатие информации. Кодирование видеоинформации. Стандарт MREG.

Термин «информация» (от лат. informatio – «разъяснение, изложение, осведомлённость») в настоящее время не имеет строгого определения и в сущности обозначает некоторые сведения о чем-либо, совокупность каких-либо данных, знаний, сообщений и т.п

Активно используемая обществом информация рассматривается как важный ресурс наряду с интеллектуальными, энергетическими и материальными ресурсами. Следует заметить, что переработка полученной информации приводит к появлению новой информации.

Передача информации обычно предполагает наличие двух объектов – источника информации и её потребителя – приёмника (адресата) информации. Чтобы информация могла быть передана от источника к приёмнику, состояние источника необходимо отразить во внешней среде, воздействующей на приёмные органы адресата. Материальные объекты, посредством которых происходит перенос информации от источника к приёмнику, называются носителями информации. Например, при общении людей информация между ними передаётся с помощью звуковых колебаний, визуальных образов, запахов и прочего и улавливается органами чувств.

ЭВМ, являясь полностью искусственной технической системой, требует принципиально иной способ введения в неё информации. Другими словами, информацию, представленную в виде, понятном для людей, необходимо преобразовать в форму, пригодную для обработки компьютером.

Преобразование множества состояний источника информации в множество состояний носителя информации называется кодированием. В

сущности, код представляет собой набор некоторых условных знаков (символов) и правил их составления. Организованная последовательность этих символов называется ***кодовой комбинацией.*** С этой точки зрения алфавит, т.е. набор букв, представляет собой код, с помощью которого можно преобразовать устную речь в рукописный или печатный текст.

В цифровой вычислительной технике применяются двоичные (бинарные) коды, для которых количество условных знаков равно двум (знаки «0» и «1»). Из этих символов составляется кодовая комбинация в виде последовательности нулей и единиц, образуя числа в двоичной системе счисления. Таким образом, всего двух символов достаточно для кодировки любой информации.

Использование двоичного кода позволяет дать количественную оценку информации, т.е. узнать, сколько информации содержит то или иное сообщение. Извлечение информации предусматривает получение ответа на интересующий нас вопрос и связано с совершением какого- либо действия.

В соответствии с вероятностным подходом к определению количества информации, два символа двоичной знаковой системы можно рассматривать как два различных возможных события. Единица для оценки количества информации при условии двоичного кодирования получила название ***бит*** (от англ. binary digit – двоичная цифра).

Следующей по величине единицей измерения количества информации является байт, представляющий собой последовательность, составленную из восьми бит, т.е. 1 байт = 2^3 бит = 8 бит.

В вычислительной технике и информатике также широко используются кратные байту единицы измерения количества информации,однако в отличие от метрической системы мер, где в качестве множителей кратных единиц применяют коэффициент 10^n, в кратных единицах измерения количества информации используется коэффициент 2^n.

Выбор этот объясняется тем, что компьютер работает с числами не в десятичной, а в двоичной системе счисления.

Кратные байту единицы измерения количества информации вводятся следующим образом:

1 Килобайт (Кбайт) = 2^{10} байт = 1024 байт;

1 Мегабайт (Мбайт) = 2^{10} Кбайт = 1024 Кбайт;

1 Гигабайт (Гбайт) = 2^{10} Мбайт = 1024 Мбайт;

1 Терабайт (Тбайт) = 2^{10} Гбайт = 1024 Гбайт и т.д.

Количество информации, которое несёт один знак алфавита, тем больше, чем больше знаков входит в этот алфавит. Количество информации V, содержащееся в сообщении, закодированном с помощью знаковой системы и

содержащем определённое количество знаков, определяется с помощью формулы

$$V = K \log_2 N ,$$

где K – количество знаков в сообщении; N – количество знаков в алфавите (мощность алфавита).

Следует заметить, что для удобства вычислений логарифмов по основанию 2 (поскольку большинство калькуляторов работают только с десятичными логарифмами), в соотношениях (1.1) – (1.3) можно использовать формулу перевода

$$\log_2 N = \frac{\log_{10} N}{\log_{10} 2} .$$

Для пояснения вышеизложенного рассмотрим несколько практических примеров.

Пусть необходимо оценить, какое количество информации можно получить после реализации одного из пяти событий. Вероятности событий составляют: p1 = 0,2; p2 = 0,1; p3 = 0,3; p4 = 0,4; p5 = 0,2.

Подставляя заданные значения вероятностей в формулу (1.1) и с учетом (1.4), получим искомое количество информации в битах:

$$I = -\sum_{i=1}^{5} p_i \frac{\log_{10} p_i}{\log_{10} 2} =$$

$$= -\left(p_1 \frac{\log_{10} p_1}{\log_{10} 2} + p_2 \frac{\log_{10} p_2}{\log_{10} 2} + p_3 \frac{\log_{10} p_3}{\log_{10} 2} + p_4 \frac{\log_{10} p_4}{\log_{10} 2} + p_5 \frac{\log_{10} p_5}{\log_{10} 2} \right) =$$

$$= -\left(0{,}2 \frac{\log_{10} 0{,}2}{\log_{10} 2} + 0{,}1 \frac{\log_{10} 0{,}1}{\log_{10} 2} + 0{,}3 \frac{\log_{10} 0{,}3}{\log_{10} 2} + 0{,}4 \frac{\log_{10} 0{,}4}{\log_{10} 2} + 0{,}2 \frac{\log_{10} 0{,}2}{\log_{10} 2} \right) =$$

$$= -(-0{,}464 - 0{,}332 - 0{,}521 - 0{,}529 - 0{,}464) = 2{,}31.$$

Полученный результат выразим в байтах:

$$I = 2{,}31/8 = 0{,}28875 \text{ байт.}$$

Рассмотрим другой пример. Пусть требуется определить количество символов в алфавите, с помощью которых было передано сообщение объёмом 4 Кбайт, содержащее 8192 знака.

Решение задачи основано на использовании формулы (1.3), однако для её использования необходимо перевести заданный объём сообщения из килобайт в биты:

$$V = 4 \text{ Кбайт} = 4096 \text{ байт} = 32768 \text{ бит.}$$

Определим количество бит, приходящееся на один символ (количество информации, содержащейся в одном символе) алфавита:

$$I = 32768/8192 = 4 \text{ бит.}$$

Тогда согласно (1.3), мощность алфавита составит $N = 2^I = 2^4 = 16$ знаков.

Кодирование текстовой информации.

Для представления информации в ЭВМ используются разные правила кодирования. Так, для ввода всех символов с клавиатуры компьютера, используют восьмиразрядные двоичные комбинации нулей и единиц. Различные кодовые комбинации соответствуют строчным и прописным буквам латинского и русского алфавитов, арабским цифрам, знакам препинания и математических операций, а также некоторым специальным символам (#, @, $) и др. Максимальное количество таких комбинаций составляет $2^8 = 256$. Например в таблице ASCII одним байтом кодируются 256 символов. Согласно этой кодировке букве **b** соответствует код 01100010, **o** – 01101111, **k** – 01101011. И слово **book**записывается четырьмя байтами – 01100010 01101111 01101111 01101011.

1.1. Структура кодовой таблицы *ASCII*

Порядковые номера символов		Назначение
десятичный код	двоичный код	
0 – 31	00000000 – 00011111	Специальные символы для управления выводом текста на экран или печать, подачи звукового сигнала, разметки текста и т.п.
32 – 127	00100000 – 01111111	Стандартная часть таблицы. Сюда входят строчные и прописные буквы латинского алфавита, десятичные цифры, знаки препинания, всевозможные скобки, математические, денежные и другие символы
128 – 255	10000000 – 11111111	Альтернативная часть таблицы (кодовая страница). В русских национальных кодировках в этой части таблицы размещаются символы русского алфавита

Для русского алфавита в настоящее время действуют пять различных кодировок кириллицы, реализованных в кодовых страницах KOI8-R, Windows, DOS, Macintosh и ISO. Вследствие этого иногда возникают проблемы с переносом текста на кириллице из одной программной системы в другую. Исторически одним из первых отечественных стандартов кодирования русских букв на компьютерах был КОИ8 (Код Обмена Информацией, 8-битный). Эта кодировка применялась ещё в 1970-е гг. на компьютерах серии ЕС ЭВМ. В настоящее время используется в русифицированных версиях UNIX-подобных операционных систем (Linux, FreeBSD, Solaris и др.).

Компьютеры фирмы Apple, работающие под управлением операционной системы Mac OS (от англ. Macintosh Operation System), используют свою собственную кодировку Mac.

Кроме того, Международная организация по стандартизации (InternationalStandards Organization, ISO) утвердила в качестве стандарта для русского языка ещё одну кодировку под названием ISO 8859-5.

Сейчас наиболее распространённой во всём мире является операционная система Microsoft Windows, в различных версиях которой основной кодировкой для русского языка является кодовая страница CP1251.

В конце 1990-х годов для решения проблем стандартизации символьного кодирования был введен международный стандарт Unicode.

Это 16-разрядная кодировка, в которой на каждый символ отводится не один, как в ASCII, а целых два байта памяти. Такая кодовая таблица допускает включение до 65 536 символов. Полная спецификация стандарта Unicode включает в себя все существующие, вымершие и искусственно созданные алфавиты мира, а также множество математических, музыкальных, химических и прочих символов.

Кодирование графической информации.

Таким способом в память компьютера осуществляется ввод текстовой информации. Ввод графической информации (рисунки, фотографии, чертежи и т.п.) также выполняется в виде двоичных кодовых комбинаций. В сущности, картинка в памяти компьютера представляет собой совокупность точек, окрашенных в два цвета (чёрно-белый рисунок) или в один из цветовых оттенков (цветное изображение).

Пусть разрешение рисунка, определяемое количеством точек – пикселей (от англ. pixels), формирующим изображение, составляет 1280 в горизонтальном и 1024 в вертикальном направлении. Тогда общее количество пикселей составит 1280 × 1024 = 1 310 720. Для обозначения каждого пикселя изображения можно использовать двоичные цифры. Если рисунок чёрно-белый, то таких цифр будет две, например, чёрной точке будет соответствовать цифра 1, а белой – 0. В этом случае весь рисунок займет в памяти 1 310 720 бит, или 163,84 Кбайт.

В случае цветного изображения, объём занимаемой им памяти существенно возрастает и зависит от количества цветовых оттенков, которые могут принимать отдельные пиксели, формирующие эту картинку. Известно, что любой цвет получается путём смешивания трёх базовых цветов: красного (Red), зелёного (Green) и синего (Blue).

Представление цвета в компьютере при помощи этих трёх составляющих получило название ***RGB-кодирования***. Каждый цвет занимает в памяти 1 бит, поэтому для хранения цвета, состоящего из трёх составляющих, требуется 3 бита двоичного кода, при этом количество цветовых комбинаций составит $2^3 = 8$. Такими характеристиками обладали первые цветные видеоадаптеры. Конечно, восьми цветовых оттенков недостаточно для формирования полноценного цветного изображения. Человеческий глаз способен улавливать огромное количество (по некоторым оценкам, до 10 млн) цветовых оттенков, поэтому вскоре для кодирования цвета добавили четвертый бит, управляющий интенсивностью (яркостью) свечения базовых цветов, что позволило использовать 16-цветную палитру. Дальнейшие исследования показали, что очень большое количество цветовых оттенков можно получить при раздельном управлении интенсивностью базовых цветов, причём для интенсивности каждого из базовых цветов можно выделять больше одного бита видеопамяти.

Количество бит, выделенных для записи цвета одного пикселя, называется ***глубиной цвета,*** а максимальное количество цветов зависит от глубины цвета. Например, для получения цветовой гаммы в 256 цветов требуется 8 бит памяти на каждый пиксель, поскольку 256 = 28. В современных компьютерах для хранения и отображения цветной графической информации в основном используется 24- или даже 32-битное RGB-кодирование, позволяющее представить 232 различных цветовых оттенков, что даже больше, чем способен различить глаз человека, поэтому за рубежом такое высококачественное кодирование графической компьютерной информации часто называют True Color (от англ. «Истинный цвет»).

Общий объём памяти, необходимый для хранения цветного изображения, легко оценить по формуле

$$I = RC,$$

где R – разрешение изображения, пикселей; C – хроматическое число(количество разрядов кодирования), бит/пиксель.

Например, в случае 32-разрядного двоичного кодирования

изображения разрешением 1024×768 = 786 432 пикселей потребуется

I = 786 432×32 = 25 165 824 бит, или 3 Мбайт памяти.

В таком виде сохраняется информация в графических файлах с расширением BMP. Но для уменьшения занимаемого объема применяются различные методы сжатия типа JPG, GIF и т.д.

Кодирование видео-информации.

Поток видеоданных можно представить в виде быстро сменяющих друг друга неподвижных кадров, и при большой скорости смены кадров,

оцениваемой в количестве кадров за секунду, необходимый объём памяти для хранения изображения можно определить, добавив в формулу (1.5) дополнительные сведения:

$$I = RCKt\ ,$$

где K – скорость смены кадров, кадр/с; t – длительность видеопотока, с.

Например, если, как и в случае предыдущего примера, для хранения одного кадра видеоинформации требуется 3 Мбайт памяти, а в течение секунды друг друга сменяют 24 кадра, то для записи потребуется I = 3×24 = 72 Мбайт памяти.

Легко видеть, что объём занимаемой памяти для хранения всего одной секунды видеоинформации достаточно велик. Нетрудно подсчитать, что видеофильм высокого разрешения продолжительностью один час для компьютерного представления потребует I = 72×3600 = 259 200 Мбайт, или 253,125 Гбайт. Такой объём занимаемой памяти очень велик даже по современным меркам (ведь на жёстком диске ПК ёмкостью 1 Тбайт уместится всего 4 таких видеофильма). Решением этой проблемы является разумный компромисс между качеством изображения и объёмом занимаемой им памяти,поэтому большинство видеокадров имеют разрешение 320×200 или 640×480 пикселей.

Кодирование звуковой информации.

Звуковую информацию также можно представить двоичным кодом.

Вначале с помощью микрофона звуковые колебания преобразуются непрерывно в изменяющийся во времени гармонический электрический сигнал. Затем такой сигнал делают прерывистым (дискретным). Этот процесс получил название дискретизации, или квантования.

Принципы дискретизации сигнала показаны на рис. 1.17. Выполняются два процесса дискретизации – по времени и по уровню. Исходный непрерывный электрический сигнал имеет бесконечное множество значений и определён в каждый момент времени. Квантование по времени состоит в том, что берутся отсчёты с интервалом времени T (рис. 1.10, а). В этом случае непрерывный аналоговый сигнал заменяется последовательностью отсчётов, величина которых может быть равна значению сигнала в данный момент времени. Академик В. А. Котельников доказал, что для передачи непрерывно изменяющихся по периодическому закону сигналов достаточно передавать их мгновенные значения с интервалом, вдвое превышающим период колебаний при максимальной частоте изменения этих сигналов.

При квантовании по уровню все возможные непрерывные значения электрического сигнала разбиваются на n уровней. Следовательно,в каждый

момент времени звуковому сигналу может быть присвоено значение максимального достигнутого в данный момент уровня.

Для простоты на рис. 1.10, б показано всего девять возможных уровней, а соответствующие им двоичные кодовые комбинации показаны на этом же рисунке справа.

Таким образом, для передачи, хранения и обработки текстовой, графической, видео- и аудиоинформации можно применять двоичные кодовые комбинации. Такое внутреннее представление информации используется в современных средствах вычислительной техники. Ввод информации от внешних источников осуществляется при помощи различных специальных технических устройств (клавиатура, сканер, видеокамера, звукозаписывающая аппаратура и др.), обладающих возможностью преобразовать входящий разнородный информационный поток в упорядоченный набор двоичных кодовых комбинаций.

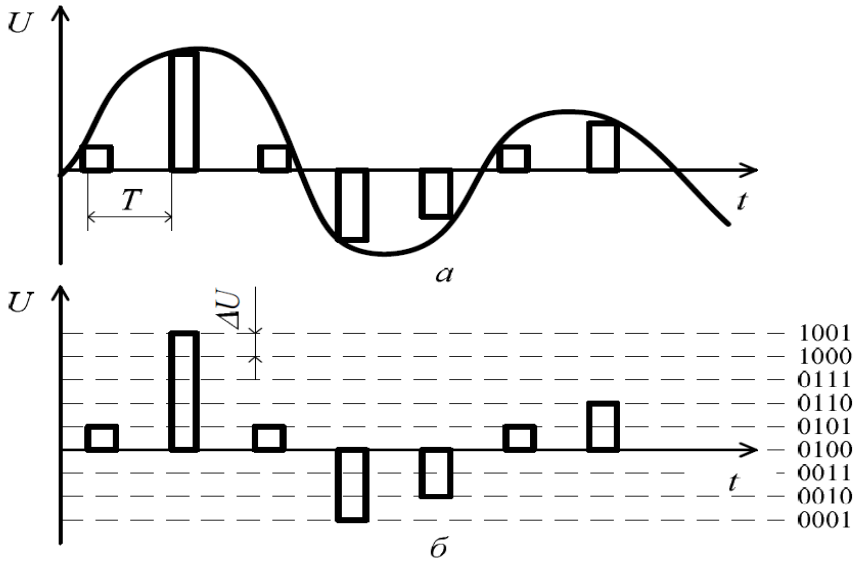


Рис. 1.10. Принципы дискретизации непрерывного сигнала:
а – квантование по времени; *б* – квантование по уровню

Стандарт MREG.

Слово **MPEG** является сокращением от Moving Picture Expert Group - названия экспертной группы ISO, действующая в направлении разработки стандартов кодирования и сжатия видео- и аудио- данных. На сегодняшний день известны следующие:

MPEG-1 предназначен для записи синхронизированных видеоизображения и звукового сопровождения на CD-ROM с учетом максимальной скорости считывания около 1.5 Мбит/с.

MPEG-2 предназначен для обработки видеоизображения соизмеримого по качеству с телевизионным при пропускной способности системы передачи

данных в пределах от 3 до 15 Мбит/с, а в профессиональной аппаратуре используют потоки скоростью до 50 Мбит/с. На технологии, основанные на MPEG-2, переходят многие телеканалы, сигнал сжатый в соответствии с этим стандартом транслируется через телевизионные спутники, используется для архивации больших объемов видеоматериала.

MPEG-3 - предназначался для использования в системах телевидения высокой четкости (HDTV) со скоростью потока данных 20-40 Мбит/с , но позже стал частью стандарта MPEG-2 и отдельно теперь не упоминается. **Примечание (не писать).** Кстати, формат **MP3**, который иногда путают с MPEG-3, предназначен только для сжатия аудиоинформации и полное название MP3 звучит как *MPEG-Audio Layer-3*.

MPEG-4 - задает принципы работы с цифровым представлением медиа-данных для трех областей: интерактивного мультимедиа (включая продукты, распространяемые на оптических дисках и через Сеть), графических приложений (синтетического контента) и цифрового телевидения.

Тема 2.1:Логические основы ЭВМ, элементы и узлы

Лекция 5. Базовые логические операции и схемы. Таблицы истинности. Схемные логические элементы ЭВМ.

Алгебра логики – это математический аппарат, с помощью которого записывают, вычисляют, упрощают и преобразовывают логические высказывания.

Создателем алгебры логики является английский математик Джордж Буль (19 век), в честь которого она названа **булевой алгеброй высказываний.**

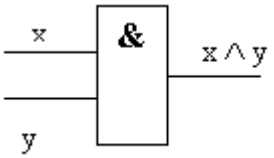
Математический аппарат алгебры логики очень удобен для описания того, как функционируют аппаратные средства компьютера, поскольку основной системой счисления в компьютере является двоичная, в которой используются цифры 1 и 0, а значений логических переменных тоже два: 1 и 0.

Логическими элементами компьютеров являются электронные схемы И, ИЛИ, НЕ, И-НЕ, ИЛИ-НЕ и др. (называемые также вентилями), а также **триггер**.

С помощью этих схем можно реализовать любую логическую функцию, описывающую работу устройств компьютера. Работу логических элементов описывают с помощью **таблиц истинности**.

Базовые логические элементы:

1. **Схема И** - реализует **конъюнкцию (логическое умножение)** двух или более логических значений.



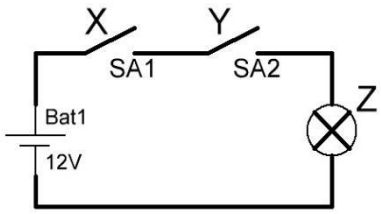


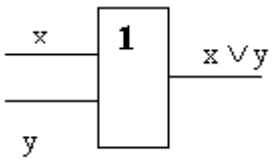
Таблица истинности		
x	y	x ***и*** y
0	0	0
0	1	0
1	0	0
1	1	1

Единица на выходе схемы И будет тогда и только тогда, когда на всех входах будут единицы. Когда хотя бы на одном входе будет нуль, на выходе также будет нуль.

Связь между выходом z этой схемы и входами x и y описывается соотношением z = x ^ y (читается как «x и y»).

Операция конъюнкции на функциональных схемах обозначается знаком &.

2. **Схема ИЛИ** - реализует **дизъюнкцию (логическое сложение)** двух или более логических значений.



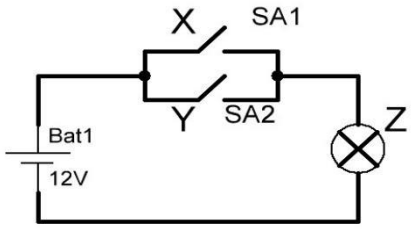


Таблица истинности		
x	y	x ***или*** y
0	0	0
0	1	1
1	0	1
1	1	1

Когда хотя бы на одном входе схемы ИЛИ будет единица, на ее выходе также будет единица. Связь между выходом z этой схемы и входами x и y описывается соотношением z = x или y.

3. **Схема НЕ (инвертор)** - реализует операцию отрицания.

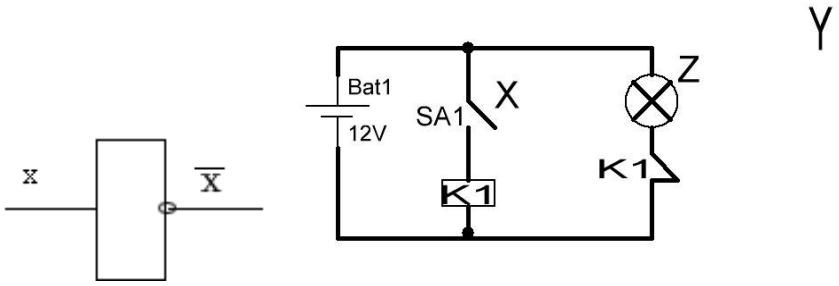
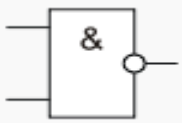


Таблица истинности	
x	***не*** x
0	1
1	0

Связь между входом x этой схемы и выходом z можно записать соотношением $Z = \overline{x}$, где x читается как «не x» или «инверсия. Если на входе схемы 0, то на выходе 1. Когда на входе 1 на выходе 0.

4. **Инверсия функции конъюнкции. Операция И-НЕ (штрих Шеффера)**



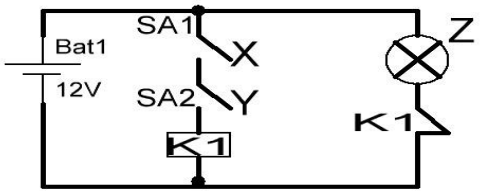


Таблица истинности		
x	y	$\overline{x \wedge y}$
0	1	1
1	0	1
1	1	0
0	0	1

Мнемоническое правило для И-НЕ с любым количеством входов звучит так: На выходе будет:

- «1» тогда и только тогда, когда ***хотя бы на одном*** входе действует «0»,
- «0» тогда и только тогда, когда на ***всех*** входах действуют «1»

5. Инверсия функции дизъюнкции. Операция ИЛИ-НЕ (стрелка Пирса)

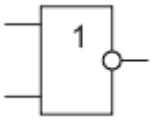
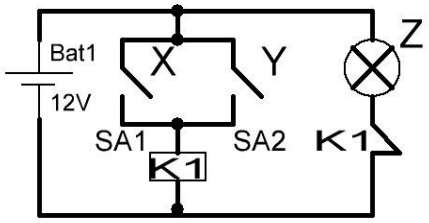


Таблица истинности		
x	y	$\overline{x \vee y}$
0	1	0
1	0	0
1	1	0
0	0	1



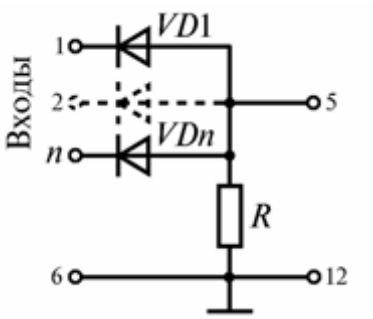
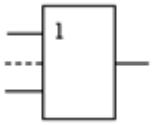
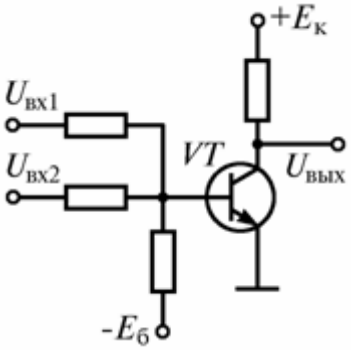
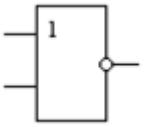
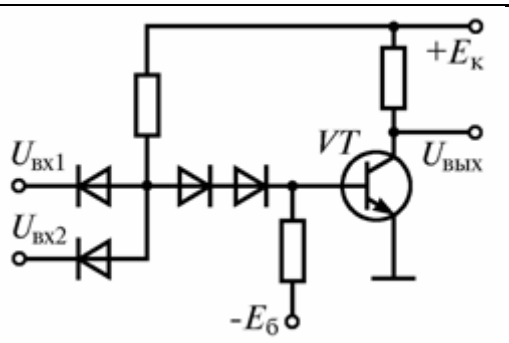
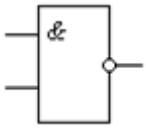
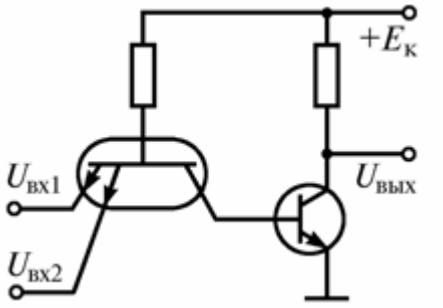
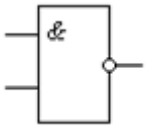
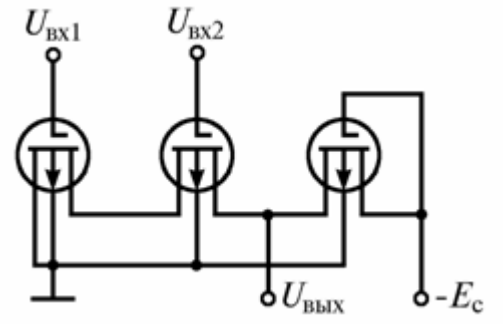
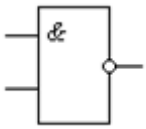
Мнемоническое правило для ИЛИ-НЕ с любым количеством входов звучит так: На выходе будет:

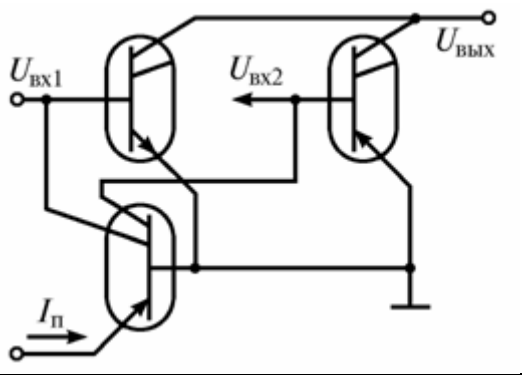
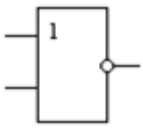
- «1» тогда и только тогда, когда на ***всех*** входах действуют «0»,
- «0» тогда и только тогда, когда ***хотя бы на одном*** входе действует «1»

Схемные логические элементы ЭВМ.

Тип логических элементов определяется совокупностью схемных и технологических признаков, характеризующих интегральные микросхемы логических элементов. Простейшие логические элементы И и ИЛИ могут быть реализованы на основе диодных ключей. Элемент НЕ обычно представляет собой транзисторный ключ с инвертирующими свойствами. Кроме рассмотренных основных логических элементов, используют комбинированные логические элементы, реализующие две (или более) логические операции, например, элементы ИЛИ – НЕ, И – НЕ. Чтобы реализовать элемент И – НЕ, к диодномуключу добавляют инвертор на транзисторе. Такая схема называется диодно-транзисторной логикой (ДТЛ), а логический элемент – ДТЛ – элементом И – НЕ. Использование различных элементов в схемах существенно расширяет ряд логических операций.

Тип логического элемента	Схема	Условное обозначение

Диодная логика		
Резистивная транзисторная логика (РТЛ)		
Диодно- транзисторная логика (ДТЛ)		
Транзисторно- транзисторная логика (ТТЛ)		
Логика на МДП-транзисторах с p- или n-каналом (p-МДПТЛ) (n-МДПТЛ)		

Интегральная инжекционная логика (И2Л)		

Лекция 6. Логические узлы ЭВМ и их классификация.

Классификация элементов и узлов ЭВМ.

ЭВМ может быть представлена как совокупность узлов, а каждый узел - как совокупность элементов.

Элемент - это наименьшая функциональная часть, на которую может быть разбита ЭВМ при логическом проектировании и технической реализации.

По функциональному назначению элементы ЭВМ могут быть разделены на:- запоминающие (для хранения одноразрядного двоичного числа);- вспомогательные (для формирования и генерации импульсов, таймеры, элементы индикаторов, преобразователи уровней и т.п.).

По типу сигналов:- аналоговые;- цифровые.

По способу представления входных и выходных сигналов:- потенциальные;- импульсные;- импульсно-потенциальные.

Узел - совокупность элементов, которая реализует выполнение одной из машинных операций.

Различают два типа узлов ЭВМ:- комбинационные;- накапливающие (с памятью).

В свою очередь комбинационные узлы включают сумматоры, схемы сравнения, шифраторы, дешифраторы, мультипликаторы, программируемые логические матрицы и т.д.

Накапливающие узлы - триггеры, регистры, счётчики и т.п.

В цифровых устройствах переменные и соответствующие им сигналы изменяются не непрерывно, а лишь в дискретные моменты времени. Временной интервал между соседними моментами времени называется тактом.

Информация в элементах ЭВМ может обрабатываться в последовательном или параллельном коде. При последовательном коде каждый временной такт предназначен для обработки одного разряда слова. При этом все разряды слова фиксируются по очереди одним и тем же элементом.

При параллельной обработке информации код слова развертывается не во времени, а в пространстве, т.к. значения всех разрядов обрабатываются одновременно за один такт.

ЭВМ 3-го поколения строились на основе базовых логических элементов(ЛЭ). Например, И-НЕ или ИЛИ-НЕ. Важнейшими характеристиками любого базового логического элемента является быстродействие и потребляемая мощность.

В зависимости от рассеиваемой мощности различают следующие ЛЭ:

- микроватные P до 300 мкВт;
- маломощные P до 3 мВт;
- средней мощности P до 30 мВт;
- мощные P свыше 30 мВт.

По величине среднего времени задержки ЛЭ разбиваются на группы:

- низкое быстродействие tз > 50 нс , P = 0,01-1 мВт;
- среднее быстродействие tз = 10-50 нс , P = 1-10 мВт;
- высокое быстродействие tз = 5-10 нс , P = 10-50 мВт;
- сверхвысокое быстродействие tз < 5 нс , P = 50-1000 мВт.

Каждый ЛЭ кроме того характеризуется величиной напряжения, соответствующим уровням логических "0" и "1" , коэффициентом объединения по входу, коэффициентом разветвления по выходу. U^0 и U^1

ЛЭ объединяются в группы (серии) интегральных микросхем, например, серии К155 , К500 , К176 и др. Для всех ЛЭ повышение быстродействия сопровождается ростом энергопотребления, а повышение плотности размещения элементов на кристалле - снижением быстродействия.

Узлы комбинированного типа.

Триггеры

Основным устройством, которое способно запомнить цифровую информацию, является триггер. Он имеет два устойчивых состояния, одно из которых принимается за 1, а другое — за 0. В вычислительной технике наибольшее распространение получили полупроводниковые триггеры, выпускаемые в виде интегральных микросхем. Они, как правило, являются двухкаскадными усилителями постоянного тока с положительной обратной связью (выход усилителя соединен с его входом).

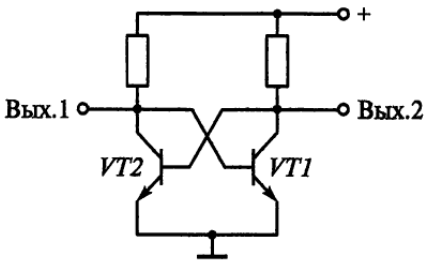


Рис. 7.5. Схема статического триггера

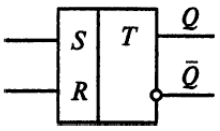


Рис. 7.6. Условное обозначение *RS*-триггера

Схема простейшего статического триггера показана на рис. 7.5. Из двух транзисторов один обязательно открыт, а другой закрыт. Если закрыт транзистор VT1, то положительный потенциал с его коллектора подается на базу транзистора VT2, и наоборот. Соединение коллектора одного транзистора с базой другого и обеспечивает положительную обратную связь. Несмотря на полную симметрию схемы, такое ее состояние, когда оба транзистора открыты, является неустойчивым и практически невозможным. Даже незначительное случайное увеличение коллекторного тока одного транзистора вызывает уменьшение положительного потенциала на его коллекторе и соответственно на базе другого транзистора. Это приводит к уменьшению коллекторного тока другого транзистора, увеличению потенциала на его коллекторе и соответственно на базе первого транзистора. В итоге первый транзистор еще больше открывается, а второй еще больше закрывается. Этот процесс протекает очень быстро (лавинообразно) и заканчивается тогда, когда первый транзистор полностью открывается (режим насыщения), а второй транзистор полностью закрывается, поскольку на его базу будет подан практически нулевой потенциал.

Перевод триггера из одного устойчивого состояния в другое осуществляется подачей положительных или отрицательных импульсов на коллектор одного или другого транзистора. При этом один из входов принимают устанавливающим триггер в состояние 1 и называют S (от англ, set — установить) а другой, устанавливающий (сбрасывающий) триггер в состояние 0, называют входом R (от англ, reset — сбросить). Такой триггер называют ЛУ-триг- гером.

Его условное обозначение показано на рис. 7.6. В сериях микросхем .ЛУ-триггеры обычно построены на двух базовых логических элементах, на которых основана вся серия, т. е. либо на двух ИЛИ— НЕ (рис. 7.7, а, б), либо на двух И—НЕ (рис. 7.7, в, г).

Поясним работу ЛУ-триггера на базе элементов ИЛИ—НЕ (см. рис. 7.7, а) с помощью диаграммы, показанной на рис. 7.7, б. После поступления сигнала 1 на вход S триггер переключается в состояние 1, если он был в состоянии 0, или сохраняет 1 на выходе Q, если он уже находился в этом состоянии. Соответственно при поступлении 1 на вход R триггер переключается в 0 или сохраняет это состояние. Исходное состояние триггера (сразу после включения и при отсутствии сигналов 1 на входах S и R) не определено, оно является случайной величиной. В отличие от схемы, представленной на рис. 7.7, а, в схеме, представленной на рис. 7.7, в, используется отрицательная логика, т. е. 1 имеет менее положительный потенциал, чем 0.

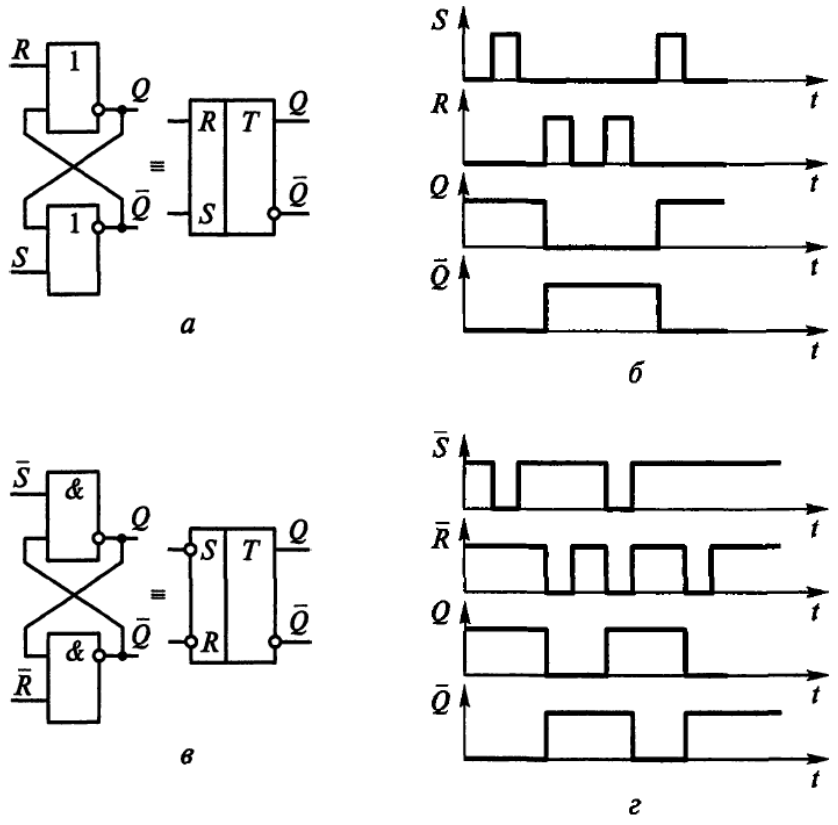


Рис. 7.7. *RS*-триггеры на базе элементов ИЛИ — НЕ и И — НЕ:
а — схема с положительной логикой и ее условное обозначение; *б* — диаграмма сигналов для схемы *а*; *в* — схема с отрицательной логикой и ее условное обозначение; *г* — диаграмма сигналов для схемы *в*

Регистры

Регистр представляет собой упорядоченную последовательность (совокупность) триггеров, число которых соответствует числу разрядов в слове. Регистр используется для хранения «-разрядного слова и выполнения

логических преобразований над ним. В регистре могут выполняться следующие микрооперации: прием (запись)слова; передача слова в другой регистр; поразрядные логические операции; сдвиг слова влево или вправо на заданное число разрядов; преобразование последовательного кода слова в параллельный и обратно; установка регистра в начальное состояние (сброс). Кроме того, регистр может осуществлять преобразование двоичного кода из прямого в обратный (когда единицы заменяются нулями, а нули — единицами), и наоборот.

Поскольку регистр предназначен для хранения двоичного числа (слова), то основу его составляют запоминающие элементы — триггеры. В каждом из них должна храниться цифра разряда числа.

В зависимости от способа ввода и вывода разряда числа различают регистры параллельные, последовательные и параллельно-последовательные.

В параллельном регистре ввод или вывод слова осуществляется в параллельной форме — одновременно для всех разрядов, в последовательном регистре разряды числа вводятся и выводятся последовательно один за другим, в параллельно-последовательном регистре ввод числа осуществляется в параллельной форме, а вывод — в последовательной, или наоборот.

Параллельный регистр. Функциональная схема параллельного регистра на RS-триггерах при однофазном способе приема числа xn...x2 x1 приведена на рис. 7.9. Поскольку сигналы, поступающие только на входы S, могут установить соответствующие триггеры только в состояние 1, но не в состояние 0, то перед приемом числа все триггеры регистра обнуляются. С этой целью по шине 0 подается сигнал на входы R всех триггеров регистра для их предварительной установки в состояние 0. Подготовка к приему новой информации составляет первый такт. Во втором такте по сигналу 1, подаваемому по шине П (Прием), двоичное число xn,,...x2 x1, всеми разрядами одновременно (параллельно) через конъюнкторы (элементы И) записывается в разряды регистра. Выдача числа в прямом коде осуществляется по сигналу 1, подаваемому по шине Впр, а в обратном — по сигналу 1, подаваемому по шине Вобр.

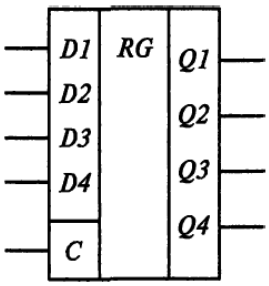


Рис. 7.10. Условное обозначение параллельного четырехразрядного регистра

Условное обозначение параллельного четырехразрядного регистра приведено на рис. 7.10, где Q1—Q4 — выходы разрядов регистра, а D1—D4— входы, с которых в регистр одновременно записываются все разряды заносимого слова; С — вход, импульс на котором разрешает запись с входов D1—D4.

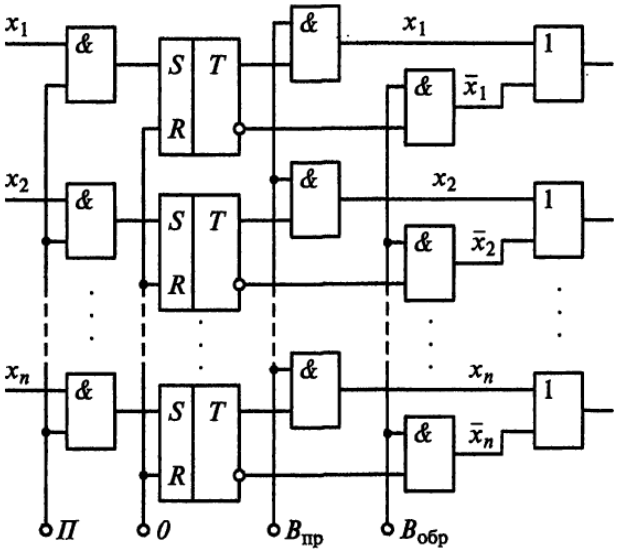


Рис. 7.9. Функциональная схема параллельного регистра на *RS*-триггерах

Последовательный регистр. В таких регистрах двоичное число вводится и выводится последовательно разряд за разрядом. Разряды самого регистра соединены последовательно. Каждый разряд выдает информацию в следующий разряд и одновременно принимает новую информацию из предыдущего. Для этого каждый разряд должен иметь два запоминающих элемента, т.е. сдвоенный или двухступенчатый триггер.В первую ступень передается информация из предыдущего разряда, одновременно вторая ступень передает свою информацию в последующий разряд. Затем информация, принятая первой ступенью, передается во вторую, а первая освобождается для приема новой информации. Двухступенчатый триггер (например, /Л--триггер, Z)-триггер) представляет собой совокупность двух запоминающих элементов, поэтому он один может составлять разряд последовательного регистра. Если в цепи таких триггеров выходы одного соединить с входами другого, то по фронту тактового импульса, подаваемого на вход С, во входную (первую) ступень каждого триггера будет заноситься информация из выходной (второй) ступени предыдущего триггера, а по спаду импульса она будет переписываться в выходную ступень. По фронту следующего тактового импульса во входной ступени триггера информация может быть заменена новой (из предыдущего триггера) без опасения, что предыдущая окажется потерянной.

Функциональная схема последовательного регистра приведена на рис. 7.11. Крайний левый триггер предназначен для хранения старшего разряда числа, а крайний правый — для хранения младшего разряда.

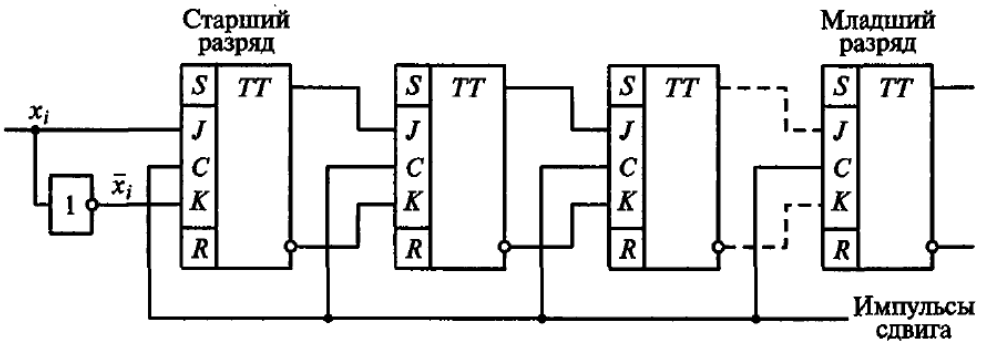


Рис. 7.11. Функциональная схема последовательного регистра

Счетчики

Счетный триггер, или Т-триггер, условное обозначение которого показано на рис. 7.14, а, имеет один вход и два выхода. Сигналы на выходах меняются на противоположные при каждом положительном перепаде напряжения на счетном входе Т. Счетный триггер может быть создан на базе динамического D-триггера, если его инверсный выход соединить с информационным входом, как показано на рис. 7.14, б.

Рассмотрим работу счетного триггера с помощью диаграммы, приведенной на рис. 7.14, в.

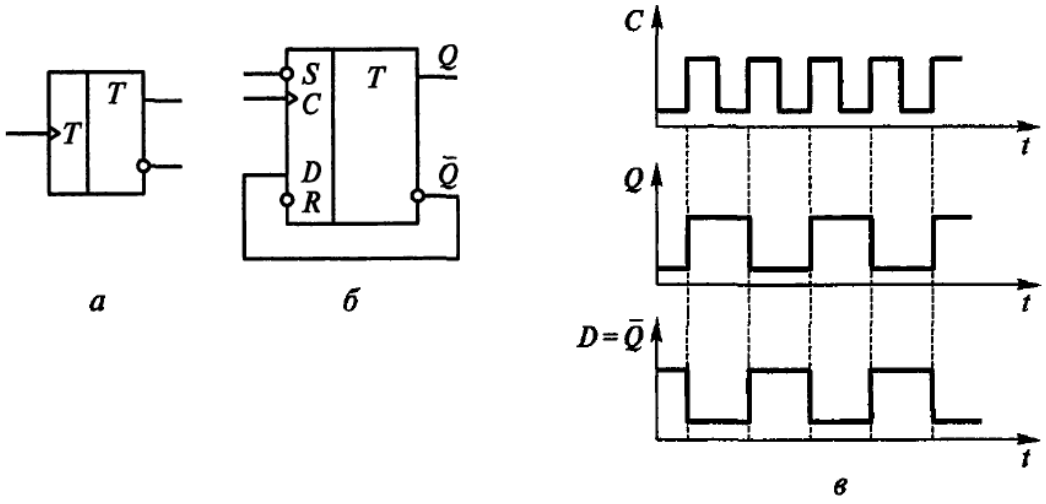


Рис. 7.14. Счетный триггер:
а — условное обозначение; *б* — схема на базе динамического *D*-триггера; *в* — диаграмма сигналов

Пусть в начальный момент времени на прямом выходе был сигнал 0, тогда на инверсном выходе и, следовательно, на входе D — сигнал 1. По фронту первого синхроимпульса сигнала 1 с входа перепишется на прямой выход, а на инверсном выходе появится 0. По фронту второго синхроимпульса этот сигнал 0 перепишется на прямой выход и будет там сохраняться до прихода третьего синхроимпульса и т.д. Обратите внимание, что частота сигналов на выходе вдвое меньше входной частоты синхроимпульсов, поэтому счетный триггер называется делителем частоты.

Для хранения информации о многоразрядном кодовом слове используются несколько триггеров, по одному на каждый разряд. В этом случае такая группа триггеров называется регистром.

Для подсчета импульсов применяют регистры, состоящие из Г-триггеров. На рис. 7.15 показан простой трехразрядный двоичный счетчик импульсов, состоящий из трех Т-триггеров, которые имеют входы R для установки нуля. Временные диаграммы сигна-лов в таком счетчике приведены на рис. 7.16, а табл. 7.6 иллюстрирует состояние триггеров. В исходном положении все триггеры находятся в состоянии 0. После первого входного импульса триггер 77 переходит в состояние 1, после второго входного импульса в состояние 1 переходит триггер 72, а 77 возвращается в состояние 0 и т.д. Из табл. 7.6 видно, что по состоянию триггеров можно определить, сколько импульсов поступило на вход к данному моменту времени. После восьмого входного импульса все три триггера переходят в состояние 0 и счет повторяется. В общем случае емкость счетчика (т. е. коэффициент пересчета) равна 2^n, где n— число триггеров в счетчике. С помощью обратных связей можно получить коэффициент пересчета меньше указанного значения.

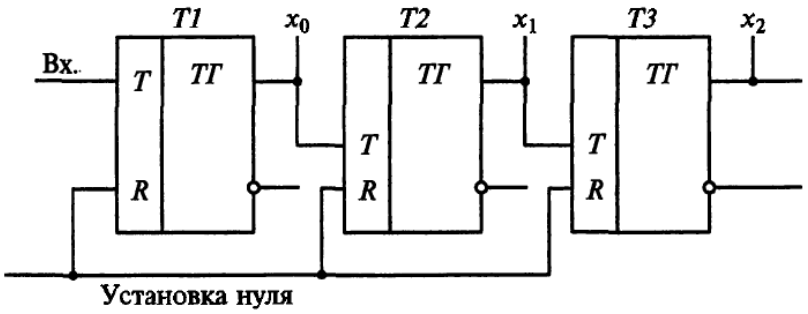


Рис. 7.15. Трехразрядный двоичный счетчик импульсов

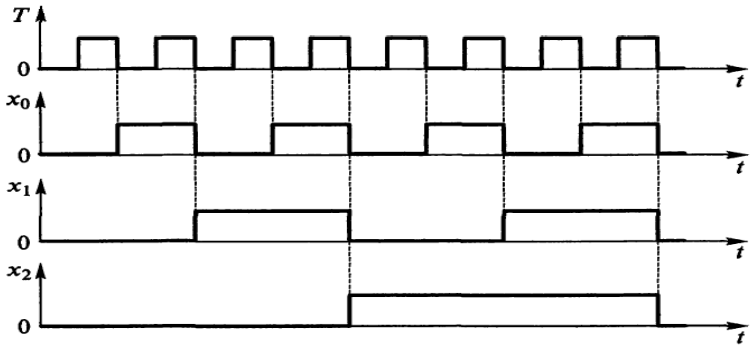


Рис. 7.16. Диаграммы сигналов в счетчике импульсов

Сумматоры

Сумматор представляет собой комбинационное цифровое устройство (КЦУ), предназначенное в основном для суммирования двоичных чисел. Кроме того, с помощью сумматора могут выполняться вычитание, умножение, деление, преобразование чисел в дополнительный код и некоторые другие

операции. Обычно сумматор состоит только из логических элементов, а результат операции направляется затем для записи в регистр.

Полусумматорами (рис. 7.17, 7.18) называют КЦУ с двумя входами (а, Ь) и двумя выходами, на одном из которых вырабатывается сигнал суммы (выход S), а на другом — сигнал переноса (выход Р). Табл. 7.7 является таблицей истинности полусумматора.

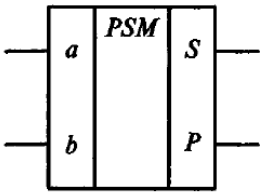


Рис. 7.17. Условное обозначение полусумматора

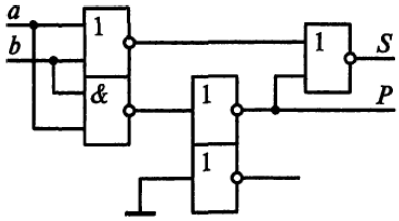


Рис. 7.18. Схема полусумматора на логических элементах

Таблица истинности полусумматора

a	*b*	*S*	*P*
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Одноразрядным сумматором (рис. 7.19,7.20) называют КЦУ с тремя входами и двумя выходами. Кроме двух входов для чисел он имеет третий вход, на который подается сигнал переноса из предыдущего разряда.

Одноразрядный сумматор является основным элементом многоразрядных сумматоров.

Он выполняет арифметическое сложение одноразрядных двоичных чисел ai и bi,- и перенос Pi-1 из предыдущего разряда с образованием на выходе суммы Si и переноса Pi, в старший разряд (табл. 7.8).

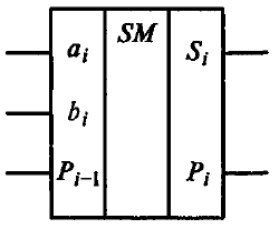


Рис. 7.19. Условное обозначение одноразрядного сумматора

Аналогичным способом могут быть построены логические схемы вычитателей. Как сумматоры, так и вычитатели предназначены для выполнения основных арифметических операций — сложения и вычитания. Имея на входе дополнительные средства для изменения знака второго аргумента, сумматор может прибавлять к первому слагаемому второе с измененным знаком, т.е. вычитать, а вычитатель — вычитать из уменьшаемого вычитаемое с измененным знаком, т.е. прибавлять. Таким образом, в арифметико-логических устройствах (АЛУ) в большинстве случаев используется только один из двух

рассматриваемых узлов, традиционно — именно сумматор, хотя по всем показателям вычитатель подобен сумматору.

Шифраторы и дешифраторы.

Основными видами преобразования информации являются шифрование и дешифрование (сжатие данных и обратное преобразование). Для реализации таких преобразований служат шифраторы и дешифраторы. Они представляют собой комбинационные устройства, поскольку состоят только из логических элементов, а элементов памяти не имеют. Шифраторы преобразуют код «1 из N» в двоичный код, а дешифраторы — двоичный код в код «1 из A'». Число разрядов двоичного кода обычно меньше N, поэтому операцию шифрования можно считать сжатием данных. Иными словами, шифратор превращает сигнал 1 на одном из нескольких входных зажимов в выходную кодовую комбинацию. Поэтому для шифратора используется также название «кодер», а на его условном обозначении пишут буквы CD. Дешифратор превращает комбинацию из нулей и единиц на входе в сигнал 1 только на одном единственном из нескольких выходных зажимов. Поэтому для дешифратора используется также название «декодер», а на его условном обозначении пишут DC.

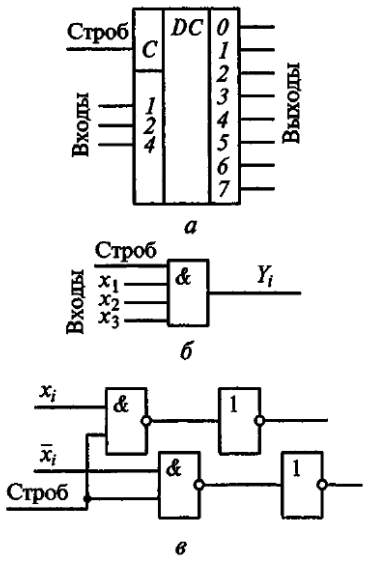


Рис. 7.22. Дешифратор «1 из 8»: *а* — условное обозначение; *б* — стробирование по выходу; *в* — стробирование по входу

Как и для других комбинационных устройств, для шифраторов и дешифраторов связь между входными и выходными сигналами можно задать с помощью логических функций или таблиц истинности. Для синхронизации выходных сигналов этих преобразователей тактовыми импульсами используют дополнительные входы. В этом случае преобразователи называют стробируемыми. ***Стробирование*** — это выделение сигнала в определенный момент времени. Наличие входов стробирования расширяет функциональные возможности шифраторов и дешифраторов.

Дешифраторы широко применяются в устройствах управления, вывода информации на цифровые индикаторы, в коммутаторах для распределения сигналов по различным цепям.

Существуют два способа стробирования дешифраторов: введением дополнительного входа в каждый элемент (стробирование по выходу — рис. 7.22, б) и блокированием всех элементов через одну из входных цепей (стробирование по входу — рис.7.22, в).

Таблица истинности дешифратора «1 из 8»

Входная кодовая комбинация				Сигнал на выходе							
C	x_1	x_2	x_3	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
1	0	0	0	1	0	0	0	0	0	0	0
1	0	0	1	0	1	0	0	0	0	0	0
1	0	1	0	0	0	1	0	0	0	0	0
1	0	1	1	0	0	0	1	0	0	0	0
1	1	0	0	0	0	0	0	1	0	0	0
1	1	0	1	0	0	0	0	0	1	0	0
1	1	1	0	0	0	0	0	0	0	1	0
1	1	1	1	0	0	0	0	0	0	0	1
0	Любые комбинации нулей и единиц			0	0	0	0	0	0	0	0

Шифратор — это комбинационное устройство, преобразующее код «1 из N» в двоичный код. Полный шифратор имеет 2^n входов и n выходов. Одно из основных применений шифратора — ввод данных с клавиатуры, при котором нажатие на клавишу с десятичной цифрой должно приводить к передаче в устройство этой цифры в двоичном коде. При нажатии любой из десяти цифровых клавиш единица появляется только на одном из десяти входов шифратора X0, X1 ..., X9. На выходе шифратора должен появиться двоичный код (y0 y1 y2 y3) введенного десятичного числа. Из таблицы истинности (табл. 7.11) видно, что в этом случае нужен преобразователь с десятью входами и четырьмя выходами, т.е. так называемый шифратор «10—4».

Табли

Десятичное число		
	X_0	X_1
0	1	0
1	0	1
2	0	0
3	0	0
4	0	0
5	0	0
6	0	0
7	0	0
8	0	0
9	0	0

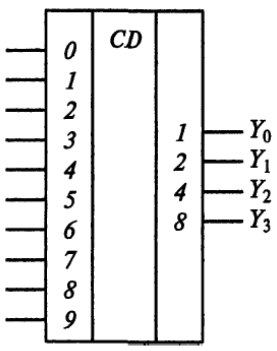


Рис. 7.23. Условное обозначение шифратора

Лекция 7. Понятие архитектуры и структуры компьютера. Принципы (архитектура) фон Неймана. Основные компоненты ЭВМ. Основные типы архитектур ЭВМ.

Компьютер — это электронный прибор, предназначенный для автоматизации создания, обработки, хранения и транспортировки данных. Архитектура ЭВМ — общее описание структуры и функций ЭВМ, ее ресурсов.

Архитектура компьютера обычно определяется совокупностью её свойств, существенных для пользователя. Основное внимание при этом уделяется структуре и функциональным возможностям компьютера.

Функциональные возможности разделяют на основные (обработка и хранение информации, обмен информацией с внешними объектами) и дополнительные (повышают эффективность основных).

Несмотря на то, что первая электронная вычислительная техника появилась ещё в конце 1940-х гг., архитектура и принципы функционирования цифровых ЭВМ, заложенные фон Нейманом, остались в целом теми же и сегодня.

Чтобы компьютер был и эффективным, и универсальным инструментом, он должен включать следующие структуры: центральное арифметико-логическое устройство (АЛУ), центральное устройство управления (УУ), «дирижирующее» операциями, запоминающее устройство, или память, а также устройство ввода-вывода информации (рис. 3). Фон Нейман отмечал, что эта с-система должна работать с двоичными числами, быть электронным, а не

механическим устройством и выполнять операции последовательно, одну за другой.

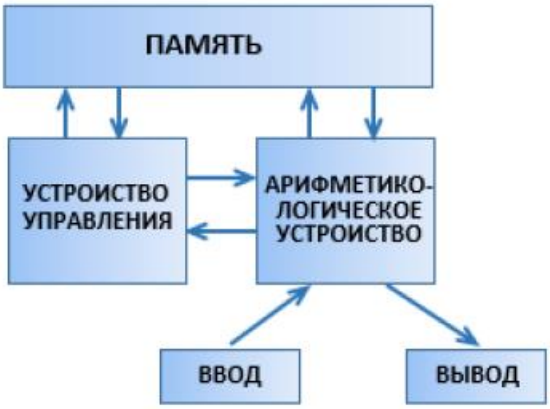


Рис. 3. Схематичное изображение[14] машины фон Неймана

К принципам, получившим название «принципы фон Неймана», относят следующие:

1. ***Принцип однородности памяти.*** Команды и данные хранятся в одной и той же па-мяти и внешне в памяти неразличимы. Распознать их можно только по способу ис-пользования; то есть одно и то же значение в ячейке памяти может использоваться и как данные, и как команда, и как адрес в зависимости лишь от способа обращения к нему. Это позволяет производить над командами те же операции, что и над числами, и, соответственно, открывает ряд возможностей.

2.***Принцип адресности.*** Структурно основная память состоит из пронумерованных ячеек, причем процессору в произвольный момент доступна любая ячейка. Двоичные коды команд и данных разделяются на единицы информации, называемые словами, и хранятся в ячейках памяти, а для доступа к ним используются номера соответствующих ячеек – адреса.

3.***Принцип программного управления.*** Все вычисления, предусмотренные алгоритмом решения задачи, должны быть представлены в виде программы, состоящей из последовательности управляющих слов – команд. Каждая команда предписывает некоторую операцию из набора операций, реализуемых вычислительной машиной. Команды программы хранятся в последовательных ячейках памяти вычислительной машины и выполняются в естественной последовательности, то есть в порядке их положения в программе. При необходимости, с помощью специальных команд, эта последователь-ность может быть изменена.

4.***Принцип двоичного кодирования.*** Согласно этому принципу, вся информация, как данные, так и команды, кодируются двоичными цифрами 0 и 1. Каждый тип информации представляется двоичной последовательностью и

имеет свой формат. Последовательность битов в формате, имеющая определенный смысл, называется полем. В числовой информации обычно выделяют поле знака и поле значащих разрядов. В формате команды можно выделить два поля: поле кода операции и поле адресов.

В упрощённом виде структурная схема любой цифровой ЭВМ обязательно содержит три основных компонента –блок центрального процессора, блок памяти и блок интерфейсов, как показано на рис. 1.9 .

Самым сложным является устройство центрального процессора. Он представляет собой программно-управляемое устройство, которое обрабатывает информацию и осуществляет её распределение между основными компонентами ЭВМ. Обработка информации процессором состоит в выполнении арифметических и логических операций над данными, представленными в виде кодовых комбинаций. Последовательность операций задается командами, также представляющими собой комбинации кодов. Упорядоченный набор команд для выполнения конкретной задачи называется ***программой.*** Процессор не только обрабатывает данные, но и управляет процессом обработки информации,контролирует шаги исполняемой программы, вызывает из запоминающего устройства очередную команду и данные, выполняет требуемую операцию, а полученный результат сохраняет в памяти и(или)отправляет на устройства вывода.

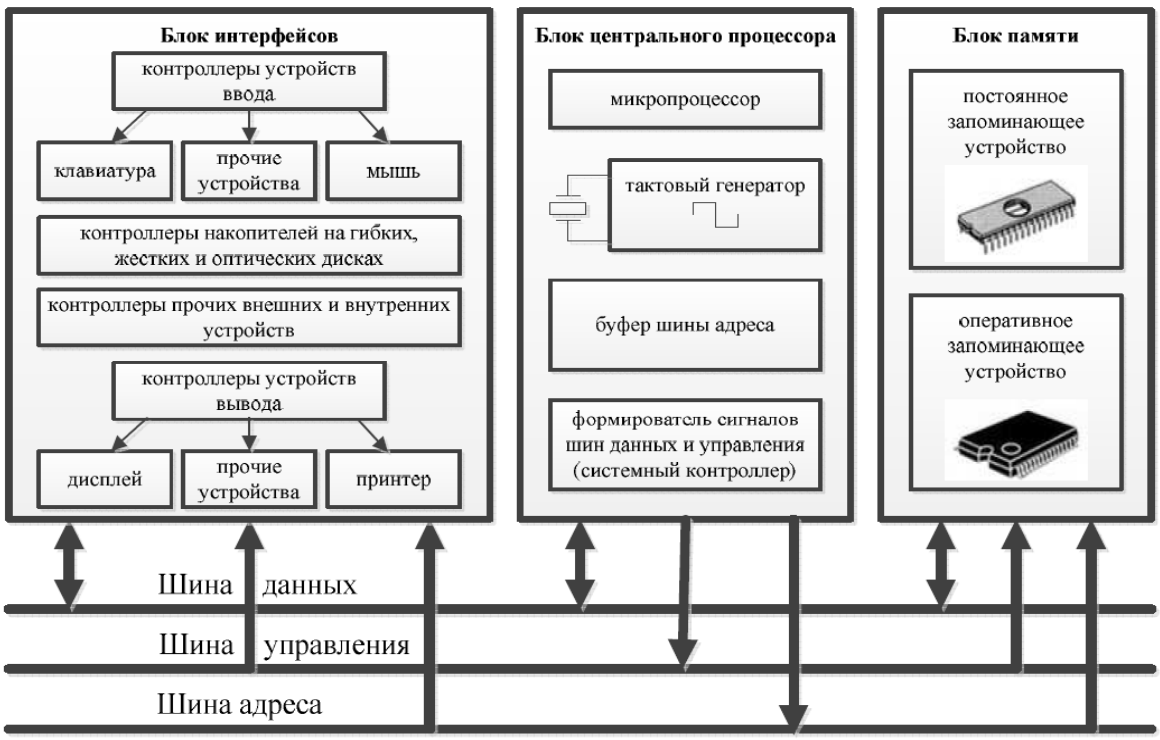


Рис. 1.9. Упрощённая структурная схема ЭВМ

Память ЭВМ состоит из постоянного запоминающего устройства (ПЗУ), которое допускает только чтение хранимой в нём информации,и оперативного запоминающего устройства, позволяющего как считывать из него данные, так и записывать новые. В сущности, память можно представить в виде совокупности однотипных ячеек, в каждой из которых хранится закодированная информация. Например, для внесения в память одного информационного бита достаточно одноразрядной ячейки, в которую можно записать логическую 1 или 0.

Разрядность ячейки памяти ЭВМ определяется разрядностью шины данных процессора. Другими словами, разрядность шины данных определяет количество разрядов, над которыми одновременно могут выполняться операции, т.е. определяет скорость передачи данных между микропроцессором, памятью и другими устройствами.

Большинство современных микропроцессоров имеют 32- или 64-разрядную шину данных. При 64-разрядной шине данных скорость передачи в два раза выше, чем у 32-разрядной шины.

Для того, чтобы выделить из массива информации, хранящейся в памяти, требуемые данные, каждой ячейке памяти присваивается специальный номер, называемый адресом. Максимальное количество ячеек памяти, к которым можно осуществлять непосредственный доступ по индивидуальному адресу, формирует объём адресного пространства, и определяется разрядностью шины адреса процессора.

Например, 32-разрядная шина адреса позволяет обращаться к 232 ячейкам памяти.

Блок интерфейсов предназначен для связи ЭВМ с аппаратуройс для ввода/вывода информации и обычно включает в себя функциональные узлы для сопряжения с клавиатурой, дисплеем, манипулятором типа «мышь», принтером и другими внешними и внутренним устройствами.

Взаимосвязь между всеми компонентами ЭВМ осуществляется центральным процессором по шине управления (системной шине).

Разрядность шины управления также непосредственно влияет на производительность ЭВМ, поэтому, чем выше разрядность, тем лучше.

Шины управления современных систем, как правило, выполняют 32- и 64-разрядными.

Главной частью ПК является системный блок, внутри которого сосредоточены основные узлы компьютера. Наиболее важным компонентом, установленным в системном блоке, является ***системная, или материнская плата.*** В англоязычной терминологии её обычно называют mainboard [6]. На этой плате размещены основные узлы ЭВМ –центральный процессор с теплоотводом, память (ПЗУ и ОЗУ), шины данных, адреса и управления,

контроллеры ввода/вывода, шин, контроллеры накопителей на гибких, жёстких и оптических дисках, контроллеры прочих внешних устройств, а также дополнительные разъёмы для плат расширения и гнёзда подключения различных внешних устройств.

Взаимодействие микропроцессора с узлами компьютера осуществляется с помощью схем, выполненных по интегральной технологии, которые получили название ***северного и южного мостов***. Эти названия обусловлены расположением этих микросхем на системной плате. Ближе к микропроцессору находится северный мост, а южный мост расположен ниже, рядом с разъёмами для подключения внешних запоминающих устройств.

Южный мост содержит контроллеры для работы с дисковыми устройствами, клавиатурой, мышью и другой аппаратурой. С северным мостом микропроцессор обменивается информацией напрямую, а с южным – через северный мост.

На системной плате имеются также разъёмы, часто называемые ***слотами*** (от англ. slot – щель, паз) для установки модулей оперативной памяти.

Оперативная память используется микропроцессором в процессе обработки информации. Обмен с ОЗУ микропроцессор осуществляет через северный мост.

Северный мост также задействован для передачи результатов обработки информации на экран дисплея, который подключён к мосту через специальную электронную схему. Эта схема размещается на отдельной плате, называемой видеокартой (видеоадаптером), или в виде СБИС на самой материнской плате. В этом случае говорят об интегрированном видеоустройстве. Видеокарта устанавливается на материнскую плату в специальный слот.

Кроме того, на системной плате обычно предусматривают несколько свободных разъёмов (от одного до пяти) для установки дополнительных плат, расширяющих возможности ПК. Типичными примерами плат расширения служат сетевые карты для обеспечения возможности работы ПК в рамках компьютерной сети и звуковые карты для качественного вывода аудиоинформации. Качественные и дорогие материнские платы, предназначенные для работы в составе ПК повышенной производительности, имеют наибольшее количество свободных слотов расширения.

Остальные компоненты, смонтированные внутри системного блока – накопители на жёстких дисках, CD- и DVD-приводы, дисководыгибких магнитных дисков и флэш-карт памяти и другие устройства,соединены с материнской платой при помощи проводов и кабелей.

Питание на системную плату подводится от блока питания, который обеспечивает питающими напряжениями 5 и 12 В все функциональные узлы,

входящие в её состав. Кроме того, от блока питания электроэнергию получают также и все остальные компоненты системного блока.

Тема 2.3:Внутренняя организация процессора.

Лекция 8. Структура процессора. Устройство управления: назначение и упрощенная функциональная схема. Регистры процессора: сущность, назначение, типы. Регистры общего назначения, регистр команд, счетчик команд, регистр флагов.

Центральный процессор персонального компьютера является самым важным и, как правило, самым дорогостоящим его компонентом, обеспечивающим все вычисления и обработку данных. Выполненный по интегральной полупроводниковой технологии центральный процессор получил название ***микропроцессора.***

Характеристики микропроцессоров как функциональных узлов включают значение тактовой частоты, формат обрабатываемых данных, количество и тип команд, методы адресации данных, число внутренних регистров общего и специального назначения и их разрядность, возможности организации и адресации стека, параметры виртуальной памяти и информационную ёмкость адресуемой памяти.

По виду обрабатываемых входных сигналов различают ***цифровые и аналоговые микропроцессоры.*** Хотя сами микропроцессоры являются цифровыми устройствами, однако они могут иметь встроенные аналого-цифровые и цифроаналоговые преобразователи. Поэтому входные аналоговые сигналы передаются в микропроцессор через АЦП, обрабатываются и после обратного преобразования в ЦАП поступают на выход.

По количеству одновременно выполняемых программ различают ***одно- и многопрограммные микропроцессоры.*** В однопрограммных микропроцессорах в один момент времени может выполняться только одна программа. Переход к выполнению следующей программы происходит только после завершения текущей. В многопрограммных микропроцессорах одновременно может выполняться несколько программ. Организация мультипрограммной работы микропроцессорных систем позволяет осуществить контроль за состоянием и управлением большим числом источников или приёмников информации.

По характеру временной организации работы выделяют ***синхронные и асинхронные микропроцессоры.*** К синхронным относят такие микропроцессоры, в которых начало и конец выполнения операций задаются устройством управления (т.е. время выполнения операций не зависит от вида выполняемых команд и величин операндов). Асинхронные микропроцессоры позволяют определить начало выполнения каждой следующей операции по сигналу окончания выполнения предыдущей.

По числу ИМС в микропроцессорном комплекте ***различают однокристальные, многокристальные и многокристальные секционные микропроцессоры.*** Однокристальные микропроцессоры получаются при реализации всех аппаратных средств процессора в виде одной ИМС большой или сверхбольшой степени интеграции. Более широко распространены многокристальные микропроцессоры, а также многокристальные секционные микропроцессоры. В таком процессоре его логическая структура разбивается на функционально законченные части и реализуется либо в виде нескольких БИС, в совокупности образующих микропроцессорный комплект, либо в виде одной микросборки, объединяющей несколько полупроводниковых кристаллов (процессорных секций).

С момента выхода первого ПК в 1981 г. процессорные технологии развивались в четырёх основных направлениях:

1) увеличение количества активных компонентов и плотности их размещения на кристалле;

2) подъём тактовой частоты;

3) повышение разрядности внутренних регистров;

4) увеличение количества ядер (т.е. процессорных секций) в одной микросхеме.

Внутреннее устройство микропроцессора

Несмотря на значительное различие по классификационным характеристикам и основным параметрам процессоров различных производителей, все они имеют сходную внутреннюю структуру, которую можно упрощённо разделить на операционный и интерфейсный блоки.

Операционный блок, выполняющий функции управления и обработки данных, содержит арифметико-логическое устройство, микропроцессорную память (за исключением сегментных регистров), модуль микропрограммного управления, объединённых в узел обработки данных, и устройство управления.

Узел обработки данных предназначен для выполнения команд. ***Устройство управления*** обеспечивает синхронизацию работы устройств микропроцессора, выработку управляющих сигналов и сигналов состояния для обмена с другими компонентами,анализ и соответствующую реакцию на сигналы других устройств ЭВМ. ***Интерфейсный блок,*** или устройство связи с магистралью содержит набор сегментных регистров микропроцессорной памяти, набор регистров команд, представляющих собой регистры памяти для хранения кодов команд, выполняемых в ближайшие такты работы и сумматор адреса. Интерфейсный блок обеспечивает формирование физического адреса памяти и адреса внешнего устройства, выбор команд из памяти, обмен данными с запоминающими устройствами, внешними устройствами, а также другими процессорами по магистрали.

Обобщённая структура типичного микропроцессора представлена на рис. 3.1. Рассмотрим назначение и особенности функционирования входящих в неё блоков.

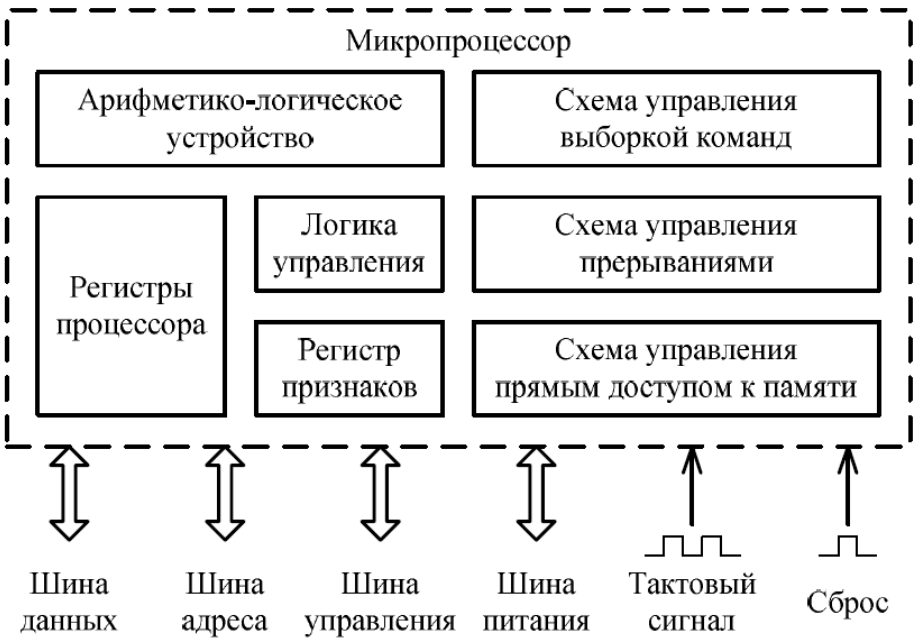


Рис. 3.1. Обобщённая внутренняя структура микропроцессора

Схема управления выборкой команд выполняет чтение команд из памяти и их дешифрацию. Ранние 8-разрядные микропроцессоры не могли одновременно выполнять текущую команду и осуществлять выборку следующей команды, но уже в 16-разрядных микропроцессорах появляется так называемый конвейер (очередь) команд, позволяющий выбирать несколько

следующих команд, пока выполняется предыдущая. Два процесса идут параллельно, что ускоряет работу процессора. Конвейер представляет собой небольшую внутреннюю память процессора, в которую при малейшей возможности (при освобождении внешней шины) записывается несколько команд, следующих за исполняемой. Читаются эти команды процессором в том же порядке, что и записываются в конвейер.

Арифметико-логическое устройство (см. рис. 3.1) предназначено для обработки информации в соответствии с полученной процессором командой. Примерами обработки могут служить логические операции дизъюнкции, конюнкции и др., т.е. побитные операции над операндами, а также различные арифметические операции. Над какими числами проводится операция, куда помещается её результат – все определяется выполняемой командой. Если команда сводится всего лишь к пересылке данных без их обработки, то АЛУ не участвует в её выполнении. Быстродействие АЛУ во многом определяет производительность процессора.

Регистры процессора (см. рис. 3.1) служат для временного хранения различных кодов: данных, адресов и служебных кодов.

Схема управления прерываниями (см. рис. 3.1) обрабатываетпоступающий на микропроцессор запрос прерывания, определяет адрес начала программы обработки прерывания (адрес вектора прерывания), обеспечивает переход к этой программе после выполнения текущей команды и сохранения в памяти текущего состояния регистров микропроцессора.

Схема управления прямым доступом к памяти (см. рис. 3.1)служит для временного отключения микропроцессора от внешних шин и приостановки работы процессора на время предоставления прямого доступа запросившему его устройству.

Логика управления (см. рис. 3.1) организует взаимодействие всех узлов микропроцессора, перенаправляет данные, синхронизирует работу микропроцессора с внешними сигналами, а также реализует процедуры ввода и вывода информации.

Устройство управления.

Процесс функционирования ЭВМ состоит из последовательности элементарных действий в ее узлах. Такие элементарные преобразования информации, выполняемые в течение одного такта сигналов синхронизации, называются ***микрооперациями (***МО).

Совокупность управляющих сигналов, вызывающих одновременно выполняемые микрооперации, образует ***микрокоманду*** (МК).

В свою очередь, последовательность микрокоманд, определяющую содержание и порядок реализации машинного цикла, принято называть микропрограммой.

Управляющие сигналы вырабатываются устройством управления, а точнее одним из его узлов — ***микропрограммным автоматом*** (МПА).

Название отражает то, что МПА определяет микропрограмму как последовательность выполнения микрокоманд.

В обобщенной структуре УУ (рис. 2.19) можно выделить две части управляющую и адресную.

Управляющая часть УУ предназначена для координирования работы операционного блока АЛУ ЭВМ, адресной части устройства управления, основной памяти и других узлов ЭВМ.

Адресная часть УУ обеспечивает формирование адресов команд и исполнительных адресов операндов в основной памяти.

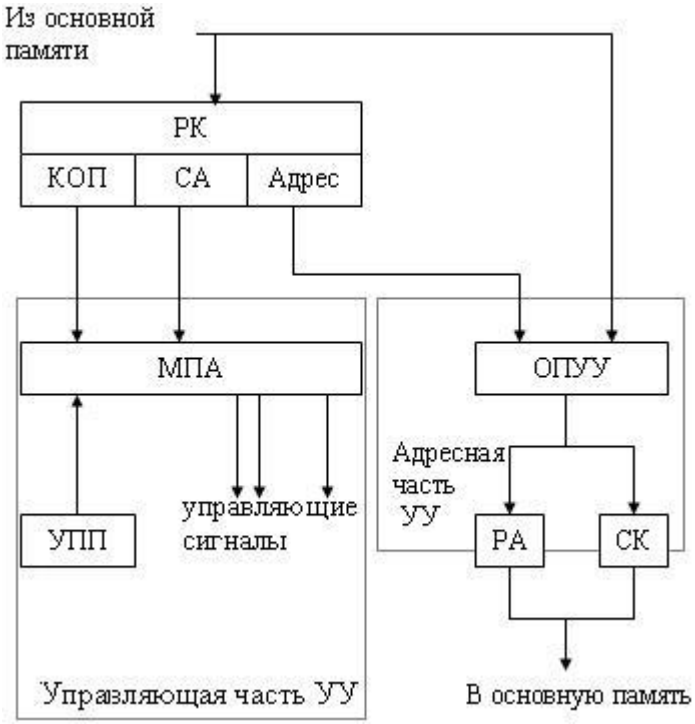


Рис. 2.19. Обобщенная структура устройства управления

Регистр команды РК - предназначен для приема очередной команды из запоминающего устройства.

Микропрограммный автомат МПА на основании результатов расшифровки операционной части команды (КОП, СА) вырабатывает определенную последовательность микрокоманд, вызывающих выполнение всех целевых функций УУ.

Узел прерываний и приоритетов УПП позволяет реагировать на

различные ситуации, связанные как с выполнением рабочих программ, так и с состоянием ЭВМ в целом.

Операционный узел устройства управления ОПУУ, называемый иначе узлом индексной арифметики или узлом адресной арифметики, обрабатывает адресные части команд, формируя исполнительные адреса операндов, а также подготавливает адрес следующей команды при выполнении команд перехода.

Регистр адреса РА используется для хранения исполнительных адресов операндов.

Счетчик команд необходим для выработки и хранения адресов команд.

Содержимое РА и СК посылается в регистр адреса основной памяти (ОП) для выборки операндов и команд соответственно.

Регистры процессора.

Регистры процессора (см. рис. 3.1) представляют собой по сути ячейки очень быстрой памяти и служат для временного хранения различных кодов: данных, адресов и служебных кодов. Операции с этими кодами выполняются предельно быстро, поэтому, чем больше внутренних регистров, тем лучше. Кроме того, на быстродействие процессора сильно влияет разрядность регистров. Именно разрядность регистров и АЛУ называется внутренней разрядностью процессора, которая может не совпадать с внешней разрядностью.

По отношению к назначению внутренних регистров существует два основных подхода.

Согласно первому подходу, каждому регистру отводится строго определённая функция. С одной стороны, это упрощает организацию микропроцессора и уменьшает время выполнения команды, но с другой – снижает гибкость, а иногда и замедляет работу программы. Например, некоторые арифметические операции и обмен с устройствами ввода/вывода проводятся только через один регистр-аккумулятор, в результате чего при выполнении некоторых процедур может потребоваться несколько дополнительных пересылок между регистрами.

Второй подход состоит в том, чтобы все (или почти все) регистры сделать равноправными, что с одной стороны, обеспечивает универсальность, но с другой стороны, приводит к усложнению структуры микропроцессора. Первого подхода придерживается компания Intel, второй подход реализуется в микропроцессорах фирмы DEC.

Набор регистров процессора приведен на рис. 3.2. Рассмотрим его состав более подробно

В первую группу входят регистры общего назначения. В современных микропроцессорах фирм Intel и AMD, начиная с модели 80386, имеются восемь

32-разрядных регистров общего назначения EAX, EBX, ECX,EDX, ESI, EDI, EBP и ESP. В первых микропроцессорах 8086/80286 доступны были лишь 16-разрядные регистры.

При необходимости возможна работа с половинами регистров, поскольку они разделены на старшую и младшую половину, называемые AH и AL, BH и BL и т.п. Физически эти регистры в микропроцессоре размещаются внутри АЛУ и имеют следующие назначения:

- *регистр-аккумулятор* (Accumulatorregister) EAX/AX/AH/AL применяется для хранения промежуточных данных;

- ***базовый регистр*** (Base register) EBX/BX/BH/BL применяется для хранения базового адреса некоторого объекта в памяти;

- ***регистр-счётчик*** (Count register) ECX/CX/CH/CL применяется в командах, производящих некоторые повторяющиеся действия;

- ***регистр данных*** (Data register) EDX/DX/DH/DL, так же как и регистр-аккумулятор, хранит промежуточные данные, причём в некоторых ситуациях его использование обязательно, а в отдельных случаях он используется неявно.

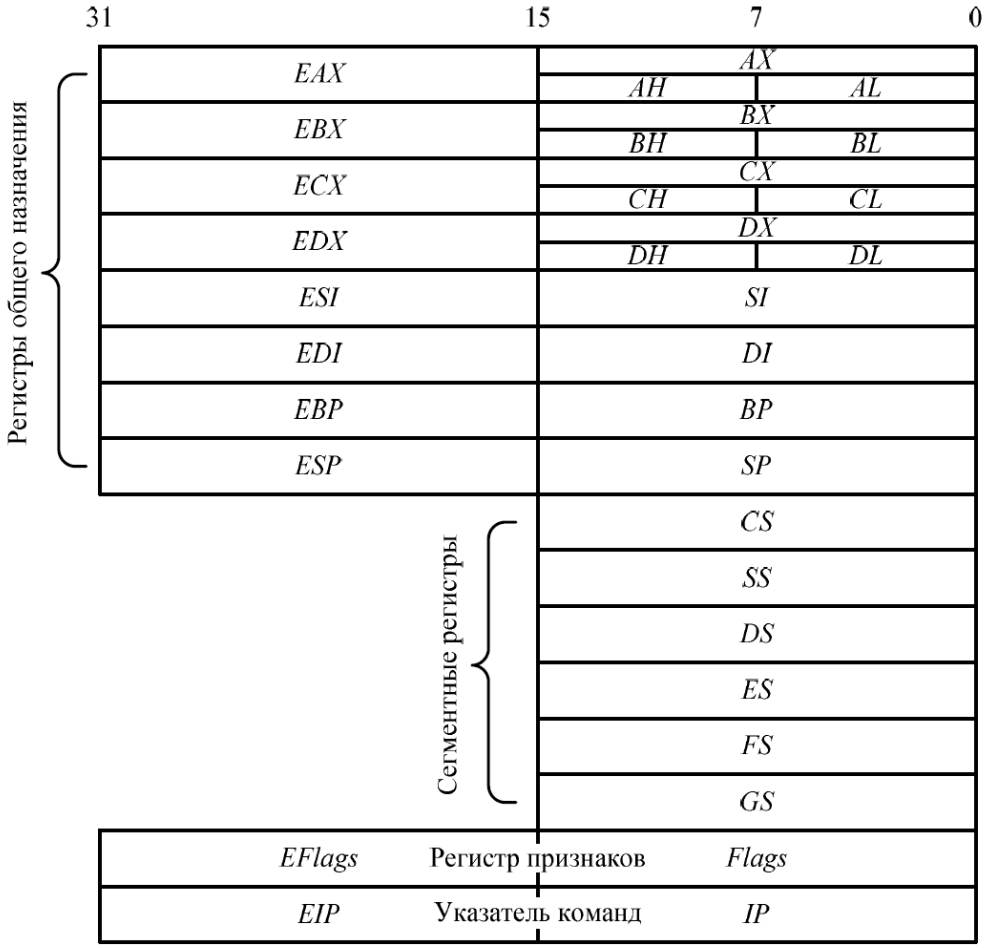


Рис. 3.2. Набор регистров микропроцессора

Следующие два регистра предназначены для поддержки так называемых цепочечных операций, в которых выполняется последовательная обработка

цепочек элементов. Каждый из этих двух регистров может иметь длину 32, 16 или 8 бит:

- регистр индекса источника (Source Index register) ESI/SI в цепочечных операциях содержит текущий адрес элемента в цепочке-источнике;

- регистр индекса приёмника (Destination Index register) EDI/DI в цепочечных операциях содержит текущий адрес в цепочке-приёмнике.

Два оставшихся регистра общего назначения предназначены для работы с особой структурой данных – стеком:

- регистр указателя стека (Stack Pointer register) ESP/SP co-
держит указатель на вершину стека в текущем сегменте стека;

- регистр указателя базы кадра стека (Base Pointer register) EBP/BP предназначен для организации произвольного доступа к данным внутри стека.

Такое разделение регистров имеется во всех современных микропроцессорах. Значительная часть внутренних операций компьютеров проводится с использованием регистров общего назначения.

Следующая группа из шести регистров помогает микропроцессору обращаться к памяти. Эти регистры получили название ***сегментных,*** поскольку каждый из них помогает обращаться к определённому сегменту (области) памяти. В первых микропроцессорах размер сегментов составлял всего 64 Кбайт, а в современных процессорах длина сегмента переменная и варьируется от одного байта до 4 Гбайт.

Регистр сегмента кода CS программы показывает, в каком месте памяти размещается программа.

Регистр сегмента данных DS локализует используемые программой данные.

Регистр дополнительного сегмента ES дополняет сегмент данных.

Регистр сегмента стека SS определяет стек компьютера.

В микропроцессорах Intel 80386 и вышеимеются ещё два дополнительных сегментных ***регистра FS и GS***, предназначенных для адресации памяти.

Регистр указателя команды IP/EIP определяет ту точку, где выполняется программа.

Регистры указателя стека SP и указателя базы BP помогают следить за информацией в стеке – области памяти, где хранится информация о текущих действиях компьютера.

Регистры индекса источника SI и индекса получателя DI помогают программам пересылать большие блоки данных из одного места в другое

Регистр признаков (регистр состояния, или регистр флагов)

Flags/EFlags (см. рис. 3.2) занимает особое место, хотя он также является внутренним регистром микропроцессора. Содержащаяся в нём информация

представляет собой двоичное слово, характеризующее различные состояния микропроцессора. Каждый бит этого слова (так называемый флаг) содержит информацию о результате предыдущей команды. Например, есть бит нулевого результата, который устанавливается в том случае, когда результат выполнения предыдущей команды – нуль, и очищается в том случае, когда результат выполнения команды отличен от нуля. Эти биты (флаги) используются командами условных переходов, например, командой перехода в случае нулевого результата. В этом же регистре иногда содержатся флаги управления, определяющие режим выполнения некоторых команд.

В качестве примера на рис. 3.3 показан состав и назначение битов регистра признаков типичного микропроцессора

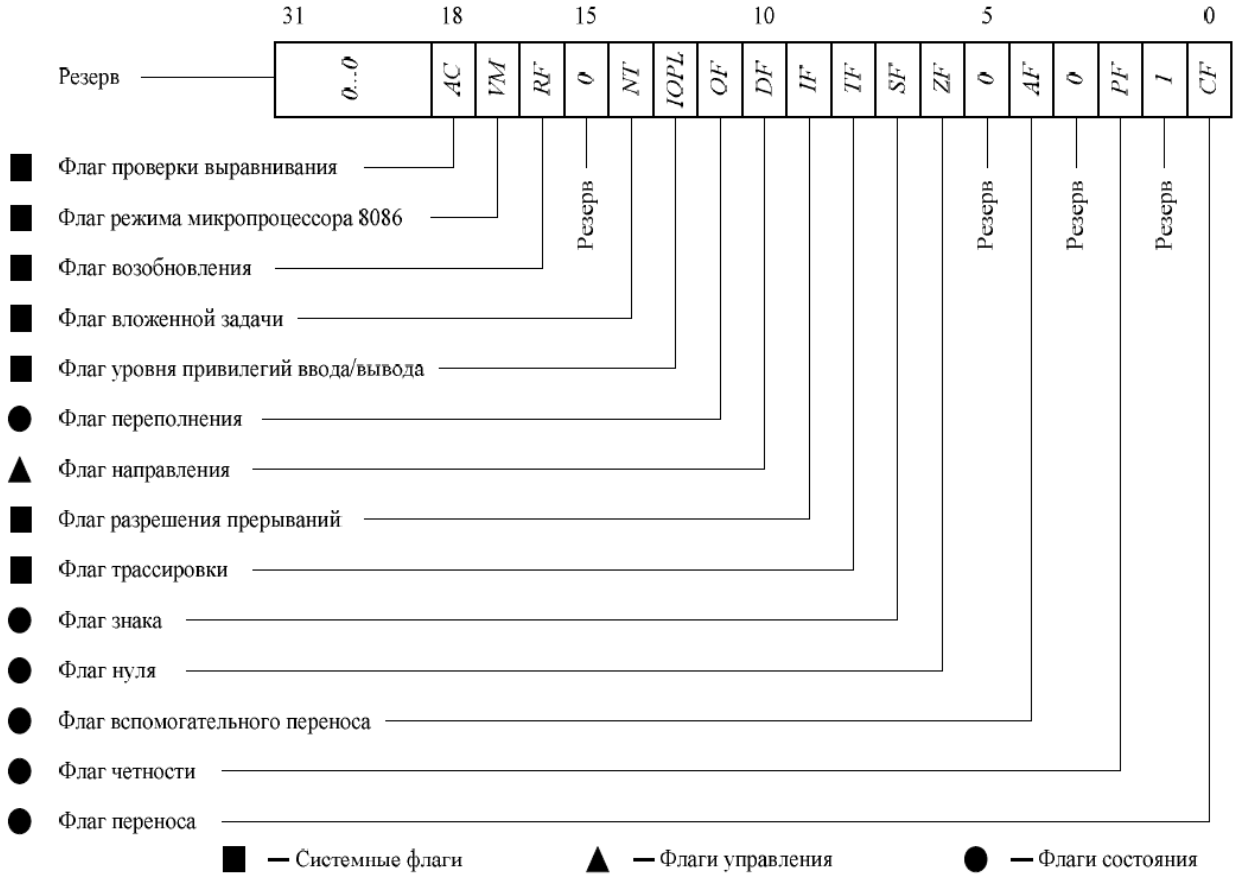


Рис. 3.3. Содержимое регистра флагов

Лекция 9. Структура команды процессора. Цикл выполнения команды. Понятие рабочего цикла, рабочего такта. Принципы распараллеливания операций и построения конвейерных структур. Классификация команд. Системы команд и классы процессоров: CISC, RISC, MISC, VLIM.

Команда микропроцессора состоит из особой инструкции, называемой кодом операции (в зарубежной литературе ей соответствует аббревиатура INS – Instruction). Разрядность команд обычно совпадает с разрядностью микропроцессора, причём форматы команд очень сильно зависят от его внутренней структуры.

Команда микропроцессора может состоять только из кода операции, когда не требуется указывать адрес операнда (т.е. тех данных, над которыми команда производит какое-либо действие), или может состоять из кода операции,адресов операндов и данных. На рисунке 3.4 показаны примеры построения команд для простейшего восьмиразрядного процессора.

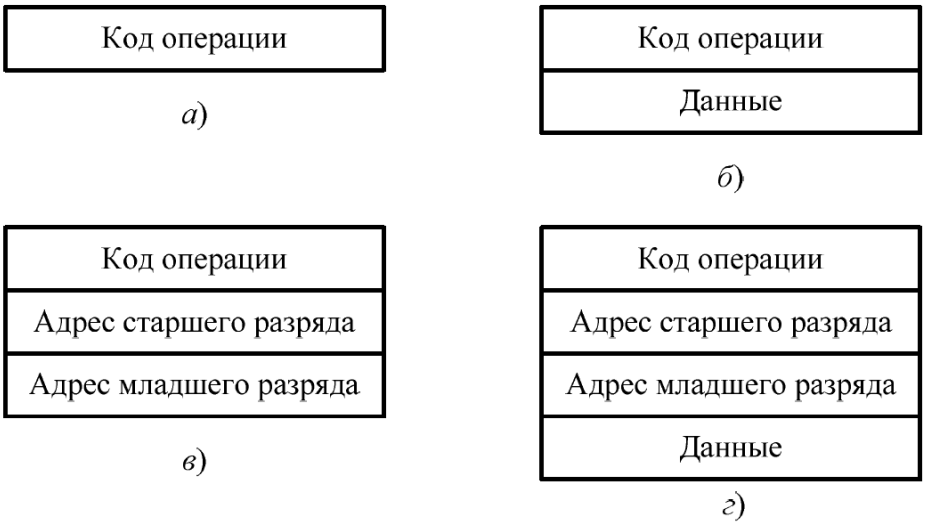


Рис. 3.4. Форматы команд микропроцессора:
а – однобайтовая команда; *б* – двухбайтовая;
в – трёхбайтовая; *г* – четырёхбайтовая

Если для кода операции используется восьмибитное двоичное слово – 1 байт, то при помощи этого слова можно закодировать 256 операций.

Система команд микропроцессора

Принципиальным достоинством микропроцессора является его программируемость. Это означает, что, подавая на вход МП команды, можно обеспечить нужную последовательность операций, т.е. реализацию определенного алгоритма.

Классификация команд МП представлена на рис. 11.1. По числу ячеек памяти, необходимых для размещения одной команды, различают команды длиной в одно, два или три слова. Команды длиной в два и три слова требуют для выборки соответственно два и три цикла обращения к памяти. В принципе возможны команды и большей длины, но практически они не используются.

Во многих случаях, в частности при сравнении МП со сходной архитектурой, оказывается полезной классификация команд в соответствии с архитектурными характеристиками МП. По этому признаку можно выделить следующие команды:

- изменения содержимого ячеек памяти;
- изменения содержимого аккумулятора;
- изменения содержимого индексного регистра;
- выполнения операций со стеком;
- выполнения операций в АЛУ;

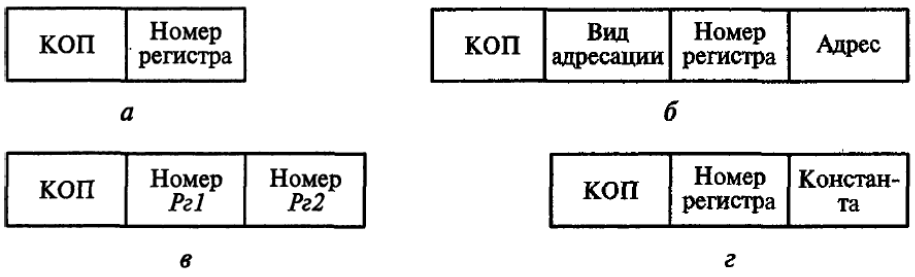


Рис. 11.1. Форматы команд микропроцессора:
а — изменения состояния регистра; *б* — выполнения операций над содержимым регистров; *в* — обращения к памяти; *г* — непосредственной адресации

• передачи управления;

• изменения содержимого регистра состояния (или регистра признаков);

• ввода — вывода.

С функциональной точки зрения команды разделяются на три большие группы:

1) передачи данных;

2) управления данных;

3) обработки данных.

Классификация команд.

Команды передачи данных. Команды этой группы обеспечивают простую пересылку информации без выполнения каких-либо операций обработки. Они подразделяются на команды, связанные с обращением к памяти, команды, связанные с обращением к регистрам, и команды ввода — вывода. Обычно программы решения задач, исходные данные и результаты операций хранятся в оперативной запоминающем устройстве (ОЗУ). Каждому элементу информации в ОЗУ отводится определенное место, которое имеет адрес.

В большинстве ЭВМ минимальной единицей адресуемой информации является байт, в связи с чем объем ОЗУ исчисляется в байтах. Достаточно часто для операндов одного байта мало, поэтому размер ячейки памяти выбирают большим, например четыре байта. Для эффективного использования всего адресуемого пространства ОЗУ МП автоматически выбирает из памяти информацию различного размера: слова (4 байт), двойные слова (8 байт), полуслова (2 байт), 1 байт, а в некоторых случаях даже отдельные биты. Используемый в командах размер информации может задаваться специальным словом в команде или в неявном виде кодом операции. При этом часть младших разрядов адреса определяет номер байта.

К командам, связанным с обращением к памяти, относятся:

• ЗАГРУЗИТЬ (ПРОЧИТАТЬ), по которой содержимое одной из ячеек памяти засылается в регистр;

• ЗАПОМНИТЬ (ЗАПИСАТЬ), по которой содержимое регистра засылается в ячейку памяти.

В этих командах, связанных с пересылкой байта или слова, должны указываться номер конкретного регистра, адрес ячейки памяти и, если необходимо, номер модуля ЗУ.

Команды, связанные с обращением к регистрам, должны указывать номер источника информации и номер регистра результата. В эту подгруппу команд передачи данных входят команды:

• ПЕРЕСЛАТЬ, по которой содержимое одного регистра пересылается в другой;

. ЗАГРУЗИТЬ НЕПОСРЕДСТВЕННО, по которой в регистр записывается константа, указанная в коде команды. Эта команда часто используется для сброса регистра в 0 или установки его в 1.

К командам ввода—вывода относятся:

• ВВОД, по которой содержимое устройства ввода засылается во внутренний регистр МП;

• ВЫВОД, по которой содержимое внутреннего регистра МП (обычно аккумулятора) пересылается в устройство вывода.

Эти команды должны следовать за командами, подготавливающими операции ввода—вывода, например за командой переключения устройства ввода—вывода (УВВ) в режим приема информации.

Команды управления. Эти команды, часто называемые командами перехода, позволяют выполнять различные действия в соответствии со значением внешних сигналов или выработанных внутри системы условий. При этом нарушается естественный порядок следования команд, при котором после выполнения очередной команды выбирается команда, расположенная в следующей по порядку ячейке памяти. Все команды управления подразделяются на команды безусловного и условного перехода.

К командам безусловного перехода относятся:

• БЕЗУСЛОВНЫЙ ПЕРЕХОД (БП), по которой в счетчик команд записывается содержимое адресного поля команды БП, т. е. обеспечивается переход в программе по адресу, указанному в команде;

• ПРОПУСТИТЬ, по которой пропускается следующая команда программы;

• БЕЗУСЛОВНЫЙ ПЕРЕХОД С ВОЗВРАТОМ (переход к подпрограмме), по которой в программный счетчик записывается новое содержимое (адрес первой команды подпрограммы), но в отличие от команды БП в памяти сохраняется состояние счетчика команд и некоторых других регистров. После выполнения подпрограммы по ее последней команде ВОЗВРАТ восстанавливается содержимое счетчика команд и всех регистров.

Команды условного перехода требуют проверки состояния какого-либо разряда регистра, флагового триггера или другого параметра. От результата проверки зависит, будет выполняться переход или нет. Обычно переход выполняется, если результат проверки соответствует указанному в команде условию. В эту подгруппу ^команд управления входят следующие команды:

• УСЛОВНЫЙ ПЕРЕХОД (УП) по адресу. В коде команды УП обязательно указывается проверяемое условие, в качестве которого в МП используются нулевое или ненулевое значение результата, положительный или отрицательный знак результата, наличие или отсутствие сигналов переноса, переполнения и др. При выполнении условия в программный счетчик записывается содержимое адресного поля команды УП, т.е. обеспечивается переход в программе по адресу, указанному в команде. При невыполнении условия управление передается следующей команде программы;

• УСЛОВНЫЙ ПЕРЕХОД С ВОЗВРАТОМ, которая отличается от команды БЕЗУСЛОВНЫЙ ПЕРЕХОД С ВОЗВРАТОМ тем, что переход к подпрограмме происходит только при выполнении указанного условия;

• УСЛОВНО ПРОПУСТИТЬ, по которой проверяется определенное условие и, если оно выполнено, пропускается следующая команда программы;

• ПРОПУСТИТЬ ПО ФЛАГУ, используемая в основном для организации взаимодействия МП и УВВ. По этой команде проверяется состояние флагового триггера;

• СРАВНИТЬ И ПРОПУСТИТЬ, по которой сравнивается содержимое двух регистров, в случае равенства пропускается следующая команда;

• ЦИКЛ, по которой содержимое определенного регистра или ячейки памяти уменьшается (или увеличивается) на единицу и, если оно станет равным нулю, пропускается следующая команда.

Команды перехода не выполняют операции с операндами и предназначены лишь для изменения порядка следования команд. Поэтому вместо сведений о местоположении операндов и результата в них содержатся сведения о местоположении очередной команды.

Команды обработки данных. Данные команды подразделяются на арифметические и логические.

К арифметическим командам относятся:

• СЛОЖИТЬ содержимое двух регистров или регистра и ячейки памяти;

• ВЫЧЕСТЬ из содержимого ячейки памяти или регистра содержимое регистра;

• УВЕЛИЧИТЬ НА 1 (или ИНКРЕМЕНТ) содержимое ячейки памяти или регистра, в частности, указателя стека, индексного регистра, аккумулятора;

• УМЕНЬШИТЬ НА 1 (или ДЕКРЕМЕНТ) содержимое ячейки памяти или регистра;• СЛОЖИТЬ С УЧЕТОМ ПЕРЕНОСА, по которой выполняется

сложение с учетом состояния триггера переноса, что позволяет легко организовывать обработку чисел большой длины;
. ВЫЧЕСТЬ С УЧЕТОМ ЗАЕМА;
• СДВИГ содержимого ячейки памяти или регистра (обычно на один разряд).

В подгруппу логических команд входят команды:
• И (или ЛОГИЧЕСКИ УМНОЖИТЬ), по которой выполняется операция конъюнкции между содержимым двух регистров или ячейки памяти и регистра;
. ИЛИ (или ЛОГИЧЕСКИ СЛОЖИТЬ), по которой выполняется операция дизъюнкции между содержимым двух регистров или ячейки памяти и регистра;
• НЕРАВНОЗНАЧНОСТЬ, по которой производится поразрядное сравнение содержимого двух регистров или ячейки памяти и регистра;
• ИНВЕРСИЯ содержимого ячейки памяти или регистра.

Команда МП состоит в общем случае из набора сведений о коде операции, адресов операндов и регистров, участвующих в операции, способов адресации. Для размещения этих сведений выделяется несколько полей, т.е. регистров, состоящих из определенного числа разрядов. Структура таких регистров и правила размещения в них команды называется форматом команды. Команды различных типов могут иметь разный формат. Каждый формат определяется кодом операции (КОП), который размещается в первом по порядку поле.

Процедура выполнения команд

Пусть при работе микроЭВМ с клавиатурой и монитором последовательно выполняются три действия (рис. 11.2): нажатие пользователем клавиши А клавиатуры; размещение буквы А (в кодированной форме) в памяти; воспроизведение буквы А на экране монитора. Такая последовательность действий (ввод—размещение—вывод) является типичной. Она многократно повторяется не только при создании с помощью ЭВМ текстовых документов, но и при наборе программ на языках программирования. Более подробная схема на рис. 11.3 позволяет понять, как именно выполняются команды.

Прежде всего рассмотрим содержимое регистров программной памяти с адресами отЮО до 105. В эти регистры предварительно были загружены три команды:

1) ВВОД — ввести данные, поступающие из порта 1;

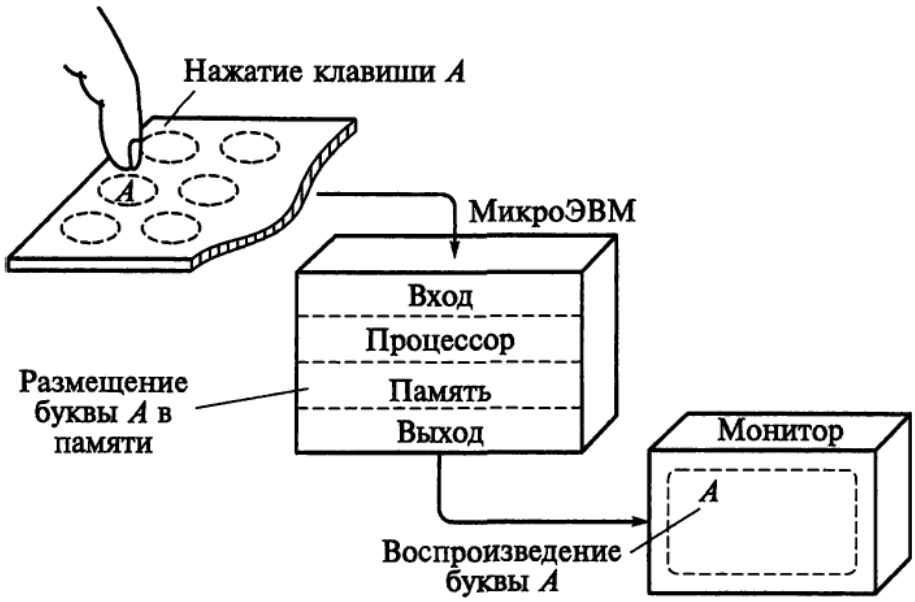


Рис. 11.2. Действия при работе микроЭВМ

2) ЗАПОМНИТЬ данные, поступающие из порта 1, т.е. записать их в ячейке памяти данных с адресом 200;

3) ВЫВОД — вывести данные через порт вывода 10.

Программа, содержащая три указанные команды, размещается в шести ячейках. Это обусловлено тем, что каждая из этих трех простых команд поделена на две части. Например, первая часть первой команды говорит, что надо выполнить операцию ВВОД (ввести данные), а вторая часть указывает, откуда подлежащие вводу данные поступают (из порта 1). Первая часть — это и есть код операции, а вторая часть — операнд. Код операции ВВОД содержится в ячейке памяти с адресом 100, код операции ЗАПОМНИТЬ — в ячейке 102, код операции ВЫВОД — в ячейке 104.

В МП на рис. 11.3 показаны только аккумулятор и регистр команд. Поскольку никаких арифметических действий эта простая программа не предусматривает, то и нет нужды показывать сумматор. Напомним, что именно МП является центром всех преобразований данных и операций.

Проследим шаг за шагом (их номера указаны на рис. 11.4 в кружках) всю процедуру выполнения трех команд.

Шаг 1. МП подает на адресную шину адрес ячейки памяти 100 и активизирует ввод считывания из программной памяти (считывание данных означает копирование информации из ячейки памяти).

Шаг 2. Программная память выставляет первую команду (ВВОД) на шину данных, а МП принимает эту кодированную информацию. Она помещается в регистр команд МП. Этой команде нужен операнд.

Шаг 3. МП подает на адресную шину адрес ячейки памяти 101. Линия управления готовит вход считывания из программной памяти.

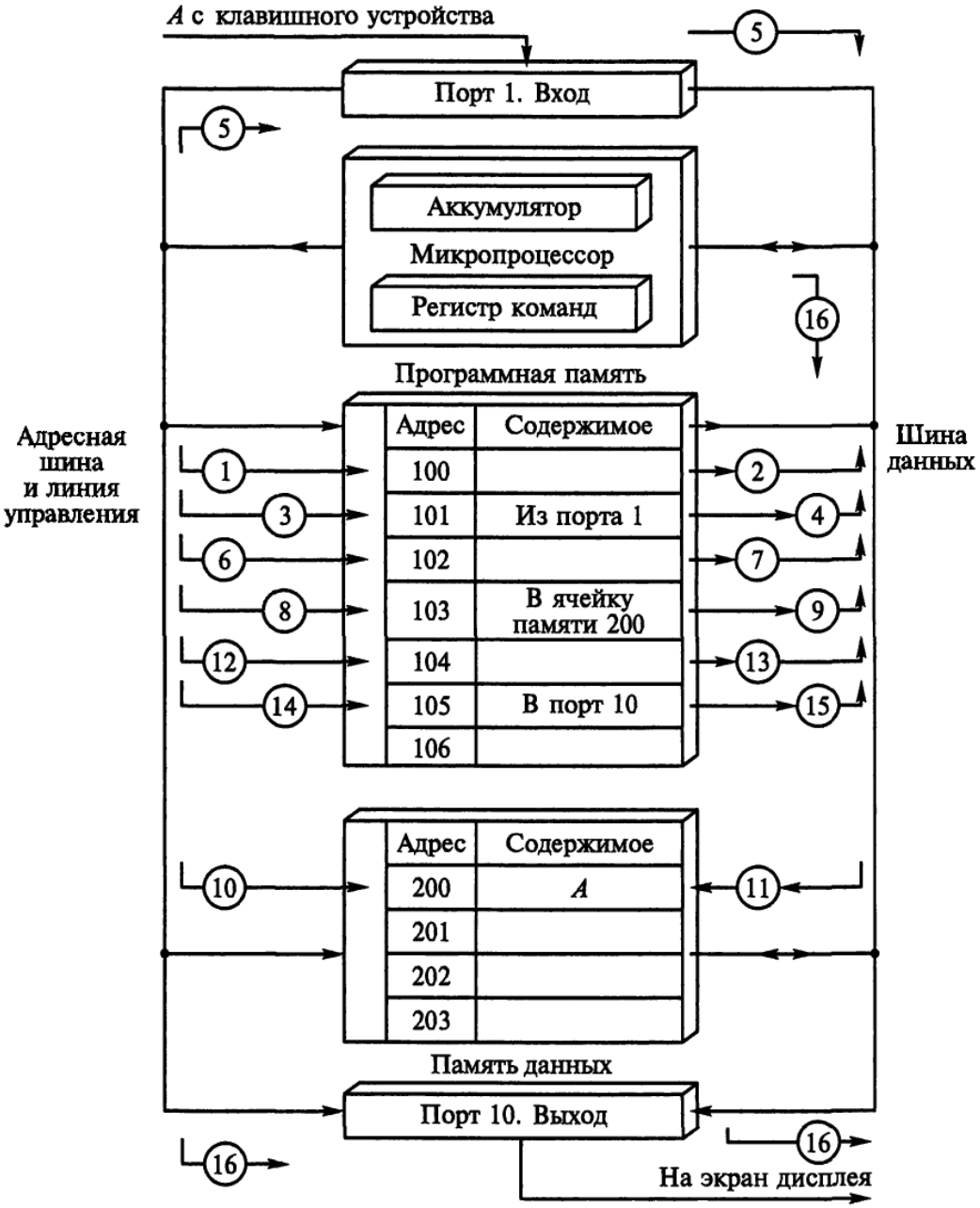


Рис. 11.3. Процедура выполнения команд

Шаг 4. Программная память помещает операнд (изъятый из порта 1) на шину данных. Этот операнд находился в ячейке памяти 101. Кодированное послание (адрес порта 1) берется с тины данных и помещается в регистр команд. МП декодирует полную команду ВВОД, как «Ввести данные, поступающие из порта 1».

Шаг 5. МП побуждает открыть порт 1 посредством адресной шины и линии управления устройством ввода. Кодированная форма А передается в аккумулятор, где и размешается.

Первая команда выполнена. Необходимо отметить, что МП все время действует в последовательности извлечение — декодирование — выполнение, т. е. сначала извлекает из программной памяти команду, затем расшифровывает (декодирует) ее и, наконец, выполняет.

Шаг 6. МП подает на адресную шину адрес ячейки памяти 102 и активизирует вход считывания посредством управляющих линий.

Шаг 7. Код команды «ЗАПОМНИТЬ данные» подается на шину данных. МП принимает его, помещает в регистр команд, декодирует и определяет, что нужен операнд.

Ш а г 8. МП подает на шину данных адрес ячейки памяти 103 и активизирует вход считывания из программной памяти.

Шаг 9. Код операнда «В ячейку памяти 200» подается из памяти (программы) на шину данных. МП принимает его и помещает в регистр команд.

Таким образом, команда «ЗАПОМНИТЬ данные, расположенные в ячейке памяти 200» полностью извлечена и декодирована. Теперь начинается процесс ее выполнения.

Шаг 10. МП подает на адресную шину адрес ячейки памяти 200 и активизирует вход записи в память (запись означает, что данные вводятся в память).

Шаг 11. МП выдает помещенную в аккумуляторе информацию (кодированная форма А) на шину данных. Это А записывается в ячейку памяти 200.

Вторая команда выполнена.

Шаг 12. МП подает на адресную шину адрес ячейки памяти
104 и активизирует вход считывания из памяти.

Шаг 13. Команда «ВЫВОД данных» подается на шину данных. МП принимает ее, помещает в регистр команд, декодирует и устанавливает, что нужен операнд.

Шаг 14. МП подает на адресную шину адрес ячейки памяти
105 и активизирует вход считывания из памяти.

Шаг 15. Код операнда «В порт 10» подается из памяти на шину данных. МП принимает его и помещает в регистр команд.

Шаг 16. МП декодирует полную команду «ВЫВОД данных в порт 10», активизирует порт 10 посредством адресной шины и управляющей линии устройства вывода. Код А (находившийся в аккумуляторе) подается на шину данных и передается портом 10 на видеотерминал

Лекция 10. Арифметико- логическое устройство (АЛУ): назначение и классификация. Интерфейсная часть процессора: назначение, состав, функционирование. Организация работы и функционирование процессора.

Арифметические и логические операции над числами (операндами, словами) выполняются в главной части процессора — арифметико-логическом устройстве.

Все арифметические действия с двумя числами (сложение, вычитание, умножение, деление) сводятся в АЛУ к операции сложения или вычитания. Поэтому в состав АЛУ обязательно входит сумматор или вычитатель (большой разницы между ними нет). Два числа (операнды) находятся в двух регистрах с соответствующими логическими схемами, их взаимодействием с сумматором руководит устройство управления. Результат выполненной операции может направляться по указанному в команде адресу, но обычно остается в специальном регистре — аккумуляторе. Процессор содержит несколько регистров, но наиболее близок к АЛУ именно аккумулятор.

На рис. 9.1 показана ***структурная схема АЛУ*** и его связи с устройством управления процессора УУП и запоминающим устройством процессора ЗУП. Аккумулятор на схеме не изображен, его роль может выполнять один из двух регистров, например Рг1.

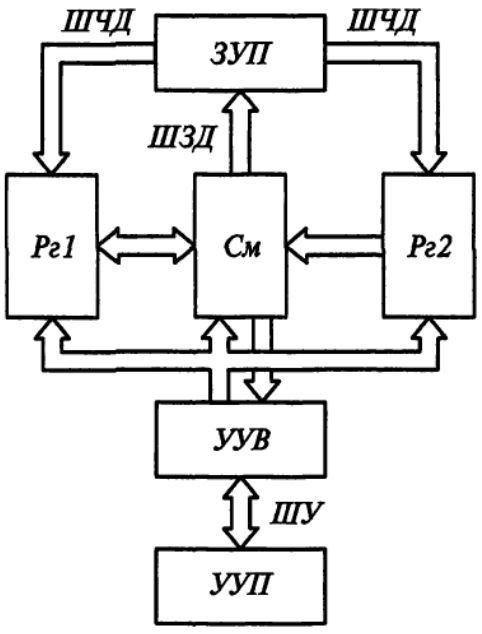


Рис. 9.1. Структурная схема арифметико-логического устройства

Сумматор См предназначен для суммирования чисел, регистры Рг1 и Рг2 — для хранения слагаемых, или уменьшаемого и вычитаемого, или множимого и множителя, или делимого и делителя (в зависимости от выполняемой операции). Устройство управления вычислениями УУВ координирует работу АЛУ, управляет последовательностью действий, необходимых при выполнении конкретной операции.

С ЗУП АЛУ связано шинами чтения данных ШЧД и записи данных ШЗД, с УУП — шиной управления ШУ, по которой в УУВпоступают тактовые импульсы, а из УУВ — сигнал об окончании вычислений.

Работает АЛУ следующим образом. Из УУП код арифметической или логической операции поступает в УУВ, где формируются сигналы, соответствующие данной операции. Затем из ЗУП выбирается первый операнд (по адресу, указанному в команде), который по ШЧД поступает на Рг1. Второй операнд, выбранный из ЗУП по второму адресу, указанному в команде, поступает также по ШЧД в Рг2. После приема обоих операндов начинается

выполнение операции, в ходе которой используются сигналы переноса, делаются необходимые сдвиги. Результат операции формируется в См. Сигналы, характеризующие признаки результата в См, называются признаками. Эти сигналы являются также составной частью понятия «состояние процессора» (подробнее см. подразд. 11.4).

По окончании формирования результата вырабатывается сигнал признака конца операции, в соответствии с которым результат из См поступает через ШЗД в ЗУП по адресу, указанному в команде. Кроме формирования результата в АЛУ могут вырабатываться различные сигналы, обозначающие признаки результата (признак переполнения, признак отрицательного результата и т.д.). Эти признаки поступают в УУП и влияют на дальнейший ход вычислительного процесса.

Основные характеристики АЛУ и его структура зависят от принятой системы счисления, способа реализации вычислительного процесса, формы представления чисел, способа представления отрицательных чисел, разрядности чисел, типа схемы АЛУ, состава операций, используемой методики вычислений и требуемого быстродействия.

В зависимости от принятой системы счисления различают АЛУ с двоичной (используется чаще всего), десятичной или двоично десятичной системой (иногда говорят — арифметикой).

В зависимости от способа реализации вычислительного процесса различают АЛУ последовательного и параллельного действия.

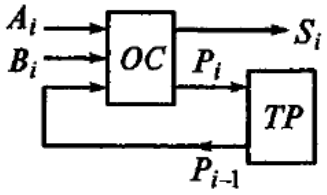


Рис. 9.2. Схема, поясняющая принцип действия АЛУ последовательного действия

В АЛУ последовательного действия каждый операнд вводится последовательно, разряд за разрядом. Операции производятся также последовательно, поразрядно. Числа представляются в виде временной последовательности сигналов и имеют один общий выход, причем каждому разряду отводится определенная временная позиция относительно заданного начала отсчета. Такое АЛУ преобразовывает временные последовательности, соответствующие обоим слагаемым, во временную последовательность, соответствующую сумме, выдаваемой по специальной цепи, начиная от младших разрядов и кончая старшими разрядами и знаком. Эту функцию обычно выполняет двоичный одноразрядный сумматор. Если его дополнить схемой хранения переносов, то он может играть роль последовательного сумматора в АЛУ последовательного действия. На рис. 9.2, поясняющем принцип действия такого АЛУ, показаны одноразрядный

сумматор ОС и триггер переноса ТР. При суммировании цифр Ai и Bi очередного i-го разряда в ОС из ТР поступает сигнал переноса Pi-1, полученный при суммировании цифр предыдущего (i - 1)-го разряда. По быстродействию АЛУ последовательного действия уступают АЛУ параллельного действия, но зато содержат меньше элементов, т.е. являются более дешевыми.

В АЛУ параллельного действия (рис. 9.3) все n разрядов каждого операнда поступают одновременно по n каналам. Действия над числами производятся также одновременно во всех n разрядах. Подобное АЛУ можно представить как n одноразрядных сумматоров, соединенных таким образом, что выход сигнала переноса Pi предыдущего одноразрядного сумматора ОС, является входом последующего одноразрядного сумматора ОСi + 1. В каждом из таких сумматоров складываются цифры соответствующего разряда чисел A и B.

В зависимости от формы представления чисел различают АЛУ, оперирующие числами с фиксированной запятой, с плавающей запятой и целыми. При одинаковой разрядности в АЛУ с плавающей запятой имеется больший диапазон представления чисел, чем в АЛУ с фиксированной запятой. При обработке чисел с плавающей запятой возможны два способа выполнения операций: последовательный и параллельный. При последовательном способе в одном и том же устройстве сначала вычисляется порядок результата, а затем его мантисса. При параллельном способе порядок и мантисса результата вычисляются в разных устройствах.

С разрядностью АЛУ связаны точность и скорость вычислений. Чем больше разрядность, тем выше точность, но меньше быстродействие. Разрядность АЛУ может быть постоянной и переменной.

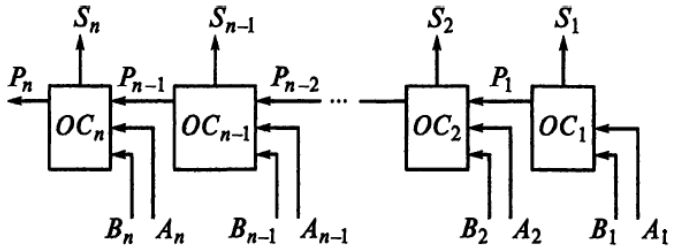


Рис. 9.3. Схема, поясняющая принцип действия АЛУ параллельного действия

Интерфейсная часть процессора: назначение, состав, функционирование.

Вся обработка информации (включая вычисления), осуществляемая ЭВМ, происходит в процессоре, который взаимодействует с другими устройствами.

Некоторые из этих устройств входят в состав ЭВМ, а некоторые не входят, т.е. являются внешними, или периферийными. Для обмена информацией между этими устройствами необходимы специальные схемы, программы и правила, которые получили название ***средства интерфейса.***

Интерфейс - это комплекс аппаратных и программных средств, обеспечивающих обмен информацией в вычислительной технике.

Работа компьютера сопровождается интенсивным обменом информацией между МП, памятью (различными ЗУ) и УВВ. В процессе выполнения программы МП принимает команды из ЗУ команд, расшифровывает их, при исполнении команд, включающих чтение и запись, обращается к ЗУ данных, а при исполнении команд ввода—вывода — к УВВ. Эффективность решения задачи оператором на компьютере в значительной степени определяется организацией этого обмена и структурой связей между МП, памятью и УВВ.

Система шин, вспомогательной аппаратуры и алгоритмов, реализованных на этом оборудовании, предназначенная для организации обмена между МП, памятью и УВВ, и называется интерфейсом. В функции интерфейса входят дешифрование адреса устройства, синхронизация обмена информацией, согласование форматов слов, дешифрование кода команды, связанной с обращением к памяти или УВВ, электрическое согласование сигналов и некоторые другие операции.

Сложность задач, возлагаемых на интерфейс, а также недостаточная мощность буферных схем, входящих в состав БИС МП, привели к распределению средств интерфейса между различными устройствами:

• устройством управления памятью и вводом—выводом, входящим в состав МП;

• непосредственно интерфейсным устройством, являющимся промежуточным звеном между МП, с одной стороны, и памятью и УВВ, с другой;

• специализированными устройствами управления (контроллерами) УВВ, предназначенными для реализации алгоритмов управления.

Организация обмена между МП и памятью или УВВ в простейших случаях возможна на основе средств, содержащихся только в МП. Недостающие средства в таких случаях реализуются программно. Более сложные ЗУ и УВВ соединяются с МП обязательно через дополнительные интерфейсные устройства.

Существуют три способа организации связи между МП и УВВ:

1) программно-управляемая передача данных;

2) использование прерываний;

3) прямой доступ к памяти.

Прежде чем подробно проанализировать эти способы, остановимся на организации связи между МП и ЗУ данных и команд.

Рассмотрим некоторые особенности организации связи МП с памятью. Под памятью будем понимать внешние по отношению к МП запоминающие устройства (в отличие от внутренних РОН, которые могут рассматриваться как СОЗУ) с произвольным доступом, выполняющие две функции:

1) хранение команд, под управлением которых работает МП;

2) хранение данных для обработки в соответствии с управляющим алгоритмом, т. е. исходных данных, промежуточных и окончательных результатов обработки.

Данные обычно хранятся в оперативном ЗУ (ОЗУ), а команды — в постоянном ЗУ (ПЗУ) или полупостоянном ЗУ (ППЗУ). Внешние выводы отдельных больших интегральных схем ЗУ, через которые МП подключается к памяти, представлены для ОЗУ на рис. 13.1, а, для ПЗУ — на рис. 13.1, б. В ОЗУ показаны две шины данных — входная и выходная. Однако многие ОЗУ имеют одну двунаправленную шину данных, направление передачи информации по которой определяется значением подаваемого на вход У управляющего сигнала Зп/Чт. В режиме записи информации (Зп/Чт - 1) эта шина работает как входная, а в режиме чтения (Зп/Чт = 0) — как выходная.

Интегральные схемы ЗУ имеют также группу управляющих входов выборки кристалла (ВК), которые могут быть прямыми илиинверсными. Эти входы являются входами схемы И, разрешающей обращение к БИС ЗУ, поэтому ОЗУ, показанное на рис.13.1, а, может записывать или выдавать информацию при ВК1 = 1 и ВК2 = 0, а ПЗУ, показанное на рис. 13.1, б, — выдавать информацию лишь при ВК1 = ВК2 = ВКЗ = 1. Выводы ВК предназначены для разрешения обращения к данной БИС при наличии нескольких из них. Различные варианты БИС ЗУ различаются числом входов ВК и наличием или отсутствием инвертирующих схем по отдельным входам ВК.Многомодульная организация является типовым способом организации памяти микроЭВМ, поскольку емкость БИС (модуля) ЗУ, как правило, меньше необходимой емкости памяти. Пример многомодульной организации памяти, содержащей к модулей ЗУ (МЗУ), с двунаправленной шиной данных представлен на рис. 13.2. Для обмена информацией между МП и памятью требуются адресации памяти, подача сигналов Зп/Чт и синхронизация работы памяти и МП.

Интерфейс ввода—вывода служит буфером между системной шиной и внешними устройствами. Используя интерфейс ввода—вывода, можно проектировать внешние устройства независимо от структуры шины, совместно

с которой он будет использоваться. Для подключения одного и того же ЗУ к двум шинам с различной структурой нужно изменить только интерфейс.

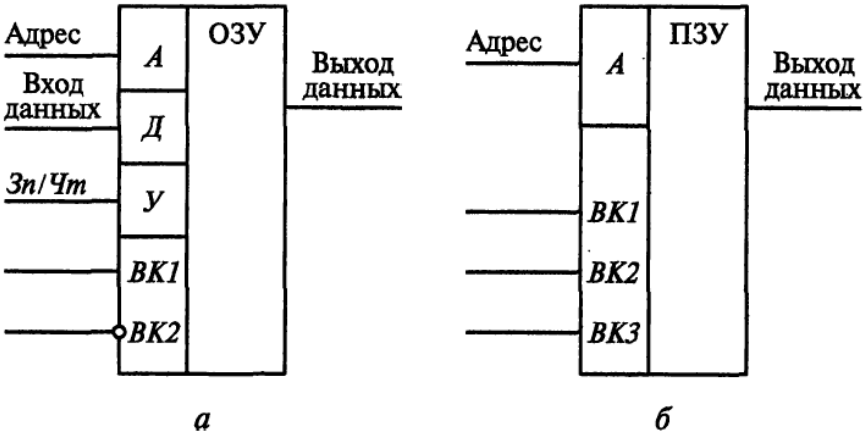


Рис. 13.1. Организация связи (интерфейса) микропроцессора с памятью: *а* — оперативной; *б* — постоянной

В общем случае процессор ЭВМ получает данные от нескольких источников (клавиатуры, датчики, устройства хранения данных и программ, другие процессоры и ЭВМ). Этот же процессор передает информацию на мониторы (дисплеи), прочие индикаторные устройства, ЗУ, вычислительные блоки, управляющие элементы системы. В тех устройствах, которые предназначены для оператора, информация выводится в удобной для человека форме: буквенно-цифровой текст и таблицы появляются на экране или бумаге; графики, схемы, рисунки, чертежи — на бумаге; звуковые сигналы, речь или музыка звучат из динамиков. В остальных устройствах информация проходит в форме электрических сигналов, причем не только в виде двоичных кодовых комбинаций, но и в виде аналогового (непрерывного) сигнала.

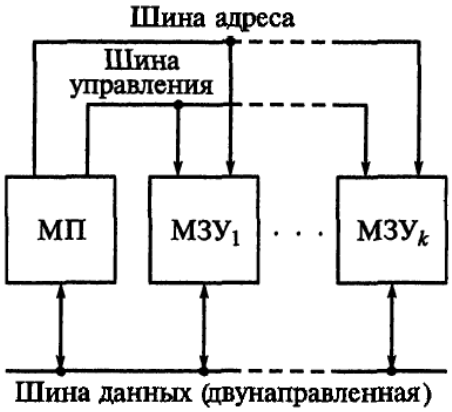


Рис. 13.2. Интерфейс с двунаправленной шиной (многомодульная организация памяти)

Передача данных в процессор и из него может выполняться параллельно или последовательно. При параллельном способе для каждого передаваемого разряда двоичного сообщения имеется отдельная физическая линия (провод), а при последовательном способе все разряды передаются по единственной линии один за другим, по очереди.

Параллельный интерфейс

При параллельном интерфейсе одновременно по нескольким параллельным проводам передается несколько бит информации аналогично

тому, как по многоразрядной шине происходит обмен данными между процессором и памятью. Поэтому параллельный интерфейс может быть 8-, 16-битным и т.д. Поскольку для передачи каждого бита используется один разряд в двоичной комбинации, то можно говорить «8-, 16-разрядный интерфейс». При параллельной передаче нескольких разрядов достигается более высокая скорость передачи информации, чем при последовательной. Для характеристики интерфейса по допустимой скорости передачи используется понятие «пропускная способность» интерфейса. Она достигает нескольких десятков мегабайт в секунду. Для увеличения пропускной способности необходимо увеличивать или число разрядов (бит), передаваемых параллельно, или тактовую частоту передачи.

Интерфейс микропроцессорных шин адреса и данных представляет собой специальные аппаратные средства, в которые (или из которых) можно пересылать данные с помощью команд чтения (или записи) памяти и чтения (или записи) сигналов УВВ. Обычно для задания режима работы интерфейсного устройства используются команды, задающие направление передачи информации и другие параметры пересылки. Эти команды управляют работой соответствующих регистров, входящих в состав аппаратуры параллельного ввода—вывода. Указанные регистры называются портами. Подсоединение к порту выполняется через соответствующий штепсельный разъем. В компьютере имеется несколько параллельных портов, через которые подключаются периферийные устройства, обозначаемые, например, LPT1, LPT2. Параллельные интерфейсные устройства сходного назначения имеют разные названия: «устройство параллельного ввода—вывода», «периферийно-интерфейсный адаптер», «универсальный интерфейсный адаптер».

На рис. 13.3 приведена структурная схема параллельного интерфейсного устройства, состоящего из единственного порта и управляющего регистра. Данные, загружаемые в управляющий регистр, определяют назначение каждого внешнего контакта порта, т.е. используется этот контакт для ввода или вывода. Подача специального входного сигнала указывает направление передачи: между процессором и портом или между процессором и управляющим регистром.

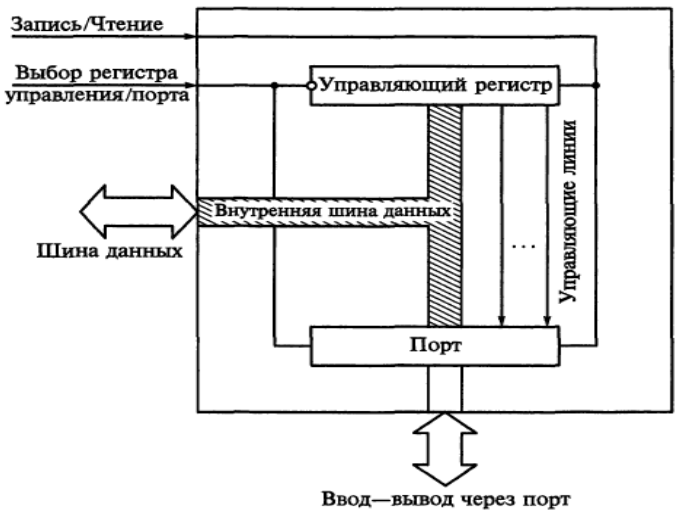


Рис. 13.3. Структурная схема параллельного интерфейсного устройства ввода — вывода

Выбор устройства передачи осуществляется выходными сигналами дешифратора, подключенного к адресной шине внутри системного блока или к шине адресов ввода—вывода, причем одновременно не может быть выбрано более одного устройства. Подобные устройства могут применяться для считывания состояния переключателей и для вывода информации на элементы с двумя состояниями, такие как световые индикаторы, реле, пускатели.

При более сложных пересылках данных в схеме ввода—вывода обычно используются сигналы, характерные для режимов работы с квитированием: сигнал готовности данных и сигнал-квитанция, служащий для подтверждения приема. На рис. 13.4 показано применение этих сигналов для управления передачей (пересылкой) данных между двумя системами (передатчиком и приемником, которые могут меняться ролями). Передатчик изменяет значения данных на выходных линиях и после небольшой задержки, в течение которой происходит установка этих значений, информирует приемник с помощью сигнала готовности данных ДГ о том, что данные для пересылки готовы. Приемник воспринимает данные и сообщает об этом посредством короткого сигнала-квитанции КВположительной полярности. Сигнал КВ может быть использован в передатчике для сброса сигнала ДГ, а в некоторых случаях и для выработки сигнала прерывания, сообщающего процессору о возможности вывода очередного элемента данных.

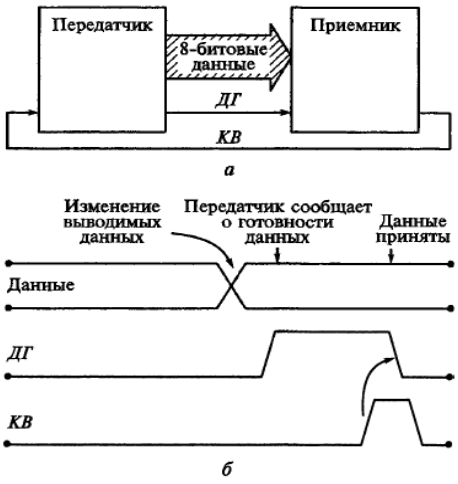


Рис. 13.4. Квитирование (управление передачей данных) при параллельных пересылках:
а — схема пересылки; *б* — временна́я диаграмма

Последовательный интерфейс

Последовательный интерфейс не обеспечивает столь высокой скорости передачи информации, как параллельный, тем не менее, он достаточно часто применяется в вычислительной технике. Объясняется это тем, что с помощью параллельного интерфейса экономически эффективно передавать информацию лишь на короткие расстояния. При значительном удалении источника от приемника возрастает стоимость кабелей и буферных устройств, что существенно повышает стоимость всей системы в целом.

Последовательный интерфейс позволяет передавать информацию, содержащую много бит, по одной линии (т. е. по одной паре проводов). Рассмотрим принцип работы последовательного интерфейса, пользуясь структурной схемой, показанной на рис. 13.5. Сначала источник данных загружает информацию в сдвиговый регистр с параллельной записью, запускает генератор тактовых импульсов и счетчик. Каждым тактовым сигналом данные в регистре сдвигаются на одну позицию вправо и поступают при этом на линию данных. Приемник состоит из еще одного сдвигового регистра, счетчика и логической схемы, которые управляются теми же тактовыми сигналами. После того как счетчик зафиксирует поступление необходимого числа тактовых импульсов, он можетинициировать параллельную передачу данных из сдвигового регистра в буферный. Этот метод последовательной передачи данных является одним из видов синхронной передачи, при которой вместе с данными необходимо передавать тактовые сигналы. Для такой последовательной передачи не требуется несколько параллельных линий передачи данных — достаточно одной, но еще одна линия нужна д ля передачи тактовых сигналов.

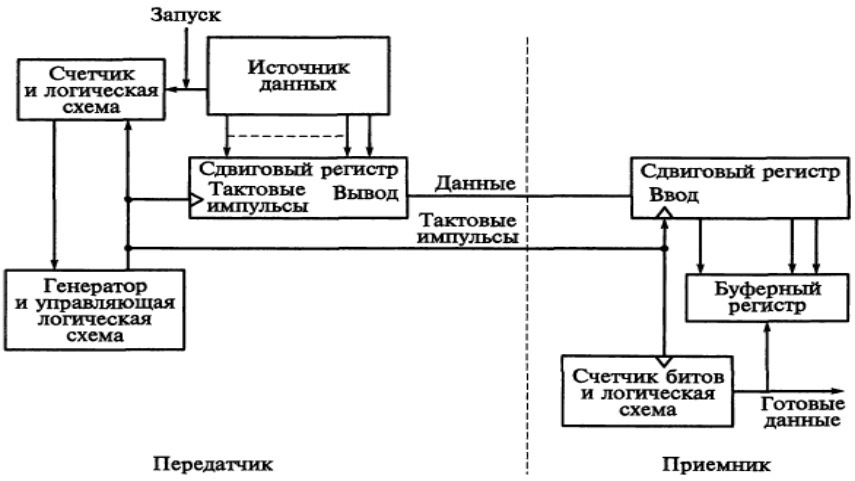


Рис. 13.5. Структурная схема синхронной последовательной передачи данных

При асинхронной последовательной передаче не требуется и отдельная линия для тактовых сигналов. Структурная схема такой передачи показана на рис. 13.6. Оба сдвиговых регистра, и на стороне источника, и на стороне приемника, управляются тактовыми импульсами, но на каждой стороне имеется собственный генератор тактовых импульсов. Оба генератора должны вырабатывать импульсы одинаковой частоты. Для того чтобы информировать приемник о начале передачи, по линии посылается так называемый стартовый бит. Затем передается блок данных определенного размера (например, 8 бит, начиная с младшего). Далее информация из сдвигового регистра приемника передается в его буферный регистр (на это выделяется определенное время), после чего можно передавать следующий блок данных, снова предварительно послав стартовый бит. Для асинхронной последовательной передачи устанавливается определенная стандартная частота. Поскольку при передаче одного блока может возникнуть ошибка внаоборот), добавляется еще один бит — контрольный, или так называемый бит четности. Значение его (0 или 1) выбирается таким образом, чтобы вместе с ним общее число единиц в блоке было четным. Например, если в блоке передана кодовая комбинация 01011000, то значение бита четности должно быть 1, так как в этом случае общее число единиц (4) будет четное. Логическая схема проводит контроль четности, и в случае выявления ошибки передача данного блока повторяется. Для определения уровней сигналов и режимов последовательной передачи используются определенные стандарты. Широко применяется разработанный Международным консультативным комитетом по телеграфии и телефонии (МККТТ) стандарт V24. Ему соответствует аналогичный американский стандарт RS232. Эти стандарты позволяют, например, объединять компьютеры'в глобальные сети с помощью существующих кабелей телефонных сетей. При этом необходимо использование дополнительных устройств

(модемов, мультиплексоров и демультиплексоров). Подробнее компьютерные сети рассмотрены в гл. 19.

При передаче данных понятия логической 1 и логического 0 являются условными, поэтому в разных стандартах и разных наборах микросхем могут быть приняты разные условия для сигналов. Пусть принято условие, что при отсутствии передачи данных в линии имеется сигнал 1. Поскольку генератор тактовых импульсов в это время не включен, то этот сигнал говорит о том, что линия готова для передачи, но самой передачи пока нет. Для информации о начале передачи сигнал в линии изменяется на 0 в тот же момент времени, когда проходит первый из тактовых импульсов. Затем начинают передаваться биты данных, по одному за каждый период тактовых импульсов, начиная от младшего разряда к старшему. Для контроля правильности передачи после старшего бита данных, как уже отмечалось, передается бит четности.

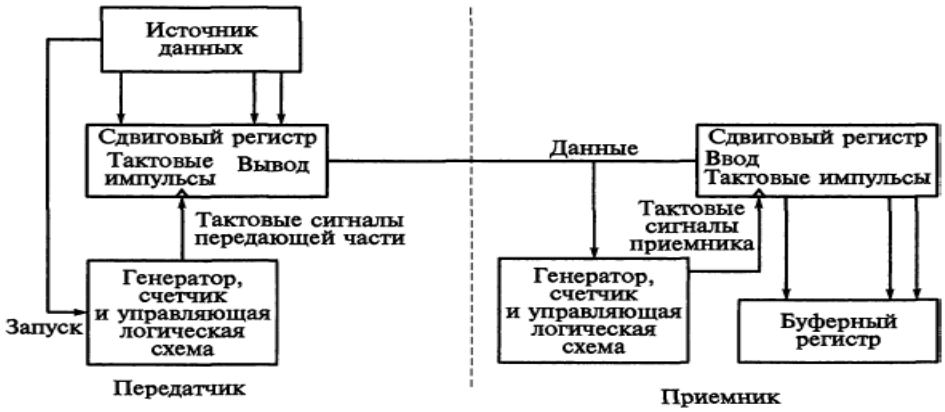


Рис. 13.6. Структурная схема асинхронной последовательной передачи данных

Интерфейсы современных персональных компьютеров

Число разнообразных устройств, которые могут работать совместно с персональным компьютером, непрерывно растет. К ним относятся дополнительные внешние запоминающие устройства (жесткие диски, компакт-диски, DVD-устройства), сканеры, цифровые фото- и видеокамеры, различные средства оргтехники и т.д. Компьютер может управлять работой различных бытовых приборов, системами охранной сигнализации, обеспечения комфортных условий в помещении, помогать в планировании и выполнении распорядка дня.

Все это требует развития средств взаимодействия компьютера с большим числом цифровых устройств. Для обеспечения такого взаимодействия нужны современные внешние интерфейсы. Рассмотрим некоторые из них.

Широкую популярность завоевал интерфейс USB (Universal Serial Bus — универсальная последовательная шина), или шина USB. Стандарт этого интерфейса был утвержден в 1996 г. по инициативе целого ряда фирм,

производящих компьютеры, другие средства и элементы вычислительной техники. Основная цель, поставленная перед разработчиками этого стандарта, заключалась в том, чтобы максимально облегчить пользователям компьютеров подключение новых периферийных устройств, их настройку и работу с ними. Такой подход получил название plug-and-play, т.е. включил и играй (без каких-либо дополнительных забот). Это означает, что должны быть предусмотрены подключение устройства к работающему компьютеру, автоматическое распознавание этого устройства сразу после подключения, подбор и установка соответствующих управляющих программ — драйверов. Перечисленные операции компьютер должен выполнить сам, не требуя от пользователя каких-либо специфических сведений.

Интерфейс USB особенно удобен для подключения дополнительных внешних ЗУ. Он позволяет иметь в качестве сменного накопителя информации не трехдюймовую дискету емкостью 1,44 Мбайт, а накопители в сотни раз большей емкости. На практике лучше всего использовать флэш-память (см. подразд. 12.8), поскольку для нее не требуется никакого дисковода.

Этот интерфейс наиболее подходит для часто подключаемых и отключаемых приборов, таких как цифровые камеры.

Подключение всех периферийных устройств к компьютеру с использованием интерфейса USB осуществляется через специальную плату внутри системного блока, которая носит название хост. В английском языке слово host имеет два значения: хозяин таверны и множество. Технический термин «хост» следует понимать как устройство, по-хозяйски управляющее многими другими устройствами.

Для того чтобы к одному порту интерфейса USB можно было подключать более одного устройства, применяются хабы. В английском языке hub — втулка (а от нее идет много спиц, т. е. связей). Корневой хаб находится внутри компьютера вместе с хостом, к которому он непосредственно подключен.

При описании интерфейса USB используют специальный термин «функция», под которым понимается логически законченное устройство, выполняющее какую-либо определенную функцию. Топология интерфейса USB представляет собой набор из семи уровней (рис. 13.7). На самом верхнем находятся хост и корневой хаб, а на самом нижнем — только функции. На каждом уровне может быть выполнено подключение и хабов, и функций, т.е. устройств.

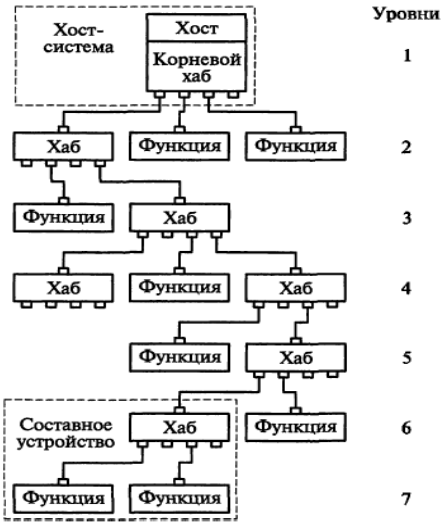


Рис. 13.7. Топология интерфейса USB

Все передачи данных по интерфейсу управляются хостом, а сами данные передаются в виде пакетов, т. е. наборов некоторого числа нулей и единиц, несущих определенную информацию. Используется несколько видов пакетов. Пакет-признак описывает тип и направление передачи данных, адрес устройства и порядковый номер конечной точки. В зависимости от типа этого пакета данные передаются от компьютера к устройству или от устройства к компьютеру. Пакет с данными содержит передаваемые данные. Пакет согласования предназначен для сообщения о результатах пересылки данных. Все эти пакеты (вместе с напряжением питания) передаются по четырехпроводному кабелю, заканчивающемуся стандартными разъемами.

Высокоскоростной последовательный интерфейс FIREWIREимеет характеристики, близкие к характеристикам интерфейсаUSB, появился раньше его, но менее распространен, поскольку более дорог. Он также обеспечивает питанием подключаемые устройства. При этом потребляемый ими ток в три раза больше, чем у USB — 1,5 А, однако общее число подключаемых устройств вдвое меньше. На практике редко требуется подключение более десятка периферийных устройств, но и в этом случае проблемой является большое число кабелей. Поэтому очень важно появление беспроводных интерфейсов для подключения дополнительных устройств.

Беспроводной интерфейс BLUETOOTH использует высокочастотный (2,4 ГГц) радиосигнал, что позволяет подсоединять устройства, находящиеся на расстоянии до 100 м от компьютера.

Беспроводной интерфейс IrDA использует оптический инфракрасный канал (сокращение «Ir» в названии этого интерфейса означает infrared — инфракрасный) для подключения устройств, находящихся на расстоянии в несколько десятков метров. При таких расстояниях это оказывается дешевле, чем использование радиосигнала.

Тема 2.4:Организация работы памяти компьютера.

Лекция 11. Иерархическая структура памяти. Основная память ЭВМ. Оперативное и постоянное запоминающие устройства: назначение и основные характеристики. Организация оперативной памяти.

Виды и характеристики запоминающих устройств

Запоминающие устройства классифицируют по ряду признаков.

По длительности хранения информации ЗУ подразделяются на долговременные и кратковременные. В свою очередь, долговременные ЗУ, или ЗУ с долговременным хранением информации, подразделяются на постоянные ЗУ (ПЗУ) и полупостоянные ЗУ (ППЗУ). Характерной чертой ПЗУ и ППЗУ является сохранение информации при отключении источников питания. В ПЗУ возможна лишь однократная запись информации, производимая либо в процессе производства, либо в результате программирования. В ППЗУ возможно многократное изменение хранимой информации при эксплуатации.

Кратковременные ЗУ используются для хранения оперативной, часто меняющейся информации. В этих ЗУ отключение источников питания приводит, как правило, к потере хранимой информации. При сокращении длительности цикла записи ППЗУ могут быть использованы и для хранения оперативной информации. В большинстве случаев ППЗУ могут применяться и в качестве ПЗУ.

По адресации эти устройства подразделяют на ЗУ с произвольной, последовательной и ассоциативной выборкой. В ЗУ с произвольной выборкой (или доступом) время обращения не зависит от адреса ячейки в устройстве. В ЗУ с последовательной выборкой для нахождения числа по определенному адресу необходимо последовательно просмотреть все ячейки, предшествующие заданной, т. е. время обращения зависит от адреса. Для поиска определенной информационной единицы в таком ЗУ необходимо сначала отыскать соответствующий массив, а затем информационную единицу в этом массиве.
Запись в устройство памяти, т. е. в ЗУ, а также чтение из ЗУ называют еще обращением к памяти. Важнейшими характеристиками ЗУ являются его информационная емкость и быстродействие.

Емкость памяти определяет число слов, которые могут храниться в памяти, и выражается в битах или байтах. Например, если память ЭВМ составляет 64 Кбайт, это означает, что она может хранить 64 х 1024 восьмиразрядных слов (1 Кбайт = 1024 байт).

Быстродействие ЗУ определяется продолжительностью обращения к нему. При записи слова время обращения к ЗУ t0 складывается из времени поиска слова tп, времени стирания ранее записанного слова tст (при необходимости) и времени записи нового слова tзп, т. е. t0 = tп + tст + tзп. При чтении время обращения к ЗУ складывается из времени поиска слова tп, времени чтения слова tчт. и времени его восстановления в памяти tвос (при необходимости), т.е. t0 = tп+ tчт+ tвос.

Работа ЗУ основана на различных физических явлениях. Наибольшее распространение в вычислительной технике получили полупроводниковые, магнитные и оптические ЗУ.

По назначению ЗУ подразделяются на сверхоперативные (СОЗУ), оперативные (ОЗУ), внешние и внутренние. Внешние ЗУ (ВЗУ) предназначаются для хранения больших массивов информации. Все математическое программное обеспечение вычислительных систем находится именно в ВЗУ. Информационная емкость ВЗУ в настоящее время составляет терабайты.

При необходимости использования той или иной программы в ходе требуемого процесса обработки эта программа предварительно передается из ВЗУ в ОЗУ. В качестве внешних ЗУ используют накопители на магнитных дисках (НМД) и на оптических дисках (НОД). При создании архивов программ и данных, их резервных копий обычно применяют накопители на магнитных лентах (НМЛ).

Для промежуточного хранения данных и программ, которыми обмениваются устройства, работающие с разной скоростью и используемые неравномерно, применяются буферные ЗУ (БЗУ). Наиболее часто они бывают необходимы в многоканальных устройствах обмена.

На рис. 12.1 в виде обобщенной схемы приведена иерархия ЗУ. В конкретных условиях число уровней иерархии, а также числоблоков СОЗУ, ОЗУ, БЗУ и ВЗУ может быть разным. На I уровне используются сравнительно небольшие по информационному объему, но быстродействующие СОЗУ, тесно связанные с процессором (Пц). К ЗУ II уровня относятся ОЗУ с произвольной выборкой, обладающие большей емкостью, чем ЗУ I уровня, но имеющие более низкое быстродействие. К ЗУ III уровня относятся БЗУ с произвольной выборкой, емкость которых занимает промежуточное положение между емкостями СОЗУ и ОЗУ. Быстродействие БЗУ должно быть высоким; часто обращение к БЗУ организовано по неявным адресам. К ГУ уровню иерархии относятся ВЗУ с плавающим временем выборки, но с большой информационной емкостью, например НМД, НОД или НМЛ.

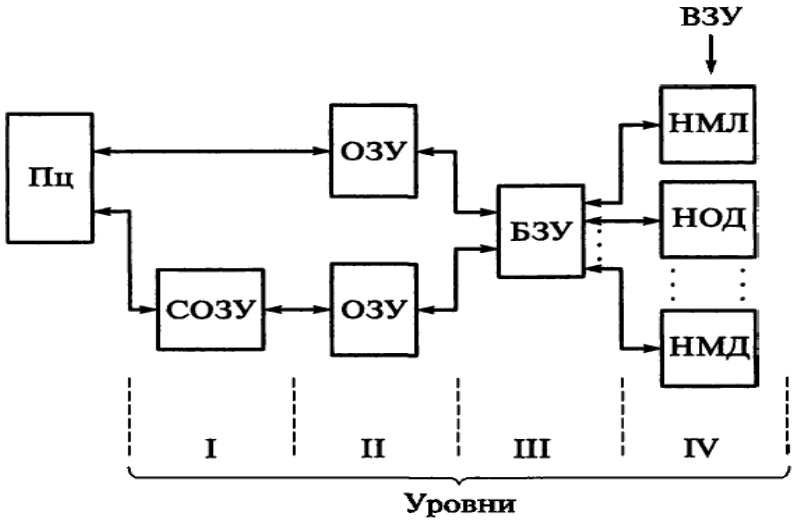


Рис. 12.1. Иерархия запоминающих устройств

Оперативные запоминающие устройства

Оперативные запоминающие устройства предназначаются для хранения данных и программ текущих вычислений, а также данных и программ, к которым следует перейти, если в ходе вычислительного процесса возникнет прерывание. Быстродействие ОЗУ соизмеримо со средним временем выполнения операций. С процессорными устройствами ОЗУ связано непосредственно или через сверхоперативные ЗУ (СОЗУ). В большинстве случаев в качестве запоминающих элементов используются триггерные схемы в полупроводниковом исполнении, объединенные в запоминающие матрицы. Быстродействие СОЗУ соизмеримо с быстродействием устройств, выполняющих самые быстрые операции над операндами. Информационная емкость СОЗУ обычно невелика. Такие устройства необходимы для того, чтобы обращение к памяти за операндами не приводило к увеличению времени выполнения операций. С целью сокращения длины линий связи СОЗУ обычно не делают самостоятельными устройствами, а вводят их в состав процессора.

В современных МП имеются СОЗУ, сформированные на том же кристалле, что и все остальные элементы МП. Расстояния от триггеров таких СОЗУ до АЛУ настолько малы, что информация передается за доли наносекунд. В новейших МП самая быстродействующая память получила название кэш 1-го уровня (см. подразд. 8.2), менее быстрое СОЗУ — кэш 2-го уровня. Емкость кэш-памяти 1-го уровня в современных ЭВМ составляет 32 Кбайт, а кэш-памяти 2-го уровня — 512 Кбайт.

Рассмотрим кратко принцип действия полупроводниковой оперативной памяти с произвольным обращением. Основу ЗУ на полупроводниковых транзисторах составляют запоминающие матрицы (ЗМ), состоящие из запоминающих элементов (ЗЭ) в виде триггерных схем. Под словом «матрица»

в данном случае понимается схема, содержащая большое число однотипных элементов,расположение которых напоминает матрицу, т. е. сетку или таблицу, имеющую строки и столбцы. Каждый элемент такой схемы является как бы ячейкой данной матрицы-таблицы. Он определяется номерами строки и столбца. Современные технологии в области микроэлектроники позволяют в одной микросхеме разместить сотни тысяч ЗЭ, т.е. в полупроводниковом ЗУ можно хранить сотни тысяч бит информации.

Организация ЗМ определяется способом выборки (опроса) ЗЭ при чтении и записи. Различают ЗУ с однокоординатной (пословной) и двухкоординатной выборками.

В матрице с однокоординатной выборкой одна строка образует слово из т разрядов. На схеме (рис. 12.2) символами А0, А1….An-1 обозначены адресные шины, а символами Р0, Р1, ..., Pm-1 — разрядные. Адресные шины связаны с каждым ЗЭ одного слова, в то время как разрядные шины — с ЗЭ одноименного разряда всех слов. Для записи слова по выбранному адресу Ai на разрядные шины Р0 — Pm-1 подается комбинация электрических сигналов 1 и 0. Эти сигналы попадают на каждый из запоминающих элементов i-й строки $ЗЭ_{i0}$, $ЗЭ_{i1}$, ..., $ЗЭ_{i(m-1)}$. При наличии в адресной шине сигнала выборки i-го слова состояние каждого из ЗЭ в этом слове может быть определено по сигналам на выходных разрядных шинах, которые подключены к разрядным усилителям считывания.

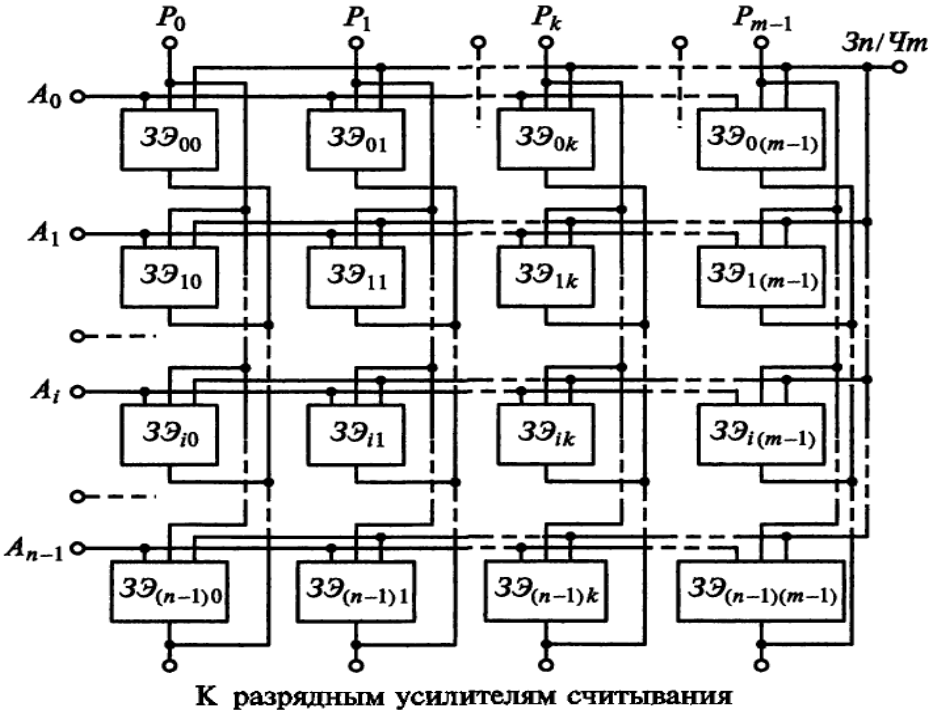


Рис. 12.2. Структурная схема матрицы ЗУ с однокоординатной выборкой

Выбор той или иной адресной шины производится дешифратором адреса (на рис. 12.2 не показан), на вход которого поступает двоичный код номера Af — номера ячейки, в которую должно быть записано или из которой должно быть считано слово. Эта ячейка имеет только одну координату — номер строки матрицы запоминающих элементов. Микросхема ОЗУ (рис. 12.3) имеет 16 запоминающих ячеек, каждая из которых может выбираться комбинацией двоичных переменных на четырех адресных входах А0 — А3 ($2^4 = 16$). Ячейка имеет четыре разряда (бита). Запись в нее осуществляется через входы D0 —D3 при наличии на входе запись — чтение (W/R) логической 1 (W/R = 1), а считывание производится с выходов Q0 — Q3 при W/R = 0. Инициализация (ввод в действие, выбор) микросхемы осуществляется логической 1, подаваемой на вход CS. При CS = 0 микросхема блокируется.

Адресная, ассоциативная и стековая организации памяти

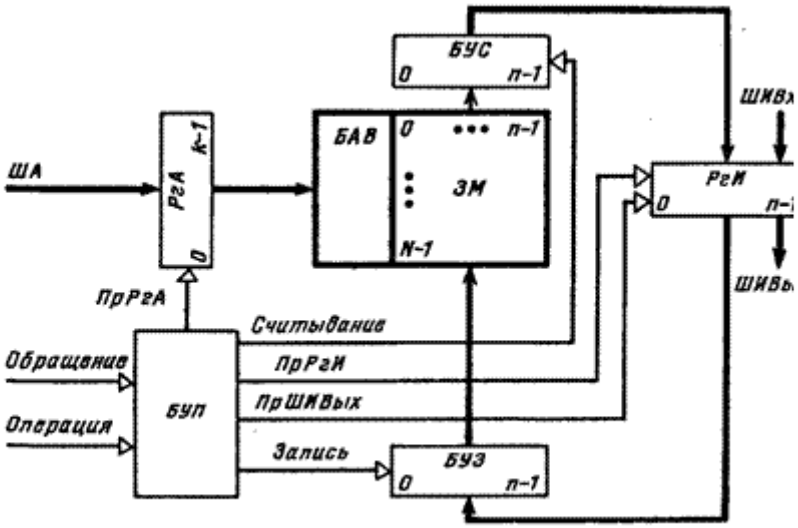
Запоминающее устройство с произвольным обращением, как правило, содержит множество одинаковых запоминающих элементов, образующих запоминающий массив (ЗМ). Массив разделен на отдельные ячейки; каждая из них предназначена для хранения двоичного кода, число разрядов в котором определяется шириной выборки памяти (в частности, это может быть одно, половина или несколько машинных слов). Способ организации памяти зависит от методов размещения и поиска информации в запоминающем массиве. По этому признаку различают адресную, ассоциативную и стековую (магазинную) памяти.

Адресная память. В памяти с адресной организацией размещение и поиск информации в ЗМ основаны на использовании адреса хранения слова (числа, команды и т. п.). Адресом служит номер ячейки ЗМ, в которой это слово размещается.

При записи (или считывании) слова в ЗМ инициирующая эту операцию команда должна указывать адрес (номер ячейки), по которому производится запись (считывание).

Типичная структура адресной памяти, содержит запоминающий массив из N-разрядных ячеек и его аппаратное обрамление, включающее в себя регистр адреса РгА, имеющий k (k » log N) разрядов, информационный регистр РгИ, блок адресной выборки БАВ, блок усилителей считывания БУС, блок разрядных усилителей-формирователей сигналов записи БУЗ и блок управления памятью БУП.

По коду адреса в РгА БАВ формирует в соответствующей ячейке памяти сигналы, позволяющие произвести в ячейке считывание или запись слова.



Цикл обращения к памяти инициируется поступлением в БУП извне сигнала Обращение. Общая часть цикла обращения включает в себя прием в РгА с шины адреса ША адреса обращения и прием в БУП и расшифровку управляющего сигнала Операция, указывающего вид запрашиваемой операции (считывание или запись).

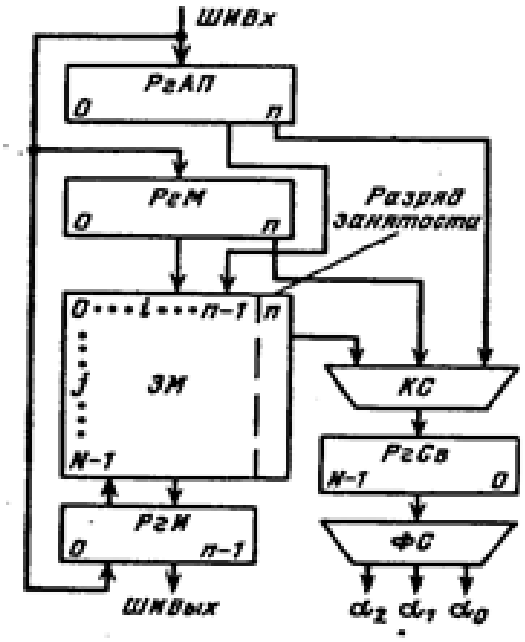
Далее при считывании БАВ дешифрирует адрес, посылает сигналы считывания в заданную адресом ячейку ЗМ, при этом код записанного в ячейке слова считывается усилителями считывания БУС и передается в РгИ. Операция считывания завершается выдачей слова из РгИ на выходную информационную шину ШИВых.

При записи помимо выполнения указанной выше общей части цикла обращения производится прием записываемого слова с входной информационной шины ШИВх и РгИ. Затем в выбранную БАВ ячейку записывается слово из РгИ.

Блок управления БУП генерирует необходимые последовательности управляющих сигналов, инициирующих работу отдельных узлов памяти.

Ассоциативная память. В памяти этого типа поиск нужной информации производится не по адресу, а по ее содержанию (по ассоциативному признаку). При этом поиск по ассоциативному признаку (или последовательно по отдельным разрядам этого признака) происходит параллельно во времени для всех ячеек запоминающего массива. Во многих случаях ассоциативный поиск позволяет существенно упростить и ускорить обработку данных. Это достигается за счет того, что в памяти этого типа операция считывания информации совмещена с выполнением ряда логических операций.

Типичная структура ассоциативной памяти представлена на рис. 4.3. Запоминающий массив содержит N (n+1)-разрядных ячеек. Для указания занятости ячейки используется служебный n-й разряд (0 — ячейка свободна, 1 — в ячейке записано слово).



По входной информационной шине ШИВх в регистр ассоциативного признака РгАП в разряды 0..n-1 поступает n-разрядный ассоциативный запрос, а в регистр маски РгМ — код маски поиска, при этом n-разряд РгМ устанавливается в 0. Ассоциативный поиск производится лишь для совокупности разрядов РгАП, которым соответствуют 1 в РгМ (незамаскированные разряды РгАП). Для слов, в которых цифры в разрядах совпали с незамаскированными разрядами РгАП, комбинационная схема КС устанавливает в 1 соответствующие разряды регистра совпадения РгСв и 0 в остальные разряды. Таким образом, значение j-го разряда в РгСв определяется выражением

$$РгСв(j)=\bigwedge_{i=0}^{n-1}\left\{\overline{РгАП[i]\oplus ЗМ[j,i]}\vee\overline{РгМ[i]}\right\}\quad(0\le j\le N-1)$$

где *РгАП*[*i*], *РгМ*[*i*] и *ЗМ*[*j, i*] — значения *i*-го разряда соответственно *РгАП, РгМ* и *j*-и ячейки *ЗМ*.

Комбинационная схема формирования результата ассоциативного обращения *ФС* формирует из слова, образовавшегося в *РгСв*, сигналы a_0, a_1, a_2, соответствующие случаям отсутствия слов в *ЗМ*, удовлетворяющих ассоциативному признаку, и наличия одного (и более) такого слова.

Формирование содержимого РгСв и сигналов a0, a1, a2 по содержимому РгАП, РгМ и ЗМ называется операцией контроля ассоциации. Эта операция

является составной частью операций считывания и записи, хотя она имеет и самостоятельное значение.

При считывании сначала производится контроль ассоциации по ассоциативному признаку в РгАП. Затем при а0 = 1 считывание отменяется из-за отсутствия искомой информации, при а1 = 1 считывается в РгИ найденное слово, при а2 = 1 в РгИ считывается слово из ячейки, имеющей наименьший номер среди ячеек, отмеченных 1 у РгСв. Из РгИ считанное слово выдается на ШИВых.

При записи сначала отыскивается свободная ячейка. Для этого выполняется операция контроля ассоциации при РгАП = 111...10 и РгМ = 00...01, при этом свободные ячейки отмечаются 1 в РгСв. Для записи выбирается свободная ячейка с наименьшим номером. В нее записывается слово, поступившее с ШИВх в РгИ.

С помощью операции контроля ассоциации можно, не считывая слов из памяти, определить по содержимому РгСв, сколько в памяти слов, удовлетворяющих ассоциативному признаку, например реализовать запросы типа сколько студентов в группе имеют отличную оценку по данной дисциплине. При использовании соответствующих комбинационных схем в ассоциативной памяти могут выполняться достаточно сложные логические операции, такие, как поиск большего (меньшего) числа, поиск слов, заключенных в определенных границах, поиск максимального (минимального) числа и др. Ассоциативная память применяется, например, в аппаратуре динамического распределения ОП.

Отметим, что для ассоциативной памяти необходимы запоминающие элементы, допускающие считывание без разрушения записанной в них информации. Это связано с тем, что при ассоциативном поиске считывание производится по всему ЗМ для всех незамаскированных разрядов и негде сохранять временно разрушаемую считыванием информацию.

Стековая память, так же как и ассоциативная, является безадресной. Стековую память можно рассматривать как совокупность ячеек, образующих одномерный массив, в котором соседние ячейки связаны друг с другом разрядными цепями передачи слов. Запись нового слова производится в верхнюю ячейку (ячейку 0), при этом все ранее записанные слова (включая слово, находившееся в ячейке 0), сдвигаются вниз, в соседние ячейки с большими на 1 номерами. Считывание возможно только из верхней (нулевой) ячейки памяти, при этом, если производится считывание с удалением, все остальные слова в памяти сдвигаются вверх, в соседние ячейки с большими номерами. В этой памяти порядок считывания слов соответствует правилу:

последним поступил — первым обслуживается. В ряде устройств рассматриваемого типа предусматривается также операция простого считывания слова из нулевой ячейки (без его удаления и сдвига слова в памяти). Иногда стековая память снабжается счетчиком стека СчСт, показывающим количество занесенных в память слов. Сигнал СчСт = 0 соответствует пустому стеку, СчСт = N - 1 — заполненному стеку.

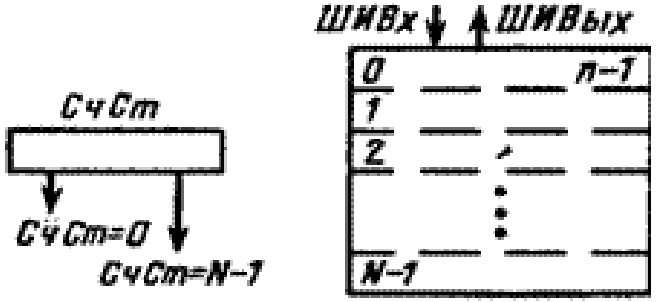


Рис. 2.3. Стековая память

Обычно стековую память организуют, используя адресную память. В этом случае счетчик стека, как правило, отсутствует, так как количество слов в памяти можно выявить по указателю стека. Широкое применение стековая память находит при обработке вложенных структур данных, при выполнении безадресных команд и прерываний.

Лекция 12. Кэш-память: назначение, структура, основные характеристики. Организация кэш-памяти: с прямым отображением, частично-ассоциативная и полностью ассоциативная.

Основная задача кэш-памяти – согласование работы быстрого процессора и медленной основной памяти. Кэш-память исполняет роль буфера между ОП и процессором (рис. 9.1). Использование кэш-памяти базируется на «принципе локальности ссылок».

Это означает, что следующее обращение к памяти в программе с большой вероятностью произойдет к тому же блоку данных, который находится в данный момент в кэш-памяти.

Кэш-память разбивается на строки по 32 байта, соответствующие одному стандартному пакетному циклу обращения к динамической памяти. На такие же строки условно разделяются и страницы основной памяти. Обмен информацией между ними осуществляется строками, даже если необходимо передать только один байт.

Процессор, выполняя команду, запрашивает операнд по некоторому адресу в адресном пространстве. Кэш-контроллер проверяет, есть ли в кэш-памяти строка данных, соответствующая запрашиваемому адресу. В случае наличия такой строки ситуация называется кэш-попадание. Если строки в кэш-памяти нет, то происходит кэш-промах, и кэш-контроллер инициирует обращение к основной памяти ОП для переписи из нее нужной строки в кэш-память.

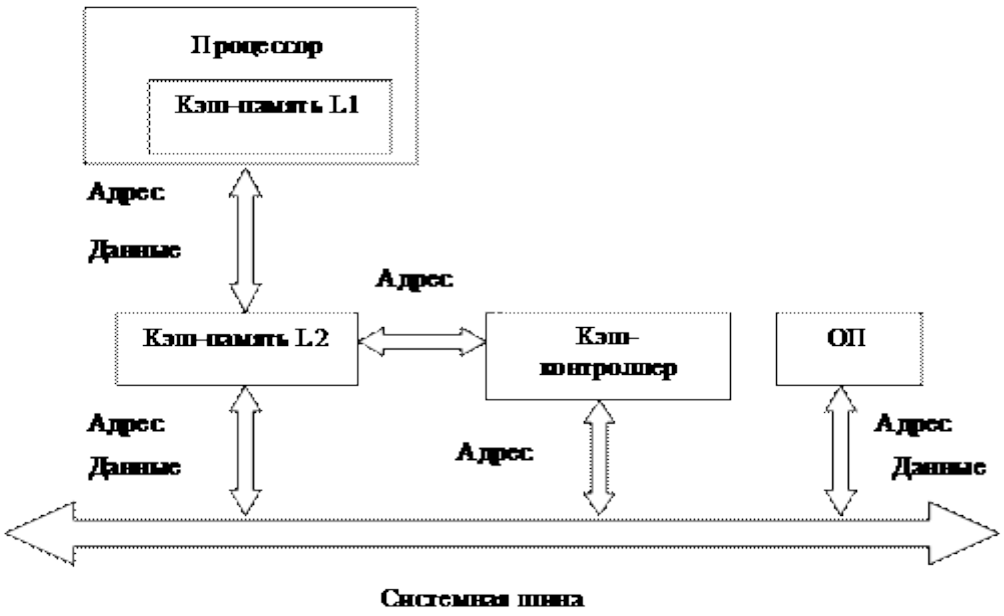


Рис. 9.1. Двухуровневая кэш-память в компьютере

В связи с этим возникает проблема замены какой-либо строки в кэше на новую строку из ОП. Для этого используют специальные дисциплины замещения строк. Эти функции выполняет кэш-контроллер.

В современных вычислительных системах используются три типа организации кэш-памяти:

- с прямым отображением;
- полностью ассоциативная;
- множественно-ассоциативная.

Кэш-память с прямым отображением. Каждая строка кэш-памяти может содержать строку основной памяти только из определенного подмножества адресов, причем эти подмножества не пересекаются. Поиск состоит из следующих шагов:

- определение, в какое из подмножеств адресов основной памяти попадает адрес строки, выработанный процессором;

-обращение к единственной соответствующей строке и сравнение ее тега с адресом от центрального процессора для определения, является ли эта строка искомой.

На рис. 9.2 приведен пример структуры кэш-памяти с прямым отображением. Для простоты дана структура ОП, содержащей 16 строк данных, и кэш-память объемом в четыре строки.

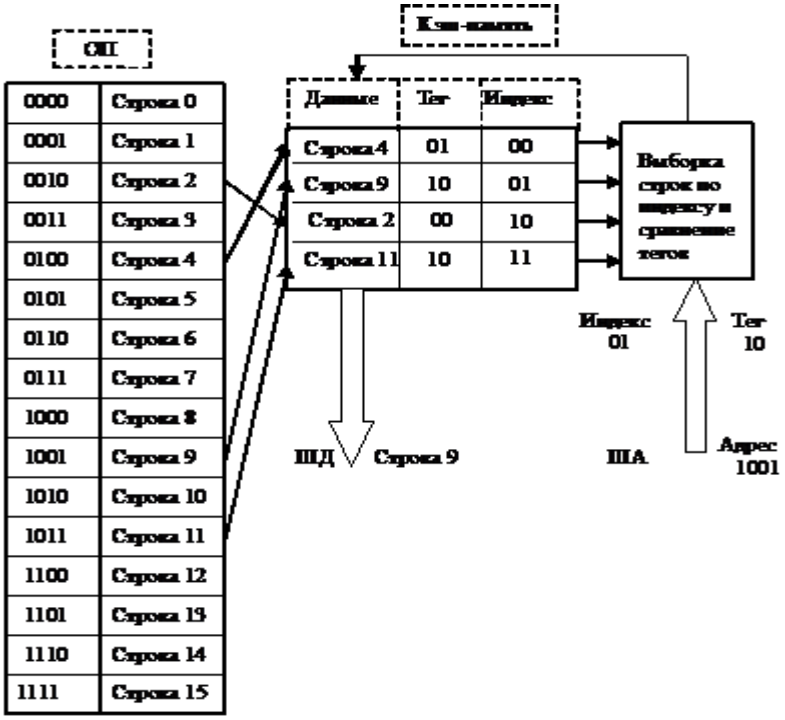


Рис.9.2. Кэш-память с прямым отображением

Все строки основной памяти, имеющие S одинаковых младших разрядов , объединяются в подмножества, которые могут отображаться в кэше в строке с индексом, равным коду этих разрядов. В нашем примере индекс образуют два младших разряда адреса ОП. Следовательно, например, строки 1, 5, 9 и 13 могут находиться только в строке кэша с индексом 01 и ни в какой другой. В общем случае если разрядность адреса ОП равна N, а разрядность индекса – n, то адресные теги содержат оставшиеся N-n разрядов адреса строки.

Преимущество описываемой кэш-памяти состоит в простоте организации и низкой стоимости. Основной недостаток – ограниченное число комбинаций строк в кэше, что приводит к увеличению процента кэш-промахов. Например, строки 5 и 9 не могут одновременно находиться в кэш-памяти, даже если есть свободные места для других индексов.

Полностью ассоциативная кэш-память. В такой памяти любая строка ОП может находиться в любом месте кэш-памяти, причем в любой комбинации с другими строками. Комбинационные схемы сравнения СС1-СС4 (рис. 9.3) одновременно анализируют все теги строк, находящихся в кэше в данный момент, и сравнивают их с адресом, поступившим с шины адреса от процессора.

При кэш-попадании строка считывается в шину данных (ШД). При кэш-промахе происходит замещение строки в кэш-памяти на требуемую строку из ОП.

При кэш-попадании строка считывается в шину данных (ШД). При кэш-промахе происходит замещение строки в кэш-памяти на требуемую строку из ОП.

Преимущество данной памяти состоит в высокой скорости считывания. Недостаток – в сложности аппаратной реализации.

Множественно-ассоциативная кэш-память. Этот вид памяти является промежуточным между двумя рассмотренными выше. В нем сочетаются простота кэша с прямым отображением и скорость ассоциативного поиска.

Кэш-память делится на непересекающиеся подмножества (блоки) строк. Каждая строка основной памяти может попадать только в одно подмножество кэша. Для поиска блоков используется прямое отображение, а для поиска внутри подмножества – полностью ассоциативный поиск. Число строк в подмножестве кэша определяет число входов (портов) самого кэша.

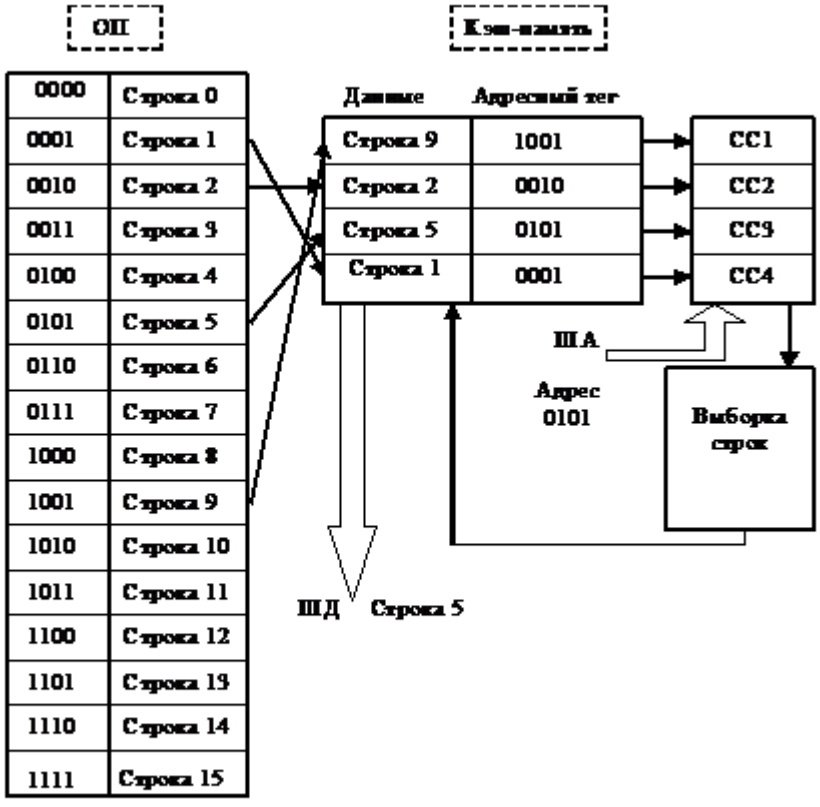
Если 2^n строк кэша разбивается на 2^s непересекающихся подмножеств, то S младших разрядов оперативной памяти показывают, в каком из подмножеств

(индексов) должен вестись ассоциативный поиск. Старшие n-s разрядов адреса основной памяти являются тегами.

Если S=0, то получим одно подмножество, что соответствует полностью ассоциативной кэш-памяти. Если S=n, то получим 2n подмножеств (то есть одна строка – одно подмножество). Это кэш-память с прямым отображением. Если 1£ S £ n-1, то имеем множественно-ассоциативную кэш-память.

На рис. 9.4 приведен пример кэша, где S=1, то есть имеются два подмножества кэш-памяти. Это двухвходовая множественно-ассоциативная кэш-память.

Физический адрес 1011, выработанный процессором, разделяется на индекс 1, равный младшему разряду, и тег 101. По индексу выбирается второе подмножество строк в кэш-памяти, а затем происходит ассоциативный поиск среди тегов строк выбранного подмножества. Найденная строка 11 с тегом 101 передается в шину данных (ШД). Ассоциативный поиск производится одновременно по всем тегам с помощью комбинационных схем сравнения СС1 и СС2.



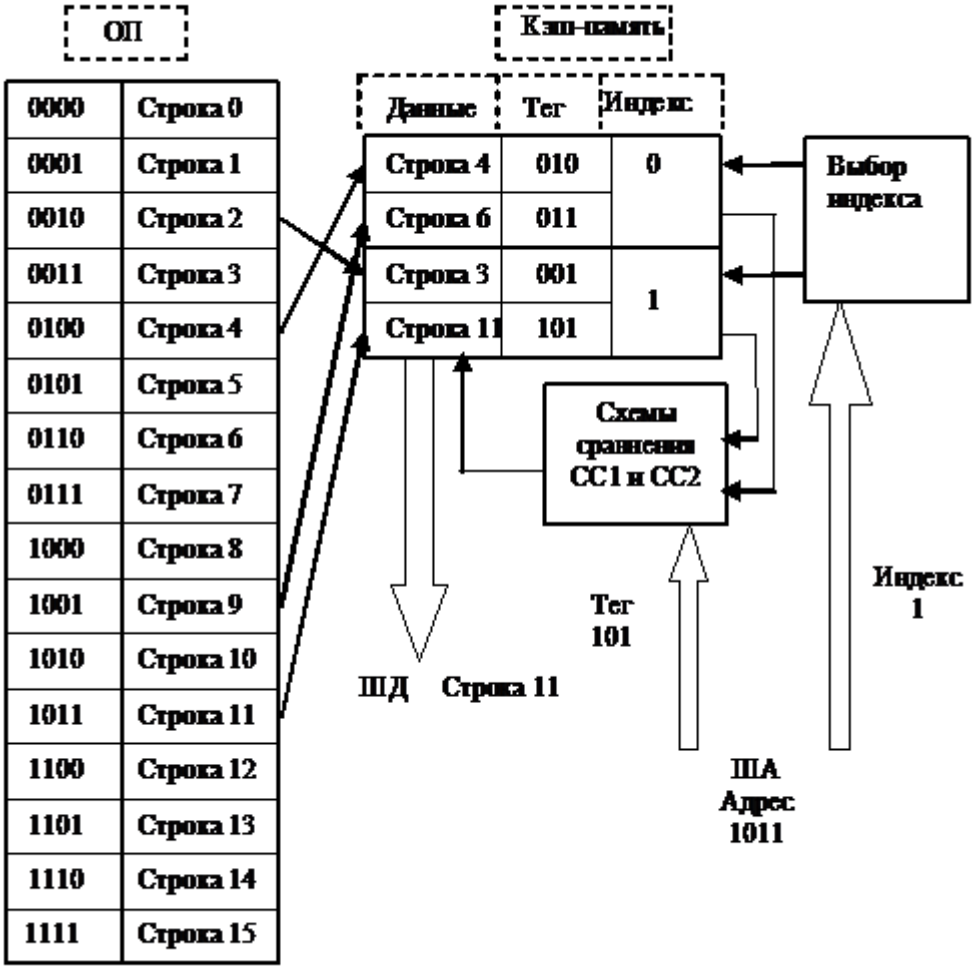
В современных процессорах используется 4-х и 8-ми входовая кэш-память. Увеличение числа ее входов приводит к быстрому увеличению

сложности аппаратной реализации той части кэша, которая обеспечивает ассоциативный поиск тегов.

Особенности записи и замещения информации в кэш-памяти. Когерентность кэш-памяти

Обращение по чтению можно начинать сразу и к кэш, и к оперативной памяти. Тогда, если информация отсутствует в кэше, к моменту установления этого факта будет уже выполнена часть цикла обращения к ОЗУ, что может повысить производительность. Если информация имеется в кэше, то обращение к оперативной памяти можно остановить.

При обращении по записи используется два метода: запись производится только в кэш или сразу и в кэш, и в ОЗУ. Эти методы получили название алгоритмов ***обратной*** WB (Write Back) и ***сквозной записи*** WT (Write Through) соответственно. Второй из них более простой, но и более медленный, хотя и гарантирует, что копии одной и той же информации в кэше и оперативной памяти всегда совпадают. Большинство ранних процессоров Intel используют именно этот алгоритм.



Алгоритм обратной записи WB более быстрый. Передача информации в ОЗУ производится только тогда, когда на место данной строки кэша передается строка из другой страницы ОП, или при выполнении команды обновления содержимого кэша. Этот алгоритм требует более аккуратного управления, поскольку существуют моменты, когда копии одной и той же информации различны в кэше и ОП. Кроме того, не каждая строка изменяется за время своего пребывания в кэше. Если изменения не было, то нет необходимости переписывать строку обратно в оперативную память. Обычно используют флаг M (modified – изменена) в памяти тэгов. Он сбрасывается в «0» при первоначальной загрузке строки в кэш и устанавливается в «1» при записи в нее информации. При выгрузке строки из кэша запись в ОП выполняется только при единичном значении флага M.

При возникновении промаха контроллер кэш-памяти должен выбрать подлежащую замещению строку. Для прямого отображения аппаратные решения наиболее простые. На попадание проверяется только одна строка, и только эта строка может быть замещена. При полностью ассоциативной или множественно-ассоциативной организации кэш-памяти имеются несколько строк, из которых надо выбрать кандидата в случае промаха. Для решения этой задачи используют следующие специальные правила, называемые алгоритмами замещения:

- FIFO (First In First Out – первый пришедший – первым выбывает);
- LRU (Least Recently Used – дольше других неиспользуемый);
- LFU (Least Frequently Used – реже других используемый);
- случайный (random).

Первый и последний методы являются самыми простыми в реализации, но они не учитывают, насколько часто используется та или иная строка кэш-памяти. При этом может быть удалена строка, к которой в самом ближайшем будущем будет обращение. Вероятность ошибки для указанных методов гораздо выше, чем у второго и третьего.

В алгоритме FIFO для замещения выбирается строка, первой попавшая в кэш. Каждая вновь размещаемая в кэше строка добавляется в хвост этой очереди. Алгоритм не учитывает фактическое ее использование. Например, первые загруженные строки могут содержать данные, требующиеся на протяжении всей работы. Это приводит к немедленному возвращению к только что замещенной строке.

Алгоритм LRU предусматривает, что для удаления следует выбирать ту строку, которая не использовалась дольше других. При каждом обращении к строке ее временная метка обновляется. Это может быть сопряжено с

существенными издержками. Однако алгоритм LRU наиболее часто используется на практике. Недостаток его заключается в том, что если программа проходит большой цикл, охватывающий множество строк, может случиться так, что строка, к которой дольше всего не было обращений, в действительности станет следующей используемой.

Одним из близких к LRU является алгоритм LFU, согласно которому удаляется наименее часто использовавшаяся строка. При этом необходимо подсчитывать количество обращений к каждой строке и контролировать его. Может оказаться, что наименее интенсивно используется та строка, которая только что записана в кэш-память и к которой успели обратиться только один раз (в то время как к другим строкам обращались больше). Она может быть удалена, что является недостатком алгоритма LFU.

Содержимое кэш-памяти меняется под управлением процессора. При этом основная память может оставаться неизменной. С другой стороны, внешние устройства могут изменять данные в ОП в режиме прямого доступа. При этом кэш-память не меняет своих данных. Еще сложнее ситуация в мультипроцессорных системах, когда несколько процессоров обращаются к общей памяти. Возникает проблема когерентности кэш-памяти.

Вычислительная система имеет когерентную память, если каждая операция чтения по адресу, выполненная каким-либо устройством, возвращает значение последней копии по этому адресу независимо от того, какое из них производило запись последним. Проблема когерентности является наиболее важной для систем с обратным копированием

Лекция 13. Динамическая память. Принцип работы.

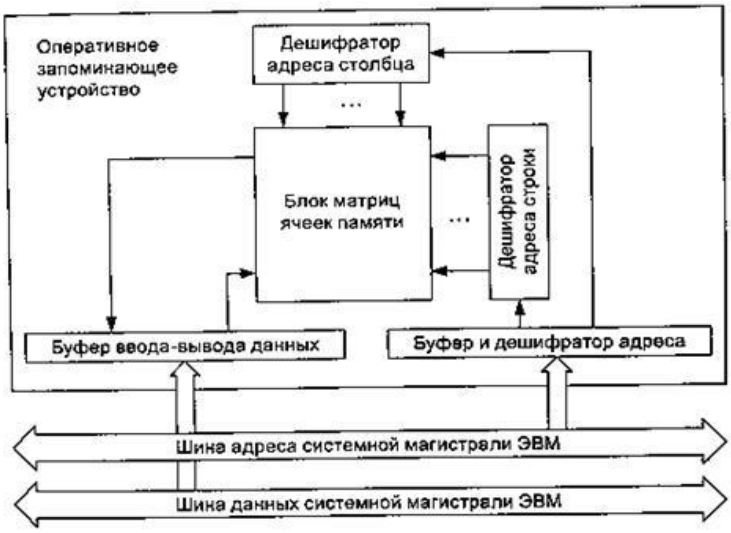
Динамическая оперативная память (DRAM – Dynamic Random Access Memory) – энергозависимая полупроводниковая память с произвольным доступом.

(На данный момент – это основной тип оперативной памяти, используемый в современных персональных компьютерах и обеспечивающий наилучший показатель отношения цена-качество по сравнению с другими типами оперативной памяти. Однако, требования к быстродействию, энергопотреблению и надежности оперативной памяти постоянно растут, и динамическая оперативная память уже с трудом соответствует современным потребностям, создаются конкурирующие типы оперативной памяти, таких как магниторезистивная оперативная память).

Память, каждая ячейка которой состоит из одного конденсатора и нескольких транзисторов. Конденсатор хранит один бит данных, а транзисторы играют роль ключей, удерживающих заряд в конденсаторе и разрешающих доступ к конденсатору при чтении и записи данных.

Однако транзисторы и конденсатор – неидеальные, и на практике заряд с конденсатора достаточно быстро истекает. Поэтому периодически, несколько десятков раз в секунду, приходится дозаряжать конденсатор. При чтении конденсатор разряжается, и необходимо его заново подзаряжать, чтобы не потерять навсегда данные, хранящиеся в ячейке памяти.

Обобщенная структурная схема ОЗУ типа DRAM:



Основным компонентом микросхемы памяти является массив элементов памяти (матрицы строк и столбцов), объеденных в блоки. Каждый элемент памяти может хранить один бит информации и имеет адрес (полный адрес, состоящий из адреса строки и адреса столбца), по которому процессор может обращаться в произвольном порядке.

Считывание информации в микропроцессорную память из модуля оперативной памяти выполняется по команде процессора, передаваемой через шину управления системной магистрали ЭВМ. По шине адреса передается адрес требуемого процессору машинного слово. В дешифраторе адреса принятый адрес расшифровывается, из него выделяют адрес столбца и адрес строки считываемого слова. Результат считывания (данные) из заданной ячейки передается в буфер данных и через шину данных возвращается в МП.

Процесс записи в ячейку памяти выполняется в обратном порядке: МП выставляет на шину данных блок данных, на шину адреса – адрес ячейки ЗУ, по шине управления - команду перемещения выставленного блока данных по указанному адресу. Получив команду, устройство управления ЗУ считывает адрес с шины адреса дешифрует его, считывает блок данных с шины данных и выполняет операцию записи в требуемую ячейку памяти.

Работа DRAM

Как видно из рисунка, основным блоком памяти является матрица памяти, состоящая из множества ячеек, каждая из которых хранит 1 бит информации.

Каждая ячейка состоит из одного конденсатора (С) и трех транзисторов.

Транзистор VT1 разрешает или запрещает запись новых данных или регенерацию ячейки.

Транзистор VT3 выполняет роль ключа, удерживающего конденсатор от разряда и разрешающего или запрещающего чтение данных из ячейки памяти.

Транзистор VT2 используется для считывания данных с конденсатора. Если на конденсаторе есть заряд, то транзистор VT2 открыт, и ток пойдет по линии AB, соответственно, на выходе Q1 тока не будет, что означает – ячейка хранит бит информации с нулевым значением. Если заряда на конденсаторе нет, то конденсатор VT2 закрыт, а ток пойдет по линии AE, соответственно, на выходе Q1 ток будет, что означает – ячейка хранит бит информации со значением “единица”.

Заряд в конденсаторе, используемый для поддержания транзистора VT2 в открытом состоянии, во время прохождения по нему тока, быстро расходуется, поэтому при чтении данных из ячейки необходимо проводить регенерацию заряда конденсатора.

Для работы динамической памяти на матрицу должно всегда поступать напряжение, на схеме оно обозначено, как Uп. С помощью резисторов R напряжение питания Uп равномерно распределяется между всеми столбцами матрицы.

Также в состав памяти входит контроллер шины памяти, который получает команды, адрес и данные от внешних устройств и ретранслирует их во внутренние блоки памяти.

1.1. Работа динамической памяти в состоянии покоя.

DRAM – память энергозависимая, поэтому работа с ней возможна только при подаче питания. На схеме подаваемое на плату питание обозначено, как Uп. Подаваемое питание распределяется между всеми столбцами матрицы памяти с помощью резисторов R.

Если память бездействует (от контроллера шины памяти не приходит никаких команд), то от дешифратора адреса строки не выдается сигнал ни на одну линию строк (S1-Sn) матрицы памяти. Соответственно, транзисторы VT1 и VT3 ячейки памяти M11 закрыты.

Следовательно, ток от подаваемого питания проходит по линии AE. Далее попадает на выходы Q1-Qm, на которых устанавливается «высокий» уровень напряжения, соответствующий значению логической «1». Но так как никаких команд от блока управления нет, то «Буфер данных» игнорирует получаемые сигналы.

Тут становится понятно, зачем нужен транзистор VT3. Он защищает конденсатор от разряда, когда к данной ячейки памяти нет обращения.

1.2. Работа динамической памяти при чтении данных и регенерации.

Будем рассматривать принцип чтения данных из динамической памяти на примере считывания данных из ячейки памяти M11:

1. Процессор запрашивает порцию данных (размер зависит от разрядности процессора, для 32-разрядного процессора минимальной единицей обмена, обычно, являются 32 бита) и выдает их адрес.
2. Контроллер шины памяти преобразует адрес в номер строки и номер столбца и выдает номер строки в дешифратор адреса строки. Дешифратор адреса строки выдает сигнал в соответствующую строку матриц памяти. Мы договорились, что в примере данные будем читать из первой ячейки памяти. Поэтому дешифратор адреса строки подаст напряжение на первую строку (S1).
3. Напряжение, поданное на строку S1, откроет транзисторы VT1 и VT3 первой ячейки памяти и соответствующие транзисторы всех остальных ячеек первой строки.
4. Дальнейшая работа памяти зависит от наличия или отсутствия заряда на конденсаторе. Рассмотрим отдельно два случая, когда на конденсаторе ячейки M11 есть заряд, и когда нет.
4.1 . В начале рассмотрим случай, когда заряд в конденсаторе есть (ячейка памяти содержит бит со значением ноль):
Так как на конденсаторе C ячейки памяти M11 есть заряд, то транзистор VT2 будет открыт, а, соответственно, ток, создаваемый входным напряжением Uп,пойдет по линии AB. В результате, на выходе Q1 первого столбца тока не будет. А это означает, что с ячейки памяти M11 считан ноль. Соответствующая информация о считанном бите с первого столбца будет записана в «Буфер данных». Для поддержания транзистора VT2 в открытом состоянии и протекания тока по линии AB расходуется заряд конденсатора C. В результате, конденсатор очень быстро разрядится, если не провести его регенерацию. Так как на выходе Q1 тока нет, то он не будет поступать и в «Блок регенерации 1», а, соответственно, на нижнем входе элемента L3 (логическое «И») будет логический ноль.
Так как мы рассматриваем случай чтения данных, то сигнал записи V1 и данные для записи D1 в «Блок регенерации 1» подаваться не будут. В остальные блоки регенерации соответствующие сигналы D1-Dm и V1-Vm также подаваться не будут. В результате, на входе элемента L1 (логическое «НЕ») будет логический «0», а на выходе – логическая «1», поэтому на входах элемента L3 (логическое «И») будет логический «0» и логическая «1». Это значит, что на выходе этого элемента будет логический «0».
На выходе логического элемента L2 (логическое «И») будет логический ноль, так как на обоих его входах напряжение отсутствует, так как от контроллера шины памяти отсутствуют команды на запись и данные для записи.
Имея на обоих входах элемента L4 (логическое «ИЛИ-НЕ») логический «0», на его выходе будем иметь логическую «1», то есть с блока регенерации пойдет

ток подзарядки конденсатора С. Так как транзистор подзарядки VT1 ячейки памяти М11 открыт, то ток подзарядки беспрепятственно пройдет в конденсатор С. Остальные ячейки памяти первого столбца имеют закрытый конденсатор подзарядки, а, следовательно, подзарядка их конденсаторов происходить не будет.

5. Параллельно с чтением и регенерацией данных первого столбца происходит по такому же алгоритму чтение данных с остальных столбцов. В результате, в буфер данных будет записано значение всех ячеек памяти первой строки.

6. С контроллера памяти на дешифратор адреса столбца выдаются номера столбцов для считывания

7. С дешифратора адреса столбцов номера столбцов передаются в «Буфер данных», откуда соответствующие данные считываются и передаются в процессор.

1.3. Работа динамической памяти при записи данных.

Будем рассматривать принцип записи данных в динамическую память на примере записи данных в ячейку памяти М11:

1. Контроллер шины памяти получает команду на запись данных, данные и адрес, куда необходимо записать эти данные.

2. Контроллер шины памяти преобразует адрес на две составляющие – номер строки и номера столбцов, и передает полученные составляющие в «Дешифратор адреса строки» и в «Дешифратор адреса столбцов». А данные передает в «Блок работы с данными».

3. Дешифратор адреса строки выдает сигнал в соответствующую строку матрицы памяти. Мы договорились, что в примере данные будем записывать в первую ячейку памяти. Поэтому дешифратор адреса строки подаст напряжение на первую строку (S1).

4. Одновременно с «Дешифратора адреса столбцов» выдаются сигналы V в столбцы, соответствующие полученному адресу. В эти же столбцы подаются сигналы D с «Блока работы с данными», уровень которых определяется значением битов записываемого слова.

5. Напряжение, поданное на строку S1, откроет конденсаторы VT1 и VT3 первой ячейки памяти и соответствующие конденсаторы всех остальных ячеек первой строки.

6. Если в ячейке М11 хранится бит со значением «0» (в конденсаторе есть заряд), то ток, создаваемый входным напряжением Uп, пойдет по линии АВ, иначе – по линии АЕ. Но нам это не важно, так как в ячейку М11 производится запись данных, а не их чтение, поэтому буфер данных будет игнорировать считанное с ячейки значение. А с выхода элемента L3 «Блока регенерации 1»

будет всегда идти логический ноль, так как с дешифратора столбцов приходит сигнал (V1) на запись данных в первый столбец.

В результате, на выходе будет логический «0», то есть ток будет отсутствовать, а, соответственно, зарядка конденсатора С идти не будет. Если до этого конденсатор С содержал заряд, то через несколько микросекунд он разрядится, пропуская ток по линии АВ. Таким образом в конденсатор С будет записан бит данных «1», соответствующий разряженному состоянию конденсатора. Если бит имеет значение «0», то на верхнем входе элемента L2 будет «0». Имея на верхнем входе логический ноль, а на нижнем – логическую единицу, на выходе элемента L2 получим логический ноль. В результате, на верхнем и нижнем входах элемента L4 имеем логические нули, что означает – на выходе элемента L4 будет логическая единица, то есть пойдет ток зарядки конденсатора. Таким образом в конденсатор С будет записан бит данных «0», соответствующий заряженному состоянию конденсатора.

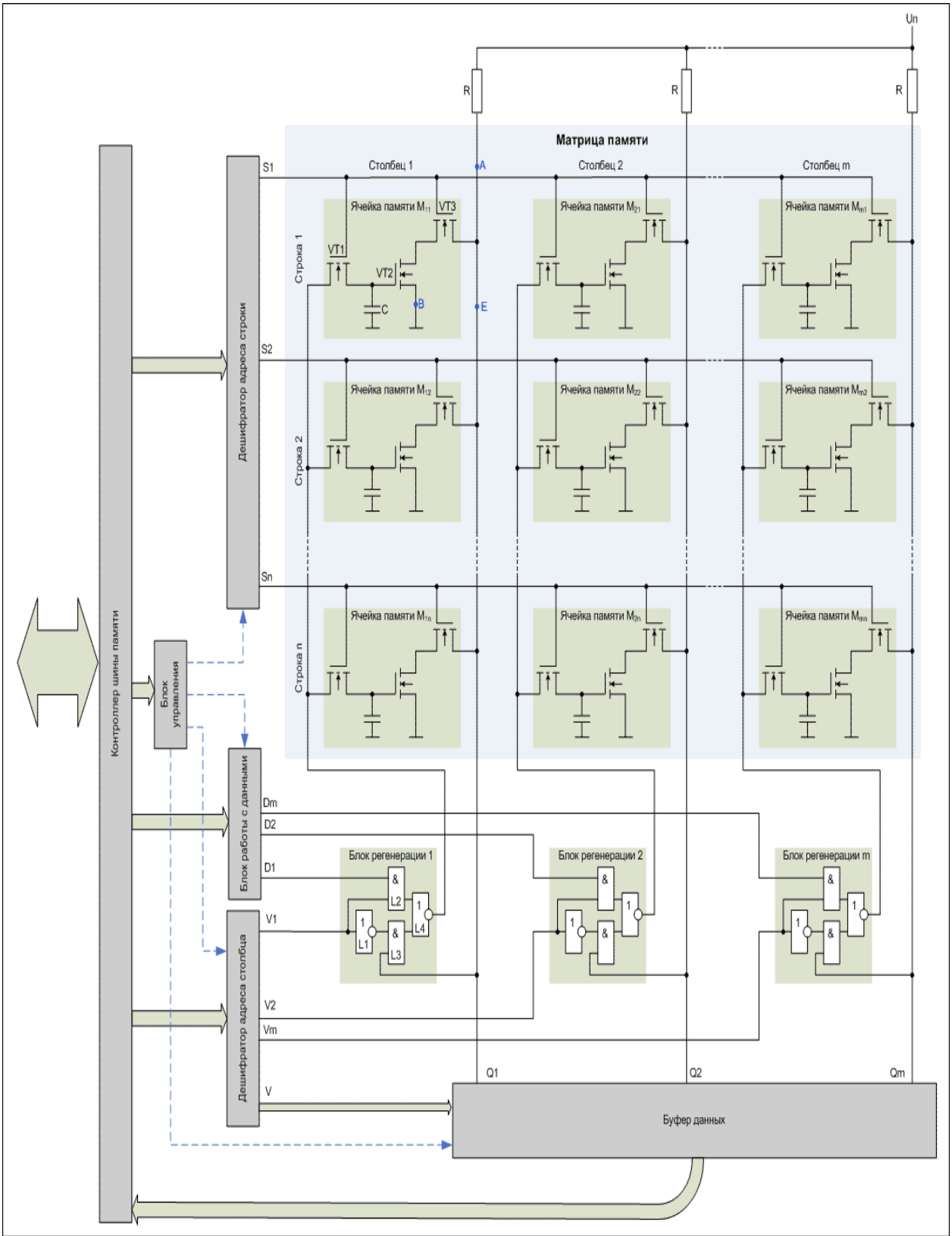


Рисунок 1. Упрощенная структурная схема динамической оперативной памяти

Модификации динамической оперативной памяти.

Все усовершенствования в работе динамической памяти были направлены на увеличение скорости работы памяти, так как скорость оперативной памяти всю историю существования вычислительной техники являлась одним из факторов, сдерживающих рост производительности ЭВМ.

✓ **PM DRAM.**

Один из первых видов оперативной памяти, используемой в персональных компьютерах, была простая динамическая оперативная память (PM DRAM – Page Mode DRAM). PM DRAM использовалась вплоть до середины 90-х годов. Недостаток: нехватка быстродействия.

✓ **FPM DRAM.**

FPM DRAM (Fast Page Mode DRAM) – быстрая страничная память. Основное ее отличие заключалось в поддержке сохраненных адресов. То есть, если новое считываемое из памяти слово находилось в той же строке, что и предыдущее слово, то обращение к матрице памяти не требовалось, а выборка данных осуществлялась из «Буфера данных» по номерам столбцов. Это позволяло в случае чтения из памяти массивов данных значительно сократить время чтения.

Недостатки : В результате, прирост производительности сильно зависел от типа программ, с которыми работала ЭВМ.

✓ **EDO-DRAM.**

EDO-DRAM (Extended Data Out DRAM) –В этом типе памяти адрес следующего считываемого слова передавался до завершения считывания линии данных памяти, то есть до того, как считанные данные из памяти были переданы процессору.

Приступить к считыванию нового слова данных, до завершения чтения предыдущего, стало возможным, благодаря вводу, так называемых, регистров – защелок, которые сохраняли последнее считанное слово даже после того, как начиналось чтение или запись следующего слова.

Достоинство: прирост производительности , достигавший 15-20%.

Недостаток: с ростом тактовой частоты работы процессора, шин и памяти стабильность работы упала.

✓ **SDRAM.**

SDRAM (Synchronous DRAM) – синхронная динамическая память с произвольным доступом. Память работает синхронно с контроллером памяти, что гарантировало завершение цикла чтения/записи строк в заданное время. Это позволяло выдавать новую команду на чтение до завершения считывания

предыдущего слова данных, будучи уверенным, что считывание завершится верно, а чтение нового слова начнется с минимальной задержкой.

Недостаток: считывание неверных данных, поэтому контроллер синхронной памяти дополнительно усложнился, обеспечивая защиту от таких ситуаций; внешняя шина памяти стала узким местом и замедляла работу.

✓ **DDR SDRAM.**

DDR SDRAM (Double Data Rate SDRAM) – синхронная динамическая память с произвольным доступом и удвоенной частотой передачи данных.

В этом типе оперативной памяти обмен данными по внешней шине идет не только по фронту тактового импульса, но и по спаду. В результате, без увеличения тактовой частоты внешней шины удваивается объем передаваемой информации.

✓ **DDR2 SDRAM.**

В памяти DDR2 SDRAM ширина внутренней шины данных была увеличена еще в два раза и стала превосходить внешнюю шину данных в четыре раза. В результате, при одной и той же тактовой частоте внешней шины памяти у памяти DDR2 SDRAM внутренняя тактовая частота была в два раза меньше, по сравнению с памятью DDR SDRAM.

Это давало огромный потенциал для увеличения производительности памяти и уменьшало энергопотребление.

✓ **DDR3 SDRAM**.

Основное направление развития памяти DDR3 SDRAM сохранилось таким же, как у DDR2 SDRAM.

Достоинства: улучшение технологического процесса и архитектуры, снижение времени цикла чтения/записи.

✓ **DDR4 SDRAM.**

Память DDR4 продолжит тенденции DDR памяти.

Достоинства: Будет увеличена ширина внутренней шины, улучшена технология производства до 32-36 нм, подняты тактовые частоты внешней и внутренней шины, а также будет снижено напряжение.

Лекция 14. Статическая память.

Статическая память (SRAM) – это энергозависимая полупроводниковая память с произвольным доступом, в которой каждый разряд хранится в триггере, позволяющем поддерживать состояние разряда без постоянной перезаписи. Для организации чтения и записи из ячейки памяти дополнительно используется три или более транзисторов.

Для того чтобы понять принцип работы статической памяти, обратимся к истокам схемотехники. И начнем с описания принципа работы триггера, изображенного на рисунке 1.

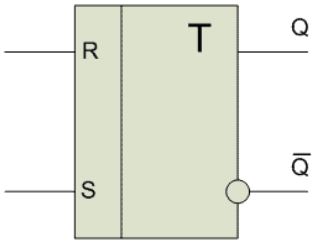


Рисунок 1. Триггер.

Триггер – это элемент памяти с двумя стабильными состояниями – «0» и «1». В установленном состоянии триггер сохраняется, пока на него подается питание.

Обычно триггер имеет два входа:

- R (Reset) – сбросить триггер (установить в состояние «0»),
- S (Set) – установить триггер в состояние «1»,

и два выхода: Q и инвертированное Q ($\overline{Q}$).

Входы R и S используются для установки состояния триггера. Если на вход S подать напряжение, соответствующее логической единице (далее просто логическую единицу), а на вход R – напряжение, соответствующее логическому нулю (далее просто логический ноль), то триггер перейдет в состояние единицы и сохранит это состояние даже, если на вход S перестать подавать сигнал.

Если на вход S подать логический ноль, а на вход R – логическую единицу, то триггер перейдет в состоянии сохранения нуля.

При подаче на оба входа логического нуля, состояние триггера не измениться.

При подаче на оба входа логической единицы, в общем случае состояние триггера будет неопределенно, то есть неизвестно, в какое состояние он перейдет.

На выходах Q и $\overline{Q}$ можно прочитать установленное состояние триггера.
Рассмотрев логику работы триггера, давайте разберемся, как же он устроен. Структурная схема триггера приведена на рисунке 2.

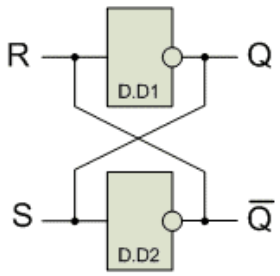


Рисунок 2. Структурная схема триггера.

(Как видно из рисунка, состоит он из двух инвертеров (логических элементов «НЕ»), причем выход одного инвертера замкнут на вход другого.
Давайте рассмотрим, как же работают эти инвертеры при подаче различных сигналов на вход).
Первый случай, на вход S подана логическая единица, а на вход R – логический ноль, то есть установка триггера в единичное состояние. И так, если на вход S подать логическую единицу, то, пройдя через инвертер D.D2, она примет значение логического нуля. Таким образом, на выходе $\overline{Q}$ будет логический ноль. На вход R был подан логический ноль, в результате, на выходе инвертера D.D1 будет логическая единица, а, соответственно, на выходе Q будет так же логическая единица.

Если сигналы с входов снять (на вход S и R подать логический ноль), то состояние триггера не изменится. Логическая единица с выхода инвертера D.D1 пойдет на вход инвертера D.D2, а логический ноль с выхода D.D2 пойдет на вход инвертера D.D1, в результате чего на выходе инвертера D.D1 будет логическая единица. То есть мы замкнули цикл, который будет продолжаться до тех пор, пока будет на триггер подводиться питание.

Рассмотрим второй случай, когда на вход S подан логический ноль, а на вход R –логическая единица, то есть сброс триггера. И так, если на вход S подать логический ноль, то, пройдя через инвертер D.D2, он примет значение логической единицы. Таким образом, на выходе $\overline{Q}$ будет логическая единица. На вход R была подана логическая единица, в результате, на выходе инвертера D.D1 будет логический ноль, а, соответственно, на выходе Q будет тот же логический ноль.

Так же, как и в первом случае, при снятии сигналов с входов R и S состояние триггера не изменится.

(Давайте теперь более подробно рассмотрим принцип работы инвертера. И так на рисунке 4 изображена структурная схема инвертера).

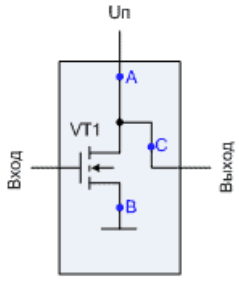


Рисунок 3. Структурная схема инвертера.

На рисунке представлена простейшая схема реализации инвертера, состоящая из одного транзистора. Давайте рассмотрим, как он работает.

На элемент всегда подается питание Uп. В результате, создаваемый ток может пойти либо по линии АВ, в этом случае на выходе инвертера ток будет отсутствовать (будет логический ноль), либо – по линии АС, в этом случае на выходе инвертера ток будет присутствовать (будет логическая единица).

По линии АВ ток пойдет, если транзистор VT1 будет открыт, а для этого необходимо подать напряжение на вход инвертера.

По линии АС ток пойдет, если транзистор VT1 будет закрыт, а это произойдет при отсутствии напряжении на входе инвертера.

Таким образом, если на вход инвертера подается логическая единица, то на выходе будет логический ноль. И, соответственно, при подаче на вход инвертера логического нуля, на выходе будет поучена логическая единица.

Устройство ячейки статической памяти.

(Теперь, зная, как работает триггер и инвертер, рассмотрим устройство ячейки статической памяти и принцип ее работы. Естественно, рассматривать мы будем простейшую ячейку памяти) .На рисунке 4 приведена упрощенная схема одного из способов организации ячейки статической памяти.

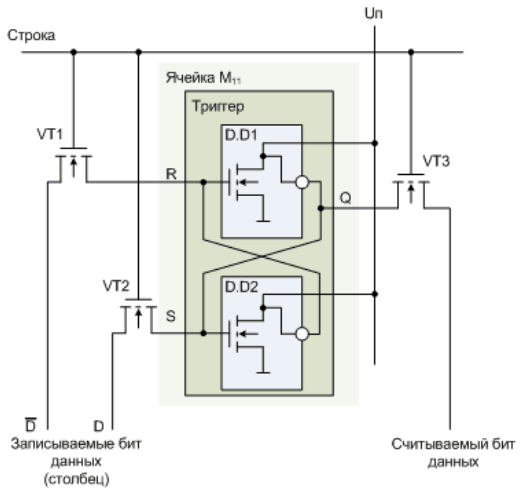


Рисунок 4. Структурная схема ячейки статической памяти (SRAM)

Как видите, она состоит из одного триггера и трех транзисторов, выполняющих роль ключей, открывающих и закрывающих доступ к ячейке памяти. Транзисторы VT1 и VT2 используются для разрешения и запрета записи в ячейку, а транзистор VT3 – для разрешения и запрета чтения.

Для **записи данных** необходимо подать напряжение в линию строки, после чего транзисторы VT1, VT2 и VT3 откроются. Затем для записи единицы необходимо подать напряжение, соответствующее логической единице, на линию D и напряжение, соответствующее логическому нулю, на линию $\overline{D}$. Для переключения триггера в **состояние хранения нуля** необходимо подать напряжение, соответствующее логическому нулю, на линию D и напряжение, соответствующее логической единице, на линию $\overline{D}$. В установленном состоянии триггер будет оставаться даже после снятия напряжения с линии строки и с линий D и $\overline{D}$ до тех пор, пока на него будет подаваться питание Uп.

Для **считывания данных** необходимо на выходы D и $\overline{D}$ подать напряжение, соответствующее логическому нулю, так как подача двух логических нулей на входы триггера не изменит его состояния, а затем подать напряжение на строку. В результате, транзистор VT3 откроется, и ток с триггера по линии Q пройдет в устройство считывания. Одновременно с транзистором VT3 откроются транзисторы VT1 и VT2. Но так как напряжение на линиях D и $\overline{D}$ соответствует логическому нулю, то оно не повлияет на состояние транзистора.

Работа статической памяти.

1. Начнем с записи данных в статическую память и рассмотрим случай записи единицы в ячейку M11.

2. В контроллер шины памяти от контроллера памяти, встроенного в северный мост материнской платы или в процессор, приходит адрес ячейки памяти и данные для записи.

3. Адрес ячейки преобразуется на две составляющие – номер строки и номер столбца. Номер строки передается в «Дешифратор адреса строки», откуда на нужную строку подается напряжение.

4. Так как мы рассматриваем запись в ячейку M11, то напряжение с дешифратора адреса строки подается на первую строку S1. В результате, транзисторы VT1, VT2 и VT3 открываются. Аналогичные транзисторы других ячеек памяти, располагающихся в этой строке, также открываются.

5. Через транзистор VT3 первой ячейки и аналогичные транзисторы других ячеек памяти первой строки пойдет ток, соответствующий состоянию триггеров этих ячеек, в «Буфер данных». Однако «Буфер данных» получаемую информацию будет игнорировать, так как у него нет сигнала от «Блока управления» на сохранение считываемых данных.

6. Параллельно с подачей напряжения на строку матрицы памяти с «Блока работы с данными» будет выдано напряжение, соответствующее записываемым данным, в «Блоки записи 1 - m», а с «Блока дешифровки адреса столбца» на соответствующие столбцы будет выдано разрешение (напряжение, соответствующее логической единице) на запись данных.

7. Блоки записи используются для запрета выдачи тока в линии D и при чтении данных и преобразования из входящих сигналов данных их инвертируемых сигналов для переключения состояния триггеров, в которые необходимо сохранить данные.

В нашем случае, запись проводится в ячейку M11, и записывается единица. Соответственно, с «Блока работы с данными» будет выдана логическая единица в «Блок записи 1», и с «Блока дешифровки адреса столбца» будет выдана логическая единица в «Блок записи 1».

Рассмотрим работу «Блока записи 1» при таких входных сигналах. И так, на входе элемента D.D3 будет логическая единица, а на выходе – логический ноль, так как элемент D.D3 – инвертер (логический элемент «НЕ»). Соответственно, на входах элемента D.D4 (логический элемент «И») будут: логический ноль и логическая единица. В результате, на выходе этого элемента будет логический ноль.

На входах элемента D.D5 (логический элемент «И») будут две логические единицы, в результате, на выходе этого элемента будет логический ноль.

Следовательно, на выходе D1 «Блока записи 1» будет напряжение, соответствующее логическому нулю, а на выходе 1 будет напряжение,

соответствующее логической единице. Эти напряжения будут поданы на все ячейки памяти первого столбца. Однако у всех ячеек, кроме первой, транзисторы, разрешающие запись, закрыты, а, следовательно, подаваемое напряжение попадет только на триггер первой ячейки и переведет его в состояние хранения единицы.

После изменения состояния триггера первой ячейки напряжение с первой строки снимается, и транзисторы VT1, VT2 и VT3 закрываются, запрещая запись и чтение из ячейки.

8. При записи нуля в ячейку памяти все происходит по той же схеме, только с «Блока работы с данными» в «Блок записи 1» будет подано напряжение, соответствующее логическому нулю. Это значит, что на выходе D1 «Блока записи 1» будет напряжение, соответствующее логической единице, а на выходе 1 будет напряжение, соответствующее логическому нулю. Эти значения напряжений переведут триггер первой ячейки памяти в состояние хранения нуля.

В установленном состоянии триггер первой ячейки останется, пока на него будет подаваться питание Uп.

9. Чтение записи происходит еще проще. От контроллера памяти приходит адрес ячеек памяти, с которых требуется считать данные, и команда на чтение.

В результате, адрес преобразуется в номер строки, и на соответствующую строку будет подано напряжение, которое откроет транзисторы разрешения/запрета чтения/записи.

Рассмотрим случай, когда данные считываются из первой ячейки. В этом случае напряжение с «Дешифратора адреса строки» будет подано в первую строку, что приведет к открытию транзисторов VT1, VT2 и VT3 ячейки M11 и всех остальных ячеек первой строки. Ток с триггера первой ячейки, через транзистор VT1, беспрепятственно пройдет в «Буфер данных». То же самое произойдет с остальными ячейками первой строки. Считанные с ячеек памяти первой строки данные сохранятся в «Буфере данных».

10. После того, как информация в «Буфере данных» будет сохранена, «Дешифратор адреса столбцов» выдаст номера столбцов, данные с которых необходимо считать, в «Буфер данных». Соответствующие данные будут переданы из микросхемы памяти в контроллер памяти, располагающийся в материнской плате или непосредственно в процессоре.

Для того чтобы при чтении данных не происходила запись в эти же ячейки, ведь транзисторы, разрешающие запись, открыты, блоки записи выдают в линии D и всех столбцов матрицы памяти напряжение,

соответствующее логическому нулю. Как видите, работа статической памяти очень похожа на работу динамической памяти, однако процесс записи и чтения гораздо быстрее, так как не тратится время на заряд и разряд конденсаторов и не требуется регенерация ячеек.

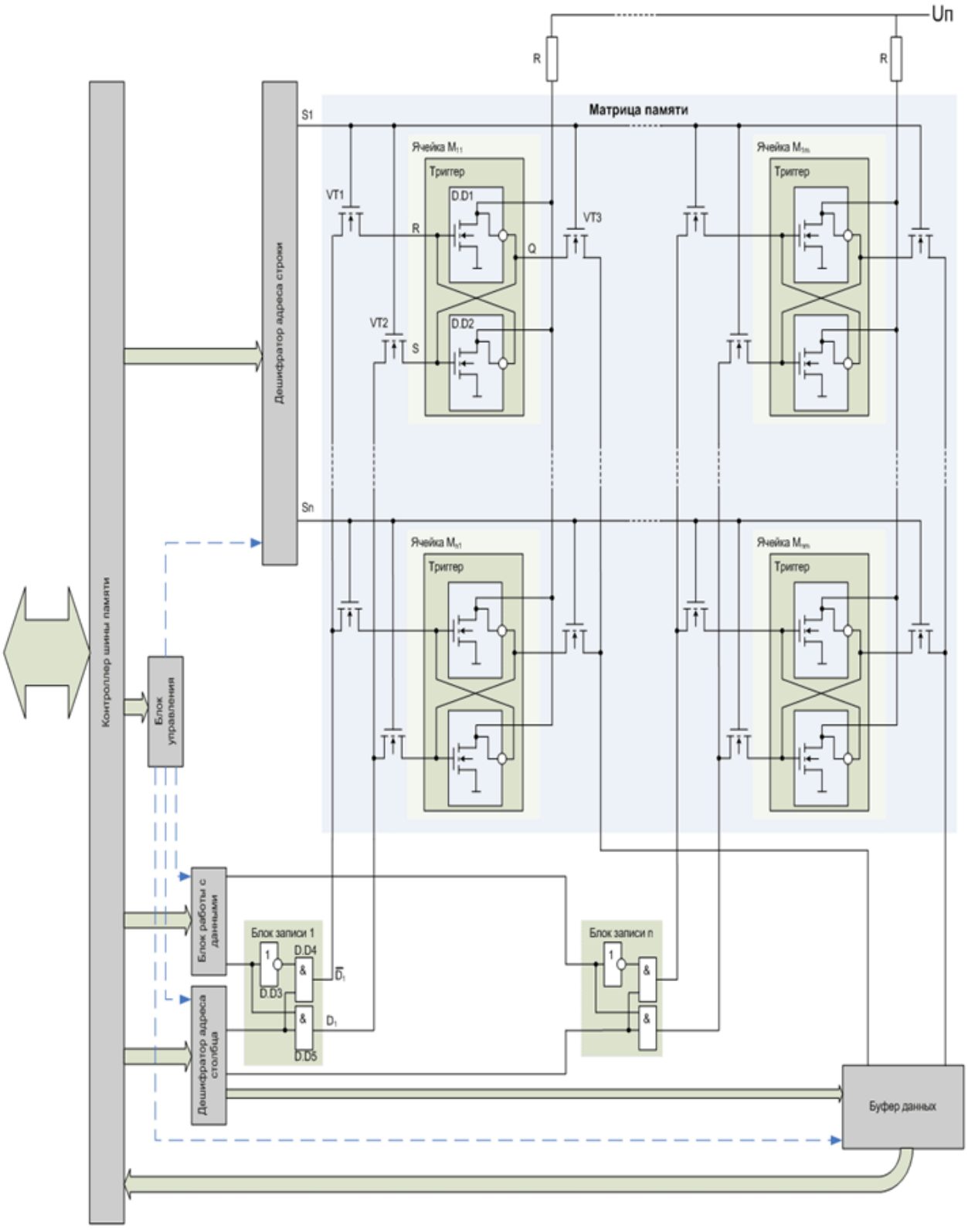


Рисунок 5. Упрощенная структурная схема статической оперативной памяти (SRAM)

Разновидности статической памяти.

Статическая память (SRAM) обычно применяется в качестве кэш-памяти второго уровня (L2) для кэширования основного объема ОЗУ. Статическая память выполняется обычно на основе ТТЛ-, КМОП- или БиКМОП-микросхем и по способу доступа к данным может быть как **асинхронной**, так и **синхронной**.

- ✓ **Async SRAM (Асинхронная статическая память)**. Это кэш-память, которая используется в течение многих лет с тех пор, как появился первый 386-й компьютер с кэш-памятью второго уровня. Обращение к ней осуществляется быстрее, чем к DRAM, и может, в зависимости от скорости процессора, использовать варианты с 20-, 15- или 10-нс доступом. Тем не менее, как видно из названия, эта память является недостаточно быстрой для синхронного доступа, что означает, что при обращении процессора все-таки требуется ожидание, хотя и меньшее, чем при использовании DRAM.
- ✓ **SyncBurst SRAM (Синхронная пакетная статическая память).** При частотах шины, не превышающих 66 МГц, синхронная пакетная SRAM является наиболее быстрой из существующих видов памяти. Причина этого в том, что, если процессор работает на не слишком большой частоте, синхронная пакетная SRAM может обеспечить полностью синхронную выдачу данных. Когда частота процессора становится больше 66 МГц, синхронная пакетная SRAM не справляется с нагрузкой и выдает данные пакетами, что существенно медленнее. К недостаткам относится и то, что синхронная пакетная SRAM производится меньшим числом компаний и поэтому стоит дороже. Синхронная пакетная SRAM имеет время адрес/данные от 8,5 до 12 нс.
- ✓ **PB SRAM (Конвейерная пакетная статическая память).** Конвейер — это распараллеливание операций SRAM с использованием входных и выходных регистров. Заполнение регистров требует дополнительного начального цикла, но, будучи однажды заполненными, регистры обеспечивают быстрый переход к следующему адресу за то время, пока по текущему адресу считываются данные.
- ✓ **1-T SRAM.** Как уже отмечалось ранее, традиционные конструкции SRAM используют статический триггер для запоминания одного разряда (ячейки).

Для реализации одной такой схемы на плате должно быть размещено от 4 до 6 транзисторов (4-T, 6-T SRAM). Фирма Monolithic System Technology (MoSys) объявила о создании нового типа памяти, в которой каждый разряд реализован на одном транзисторе (1-T SRAM).

Специальная память.

К устройствам специальной памяти относятся постоянная память (ROM), перепрограммируемая постоянная память (Flash Memory), видеопамять и некоторые другие виды памяти.

Постоянная память (ПЗУ, англ. ROM, Read Only Memory — память только для чтения) — энергонезависимая память, используется для хранения данных, которые никогда не потребуют изменения. Содержание памяти специальным образом “зашивается” в устройстве при его изготовлении для постоянного хранения. Из ПЗУ можно только читать.

Перепрограммируемая постоянная память (Flash Memory) — энергонезависимая память, допускающая многократную перезапись своего содержимого с дискеты.

Прежде всего в постоянную память записывают программу управления работой самого процессора. В ПЗУ находятся программы управления дисплеем, клавиатурой, принтером, внешней памятью, программы запуска и остановки компьютера, тестирования устройств.

Важнейшая микросхема постоянной или Flash-памяти — модуль BIOS.

Для хранения графической информации используется **видеопамять.**

Видеопамять (VRAM) — разновидность оперативного ЗУ, в котором хранятся закодированные изображения. Это ЗУ организовано так, что его содержимое доступно сразу двум устройствам — процессору и дисплею. Поэтому изображение на экране меняется одновременно с обновлением видеоданных в памяти.

Базовая система ввода/вывода (BIOS): назначение, функции, модификации.

BIOS (Basic Input/Output System) – это базовая система ввода-вывода. В состав этой системы входят различные программы ввода-вывода, которые обеспечивают взаимодействие между операционной системой, прикладными программами с одной стороны и устройствами, входящими в состав компьютера (внутренними и внешними) с другой.

(Первоначально она предназначалась для осуществления тестирования компьютера при включении – так называемых **POST**(Power On Self Test) или BIST(Built In Self Test)-процедур, и обеспечивания последующей загрузки ОС).

В настоящее время BIOS представляет собой сложную систему, состоящую из большого количества утилит, предназначенных для автоматического распознавания установленного на компьютер оборудования, его настройки и проверки функционирования. Вызов программ BIOS, как правило, осуществляется через программные или аппаратные прерывания. При включении питания компьютера BIOS тестирует (POST -- Power-On-Self-Test) компоненты системы -- процессор, память, приводы дисков (как жестких, так и флоппи-дисководов), клавиатуру т.д.

BIOS реализован в виде микросхемы, установленной на материнской плате компьютера. (Заметим, что название ROM BIOS в настоящее время не совсем справедливо, ибо "ROM" предполагает использование постоянных запоминающих устройств (Read Only Memory), а для хранения кодов BIOS в настоящее время применяют в основном перепрограммируемые запоминающие устройства). Наиболее перспективной для хранения системы BIOS является **флэш-память**. Она позволяет модифицировать функции для поддержки новых устройств, подключаемых к компьютеру.

Система BIOS неразрывносвязанас **CMOS RAM** (CMOS -- Complementary Metal Oxide Semiconductor). Это память, в которой хранится информация о текущих установках BIOS (время, количество памяти, типы жестких дисков и т.д.). В этой информации нуждаются программные модули системы BIOS.

Основными производителями BIOS являются фирмы AWARD и AMI.

Тема 2.5:Интерфейсы.

Лекция 15. Организация взаимодействия ПК с периферийными устройствами. Чипсет: назначение и схема функционирования.

Организация взаимодействия ПК с периферийными устройствами.

Для обмена данными компьютер и периферийное устройство (ПУ) оснащены внешними **интерфейсами** или **портами**. В данном случае к понятию "интерфейс" относятся:

- электрический разъем;
- набор проводов, соединяющих устройства;
- совокупность правил обмена информацией по этим проводам.

Со стороны компьютера логикой передачи сигналов на внешний интерфейс управляют:

- контроллер ПУ - аппаратный блок, часто реализуемый в виде отдельной платы;
- драйвер ПУ - программа, управляющая контроллером периферийного устройства.

Со стороны ПУ интерфейс чаще всего реализуется аппаратным устройством управления ПУ.

Обмен данными между ПУ и компьютером, как правило, является двунаправленным. Так, например, даже принтер, который представляет собой устройство вывода информации, возвращает в компьютер данные о своем состоянии.

Таким образом, по каналу, связывающему внешние интерфейсы, передается следующая информация:

- данные, поступающие от контроллера на ПУ, например байты текста, который нужно распечатать на бумаге;
- команды управления, которые контроллер передает на устройство управления ПУ; в ответ на них оно выполняет специальные действия, например выталкивает из принтера лист бумаги;

- данные, возвращаемые устройством управления ПУ в ответ на запрос от контроллера, например данные о готовности к выполнению операции.

Рассмотрим последовательность действий, которые выполняются в том случае, когда некоторому приложению требуется напечатать текст на принтере. Со стороны компьютера в выполнении этой операции принимает участие, кроме уже названных контроллера, драйвера и приложения, еще один важнейший компонент - операционная система. Итак, последовательность действий такова:

1. Приложение обращается с запросом на выполнение операции печати к операционной системе. В запросе указываются: адрес данных в оперативной памяти, идентифицирующая информация принтера и операция, которую требуется выполнить.
2. Получив запрос, операционная система анализирует его, решает, может ли он быть выполнен, и если решение положительное, то запускает соответствующий драйвер, передавая ему в качестве параметров адрес выводимых данных.
3. Драйвер передает команды и данные контроллеру, который помещает их в свой внутренний буфер.
4. Контроллер перемещает данные из внутреннего буфера во внешний порт.
5. Контроллер начинает последовательно передавать биты в линию связи, представляя каждый бит соответствующим электрическим сигналом. Чтобы сообщить устройству управления принтера о том, что начинается передача байта, перед передачей первого бита данных контроллер формирует стартовый сигнал специфической формы, а после передачи последнего информационного бита - стоповый сигнал. Эти сигналы синхронизируют передачу байта.
6. Устройство управления принтера, обнаружив на соответствующей линии стартовый бит, выполняет подготовительные действия и начинает принимать информационные биты, формируя из них байт в своем приемном буфере. Наконец, принятый байт обрабатывается принтером - выполняется соответствующая команда или печатается символ.

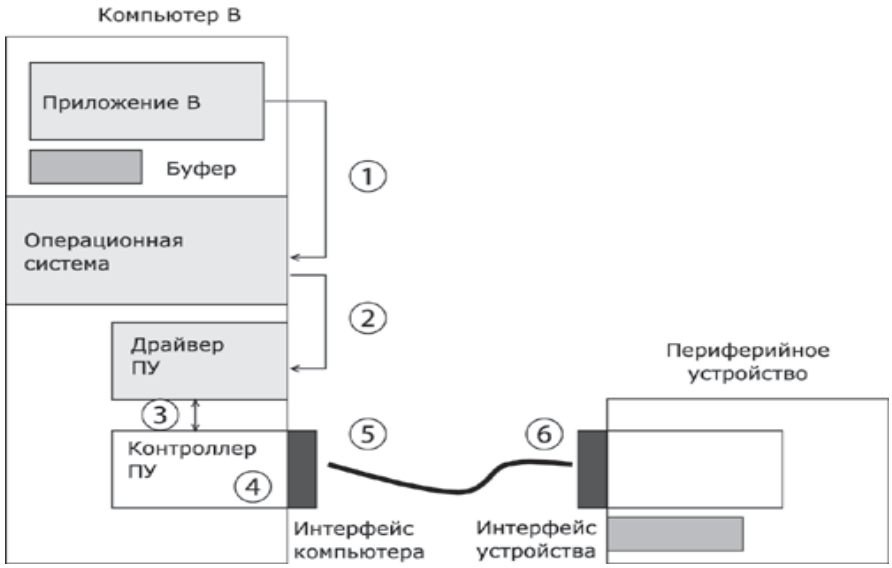


Рис. 3.1. Связь компьютера с периферийным устройством.

Чипсет: назначение и схема функционирования.

Чипсет или набор системной логики - это основной набор микросхем материнской платы, обеспечивающий совместное функционирование центрального процессора, ОЗУ, видеокарты, контроллеров периферийных устройств и других компонентов, подключаемых к материнской плате.

Системные чипсеты и контроллеры представляют собой логические схемы, которые образуют "интеллект" материнской платы. Они действуют как "регулировщики уличного движения", управляя передачами данных между процессором, кэшем, системными шинами, и периферийными устройствами, подключенными к шинам ISA и PCI. Кроме того, чипсет управляет обменом данными с жесткими дисками и другими устройствами, подключенным к каналам IDE. Таким образом, чипсет управляет передачами данных практически между всеми основными компонентами PC. Поскольку передачи данных влияют на работу и производительность стольких компонентов PC, чипсет определяет качество PC, его возможности и быстродействие.

Чипсет (chipset) - это просто набор микросхем (set of chips). В прошлом большинство функций чипсета выполняло множество меньших микросхем контроллеров. Для каждой функции (а часто и нескольких) применялся отдельный контроллер - для управления кэшем, выполнения прямого доступа к памяти (Direct Memory Access - DMA), обработки прерываний, передач данных по шинам ввода-вывода и т.д. Со временем эти микросхемы были интегрированы в набор микросхем, который и называется чипсетом. Ситуация напоминает эволюцию самого процессора - в прошлом

многие компоненты процессора Pentium были представлены отдельными микросхемами.

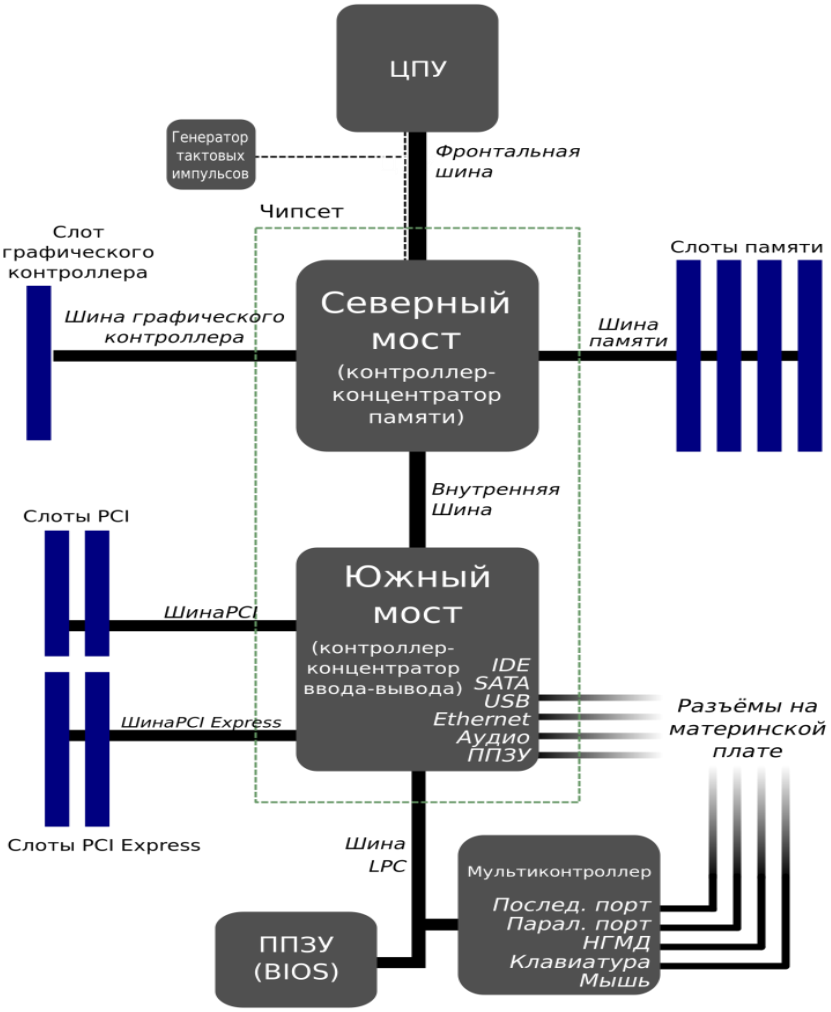
Имеется несколько преимуществ такой интеграции, главными из которых являются снижение стоимости и лучшая совместимость - чем больше функций реализуются одной микросхемой или группой микросхем одного производителя, тем проще проектирование и тем меньше вероятность возникновения проблем. Иногда микросхемы чипсета называются специализированными интегральными схемами (Application-Specific Integrated Circuits - ASIC), но имеется и множество других типов таких микросхем. По существу чипсет представляет собой набор микросхем интеллектуальных контроллеров, который имеется на любой материнской плате. На практике оказалось, что при разработке нового процессора требовалось проектировать для него новый чипсет. На следующем рисунке наглядно показаны функции чипсета, которые сводятся к объединению множества компонентов PC.

Фирма Intel также называет свои чипсеты "PCIsets" и "AGPsets" в соответствии с системной шиной, на которую рассчитаны чипсеты.

Контроллеры тесно взаимодействуют с центральным процессором, управляя шинами PC, как показано на рисунке слева. Без чипсета ни системная память, ни шины ввода-вывода не могли бы функционировать совместно с центральным процессором.

Обычно системный чипсет не интегрирует все схемы, требующиеся на материнской плате. Большинство материнских плат содержат следующие контроллеры: сам системный чипсет; контроллер клавиатуры, который управляет не только клавиатурой, но и мышью PS/2; микросхему "Super I/O", которая управляет вводом и выводом последовательных портов, параллельного порта, гибких дисков и жестких дисков с интерфейсом IDE; дополнительные встроенные контроллеры, которые обычно размещаются на платах расширения: наглядными примерами служат контроллеры видео, звука, сетевые контроллеры и SCSI-контроллеры.

Конструктивно чипсет состоит из **северного моста, южного моста и соединяющей их внутренней шины.**



Северный мост (Northbridge) - это системный контроллер, являющийся одним из элементов чипсета материнской платы, отвечающий за работу с оперативной памятью (RAM), видеоадаптером и процессором (CPU). Северный мост отвечает за частоту системной шины, тип оперативной памяти и ее максимально возможный объем. Одной из основных функций северного моста является обеспечение взаимодействия системной платы и процессора, а также определение скорости работы. Частью северного моста во многих современных материнских платах является встроенный видеоадаптер. Таким образом, функциональная особенность северного моста являет собой еще и управление шиной видеоадаптера и ее быстродействием. Также северный мост обеспечивает связь всех вышеперечисленных устройств с южным мостом.

Южный мост (Southbridge) - это функциональный контроллер, известен как контроллер ввода-вывода или ICH (In/Out Controller Hub). Отвечает за так называемые "медленные" операции, к которым относится отработка взаимодействия между интерфейсами IDE, SATA, USB, LAN, Embeded Audio и северным мостом системы, который, в свою очередь, напрямую связан с процессором и другими важными компонентами, такими как оперативная

память или видеоподсистема. Также южный мост отвечает за обработку данных на шинах PCI, PCIe и ISA (в старых моделях системных плат).

Примечание: *Термином "чипсет" часто называют также основные схемы обработки на многих видеокартах. Принцип здесь тот же: сверхбольшая интегральная схема используется для выполнения набора функций. Однако это совершенно другой тип чипсета, которые необходимо отличать от системного (находящегося на материнской плате) чипсета.*

Чипсеты выпускают несколько компаний, например SIS, VIA и Opti, но наиболее популярны чипсеты Triton фирмы Intel. Появление чипсетов Triton произвело революцию на рынке материнских плат и почти все производители стали ориентироваться на чипсеты Triton. Эти чипсеты позволяют максимально использовать возможности процессора и шины PCI, имеют встроенную поддержку ведущего интерфейса EIDE, улучшенный мост шины ISA и могут работать с различными видами системной памяти RAM.

Функции и возможности чипсета

1.Поддержка чипсетом процессора

- *поддержка класса процессора и оптимизация.* Чипсет разрабатывается для конкретного набора процессоров. В общем, большинство чипсетов поддерживает только один класс, или поколение, процессоров.

- ***поддержка быстродействия (скорости) процессора.*** При повышении скорости процессора требуется, чтобы схемы управления чипсета работали с соответствующей скоростью.

- ***Поддержка нескольких процессоров*** Некоторые чипсеты поддерживают возможность размещения на материнской плате двух или четырех процессоров. Схемы чипсета координируют действия процессоров таким образом, что они не "мешают" друг другу, а совместно с операционной системой разделяют нагрузку между несколькими процессорами ради повышения производительности.

2. Поддержка чипсетом кэша

- ***Емкость (размер) вторичного кэша*** Чипсет определяет, каким образом поддерживается L2-кэш.

- ***Тип вторичного кэша*** Сейчас применяются три типа L2-кэша; в порядке повышения производительности ими являются: асинхронный кэш, синхронно-пакетный кэш и конвейерно-пакетный кэш. Для каждого типа кэша требуются различные схемы управления, поэтому каждый тип должен явно поддерживаться чипсетом.

- *Политика записи вторичного кэша* Имеются две "политики" (policy) управления записью в память в кэшируемой системе: кэш со сквозной записью (write through cache) означает, что записываемые в память данные передаются в

память, как только процессор выдает данные; кэш с обратной записью (write back cache) означает, что процессор записывает данные только в кэш, а затем в благоприятное время они записываются в память. Производительность кэша с обратной записью выше, чем кэша со сквозной записью.

- Кэшируемость системной памяти Характеристики чипсета управляют максимальной емкостью памяти, которую может кэшировать система. Емкость кэшируемой памяти отличается от общей емкости памяти PC.

3.Поддержка чипсетом памяти

- Поддержка максимальной памяти Чипсет определяет максимальную емкость памяти RAM на материнской плате.

- Технология DRAM Чипсет определяет, можно ли использовать на материнской плате память FPM, EDO, BEDO или SDRAM. Изменение типа памяти влияет на операции считывания и записи, которыми управляет чипсет.

- Конструктивное оформление DRAM Память для настольных PC выпускается в виде модулей SIMM и DIMM. Схемы чипсета также определяют, сколько слотов модулей SIMM или DIMM может иметь материнская плата. Лучшие чипсеты поддерживают больше слотов, обеспечивая большую гибкость в модернизации PC.

- Поддержка паритета и исправления ошибок Возможность исправления ошибок встраивается в схемы управления памятью чипсета.

4.Управление временной диаграммой

Одна из важных функций чипсета состоит в управлении операциями считывания и записи памяти, а также передачами данных от процессора на локальную шину (обычно PCI и/или AGP). Качество чипсета, тип кэша, скорость системной памяти и тип используемого процессора определяют эффективность передач данных в процессор и из процессора, а она влияет на общую производительность PC.

- ***Дешифрирование адреса*** чипсет выполняет функцию преобразования запросов данных и команд от процессора в адреса ячеек памяти, в которых находится запрашиваемая информация.

- ***Передача данных кэша и памяти*** Когда процессор запрашивает информацию из памяти, вначале проверяется, не находится ли она в кэше (так как кэш намного быстрее памяти). Если информация содержится в кэше, она считывается именно оттуда, а обращение к памяти отменяется. В противном случае информация считывается из памяти и передается в процессор для обработки и в кэш в надежде, что она вскоре потребуется процессору. Чипсет управляет временной диаграммой этих передач.

- Буферирование шины Чипсет управляет потоком информации от локальной шины (PCI в PC с процессором класса Pentium) в память, а также от шины PCI

непосредственно в процессор. В PC с ускоренным графическим портом (Accelerated Graphics Port - AGP) чипсет также координирует передачи между процессором и видеокартой.

- Временная диаграмма системной памяти Так как память в большинстве случаев работает медленнее процессора, процессор часто должен ожидать нужной ему информации из памяти. Состояние ожидания (wait state) представляет собой такт синхронизации (tick), когда процессор простаивает в ожидании данных из памяти. Задача чипсета состоит в том, чтобы свести ожидание к минимуму; при необходимости он вставляет такты ожидания, чтобы процессор не опережал системных кэша или памяти. Чем быстрее системные кэш и память, тем меньше состояний ожидания приходится вводить и тем выше производительность PC.

-Автоматическое обнаружение памяти (memory autodetection)Большинство современных чипсетов способны автоматически обнаруживать тип и быстродействие памяти, а затем соответственно скорректировать состояния ожидания для достижения максимальной производительности без потери данных.

5. Управление периферийными устройствами и шинами ввода-вывода

В большинстве современных PC используются две шины: шина ISA (Industry Standard Architecture) для медленных периферийных устройств и совместимости со старыми компонентами и шина PCI (Peripheral Component Interconnect) - скоростная локальная шина для жестких дисков, видеокарт и других быстродействующих устройств. Есть также порт AGP для видеокарты.

Чипсет управляет этими шинами и обменивается данными между шинами и процессором и памятью. Характеристики чипсета определяют, какие типы шин поддерживает система, с какой скоростью они могут работать и какие дополнительные возможности они могут иметь.

6.Поддержка управления мощностью

Большинство новых чипсетов поддерживает набор возможностей, которые снижают мощность, потребляемую PC во время бездеятельности. Разработка этих возможностей стимулировалась желанием уменьшить потребление мощности PC, которые длительное время остаются включенными, но ничего не делают, а также лаптопов, которые после заряда батареи должны работать как можно более продолжительное время.

Лекция 16. Системная шина и ее параметры. Интерфейсные шины и связь с системной шиной. Системная плата: архитектура и основные разъемы.

Системная шина и ее параметры.

В основе устройства ЭВМ лежит **системная шина**, которая служит для обмена командами и данными между компонентами ЭВМ, расположенными наматеринской плате. ПУ подключаются к шине через контроллеры. Такая архитектура ЭВМ называется открытой, так как легко может быть расширена за счет подключения новых устройств. Передача информации по системной шине также осуществляется по тактам.

Системная шина обеспечивает три направления передачи информации:

1) между МП и ОЗУ;
2) между МП и контроллерами устройств;

3) между ОЗУ и внешними устройствами.

Характеристиками системной шины являются количество **обслуживаемых ею устройств** и ее **пропускная способность**, то есть максимально **возможная скорость передачи информации**. Пропускная способность шины зависит от следующих параметров:

- **разрядность или ширина шины** – количество бит, которое может быть передано по шине одновременно (существуют 8-, 16-, 32- и 64-разрядные шины);
- **тактовая частота шины** – частота, с которой передаются биты информации по шине.

Наиболее распространенные шины:

PCI – самая распространенная системная шина. Быстродействие шины не зависит от количества подсоединенных устройств. Поддерживает следующие режимы:

AGP – магистраль между видеокартой и ОЗУ. Разработана, так как параметры шины PCI не отвечают требованиям видеоадаптеров по быстродействию. Шина работает на большей частоте, что позволяет ускорить работу графической подсистемы ЭВМ.

Сейчас определение системной шины устарело, а функции ее выполняет чипсет компьютера.

Системная плата: архитектура и основные разъемы.

Центральным конструктивным узлом ПЭВМ, определяющим вместе с процессором ее архитектуру и базовые характеристики, является системная плата (system board), или материнская плата (motherboard), или основная (главная) плата (main board).

Системная плата представляет собой печатную плату, на которой смонтированы все электронные составные части компьютера: процессор; ОЗУ; BIOS; набор системных и вспомогательных микросхем, контроллеров ввода-вывода; память CMOS с автономным питанием.

Системная плата содержит ряд коммутационных элементов: слоты расширения; разъемы для подключения интерфейсных кабелей клавиатуры, мыши, жестких дисков, оптических дисководов, последовательного и параллельного портов, шины USB; преобразователь напряжения для питания ядра процессора и ряд других компонентов, необходимых для работы ПЭВМ.

В основном современные платы состоят из шести печатных слоев, включающих в себя три или четыре слоя сигнальных дорожек, пластину заземления (ОВ), соединенную с корпусом ПЭВМ и экранирующую перекрестные помехи от высокочастотных цепей, и один или два слоя проводников питания. В верхнем слое размещаются контактные площадки для распайки компонентов. В платах предусматриваются отверстия для винтового крепления к боковой стенке системного блока. От климатических воздействий плата защищена водостойким диэлектрическим лаком.

Одной из характеристик материнской платы являются ее типоразмер — ***форм-фактор,*** определяющий расположение процессора и разъемов расширения, габаритные размеры и точки крепления платы, а также тип разъема питания платы и питающие напряжения. Кроме того, форм-фактор платы предопределяет используемый тип корпуса и блока питания.

ATX-платы имеют более совершенную конструкцию, позволяя устанавливать до шести слотов расширения с платами полной длины, обеспечивая удобный доступ к компонентам и хорошее их охлаждение. В настоящее время все системные платы выпускают только в формате ATX (рис. 2).Внедрение форм-фактора ATX позволило установить на системную плату разъемы портов ввода-вывода, что привело к существенному снижению количества соединительных проводов внутри корпуса ПЭВМ. Кроме того, упростился доступ к модулям ОЗУ при их замене или наращивании объема. Кроме того, разъемы контроллеров накопителей переместились практически вплотную к дисководам, что позволило сократить длину используемых кабелей и тем самым повысить надежность и помехозащищенность интерфейсных связей.

Все внешние разъемы (порты) располагаются на двух уровнях и распаяны на крае печатной платы. Для них на тыльной стенке корпуса системного блока предусмотрено специальное отверстие.

Современные платы рассчитаны в основном на применение памяти DDR SDRAM, поэтому используют для установки модулей ОЗУ несколько гнездовых разъемов типа DIMM на 184 контакта, которые имеют по краям специальные фиксирующие пластмассовые • защелки.

В большинстве системных плат используют двухуровневую структуру, образованную двумя микросхемами системной логики — северного и южного мостов, через которые осуществляете я управление работой контроллеров и других компонентов ПЭВМ.

Основным узлом системной логики является ***быстродействующая микросхема северного моста***, которая обеспечивает единственный интерфейс

между процессором и остальной частью системной платы, работающий на полной тактовой частоте шины процессора. Северный мост имеет значительное энергопотребление и поэтому требует пассивного охлаждения (установки радиатора).

Микросхема южного моста имеет меньшее быстродействие, обычно реализует интерфейсы двухканального контроллера жестких дисков и USB-портов, содержит память CMOS и схему часов, атакже компоненты, необходимые для функционирования шины PCI, например контроллер прямого доступа к памяти.

Микросхема ввода-вывода SuperI/O в большинстве системных плат обычно реализует функции контроллеров, которые раньше размещались на отдельных платах расширения ISA, и обеспечивает подключение:

■ дисковода гибких дисков;
■ двух быстродействующих последовательных COM-портов;
■ многорежимного параллельного порта;
■ клавиатуры и мыши.

С развитием технологий производители объединяют все больше функций в основном наборе микросхем системной логики, поэтому необходимость в микросхеме Super I/O постепенно исчезает. Схемы Super I/O и южного моста объединяются, благодаря чему уменьшается количество компонентов на системной плате и освобождается свободная площадь на плате.

В платах ATX используется 20-контактный разъем питания с ключом, исключающим неправильное подсоединение, через который подаются напряжения ±12, ±5 и +3,3 В, а также сигналы программного включения и выключения питания ПЭВМ. Двигатели дисковых накопителей и кулеры процессоров питаются от напряжения 12 В.

Для процессоров с двойным напряжением питания — основным и ядра — требуется дополнительный ***вторичный источник напряжения VRM*** (Voltage Regulator Module — модуль регулирования напряжения), в котором применяются два метода преобразования: линейный и импульсный. Линейный источник напряжения использовался в старых платах, имел малый коэффициент полезного действия (КПД), так как понижал входное напряжение за счет его падения на регулирующем элементе (мощном транзисторе) и рассеивания в виде теплоты. С уменьшением выходного напряжения росла тепловая мощность, рассеиваемая такими преобразователями, поэтому они имели массивные радиаторы и значительно ухудшали температурный режим компонент материнской платы.

В современных материнских платах стали использовать импульсные источники с катушками индуктивности и высоким КПД. Применяются и

комбинированные варианты питания с двухступенчатой схемой: сначала импульсный преобразователь понижает напряжение с 12 В, поступающее через разъем ATX12V, до 3,3 В, а затем линейный преобразователь регулирует это значение номинала 1,4... 2,0 В, который зависит от типа процессора. Чтобы использовать на системной плате процессоры с разными номиналами питания ядра, с помощью переключателя, устанавливаемого на плате, можно вручную формировать напряжения с шагом 0,1 В или через BIOS.

В современных материнских платах используется автоматическое определение номинала питания, когда схема VRM сама определяет требуемое напряжение по контактам процессора. Для контроля напряжения на платах устанавливают светодиоды.

К дополнительным компонентам относятся датчики, выдающие информацию о температуре процессора, материнской платы, скорости вращения вентилятора и др. Большинство современных материнских плат аппаратно и программно поддерживают несколько экономичных режимов с пониженным энергопотреблением (например, ждущий режим).

Обязательным устройством материнской платы является ***тактовый генератор,*** который выдает сетку рабочих частот для синхронизации всех электронных устройств ПЭВМ. Для селективной установки номиналов частот, требуемых процессору, ОЗУ, видеокарте и слотам PCI, используется специальный ползунковый коммутатор, который меняет коэффициент их изменения.

В компьютерах применяется деление опорной частоты (частота шины FSB) генератора для синхронизации других шин и внутреннее умножение этой частоты в процессоре. Для каждой шины тактовый генератор может поддерживать как один, так и несколько коэффициентов деления, который меняется через 33 МГц относительно опорной частоты. Для обеспечения работы различных типов процессоров используются множители частоты для шины FSB.

После контроля допусков номиналов напряжений и при наличии сигнала исправного состояния от блока питания ПЭВМ тактовый генератор формирует сигнал начальной установки для перевода процессора в рабочий режим.

Для уменьшения длины шлейфов разъемы контролеров накопителей располагаются на плате ближе к отсекам для крепления дисководов.

На материнской плате обычно размещают дополнительные контроллеры (модемный, сетевой, SCSI, звука и т.д.). Встроенные звукопреобразующие средства обычно позволяют обеспечить шестиканальную акустику. В этом

случае на плате имеются разъемы (линейный вход-выход, микрофон), а также MIDI-порт.

Слот AGP используется только для установки видеокарт. Иногда материнские платы поставляются с разъемом AGP Pro для видеокарт с повышенными энергопотреблением и тепловыделением.

Другие дополнительные функциональные узлы ПЭВМ выполняются на отдельных печатных платах, которые устанавливаются в разъемы расширения PCI (рис. 4.5).

Стандарт PCI предлагает три вида плат для ПЭВМ разных типов и с различным напряжением питания. Платы с напряжением 5 В предназначены в основном для стационарных компьютеров, а с напряжением 3,3 В — для портативных компьютеров.

Технология Plug and Play значительно упростила процесс установки и конфигурирования новых устройств. Пользователю необходимо лишь вставить плату в свободный разъем PCI, а система BIOS автоматически выделит необходимые ресурсы. Во время определения конфигурации ПЭВМ BIOS проверяет наличие устройств на PCI-шине, назначает для них адреса и разрешает их инициализацию.

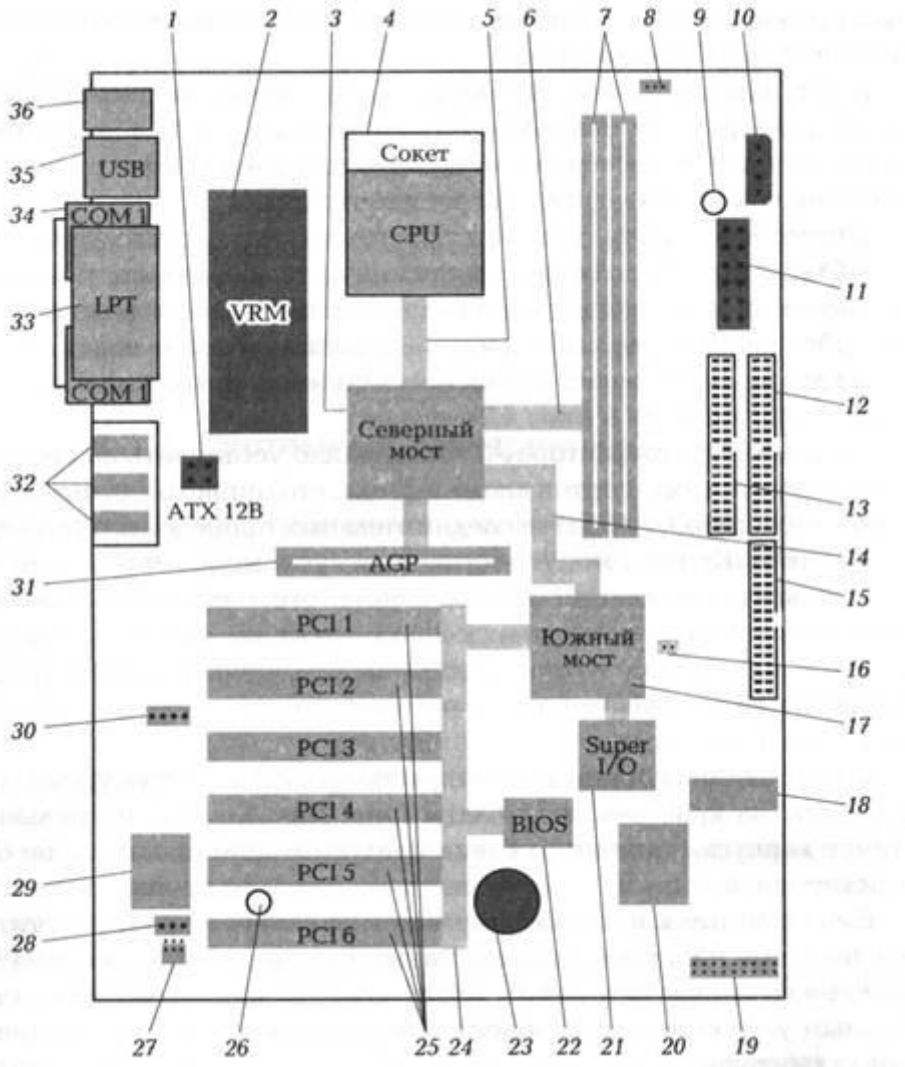
Для хранения оперативных настроек на системную плату устанавливается ***ЗУ CMOS*** с автономным источником питания, поэтому ее содержимое не стирается после выключения ПЭВМ. Из-за очень малого потребления энергии время надежного хранения информации в памяти составляет свыше 15 лет и зависит от емкости гальванического источника напряжением 3 В, который питает и схему часов. Если на плате установлен аккумулятор, то он будет постоянно подзаряжаться при включенном компьютере.

Внося изменения в CMOS с помощью ***меню BIOS***, пользователь может вручную настроить работу материнской платы, установить параметры гибких и жестких дисков, интервалы времени, по истечении которого компьютер переходит в режим ожидания, параметры питания и многое другое.

В экстремальных случаях (забыт пароль) сброс информации в CMOS осуществляется с помощью перемычки, отключающей питание от микросхемы.

Для подключения индикаторов, кнопок и динамика, расположенных на корпусе системного блока, на материнской плате имеются миниатюрные разъемы-вилки. Подобные же разъемы служат как контакты для перемычек при задании аппаратной конфигурации системы и для соединения платы с аудиовыходом оптических дисководов.

Схема размещения конструктивных элементов на системной плате формата ATX:



1 — разъем питания VRM;2 — узел VRM питания ядра процессора;3 — северный мост (хаб);4 — сокет (гнездо) процессора;5 — процессорная шина;6 — шина ОЗУ;7 — модули ОЗУ;8— регулятор питания ОЗУ;9 — светодиод контроля питания видеокарты;10 — вспомогательное питание платы;11 — основное питание стандарта ATX;12 — разъем подключения ведущего НМЖД;13 — разъем подключения ведомого НМЖД;14 — шина взаимодействия мостов;15 — разъем НМГД;16 — сброс CMOS;17 — южный мост (хаб);18 — коммутатор частот процессора и шин;19 — разъем управления ПЭВМ;20 — контроллер накопителей;21 — контроллер ввода-вывода (SuperI/O);22 — BIOS;23 — автономное питание CMOS;24 — PCI- шина;25 — слоты расширения;26 — индикатор подачи напряжения на системную плату;27 — разъем цифрового звука;28 — разъем модема;29 — контроллер аудио;30 — разъем аналогового выхода оптического дисковода;31 — слот видеокарты;
32 — разъемы аудио;33 — параллельный порт;34 — COM-порты;35 — USB-порты;36 — разъемы подключения клавиатуры и мыши

Лекция 17. Внутренние интерфейсы ПК: шины ISA, EISA, VCF, VLB, PCI, AGP и их характеристики. Интерфейсы периферийных устройств IDE и SCSI. Последовательный порт стандарта RS-232: назначение, структура кадра данных, структура разъемов. Интерфейс стандарта 802.11 (Wi-Fi).

Внутренние интерфейсы ПК: шины ISA, EISA, VCF, VLB, PCI, AGP и их характеристики.

ISABUS(IndustryStandardArchitecture - Промышленный Стандарт Архитектуры) - стандартные шины IBMPCXT (8 бит) и AT (16 бит).

Шина XT имеет: 8-битовую шину данных;20-битовую шину адреса, что позволяет адресоваться к 220 бит (1 Мбайт) памяти; три канала прямого

доступа к памяти (DMA);тактовую частоту 8 МГц; пропускную способность 4 Мбайт/с; 62-контактный разъем.9--

В настоящее время XT практически не применяется. В компьютерах AT шину расширили до 16 бит данных и 24 бит адреса. В таком виде она существует и поныне как самая распространенная шина для периферийных адаптеров.

Шина AT имеет:6-битовую шину данных;24-битовую шину адреса, что позволяет адресовать 16 Мбайт памяти;8 каналов прямого доступа (DMA);тактовую частоту 8-16 МГц.

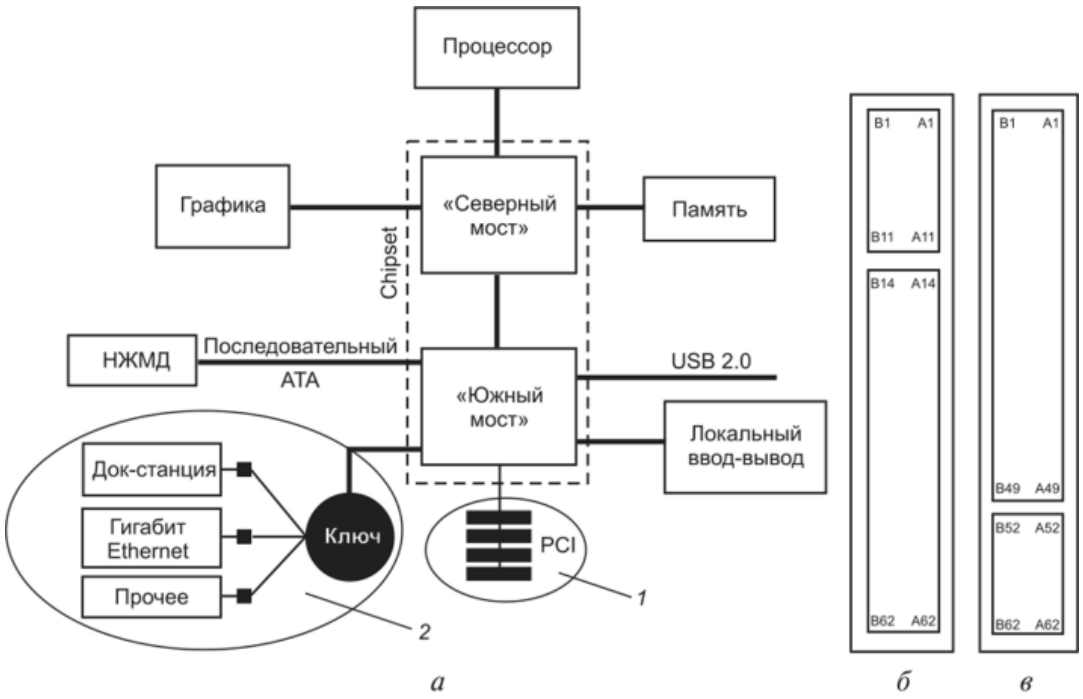
Шина EISA явилась «асимметричным ответом» производителей клонов PC на попытку IBM поставить рынок под свой контроль путем выпуска MCA. В сентябре 1988 года производители компьютеров - Compaq, Wyse, AST Research, Tandy, Hewlett-Packard, Zenith, Olivetti, NEC и Epson - представили совместный проект: 32-разрядное расширение шины ISA с полной обратной совместимостью. Основные характеристики новой шины:32-разрядная передача данных;максимальная пропускная способность 33 Мбайт/с;32-разрядная адресация памяти позволяла адресовать до 4 Гбайт;поддержка многих активных устройств (bus master);возможность задания уровня двухуровневого (edge-triggered) прерывания (что позволяло нескольким устройствам использовать одно прерывание, как и в случае многоуровневого (level-triggered) прерывания);автоматическая настройка плат расширения.

Шина PCI (peripheral component interconnect bus — взаимосвязь периферийных компонентов)

Разработка шины PCI закончилась в июне 1992 года как внутренний проект корпорации Intel. Основные возможности шины следующие:

синхронный 32- или 64-разрядный обмен данными (64-разрядная шина в настоящее время используется только в Alpha-системах и серверах на базе процессоров Intel Xeon). При этом для уменьшения числа контактов (и стоимости) используется мультиплексирование, то есть адрес и данные передаются по одним и тем же линиям;частота работы шины 33 или 66 МГц (в версии 2.1) позволяет обеспечить широкий диапазон пропускных способностей (с использованием пакетного режима);полная поддержка многих активных устройств (например, несколько контроллеров жестких дисков могут одновременно работать на шине);спецификация шины позволяет комбинировать до восьми функций на одной карте (например, видео, звук и так далее).

Архитектуры шин PCI (1) и PCX (2)



а - разъем 32-разрядной шины с напряжением питания 5 В; б - то же с напряжением питания 3.3 В; в - типичное PCI-устройство

Известны также более поздние разновидности - PC1-X и PCI-Express, кроме того, к данному типу относится и PCMCIA - стандарт на шину для ноутбуков. Она позволяет подключать расширители памяти, модемы, контроллеры дисков и стримеров, SCSI-адаптеры, сетевые адаптеры и другие.

Внутренний интерфейс пк: шины AGP, PCI-E и их характеристики.

Появление ***шины AGP*** в начале 1998 года было своеобразным прорывом в области графических работ. При частоте шины в 66 МГц она была способна передавать два блока данных за один такт. Пропускная способность шины составляет 500 Мбайт/с (V2.0) при двух режимах работы: DMA и Execute. На сегодняшний день существует стандарт (поддерживаемый новыми чипсетами Intel и Via) AGP4x, позволяющий повысить пропускную способность до 1 Гбайт/с.

AGP функционирует на скорости процессорной шины (FSB). При тактовой частоте 66 МГц, например, это в 2 раза выше, чем скорость PCI, и позволяет достигать пиковой пропускной способности в 264 Мбайт/с. В графических картах, специально спроектированных для AGP, передача происходит как по переднему, так и по заднему фронту тактовых импульсов центрального процессора, что позволяет при частоте 133 МГц осуществлять передачу со скоростью до 528 Мбайт/с (это называется «2-х графика»). В дальнейшем была выпущена версия AGP 2.0, которая поддерживала «4-х графику» или четырехкратную передачу данных за один такт центрального процессора.

Стандарт PCI-E определяет гибкий, масштабируемый, высокоскоростной, последовательный, интерфейс, программно-совместимый с PCI. Это устраняет потребность в шинном арбитраже, обеспечивает низкое время ожидания и упрощает быстрое подключение-отключение системных устройств.

Интерфейс PCI-E включает пары проводов - каналы (lane), и единственная пара (PCI-E-lane) представляет собой интерфейс PCI-E 1x (800 Мбайт/с). Каналы могут быть соединены параллельно, и максимум (32 канала – PCI-E 32x) обеспечивает полную пропускную способность 16 Гбайт/с.

Одним из направлений развития PCI-E является замена AGP. При этом данные технологии характеризуются следующими особенностями:

· AGP (от англ. Accelerated Graphics Port, ускоренный графический порт) - разделение полос пропускания для записи и чтения; общая полоса пропускания - 2 Гбайт/с; оптимизировано для однозадачного режима.

· PCI Express (англ. Peripheral component interconnect, дословно — взаимосвязь периферийных компонентов) - выделенные полосы для ввода и вывода; общая полоса пропускания до 8 Гбайт/с; оптимизировано для многозадачного режима.

Интерфейсы периферийных устройств IDE и SCSI. Современная модификация и характеристики.

IDE (IntegratedDeviceElectronics) - интерфейс устройств со встроенным контроллером. При создании этого интерфейса разработчики ориентировались на подключение дискового накопителя. За счет минимального удаления контролера от диска существенно повышается быстродействие.

Физически интерфейс IDE реализован с помощью плоского 40-жильного кабеля, на котором могут быть разъемы для подключения одного или двух устройств. Общая длина кабеля не должна превышать 45 сантиметров, причем между разъемами должно быть расстояние не менее 15 сантиметров.

Для обозначения IDE в настоящее время широко используется аббревиатура PATA (ParallelATA), которая подчеркивает, что для передачи данных используется параллельный интерфейс

Подключение PATA позволяет подключать два устройства на канал.

Самые последние версии могут иметь поддержку скорости передачи данных до 133 МБ/с.

SATA использует последовательный порт и поддерживает технологию «горячего» подключения. С помощью технологии Plug and Play, компьютерные компоненты могут быть заменены без выключения системы

Кабель данных имеет 9 контактов и длину не более метра. К самому разъему значительно проще и удобнее подключать устройства. К тому же с

появлением SATA, можно забыть про разграничение устройств на Master и Slave.

Первые SATA поддерживали скорость в 1.5 Гбит/с. Современные версии поддерживают скорость передачи данных в 3 Гбит / с и до 6 Гбит / сек.

SCSIэтот стандарт был поддержан многими производителями и лидерами отрасли того времени. Самая первая версия использовала 50-контактный плоский разъем. В то время как первые разъемы SCSI использовали параллельные интерфейсы, более современные SCSI работают через последовательный интерфейс. Последовательный интерфейс SCSI, по сравнению с параллельным, обеспечивает более высокую скорость передачи данных. SCSI может быть либо установлен на материнской плате физически, либо может быть реализован с помощью адаптеров.

Современные версии могут передавать данные до 80 мегабайт / сек. Современные устройства SCSI имеют обратную совместимость, т.е. если подключено устройство старшей версии, то шина SCSI будет по-прежнему поддерживать его, хотя скорость передачи данных может быть сниженной.

Последовательный портстандарта RS-232-C

Интерфейс RS-232-C разработан EIA. (Назначение в предыдущей части).

Последовательная передача данных состоит в побитовой передаче каждого байта цифровой информации, в форме кадра данных, содержащего сигнал начала передачи (Start), сигнал окончания передачи (Stop) и информационные биты.

Структура кадра данных при передаче байта информации в стандарте RS-232-C

Бит ST сигнализирует о начале передачи данных, затем передается информационные биты - вначале младшие, потом старшие.

Иногда используется контрольный бит P, которому присваивается такое значение, чтобы общее число единиц или нулей было четным или нечетным. Это применяется для контроля правильности передачи кадра. Приемное устройство проверяет кадр на четность и при несовпадении с ожидаемым значением передает запрос о повторе передачи кадра. Бит (или биты) SP сигнализирует об окончании передачи байта.

Принимающее и передающее устройства должны применять одинаковые форматы.

Разъем для подключения последовательного порта может содержать 25 или 9 выводов (соответственные обозначения - D25 и D9). Только два провода этих разъемов используются для передачи и приема данных, остальные отведены для вспомогательных и управляющих сигналов.

Интерфейс стандарта 802.11 (Wi-Fi).

Wi-Fi — торговая марка организации Wi-Fi Alliance для локальных беспроводных сетей WLAN на базе спецификаций семейства IEEE 802.11. Под аббревиатурой Wi-Fi (от английского словосочетания Wireless Fidelity — «беспроводное качество») понимают целое семейство стандартов передачи цифровых потоков данных по радиоканалам.

IEEE 802.11 — набор стандартов связи для коммуникации в WLAN в частотных диапазонах 0,9; 2,4; 3,6 и 5 ГГц.

802.11 — изначальный 1 Мбит/с и 2 Мбит/с, 2,4 ГГц и ИК стандарт (1997).

802.11a — 54 Мбит/с, 5 ГГц стандарт (1999, выход продуктов в 2001).

802.11b — улучшения к 802.11 для поддержки 5,5 и 11 Мбит/с, 2,4 ГГц (1999).

802.11g — 54 Мбит/с, 2,4 ГГц стандарт (обратно совместим с b) (2003).

802.11n — увеличение скорости передачи данных (600 Мбит/с). 2,4 или 5 ГГц. Обратно совместим с 802.11a/b/g (сентябрь 2009).

802.11u — взаимодействие с не-802 сетями (например, сотовыми).

802.11v — управление беспроводными сетями.

802.11y — дополнительный стандарт связи, работающий на частотах 3,65–3,70 ГГц. Обеспечивает скорость до 54 Мбит/с на расстоянии до 5000 м на открытом пространстве.

802.11ac — Скорость передачи данных — до 6,77 Гбит/с для устройств, имеющих 8 антенн. (январь 2014).

802.11ad — новый стандарт с дополнительным диапазоном 60 ГГц (частота не требует лицензирования). Скорость передачи данных — до 7 Гбит/с.

802.11ah — стандарт, предназначенный для работы в диапазоне ~900 МГц. Предполагается, что он позволит обеспечить скорость до 40 Мбит/с. Основное назначение — интернет-вещей. (2007). В 2016 Wi-Fi Alliance анонсировал расширение стандарта, получившее название Wi-Fi HaLow

Первый стандарт 802.11 предусматривал два типа среды передачи: радиочастота 2,4 ГГц и инфракрасный диапазон 850–950 нм. ИК-устройства не были широко распространены и в будущем развития не получили. В диапазоне 2,4 ГГц было предусмотрено два способа расширения спектра:

- методом скачкообразного изменения частоты (FHSS)
- методом прямой последовательности (DSSS).

В первом случае все сети используют одну и ту же полосу частот, но с различными алгоритмами перестроения. Во втором случае появляются частотные каналы от 2412 МГц до 2472 МГц с шагом 5 МГц, сохранившиеся по сей день.

На смену 802.11 пришёл ***стандарт 802.11b,*** в котором скорость передачи данных была увеличена до 5,5; 11 и 22 (опционально) Мбит/с. Увеличение скорости было достигнуто путём уменьшения избыточности помехоустойчивого кодирования с 1/11 до 1/2 и даже 2/3 за счёт внедрения блочных и сверточных кодов. Кроме того, максимальное число ступеней модуляции было увеличено до 8 на символ (3 бита на 1 бод). Ширина канала и используемые частоты не изменились, но при уменьшении избыточности и увеличении глубины модуляции выросли требования к соотношению сигнал/шум. Из-за невозможности увеличения мощности устройств по причине экономии энергии мобильных устройств и законодательных ограничений,пришлось сократить зону обслуживания на новых скоростях. Площадь обслуживания на унаследованных скоростях 1–2 Мбит/с не изменилась. От способа расширения 2 из 21 спектра методом скачкообразной перестройки частоты было решено полностью отказаться. Больше в семействе Wi-Fi он не использовался.

Следующее увеличение скорости до 54 Мбит/с было реализовано в ***стандарте 802.11a***Увеличение скорости в основном было достигнуто за счёт увеличения глубины модуляции до 64 уровней на символ (6 бит на 1 бод). Кроме того, была радикально пересмотрена радиочастотная часть: расширение спектра методом прямой последовательности было заменено на расширение спектра методом разделения последовательного сигнала на параллельные ортогональные поденсущие (OFDM).

Стандарт 802.11g был утверждён в октябре 2002 года. Он стал компиляцией 802.11a и 802.11b в диапазоне 2,4 ГГц: в нём поддерживались скорости обоих стандартов. Этот стандарт предусматривает использование диапазона частот 2,4 ГГц, обеспечивая скорость соединения до 54 Мбит/с (при модуляции OFDM) и гарантируя обратную совместимость со стандартом 802.11b, для чего он поддерживает также режим модуляции DSSS (скорость при этом 11 Мбит/с).

В стандарте 802.11n (в обоих диапазонах 2,4 и 5 ГГц) скорость была увеличена до 72 Мбит/с за счёт уменьшения защитных интервалов между передаваемыми символами. Для увеличения пропускной способности можно было объединить два канала по 20 МГц и получить 150 Мбит/с. При этом в диапазоне 2,4 ГГц может поместиться всего один расширенный канал в 40 МГц. Также, согласно стандарту,если в диапазоне 2,4 ГГц на котором

используется канал удвоенной ширины появляется устройство, работающее на канале стандартной ширины, то устройство 802.11n обязано перейти на работу с каналом стандартной ширины. Соответственно, использовать каналы по 40 МГц рекомендуется только в диапазоне 5 ГГц. Для сосуществования каналов шириной 20/40 МГц точка доступа стандарта 802.11n должна переходить на другой канал или переключаться на использование канала шириной в 20 МГц, если соседняя точка доступа начинает передачу в одной из половин канала 40 МГц. Главным недостатком широких каналов является большее влияние на них помех и, соответственно, меньшее расстояние передачи данных.

Ещё одним способом повышения скорости стала ***технология MIMO:*** использование нескольких приёмопередатчиков, работающих на одной и той же частоте. Разделение каналов происходит за счёт пространственного разнесения антенн и математических операций над сигналом, принятым на разные антенны: он будет различаться в силу многолучевого распространения радиоволн. Стандарт 802.11n поддерживает MIMO 4x4:4 (четыре независимых канала) и обеспечивает скорость до 600 Мбит/с.

Стандарт 802.11ac предусматривает максимальную скорость до 6,93 Гбит/с, однако фактически такая скорость ещё не достигнута ни на одном оборудовании, представленном на рынке. Увеличение скорости достигнуто за счёт увеличения полосы пропускания до 80 и даже до 160 МГц. Такая полоса не может быть предоставлена в диапазоне 2,4 ГГц, поэтому стандарт 802.11ac функционирует только в диапазоне 5 ГГц. Технология MIMO была усовершенствована: впервые в стандарте Wi-Fi информация в одной сети может передаваться двум абонентам одновременно с использованием различных пространственных потоков.

Принцип действия

Обычно схема Wi-Fi сети содержит не менее одной точки доступа и не менее одного клиента. Принцип работы Wi-Fi базируется на использовании радиоволн, а сам обмен данными напоминает переговоры по радиосвязи. Обычно схема Wi-Fi-сети содержит не менее одной точки доступа и не менее одного клиента. Также возможно подключение двух клиентов, когда точка доступа не используется, а клиенты соединяются посредством сетевых адаптеров «напрямую». Адаптеры на каждом компьютере преобразуют цифровые данные в радиосигналы, которые передаются на другие сетевые устройства. Они же преобразуют входящие радиосигналы от внешних сетевых устройств в цифровые данные. Радиопередатчики и приемники одной Wi-Fi-сети работают на одних и тех же частотах и используют один и тот же вид модуляции данных в радиоволны.Wi-Fi-сети работают в специальных

диапазонах радиочастот «2,4» и «5» ГГц, которые зарезервированы в большинстве стран мира под так называемые нелицензируемые радиослужбы, то есть такие, которые можно использовать, не получая лицензию на радиостанцию.

Для подключения к сети необходимо знать идентификатор сети. Точка доступа передаёт его с помощью специальных сигнальных пакетов на скорости 0,1 Мбит/с каждые 100 миллисекунд. Поэтому 0,1 Мбит/с — наименьшая скорость передачи данных Wi-Fi. Зная идентификатор сети, клиент может выяснить, возможно ли подключение к данной точке доступа. При попадании в зону действия двух точек доступа с идентичными идентификаторами приёмник может выбирать между ними на основании данных об уровне сигнала. Стандарт Wi-Fi даёт клиенту полную свободу при выборе критериев для соединения.

Роутер получает трафик через сетевой кабель, преобразовывает его в радиоволны и распространяет их «по воздуху» в виде радиосигналов сверхвысокой частоты с определёнными параметрами. Приёмник «ловит» эти волны и декодирует их (расшифровывает, извлекает из них информацию, которая кодируется несущей частотой).

РЕФЕРАТЫ: «Параллельные порты», « Последовательные порты».

Тема 2.6: Режимы работы процессора.

Лекция 18. Режимы работы процессора. Адресация памяти реального режима.

Все 32-разрядные процессоры Intel (и совместимые с ними) начиная с 80386-го могут выполнять программы в нескольких режимах. Режимы процессора предназначены для выполнения программ в различных средах; в

разных режимах возможности МП неодинаковы, потому что команды выполняются по-разному.

В зависимости от режима процессора изменяется схема управления памятью системы и задачами. Процессоры могут работать в трех режимах:

- Реальный режим (16-разрядное программное обеспечение).
- Режим IA-32:
- • защищенный режим (32-разрядное программное обеспечение);
- • виртуальный реальный режим (16-разрядное программное обеспечение в 32-разрядной среде).
- Расширенный 64-разрядный режим IA-32e (также называемый AMD64, x86-64 и EM64T):
- • 64-разрядный режим (64-разрядное программное обеспечение);
- • режим совместимости (32-разрядное программное обеспечение).

Реальный режим. В первоначальном IBM PC использовался процессор i8086, который мог выполнять 16-разрядные команды, применяя 16-разрядные внутренние регистры, и адресовать только 1 Мбайт (220 байт) памяти, используя 20 разрядов для адреса. Все программное обеспечение PC первоначально было предназначено для этого процессора; оно было разработано на основе 16-разрядной системы команд и модели памяти объемом 1 Мбайт. Например, DOS, все программное обеспечение DOS, Windows от 1.x до 3.x и все приложения для Windows от 1.x до 3.x написаны в расчете на 16-разрядные команды.

Более поздние процессоры, например могли также выполнять те же самые 16-разрядные команды, что и первоначальный i8086, но намного быстрее..

Для программного обеспечения этого типа обычно используется однозадачный режим, т. е. одновременно может выполняться только одна программа. Нет никакой встроенной защиты для предотвращения перезаписи ячеек памяти одной программы или даже операционной системы другой программой; это означает, что при выполнении нескольких программ вполне могут быть испорчены данные или код одной из них, а это может привести всю систему к краху (или останову).

Защищенный режим. Несмотря на то, что процессор i80286, как и i8086, является 16-разрядным, он (в отличие от последнего) может работать в новом — защищенном — режиме и имеет аппаратурную поддержку многозадачных операционных систем, значительно ускоряющую и упрощающую процесс переключения задач.

Адресная шина была увеличена с 20 до 24 разрядов, что привело к расширению адресного пространства с 1 до 16 Мбайт (224 байт).

В защищенном режиме программа может записывать данные только в те области памяти, которые выделены ей операционной системой. Это повышает надежность работы мультизадачных и, в частности, мультипользовательских операционных систем.

Помимо расширения адресного пространства до величины в 4 Гбайта (232 байт) в них реализована концепция страничной виртуальной памяти, возможной только в защищенном режиме.

Механизм страничной виртуальной памяти позволяет разместить часть оперативной памяти на диске. При этом размер виртуальной памяти, предоставляемой программам, ограничивается размером свободного пространства на диске.

Виртуальный реальный режим. Этот режим реализуется в рамках защищенного режима (процессор может переключиться в виртуальный режим только из защищенного режима). В виртуальном режиме процессор способен выполнять программы, составленные для процессора i8086, находясь в защищенном режиме и используя аппаратные средства защищенного режима: мультизадачность, изолирование адресных пространств отдельных задач друг от друга, страничная виртуальная память.

Адресация памяти реального режима

Вы наверное знаете, что для работы с памятью используются две шины - шина адреса и шина данных. Физически память устроена таким образом, что возможна адресация как 16-битовых слов, так и отдельных байтов памяти.

В любом случае так называемый физический адрес передаётся из процессора в память по шине адреса. Ширина шины адреса определяет максимальный объём физической памяти, непосредственно адресуемой процессором. На рис. 1 показана схема взаимодействия процессора и памяти через шины адреса и данных.

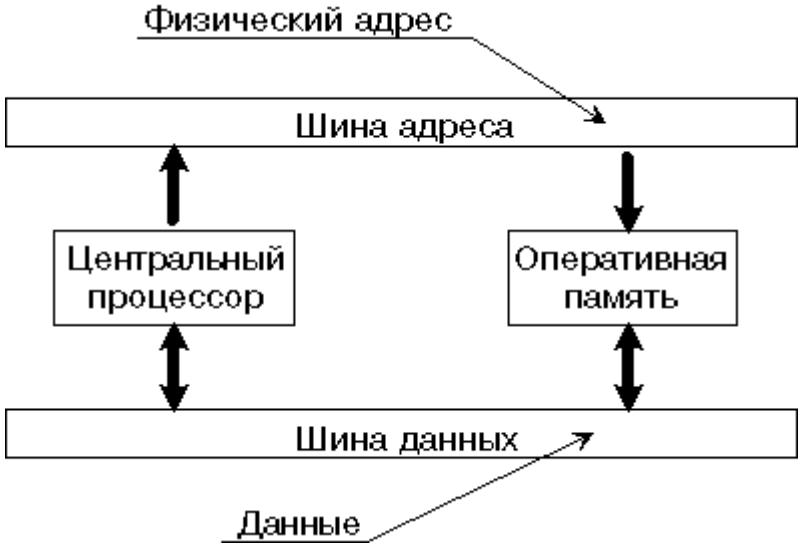


Рис. 1. Шина адреса и шина данных

Например, компьютер IBM XT оснащён 20-разрядной шиной адреса и 16-разрядной шиной данных. Это означает, что имеется возможность адресоваться к 216 байтам памяти, т.е. к 1 мегабайту памяти. Причём возможно адресоваться к байтам и словам размером в 16 бит.

Так как адреса принято записывать в шестнадцатеричной форме, то мы можем записать диапазон физических адресов для 20-разрядной шины адреса следующим образом:

00000h <= [физический адрес] <= FFFFFh

Таким образом, для представления физического адреса в компьютерах IBM PC и IBM XT используется двадцать двоичных или пять шестнадцатеричных разрядов.

Однако все регистры процессора i8086 являются 16-разрядными. Возникает проблема представления 20-разрядного физического адреса памяти при помощи содержимого 16-разрядных регистров.

Для разрешения этой проблемы используется двухкомпонентный логический адрес.

Логический адрес состоит из 16-разрядных компонент: компоненты сегмента памяти и компоненты смещения внутри сегмента.

Для получения 20-разрядного физического адреса к сегментной компоненте приписывается справа четыре нулевых бита (для расширения до 20 разрядов), затем полученное число складывается с компонентой смещения. Перед сложением к компоненте смещения слева дописывается четыре нулевых бита (также для расширения до 20 разрядов). Эту процедуру иллюстрирует рис. 2.

Рис. 2. Адресация памяти в реальном режиме.

Логический адрес принято записывать в форме <сегмент:смещение>.
Например, пусть у нас есть логический адрес 1234h:0123h. Сегментная компонента равна 1234h, компонента смещения - 0123h. Вычислим физический адрес, соответствующий нашему логическому адресу:

- расширяем до 20 бит сегментную компоненту, дописывая справа 4 нулевых бита, получаем число 12340h;
- расширяем до 20 бит компоненту смещения, дописывая слева 4 нулевых бита, получаем число 00123h;
- для получения физического адреса складываем полученные числа: 12340h + 00123h = 12453h.

Очевидно, что одному физическому адресу может соответствовать несколько логических. Фактически в схеме адресации памяти реального режима вся память как бы разбивается на сегменты.

Лекция 19. Основные понятия защищенного режима. Адресация в защищенном режиме. Дескрипторы и таблицы. Системы привилегий. Защита.

Основные понятия защищенного режима.

В защищенном режиме программа может записывать данные только в те области памяти, которые выделены ей операционной системой. В хорошо спроектированной операционной системе полностью исключает такую ситуацию, когда после запуска одним пользователем недостаточно отлаженной программы приходится перезапускать всю систему.

Следующие модели процессоров фирмы Intel — i80386, i80486 и i80586 (Pentium) были 32-разрядными. Помимо расширения адресного пространства до величины в 4 Гбайта (232 байт) в них реализована концепция страничной виртуальной памяти, возможной только в защищенном режиме.

Механизм страничной виртуальной памяти позволяет разместить часть оперативной памяти на диске. При этом размер виртуальной памяти, предоставляемой программам, ограничивается размером свободного пространства на диске.

Перечислим кратко основные преимущества, которые получает программа, работающая в защищенном режиме процессора:

· возможность непосредственной адресации памяти за пределами первого мегабайта;

· для процессоров i80x86 реализован механизм страничной виртуальной памяти, позволяющий программам работать с памятью, размер которой может быть много больше физической оперативной памяти, установленной в компьютере;

· аппаратная поддержка позволяет создавать на основе процессоров, работающих в защищенном режиме, высокопроизводительные системы.

Адресация в защищенном режиме.

Линейная адресация памяти — схема адресации памяти компьютера в защищенном режиме (начиная с Intel 80386 и других совместимых x86-процессорах). Используется большинством современных многозадачных ОС.
Благодаря механизму линейной адресации можно создавать любое (ограниченное только размерами оперативной памяти) количество независимых виртуальных *адресных пространств*. Причём каждая *страница* линейного адресного пространства может находиться по любому физическому адресу или даже быть выгруженной на диск.

В защищённом режиме, также как и в реальном, существуют понятия логического и физического адреса. Логический адрес в защищённом режиме (иногда используется термин "виртуальный адрес") состоит из двух 16-разрядных компонент - селектора и смещения. Селектор записывается в те же сегментные регистры, что и сегментный адрес в реальном режиме. Однако преобразование логического адреса в физический выполняется не простым сложением со сдвигом, а при помощи специальных таблиц преобразования адресов.
В первом приближении можно считать, что для процессора i80286 селектор является индексом в таблице, содержащей базовые 24-разрядные физические адреса сегментов. В процессе преобразования логического адреса в физический процессор прибавляет к базовому 24-разрядному адресу 16-разрядное смещение, т.е. вторую компоненту логического адреса.

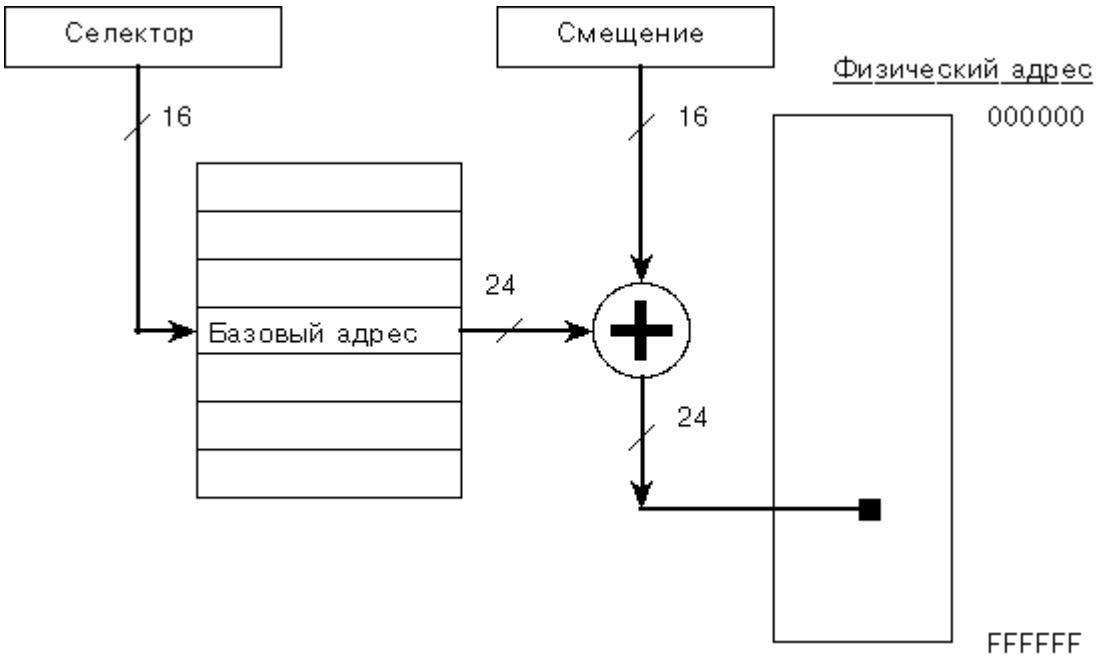


Рис. 4. Упрощённая схема преобразования логического адреса в физический в защищённом режиме.

Такая схема формирования физического адреса позволяет непосредственно адресовать 16 мегабайт памяти с помощью 16-разрядных компонент логического адреса.

Дескрипторы и таблицы

Заметьте, что селектор - это не сегментный адрес. Это индекс, с помощью которого процессор извлекает из специальной таблицы 24-разрядный базовый адрес сегмента. В реальном режиме мы имеем дело с сегментным адресом и смещением, а в защищённом - с селектором и смещением.

На самом деле не все 16 бит селектора используются для индексации по таблице базовых адресов. В качестве индекса выступают старшие 13 бит. Два младших бита (бит 0 и бит 1) используются системой защиты памяти, о чём мы подробно поговорим в следующем разделе. Бит 2 позволяет выбирать для преобразования адреса один из двух типов таблиц преобразования адресов.

 Полный формат селектора показан на рис. 5.

Рис. 5. Формат селектора адреса.

На этом рисунке два младших бита обозначены как RPL (Requested Privilege Level). Это поле является запрошенным программой уровнем привилегий. Поле TI (Table Indicator) состоит из одного бита. Если этот бит равен нулю, для

преобразования адреса используется так называемая глобальная таблица дескрипторов GDT (Global Descriptor Table), в противном случае - локальная таблица дескрипторов LDT (Local Descriptor Table).

Таблица дескрипторов - это просто таблица преобразования адресов, содержащая базовые 24-разрядные физические адреса сегментов и некоторую другую информацию. То есть каждый элемент таблицы дескрипторов (дескриптор) содержит 24-разрядный базовый адрес сегмента и другую информацию, описывающую сегмент.

Таблица GDT - единственная в системе. Обычно в ней находятся описания сегментов операционной системы. **Таблиц LDT** может быть много. Эти таблицы содержат описания сегментов программ, работающих под управлением операционной системы, т.е. отдельных задач. В каждый данный момент времени процессор может использовать только одну таблицу LDT.

Процессор имеет два регистра, предназначенных для адресации используемых в настоящий момент таблиц GDT и LDT. Регистр GTDR описывает расположение и размер таблицы GDT, а регистр LDTR содержит ссылку на использующуюся в настоящее время таблицу LDT.

На рис. 6 показана уточнённая схема преобразования адресов в защищённом режиме. Из рисунка видно, что регистры процессора GDTR и LDTR определяют расположение в памяти таблиц GDT и LDT соответственно. Таблицы GDT и LDT содержат дескрипторы, описывающие сегменты памяти. В этих дескрипторах, помимо другой информации (заштрихованная область) содержится 24-разрядный базовый адрес сегмента.

Старшие 13 битов селектора (индекс) выбирают элемент из таблицы GDT или LDT в зависимости от состояния бита TI селектора.

Извлечённый из таблицы дескрипторов базовый адрес сегмента складывается процессором для получения 24-разрядного физического адреса.

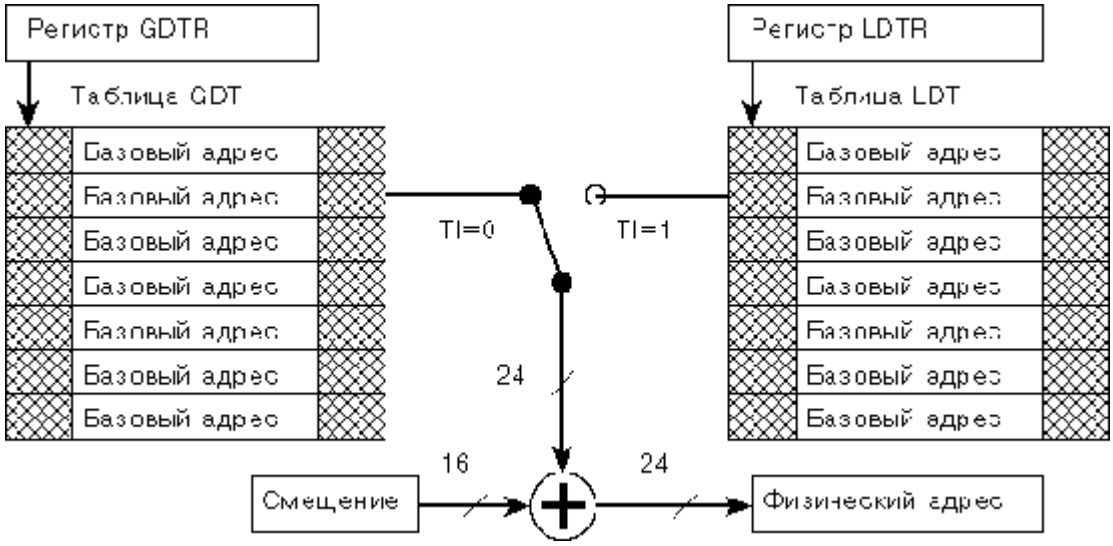


Рис. 6. Уточнённая схема преобразования адресов.

Селектор 0000h адресует самый первый дескриптор в глобальной таблице дескрипторов GDT. Поле RPL для этого дескриптора равно 0, поле TI также равно 0. Селектор 0008h указывает на второй элемент таблицы GDT, а селектор 0014h указывает на третий дескриптор в локальной таблице дескрипторов LDT, т.к. поле TI в нём равно 1.

Системы привилегий. Защита.

По мере повышения нашей зависимости от компьютеров пользователям необходимы все более надежные и защищенные системы, способные выполнять несколько задач одновременно.

Для того, чтобы прикладная программа пользователя не смогла разрушить систему, каждая группа программ выполняется на своем уровне привилегий.

Привилегия - это разрешение на выполнение конкретной операции над конкретным объектом или объектами. При этом ошибки в программах, имеющих низкий уровень привилегий, никак не отражаются на работе программ, работающих на более высоком уровне привилегий.

Простейшие уровни привилегий, реализованные в ЭВМ 60-х годов - это работа в режимах пользователя (User) и системы, или супервизора (Supervisor). Эта двухуровневая модель хорошо зарекомендовала себя в системах, когда одна большая ЭВМ (операционная система, процесс и т.п.) обслуживала нескольких равноправных пользователей. Двухуровневая система привилегий работает на компьютерах Apple вплоть до настоящего времени и в микропроцессорах x86 при страничной организации памяти.

С появлением современных технологий работы вычислительной техники двух уровней привилегий стало не хватать. Предвосхищая эту проблему, фирма Intel предложила в своем процессоре i80386 четырехуровневую систему привилегий на уровне сегмента.

На каждом уровне привилегий проверяется:

1. Может ли программа выполнить указанную подпрограмму?
2. К данным каких программ может обратиться та или иная программа?
3. Имеет ли программа право передавать управлению внешнему процессу и какому именно?

Рассмотрим накладываемые ограничения:

1. Привилегированные команды, управляющие сегментацией или влияющие на механизм защиты, могут работать только на нулевом уровне привилегий.

2. Программам не разрешается считывать/записывать элементы данных, которые имеют более высокий уровень привилегий. Однако программы могут использовать данные на своем и более низком уровне привилегий.
3. Передача управления внешним процедурам возможна, только если они имеют тот же уровень привилегий, что и исходный процесс.

Лекция 20. Переключение задач. Страничное управление памятью. Виртуализация прерываний. Переключение между реальным и защищенным режимами.

Страничное управление памятью.

Распределение памяти разделами.

Этот простейший способ управления оперативной памятью состоит в том, что память, незанятая ядром ОС, разбивается на несколько непрерывных областей фиксированной величины, называемых разделами.

В каждом разделе в каждый момент времени может располагаться по одному процессу. Очередной новый процесс, поступивший на выполнение, помещается либо в общую очередь, либо в очередь к некоторому разделу.

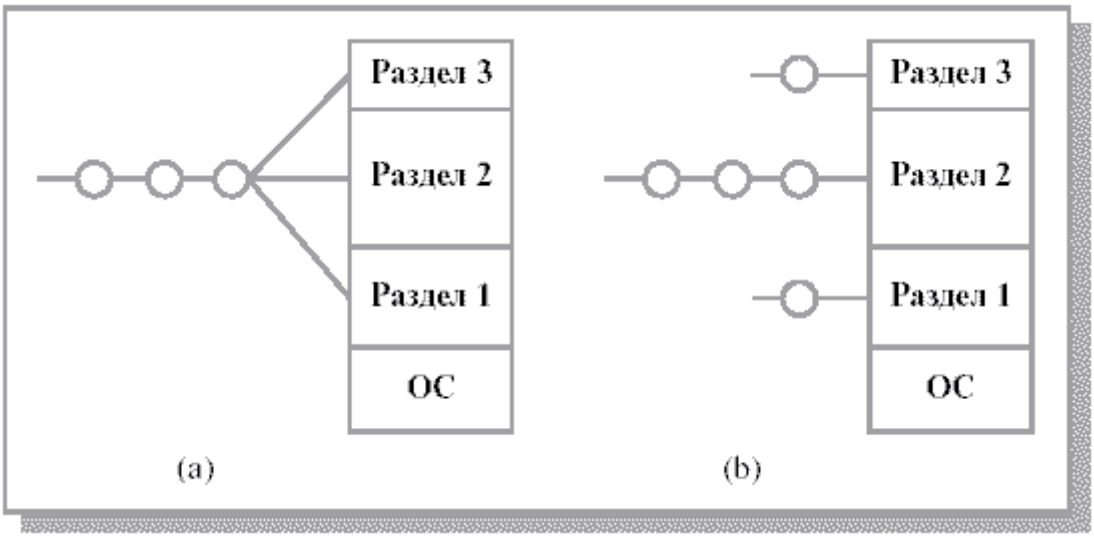


Схема с фиксированными разделами: (a) – с общей очередью процессов, (b) – с отдельными очередями процессов

Описанные выше схемы недостаточно эффективно используют память, поэтому в современных схемах управления памятью не принято размещать процесс в оперативной памяти одним непрерывным блоком.)

Логическое адресное пространство каждого процесса делится на части одинакового, фиксированного для данной системы размера, называемые виртуальными страницами.

Вся оперативная память также делится на части такого же размера, называемые физическими страницами (или блоками).

Для упрощения механизма преобразования адресов размер страницы обычно выбирается, равным степени двойки 2^k : 512, 1024 и т.д. Типичный размер страницы составляет несколько килобайт, например 4096 байт (4 Кбайт). Страницы не перекрываются.

Говорят, что оперативная память разбивается на физические страницы, а программа – на виртуальные.

В общем случае размер логического адресного пространства не является кратным размеру страницы, поэтому последняя страница каждого процесса дополняется фиктивной областью.

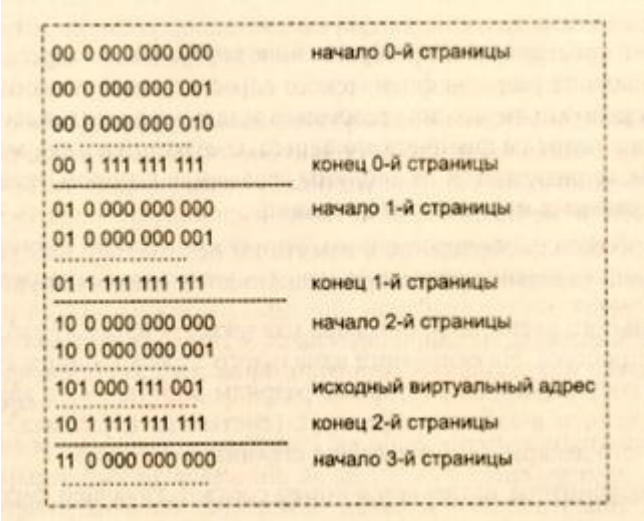
Передача информации между памятью и диском всегда осуществляется целыми страницами.

Логический адрес в страничной системе это упорядоченная пара (Nвс, Овс), где Nвс – порядковый номер страницы процесса в ЛАП, Овс - смещение в пределах страницы Nвс, на которой размещается адресуемый элемент.

Физический адрес - упорядоченная пара (Nфс, Офс), где Nфс – порядковый номер страницы процесса в ФАП, Офс - смещение в пределах страницы Nфс.

Из этого следует, что смещение Офс может быть получено простым отделением младших разрядов в двоичной записи адреса, а оставшиеся старшие разряды адреса представляют собой двоичную запись номера страницы.

Пусть размер страницы равен 1 Кбайт (2^{10}). Тогда из двоичной записи адреса 101 000 111 001 можно определить Nфс = 10, Офс = 1 000 111 001 (байт).



Двоичное представление адресов

Из рисунка видно, что номер страницы и ее начальный адрес легко могут быть получены один из другого дополнением или отбрасыванием нулей, соответствующих смещению.

Разбиение адресного пространства на страницы осуществляется ОС незаметно для пользователя. Поэтому адрес является двумерным лишь с точки зрения операционной системы, а с точки зрения пользователя адресное пространство процесса остается линейным.

При создании процесса ОС загружает в оперативную память несколько его виртуальных страниц, необходимых для старта процесса. Копия всего логического адресного пространства процесса находится на диске. Смежные виртуальные страницы не обязательно располагаются в смежных физических страницах.

Если текущей страницы в главной памяти нет, она должна быть переписана (подкачана) из внешней памяти.

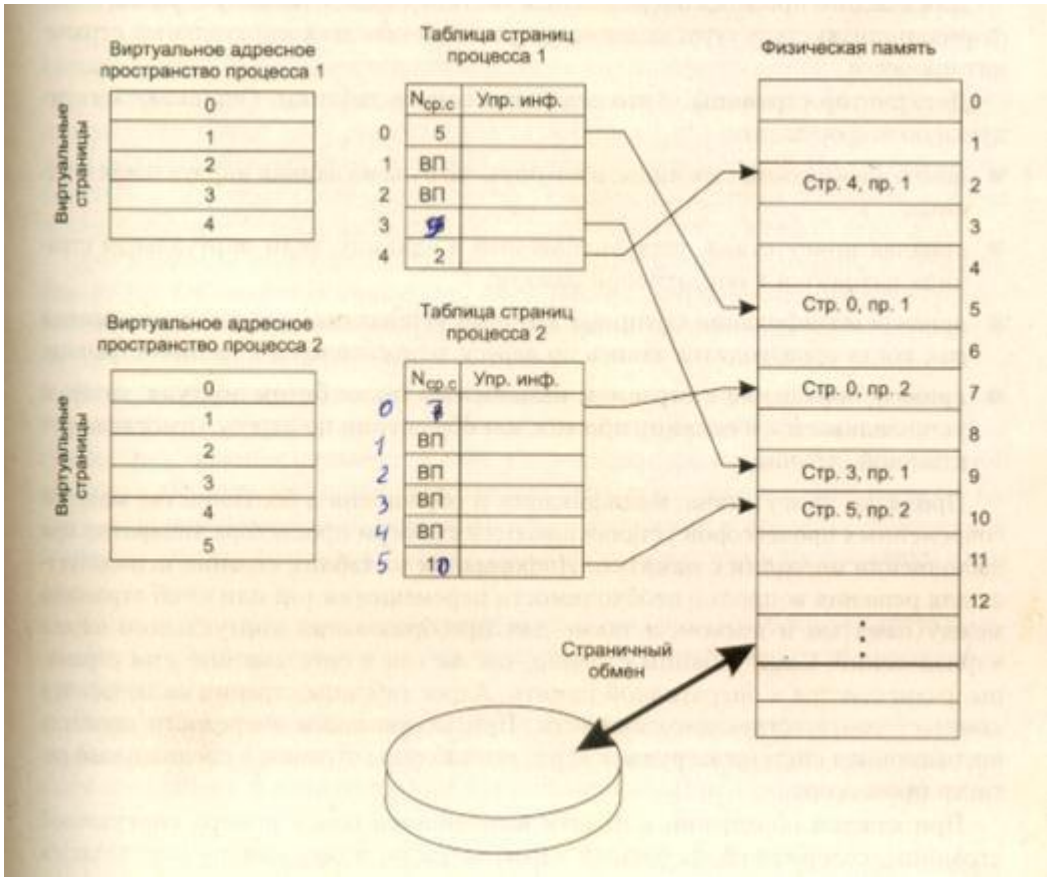


Схема страничного распределения памяти

Система отображения виртуальных адресов процессов в физические сводится к отображению виртуальных страниц в физические и представляет собой таблицу страниц. Для каждого процесса ОС создает информационную структуру - таблицу страниц. Каждая запись этой таблицы кроме служебной **(управляющей)** информации содержит номер физической страницы, в которую загружена соответствующая виртуальная страница.

Информация из таблицы страниц используется для решения вопроса о необходимости перемещения той или иной страницы между памятью и диском, а также для преобразования логического(виртуального) адреса в физический. Сами таблицы страниц, так же как и описываемые ими страницы, размещаются в оперативной памяти. Адрес таблицы страниц включается в контекст соответствующего процесса.

Реферат про ОС

Тема 2.7: Основы программирования процессора.

Лекция 21. Основы программирования процессора. Выбор и дешифрация команд. Обработка данных и их запись. Выработка управляющих сигналов.

Назначение, функции и состав операционных систем

Общение пользователя с ЭВМ осуществляется с помощью операционной системы (ОС). По мере развития вычислительной техники ОС развивались и совершенствовались, делая это общение более удобным для пользователя.

Операционная система представляет собой комплекс программ, обеспечивающих общее управление работой вычислительной машины и реализующих наиболее общие алгоритмы обработки информации. Основные функции ОС заключаются в упорядочении, сортировке, редактировании обрабатываемой и хранящейся в машине информации, трансляции прикладных программ, составленных на различных языках программирования, в воспринимаемые машиной (понятные ей) команды — машинные коды. Оператору (или пользователю) вычислительной машины ОС обеспечивает удобное обращение к различным разделам информации, к программам, решающим конкретные задачи, различным устройствам ввода—вывода, периферийным устройствам, компьютерным сетям.

Операционная система обеспечивает пользователю удобный способ общения (интерфейс) с устройствами компьютера.

При работе на компьютере используются программы трех категорий:

1) прикладные программы, непосредственно обеспечивающие выполнение необходимых пользователям работ (создание и редактирование текстов, создание и обработка графических изображений, обработка информационных массивов, создание электронных таблиц, математические вычисления, запись, воспроизведение и обработка музыки и т.д.);

2) системные программы, выполняющие различные вспомогательные функции (копирование и архивирование используемой информации, выдача справочной информации о компьютере, проверка работоспособности устройств компьютера, установка новых устройств и замена действующих, размещение информации в запоминающих устройствах и т.д.);

3) инструментальные программы, или системы, обеспечивающие создание новых программ для компьютера.

Четких границ между этими тремя категориями нет. Системные программы непрерывно совершенствуются, в них включаются дополнительные возможности, позволяющие выполнять задачи прикладного характера, например, редактировать тексты, выполнять расчеты на калькуляторе, создавать программы для часто выполняемых конкретным пользователем задач

(так называемые макрокоманды или макросы). Эти программы представляют собой удобную среду для оператора, не имеющего специальной подготовки для работы на вычислительной машине. В связи с широким распространением персональных компьютеров число таких операторов (их называют пользователями) вполне возможно уже достигло миллиарда. Операционные системы, усовершенствованные для огромного круга пользователей, получили название операционные оболочки.

Благодаря операционным оболочкам круг пользователей ЭВМ расширяется, поскольку не требуется наличие каких-то специфических знаний для общения с компьютером. Операционные оболочки обеспечивают пользователю более наглядные средства для выполнения часто использующихся действий. Это, прежде всего, широкий набор средств для вывода изображений (включая тексты) на экран, внесения различных изменений в эти изображения, построения меню, окон на экране и т.д. Кроме того, операционные оболочки предоставляют дополнительные возможности для запускаемых программ, позволяют одновременно выполнять несколько программ, т. е. обеспечивают режим мультипрограммирования, предоставляют расширенные средства для обмена между программами, упрощают создание графических программ.

Проследить за направлением совершенствования ОС можно на следующем примере. В одной из первых ОС для компьютеров MS- DOS (первая версия была создана в 1981 г. фирмой Microsoft) для вызова нужной информации необходимо было набирать на клавиатуре определенное сочетание букв. В пришедшей ей на смену системе Norton Commander достаточно было выделить на экране требуемую строку из предложенного набора. Сейчас наиболее популярной является система Windows, при пользовании которой курсор с помощью мыши устанавливается на условное изображение требуемого раздела информации и вызов выполняется значительно быстрее и проще. По мере своего совершенствования ОС требуют все большего объема памяти. Если раньше она измерялась килобайтами, то теперь требуются десятки мегабайт.

Операционная система освобождает пользователя от выполнения часто повторяемых типовых операций, таких как запуск и.завершение программ, запись и чтение информации на магнитных дисках, ввод информации с клавиатуры и вывод ее на экран, отсчет времени и количества информации, определение причин некоторых сбоев в работе аппаратуры. Все программы и данные хранятся на магнитных дисках в виде файлов. Английское слово file имеет несколько значений: очередь, ряд, картотека, регистратор (для бумаг). В вычислительной технике файл означает блок однотипной информации, которому присвоено определенное имя. Файл — это информация, которая

используется целиком. Его можно изменять, редактировать, при необходимости его можно объединять с другими файлами или даже разбивать на несколько файлов. Но работать с ним можно, только вызвав его целиком. Размер файла определяется количеством содержащейся в нем информации. Самые маленькие текстовые файлы имеют объем несколько байт, а самые большие — сотни килобайт и больше. Файл одной цветной фотографии может занимать около 10 Мбайт. Очень большой файл неудобен в использовании: он может не поместиться в запоминающем устройстве малой емкости (например, на гибком диске), много времени уходит на его вызов с жесткого диска.

Вся информация в ЭВМ состоит из некоторого, как правило, очень большого числа файлов. Для того чтобы облегчить поиск нужного файла, используется иерархическая (или древовидная) система. Операционная система должна знать, где находится тот или иной файл. Поэтому на каждом диске кроме самих файлов находятся еще несколько каталогов (или директорий, или папок), объединяющих группы файлов. Каталог также имеет имя, которое, как правило, отражает общее назначение файлов, которые данный каталог объединяет. В свою очередь несколько каталогов могут быть объединены в каталог более высокого уровня, которому присваивается определенное имя, и т.д. Каталог самого высокого уровня называется корневым каталогом. Вызывая с помощью ОС информацию о содержании жесткого диска, пользователь видит именно корневой каталог, который можно сравнить с оглавлением книги, где перечислены все разделы. Внутри каждого раздела имеются главы — это подкаталоги, т.е. каталоги более низкого уровня. Внутри главы есть параграфы, которые и являются аналогами файлов. Такую структуру называют древовидной, или деревом (рис. 15.1): от корня ЭВМ ствол ведет к крупным ветвям (дискам), от них отходят ветки (каталоги), а с ветками связаны листья (файлы).

Файловая система размещения информации была заложена уже в первых ОС. Эти системы обеспечивали управление взаимодействием процессора с устройствами ввода—вывода и всей информацией, размещаемой на дисках. Поэтому они назывались дисковыми операционными системами (ДОС).

Операционная система ДОС состоит из нескольких частей: базовой, резидентной и загружаемой.

Базовая система ввода—вывода (BIOS — Basic Input/Output System) находится в постоянной памяти (ПЗУ) компьютера. Эта часть ОС является неотъемлемой частью компьютера. Ее назначение состоит в выполнении наиболее простых и универсальных действий ОС, связанных с осуществлением ввода—вывода. Базовая система ввода—вывода содержит также тест функционирования компьютера, проверяющий работу памяти и других

устройств компьютера при включении его электропитания. Кроме того, BIOS (по-русски так и говорят — БИОС) содержит программу вызова загрузчика ОС. Таким образом, БИОС — это программы ОС, хранящиеся в постоянной памяти компьютера.

Вводя с помощью клавиатуры команды ДОС, пользователь может выполнять некоторые изменения в настройке отдельных узлов компьютера, обрабатывать свои файлы, запускать требуемые программы. В промежутках между выполнением запускаемых пользователем команд ДОС находится в состоянии ожидания команд пользователя. Когда компьютер выключен, ОС хранится на жестком магнитном диске в виде так называемых системных файлов. После включения компьютера системные файлы загружаются в оперативную память, где и находятся в течение всего времени работы компьютера. Эта часть ОС называется резидентной.

Часть программ ДОС остается на жестком диске. Они загружаются в оперативную память лишь по мере необходимости и освобождают ее после отработки. Эта часть ОС называется загружаемой.

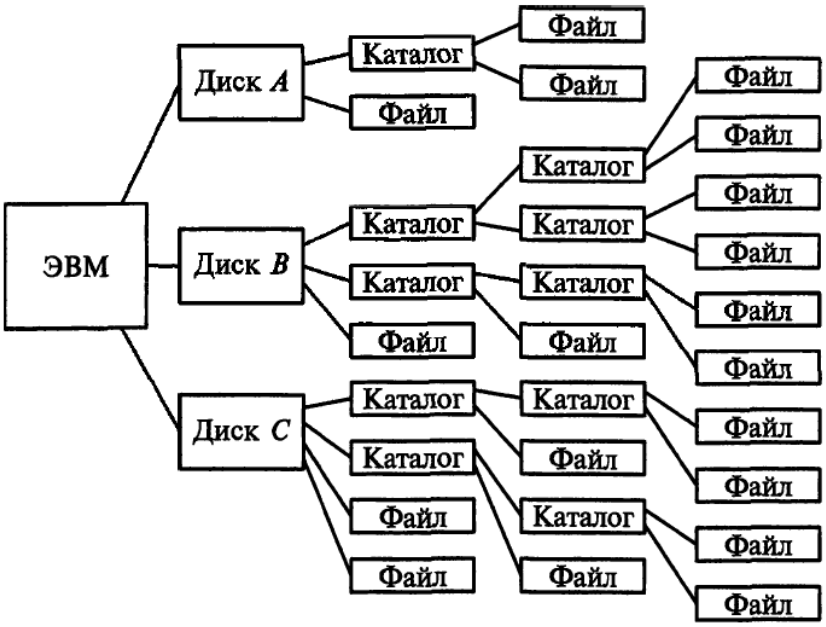


Рис. 15.1. Дерево файлов

Принципы программирования

Задачей программирования в широком смысле слова является составление самых разнообразных программ для ЭВМ, с помощью которых выполняются вычисления, обрабатывается информация, вырабатываются сигналы, управляющие ходом технологических и производственных процессов, и

осуществляются иные действия, которые облегчают умственную работу человека.

Действительно, работа ЭВМ во многом напоминает работу человеческого мозга. С помощью ЭВМ решаются сложнейшие интеллектуальные задачи. Благодаря огромному объему памяти и высокому быстродействию (по этим показателям вычислительная техника уже превзошла возможности не только обычных, но и самых одаренных людей) ЭВМ выполняет некоторые из таких задач более успешно, чем человек, но происходит это в соответствии с программой, которую составил человек. И хотя ЭВМ уже выигрывает в шахматы у чемпиона мира, нельзя сказать, что машина «умнее» человека. Есть целые области интеллектуальной деятельности человека, где основными факторами успеха являются талант, творческие способности, вдохновение.

Различают три основных вида программирования для вычислительной техники: ручное, автоматическое, системное.

Ручное программирование для ЭВМ заключается в составлении человеком программ на машинном языке. Машинный язык — это язык команд конкретной машины, на которой будет решаться данная задача. Каждая команда задает машине информацию об одной операции (например, сложении, умножении и т.д.), адреса исходных чисел и результата операции. Адресами являются номера ячеек памяти, в которых хранятся эти числа.

Последовательность команд называется ***программой.*** Команды выполняются в том порядке, в каком они написаны в программе, за исключением случаев, когда имеются так называемые команды перехода, указывающие номер команды в программе, к которой нужно перейти после проверки какого-либо условия.

Основными приемами программирования являются построение циклов, подпрограмм и модификация программ.

Модификация команд — это изменение адресов в команде, обеспечивающее применение данной команды для операции над величинами, находящимися в других ячейках памяти. Для решения часто встречающихся типовых задач или их отдельных частей составляют стандартные подпрограммы. Из них формируют библиотеку стандартных подпрограмм, которую используют при составлении программ для решения новых задач. Ответственным этапом программирования является так называемая ***отладка программ,*** заключающаяся в пробном решении на машине задач, результаты решения которых уже известны.

Перед написанием программы на языке машины составляют алгоритм задачи, определяющий общий ход вычислительного процесса, а также

распределение памяти для данных (исходных, промежуточных и окончательных) в запоминающих устройствах машины.

Первоначально ЭВМ предназначались для вычислений, т. е. для выполнения математических операций при решении задач. Понятие алгоритм относилось именно к последовательности выполнения математических операций. В настоящее время вычислительная техника применяется для обработки (преобразования) самой разнообразной информации, заданной не только в виде числовых величин, но и в виде графических образов, звуковых сигналов, а в более широком смысле — в виде любых сигналов, которые могут восприниматься как человеком, так и любым чувствительным прибором. Поэтому под алгоритмом теперь понимается последовательность действий, обеспечивающая преобразование объекта из исходного состояния в требуемое конечное. Преобразование заключается в выполнении нескольких простых операций, каждая из которых является командой. Алгоритм позволяет формализовать процесс достижения цели. Исполнитель (ЭВМ или человек) может выполнять алгоритм, не вникая в содержание поставленной задачи и даже ничего не зная о поставленной цели, поскольку при строгом следовании алгоритму задача все равно будет решена, а цель достигнут.

Для наглядности алгоритма используют его графическое изображение, которое называется схемой алгоритма.

На рис. 17.1 показана схема линейного алгоритма, по которому команды выполняются одна за другой. Так решаются простые задачи.

Во многих случаях линейная последовательность выполнения команд невозможна. В зависимости от полученного промежуточного результата выбирается та или иная последовательность действий. Схема такого алгоритма, называемого ветвлением, показана на рис. 17.2.

Рис. 17.1. Схема линейного алгоритма

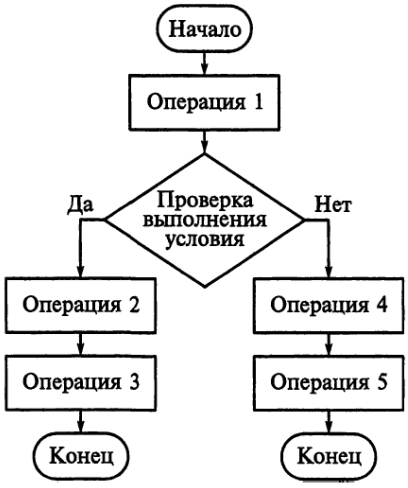


Рис. 17.2. Схема алгоритма типа ветвление

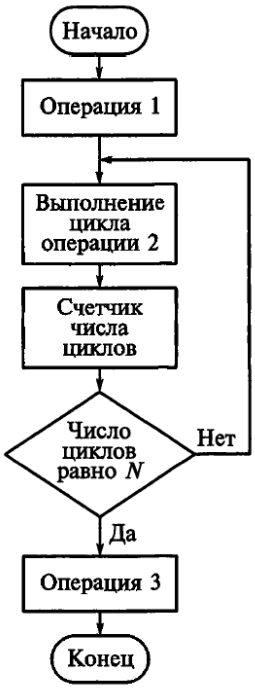


Рис. 17.3. Схема циклического алгоритма

Проверка промежуточного результата на соответствие заданному условию осуществляется в элементе, который изображен в виде ромба. Само условие записывается внутри ромба. При выполнении условия путь к следующей команде указывает стрелка с надписью «Да»; если условие не выполняется, то путь к следующей команде указывает стрелка с надписью «Нет». Если требуется выбирать более чем из двух вариантов, на схеме используют большее число ромбов, но из каждого ромба могут выходить только две стрелки (да или нет, третьего не дано).

При решении некоторых задач порой требуется выполнение какой-либо команды (илигруппы команд) многократно. При этом используется цикличный алгоритм, схема которого показана на рис. 17.3. Переход к новой операции происходит после достижения заданного числа циклов N. Возможен выход из цикла и после того, как результат будет соответствовать поставленному условию.

Лекция 22. «Основные команды процессора: арифметические и логические команды перемещения, сдвига, сравнения, команды условных и безусловных переходов, команды ввода/вывода.»

Основные типы устройств ввода—вывода

Устройства ввода—вывода предназначены для ввода исходных данных и программ в ЭВМ и для вывода результатов обработки информации. Кроме того, УВВ выполняют необходимые при этом преобразования данных из одной формы представления в другую. Поэтому при вводе информации от устройств с непрерывным (аналоговым) сигналом используются аналого-цифровые преобразователи, которые были рассмотрены в гл. 7. Соответственно при выводе результатов на аналоговое устройство используются цифроаналоговые преобразователи, также описанные в гл. 7.

Для непосредственного ввода данных в компьютер чаще всего используется клавиатура (рис. 14.1). С ее помощью вводится символьная информация, состоящая из букв и цифр, а также знаков препинания и некоторых специальных знаков. При нажатии клавиш происходит замыкание контактов. Клавиш, т. е. контактных элементов, более 100, и надежность (вероятность безотказной работы) клавиатуры в очень большой степени зависит от конструкции каждого из них.

Человек в быту и на производстве очень часто использует самые различные кнопки для включения электрических приборов. Разработано много разных кнопок и собрано много данных об ихнадежности, которая в значительной степени определяется материалом контактов. Для кнопок с серебряными контактами гарантируется 4 млн срабатываний. Если бы такие довольно дорогие кнопки использовались в клавиатуре, с которой вводятся данные (цифровые и текстовые), то клавиатура отказывала бы очень часто. Возьмем для примера среднюю скорость набора данных 200 знаков в минуту (кстати, квалифицированная машинистка набирает текст со скоростью в полтора раза большей). Некоторые буквенные клавиши используются чаще, чем другие. Для цифровых данных можно считать равновероятным появление любой из десяти цифр. Примем для простоты, что вероятность нажатия одной и той же клавиши равна 0,1. Это означает, что за час работы один и тот же контактный элемент срабатывает 200 х 60 х 0,1 = 1 200 раз. Поделив 4 млн на 1 200, получим примерно 3 тыс. часов наработки на отказ одной клавиши. А ведь их, как уже говорилось, более сотни. Значит, такая клавиатура выходила бы из строя каждые два месяца. Поэтому в клавиатуре компьютера необходимо применять контактные элементы очень высокой надежности. Пока для этой

цели используются разные конструкции. В одной, например, контакты 1 (рис. 14.2), изготовленные методом печатного монтажа, замыкаются проводящей шайбой 3. Шайба закреплена на пластиковом мостике 2, который прогибается (деформируется) при нажатии пальцем на клавишу 4, а после прекращения нажатия возвращается в исходное положение. В другой конструкции посеребренные шайбы размещаются на пленке, расположенной поверх печатной платы клавиатуры. Перспективно применение бесконтактных клавиш на герконах (герметизированных контактах из упругого ферромагнитного материала, которые замыкаются под влиянием магнитного поля), и особенно сенсорных контактов, работа которых основана на изменении емкости конденсатора при прикосновении пальца.

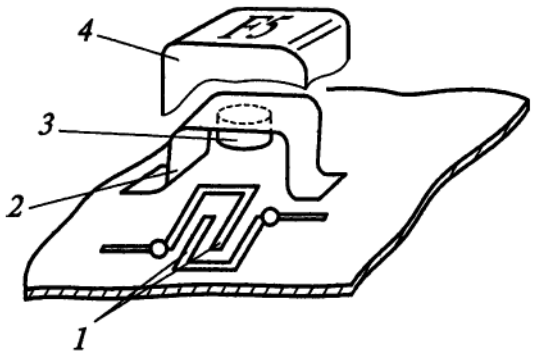


Рис. 14.2. Контактный элемент клавиши:

1 — контакты; *2* — пластиковый мостик; *3* — шайба; *4* — клавиша

Часть клавиш клавиатуры предназначена для перемещения курсора на экране монитора. Для более эффективного выполнения этого действия используется специальный манипулятор, который получил название «мышь» (рис.

14.3). При перемещении мыши ладонью по поверхности рабочего стола (а лучше — по поверхности специального коврика) вращается шарик 1, который через ролик 6 передает вращение на диск 4 с прорезями. Поворот этого диска и преобразуется вэлектрический сигнал с помощью фотодатчика, состоящего из осветителя 5 и фотоприемника 3. Мышь имеет также две или три кнопки управления 2. Манипуляторы такого назначения выпускаются разными фирмами и могут иметь разное конструктивное исполнение.

Все большее распространение получают устройства ввода с помощью сенсорных устройств, чувствительных к прикосновению пальца пользователя. Например, справочную информацию на вокзале, в магазине, в библиотеке клиент может получить, прикасаясь пальцем к экрану монитора, на котором отмечены различные разделы справочной системы. Этот монитор имеет специальный сенсорный экран. Для портативных компьютеров, на которых приходится работать в условиях ограниченного пространства, когда применение мыши оказывается неудобным, можно использовать сенсорную панель (рис. 14.4, а).

Принцип действия сенсорных устройств ввода поясняет рис.

14.4, б. Металлические проводники 1, разделенные тонкой изолирующей прокладкой из лавсановой пленки 2, образуют сетку, которая представляет

собой набор очень большого числа маленьких конденсаторов. Так как человеческое тело является хорошим проводником, то при приближении руки к поверхности сенсорной панели (или экрана) происходит изменение электрического поля, а следовательно, и емкости этих конденсаторов. Измеряя изменение емкости каждого конденсатора в сетке, можно точно определить координаты пальца на поверхности панели (экрана) и даже приблизительно оценить давление, оказываемое на панель.

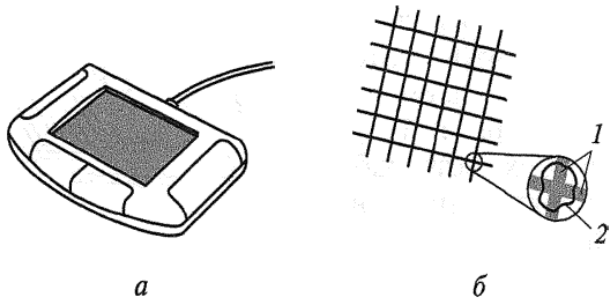


Рис. 14.4. Сенсорная панель:
***а* — внешний вид; *б* — принцип действия**

Таким образом, прижав палец к сенсорной панели и передвигая его по ее поверхности, пользователь может перемещать курсор так же, как и с помощью мыши. Как и мышь, сенсорная панель может иметь кнопки для выбора действия после установки курсора, но возможно сделать выбор непосредственным нажатием пальца на сенсорную панель, поскольку емкость конденсатора зависит и от давления пальца.

Для непосредственного ***ввода текстовой информации*** используются автоматические читающие устройства разной степени сложности. Специализированные устройства разработаны для прочтения почтового индекса на конвертах, считывания штрихового кода на отдельных изделиях и бланках документов, меток различных опросных листов и т.п.

Наиболее универсальным средством ввода текстовой и графической информации является сканер, который преобразует видеоинформацию на бумаге (текст, рисунки, слайды, фотографии) в электрический сигнал. С помощью специальной программы компьютер анализирует этот сигнал, определяет соответствие отдельных его элементов определенным символам (т.е. буквам, цифрам, знакам препинания) и воспринимает как текстовый файл, который может быть обработан, отредактирован и записан на диск.

В системах автоматического проектирования для ввода контурных изображений большого формата (чертежей, карт, схем) используются графические планшеты.

Неотъемлемой частью любого сканера являются АЦП, предназначенные для преобразования непрерывно изменяющихся значений напряжения,

получаемых с помощью ПЗС или ФЭУ, в числа, соответствующие оттенкам цвета или градациям серого. Качество сканирования напрямую связано с разрядностью используемого в сканере АЦП. В черно-белых (двухуровневых) сканерах аналогичное преобразование выполняет компаратор, сравнивая зафиксированное значение напряжения с опорным.

Печатающие устройства

Самым распространенным печатающим устройством является принтер. Существует большое разнообразие принтеров, отличающихся конструкцией, характеристиками, принципом действия. Принтеры печатают и текстовые, и графические материалы, т.е. рисунки и схемы.

Для вывода на бумажный носитель крупногабаритной графической информации (чертежи, географические карты, большие электрические схемы и т. п.) используются графопостроители, или плоттеры. Скорость печати у них несколько ниже, чем у хороших принтеров, но их применение зачастую оказывается экономичнее, чем применение для аналогичных целей хорошего принтера. Для изготовления рекламных материалов используются устройства, позволяющие отпечатать изображение размером в десятки метров.

По принципу действия различают матричные, струйные, лазерные, светодиодные и термические принтеры.

Лазерные принтеры. Большинство изготовителей лазерных принтеров использует механизм печати, который применяется в ксероксах. Например, в принтерах HP и QMS используется механизм печати ксероксов Canon.

Основным конструктивным элементом лазерного принтера является вращающийся фотонаборный барабан 8 (рис. 14.10), с помощью которого изображение переносится на бумагу. Барабан представляет собой металлический цилиндр, покрытый тонкой пленкой светопроводящего полупроводника. Обычно в качестве такого полупроводника используется оксид цинка. По поверхности барабана равномерно распределяется статический заряд. Для этого служит тонкая проволока или сетка 11, называемая корони- рующим проводом. На этот провод подается высокое напряжение, вызывающее возникновение вокруг него светящейся ионизированной области, называемой короной. Лазер 5, управляемый микроконтроллером, генерирует тонкий световой луч, отражающийся от вращающегося зеркала 7.

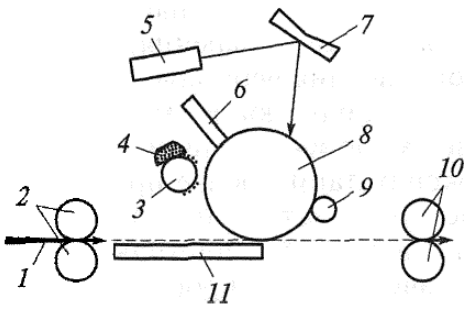


Рис. 14.10. Функциональная схема лазерного принтера:

1 — бумага; *2* — валики; *3* — барабан-девелопер; *4* — резервуар с тонером; *5* — лазер; *6* — провод разряда; *7* — зеркало; *8* — фотонаборный барабан; *9* — ролик очистки; *10* — фиксирующие ролики; *11* — коронирующий провод

Этот луч, приходя на фотонаборный барабан 8, изменяет его электрический заряд в точке

прикосновения. У некоторых типов принтеров потенциал поверхности барабана в таких точках уменьшается с 900 до 200 В. Таким образом, на барабане возникает скрытая копия изображения.

На следующем этапе на фотонаборный барабан 8 наносится тонер — мельчайшая красящая пыль. Под действием статического заряда частицы тонера притягиваются к поверхности барабана в точках, подвергшихся облучению, и формируют изображение. Бумага 1 втягивается с подающего лотка и с помощью валиков 2 перемещается к барабану. Перед самым барабаном бу-маге сообщается статический заряд. Затем бумага соприкасается с барабаном и благодаря своему заряду притягивает частицы тонера с барабана.

Для фиксации тонера бумага вновь заряжается и пропускается между двумя фиксирующими роликами 10, имеющими температуру около 180 °С. После передачи изображения на бумагу барабан полностью разряжается, очищается от прилипших частиц тонера и готов для нового процесса печати.

В цветном лазерном принтере изображение формируется на светочувствительной фотоприемной ленте последовательно для каждого цвета (циан — cyan, пурпурный — magenta, желтый — yellow, черный — black). Лист печатается за четыре прохода, что сказывается на скорости печати, имеются четыре резервуара для тонеров, выполняются несколько циклов проявления. Принтеры такого класса оборудованы имеющей большой объем памятью, процессором и, как правило, собственным винчестером.

Светодиодные принтеры. Альтернативой лазерному принтеру является так называемый светодиодный принтер, или LED-принтер (Light Emitting Diode). Вместо лазерных лучей, управляемых с помощью механизма зеркал, барабан освещает неподвижная диодная линейка, состоящая из 2 500 светодиодов, которая выделяет не каждую точку, а целую строку на сканируемой странице. Светодиодные принтеры (в отличие от лазерных) не выделяют озон, но качество отпечатков у них хуже, чем у лазерных.

Термические принтеры. Конструкция цветных лазерных принтеров еще не достигла совершенства. Для получения цветного изображения с качеством, близким к фотографическому, или изготовления допечатных цветных проб иллюстраций журналов и альбомов используются термические принтеры.

В настоящее время распространение получили три технологии цветной термопечати:

1) струйный перенос расплавленного красителя (термопластичная печать);

2) контактный перенос расплавленного красителя (термовосковая печать);

3) термоперенос красителя (сублимационная печать).

Общим для последних двух технологий является нагрев красителя и перенос его на бумагу (пленку) в жидкой или газообразной фазе.

Устройства отображения информации

Для отображения информации в вычислительной технике широко используются индикаторные устройства. Чаще всего применяются оптические индикаторы, поскольку с помощью зрения человек получает более трех четвертей всего воспринимаемого им объема информации. Кроме оптических используются звуковые (акустические) индикаторы. Например, для сигнализации об аварийных ситуациях наиболее приемлемы звуковые сигналы в сочетании с привлекающими внимание персонала световыми (т. е. оптическими) сигналами.

Рассмотрим оптические индикаторные устройства, в дальнейшем называемые для краткости индикаторами. Различают активные и пассивные оптические индикаторы. К активным относятся лампы накаливания, газоразрядные приборы, кинескопы и другие устройства, излучающие свет в видимой части спектра. К пассивным индикаторам относятся те устройства, которые сами не излучают свет, а лишь отражают свет внешних источников. К ним относятся шкалы измерительных приборов, цифровые индикаторы (например, счетчика электроэнергии в квартире или счетчика километров на приборной панели автомашины), жидкокристаллические индикаторные панели.

Одним из наиболее простых и распространенных активных индикаторов является светодиод. Принцип его действия основан на том, что при протекании прямого тока через полупроводниковый диод происходит излучение фотонов (т.е. световой энергии). Кремниевые и германиевые диоды излучают в невидимом глазом диапазоне длин волн, а вот диоды на основе арсенида-фосфида галлия (GaPAs) — в диапазоне длин волн 0,58...0,65 мкм. Такое излучение человек воспринимает как желтый (0,58 мкм), оранжевый (0,63 мкм) или красный (0,65 мкм) свет. Светодиод, изготовленный на основе фосфида галлия (GaP), излучает зеленый свет (0,56 мкм), а на основе арсенида галлия (GaAs) — инфракрасный (0,90 мкм), который хотя и является невидимым для человеческого глаза, но удобен для дистанционного управления объектами. В зависимости от числа и пропорций примесей можно изменять длину волны максимума излучения, т. е. цвет свечения светодиода.

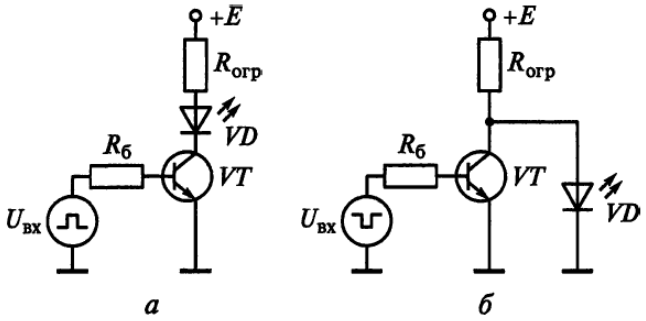


Рис. 14.11. Схемы включения светодиодов:
а — для высокого уровня входного сигнала $U_{вх}$; *б* — для низкого уровня

Быстродействие светодиодов очень высокое: при подаче скачкообразного входного сигнала яркость диода изменяется за сотую долю миллисекунды.

Входным сигналом для светодиода является прямой ток. От его величины зависит яркость свечения. Хорошая видимость даже при дневном свете обеспечивается при прямом токе от 5 до 20 мА. При этом напряжение на светодиодах составляет 2...3 В. Светодиоды по своим параметрам хорошо согласуются с транзисторными и интегральными схемами. На рис. 14.11 показаны схемы включения светодиодов VD с помощью транзисторного ключа. Поскольку транзистор VT обладает усилительными свойствами, ток, поступающий от источника сигнала напряжением Um через резистор базы Д, в десятки раз меньше прямого тока светодиода.

Сопротивление резистора $R_{огр}$, ограничивающего прямой ток светодиода, подсчитывается по формуле

$$R_{огр} = (E - U_{VD})/I_{VD}.$$

Светодиоды выпускаются в точечном, линейном и цифрознаковом исполнениях. Среди цифрознаковых наибольшее распространение получили семисегментные цифровые светодиодные индикаторы (рис. 14.12). Стилизованное изображение цифры составляется из семи светодиодных сегментов (a, Ь, c, d, e, /, g), расположенных в виде цифры 8. При подаче сигналов на определенные сегменты высвечивается требуемая цифра. Например, для высвечивания цифры 5 необходимо подать сигналы на сегменты a, f g, c, d. Из нескольких таких индикаторов может быть составлен многоразрядный индикатор.

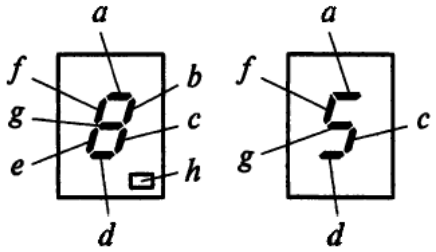


Рис. 14.12. Семисегментный цифровой светодиодный индикатор

При этом для отделения целой части десятичного числа от дробной используется сегмент А.Линейный

светодиодный индикатор представляет собой интегральную схему в виде светящегося столбика, образованного последовательно включенными светодиодными сегментами, и блока управления. Внешне такой индикатор выглядит как линейная шкала. Он служит для отображения непрерывно меняющейся информации и является аналогом стрелочного измерительного прибора. Подобные устройства используются в многоканальных системах для индикации однотипной информации. Несколько расположенных рядом линейных шкал очень удобны для восприятия оператором.

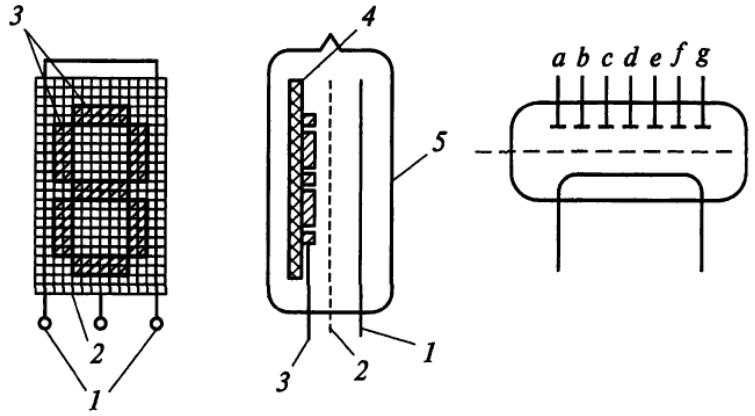


Рис. 14.13. Люминесцентный индикатор:
1 — катод; *2* — сетка; *3* — аноды; *4* — подложка; *5* — стеклянный корпус

Люминесцентный индикатор также относится к типу активных. Он представляет собой электронную вакуумную лампу с катодом 1 (рис. 14.13), управляющей сеткой 2 и несколькими анодами 3 на подложке 4. Аноды покрыты слоем люминофора, который светится, если на него попадает поток электронов, испускаемых катодом. Катод выполнен в виде двух тонких вольфрамовых нитей, натянутых параллельно анодам. Между катодом и анодами находится плоская сетка. На катод подается напряжение накала, он нагревается и испускает поток электронов. На сетку и аноды подаются положительные (по отношению к катоду) напряжения. Поток электронов из катода устремляется к положительно заряженной сетке, пролетает ее по инерции и попадает в ускоряющее поле тех анодов, на которые подано напряжение. Когда разогнавшиеся до большой скорости электроны достигают анодов, их кинетическая энергия переходит в световую энергию излучаемых люминофором квантов света (как и в обычной электронно-лучевой трубке).

К типу пассивных относится ***жидкокристаллический индикатор.*** Считывание с него информации возможно лишь при наличии внешнего освещения — естественного или искусственного. Принцип действия таких индикаторов основан на изменении степени прозрачности органических жидкокристаллических веществ, находящихся в электрическом поле.

Рис. 14.14. Жидкокристаллический индикатор:
1 — стеклянные пластины; *2* — прокладка; *3* — электроды-сегменты; *4* — прозрачный электрод

Конструктивно жидкокристаллический индикатор (рис. 14.14) ВЫПОЛНен в виде двух плоских кладка; разделенных по периметру прокладкой 2. На внутреннюю поверхность одной пластины наносят прозрачные проводящие электроды-сегменты 3, форма и взаимное расположение которых будут определять индицируемые знаки. На всю вторую пластину 1 наносят проводящий прозрачный электрод 4. Пространство между пластинами заполняют жидкокристаллическим веществом, толщина слоя которого составляет примерно 10 мкм. Собранный таким образом пакет из стеклянных пластин, электродов и жидкого кристалла герметизируют. Выводы от электродов проходят через герметик. Для управления индикатором между общим электродом и электродами-сегментами подается напряжение 5...15 В

Большие экраны, созданные с помощью плазменной технологии, позволяют весьма значительно уменьшить размеры монитора. Плоский (толщиной менее 10 см) и легкий ***монитор с плазменным экраном*** можно разместить на стене или под потолком, что особенно удобно для оснащения компьютерных справочноинформационных систем. Основное достоинство плазменного экрана заключается в том, что каждая его светящаяся точка (пиксел) сразу окрашивается в красный, зеленый и голубой цвета, что в несколько раз сокращает размеры носителя цветовоспроизведения. Заряженные электроды, расположенные между стеклянными пластинами, вызывают появление мельчайших ячеек инертного газа, служащего для изменения состояния плазмы.

Принцип действия элементарной ячейки плазменного экрана поясняет рис. 14.16. Для того чтобы ячейка стала светиться, на соответствующие электроды 1, в точке пересечения которых эта ячейка находится, подается высокое переменное напряжение. Газ в ячейке переходит в состояние плазмы, т. е. его атомы разделяются на ионы и электроны. В зависимости от полярности напряжения они попеременно собираются у электродов по разные стороны (электроны — там, где «плюс», ионы — там, где «минус») ячейки. Для

«зажигания» на сканирующий электрод 7 подается импульс, одноименные потенциалы складываются и напряженность электростатического поля удваивается. Происходит разряд — часть ионов отдает энергию в виде излучения квантов света в ультрафиолетовом диапазоне. В свою очередь, флуоресцирующее покрытие 2 ячейки начинает излучать свет в видимом диапазоне, который и воспринимает наблюдатель. Каждый пиксел состоит из трех таких ячеек: красной, зеленой и голубой. Почти все вредное для глаз ультрафиолетовое излучение поглощается наружным стеклом 3. Яркость свечения каждой ячейки определяется величиной управляющего напряжения, а цвет пиксела является результатом смешения трех цветов разной яркости.

Плазменные экраны превосходят обычные по яркости более чем в два раза, а по контрастности — в 10 раз. По всей рабочей поверхности экрана сохраняется высокая четкость изображения. В отличие от мониторов с электронно-лучевыми трубками мониторы с плазменными экранами не подвержены помехам от электромагнитных полей. Это очень важное свойство плазменных экранов, позволяющее использовать их на промышленных предприятиях

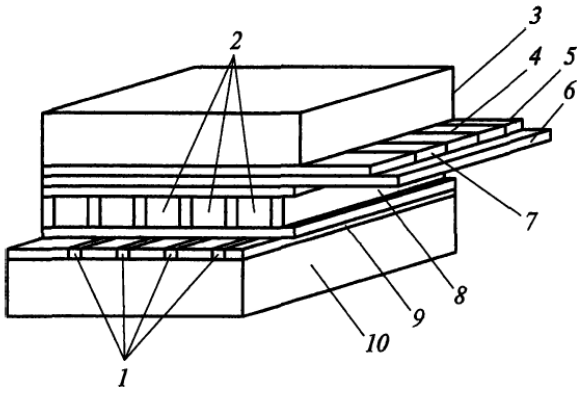


Рис. 14.16. Ячейка плазменного экрана:

1 — управляющие электроды; *2* — флуоресцирующее покрытие ячеек; *3* — наружное стекло; *4* — питающий электрод; *5*, *9* — слои диэлектрика; *6* — защитный слой; *7* — сканирующий электрод; *8* — разделительная перегородка; *10* — внутреннее стекло

Лекция 23. Подпрограммы. Виды и обработка прерываний. Этапы компиляции исходного кода в машинные коды и способы отладки. Использование отладчиков.

Подпрограмма (процедура) — поименованная или иным образом идентифицированная часть компьютерной программы, содержащая описание определённого набора действий. Подпрограмма может быть многократно *вызвана* из разных частей программы.

Назначение: Подпрограммы изначально появились как средство оптимизации программ по объёму занимаемой памяти — они позволили не повторять в программе идентичные блоки кода, а описывать их однократно и вызывать по мере необходимости. К настоящему времени данная функция подпрограмм стала вспомогательной, главное их назначение — структуризация программы с целью удобства её понимания и сопровождения.

Виды:

В языках программирования высокого уровня используется два типа подпрограмм: процедуры и функции.

- **Функция** — это подпрограмма специального вида, которая, кроме получения параметров, выполнения действий и передачи результатов работы через параметры имеет ещё одну особенность— она *всегда должна возвращать результат*.
- **Процедура** — это независимая именованная часть программы, которую после однократного описания можно многократно вызвать по имени из последующих частей программы для выполнения определенных действий.

Виды и обработка прерываний.

Прерывание— сигнал, сообщающий процессору о наступлении какого-либо события. При этом выполнение текущей последовательности команд приостанавливается, и управление передаётся обработчику прерывания, который реагирует на событие и обслуживает его, после чего возвращает управление в прерванный код.

В зависимости от источника возникновения сигнала прерывания делятся на:

- ***асинхронные, или внешние (аппаратные)*** — события, которые исходят от внешних источников (например, периферийных устройств) и могут произойти в любой произвольный момент: сигнал от таймера, сетевой карты или дискового накопителя, нажатие клавиш клавиатуры, движение мыши.
- ***синхронные, или внутренние*** — события в самом процессоре как результат нарушения каких-то условий при исполнении машинного кода: деление на ноль

или переполнение стека, обращение к недопустимым адресам памяти или недопустимый код операции;

- ***программные*** (частный случай внутреннего прерывания) — инициируются исполнением специальной инструкции в коде программы. Программные прерывания как правило используются для обращения к функциям встроенного программного обеспечения, драйверов и операционной системы.

Механизм обработки прерываний независимо от архитектуры вычислительной системы включает следующие элементы:

1. Установление факта прерывания (прием сигнала на прерывание) и идентификация прерывания.
2. Запоминание состояния прерванного процесса. 3. Управление аппаратно передаётся подпрограмме обработки прерывания.
4. Сохранение информации о прерванной программе, которую не удалось спасти с помощью действий аппаратуры. В некоторых вычислительных системах предусматривается запоминание довольно большого объёма информации о состоянии прерванного процесса.
5. Обработка прерывания. Эта работа может быть выполнена той же подпрограммой, которой было передано управление, но в ОС чаще всего она реализуется путем последующего вызова соответствующей подпрограммы.
6. Восстановление информации, относящейся к прерванному процессу (этап, обратный шагу 4).
7. Возврат в прерванную программу.

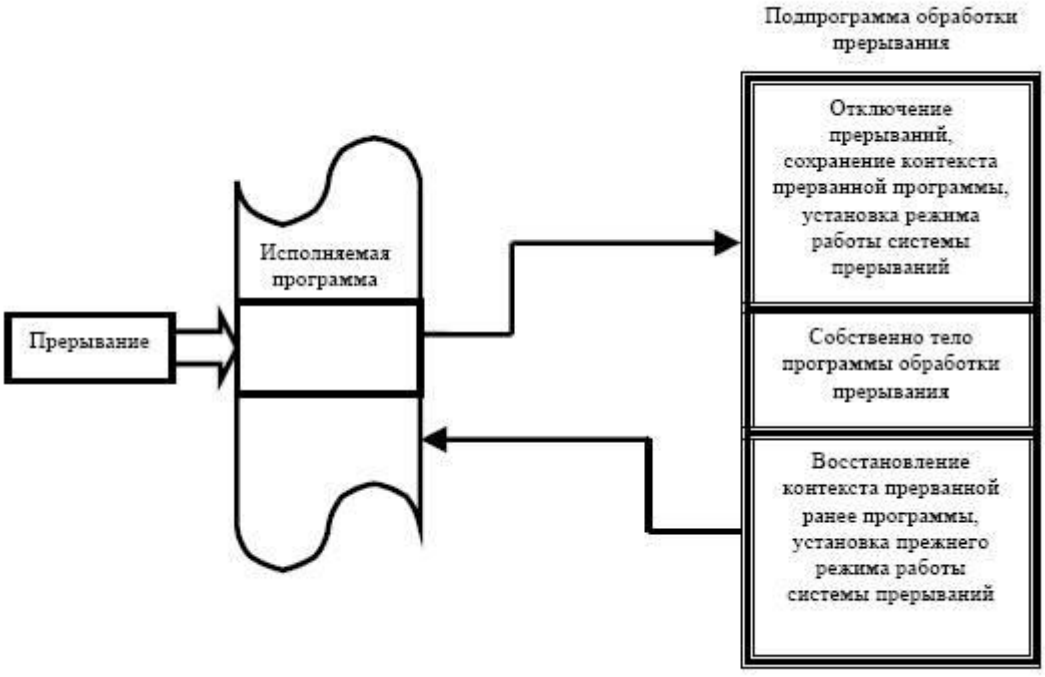


Рис. 13. Обработка прерывания

На рисунке 13 показано, что при возникновении запроса на прерывание естественный ход вычислений нарушается и управление передаётся программе обработки возникшего прерывания. При этом средствами аппаратуры сохраняется (как правило, с помощью механизмов стековой памяти) адрес той команды, с которой следует продолжить выполнение прерванной программы. После выполнения программы обработки прерывания управление возвращается прерванной ранее программе посредством занесения в указатель команд сохранённого адреса команды. Однако такая схема используется только в самых простых программных средах.

Этапы компиляции исходного кода в машинные коды и способы отладки.

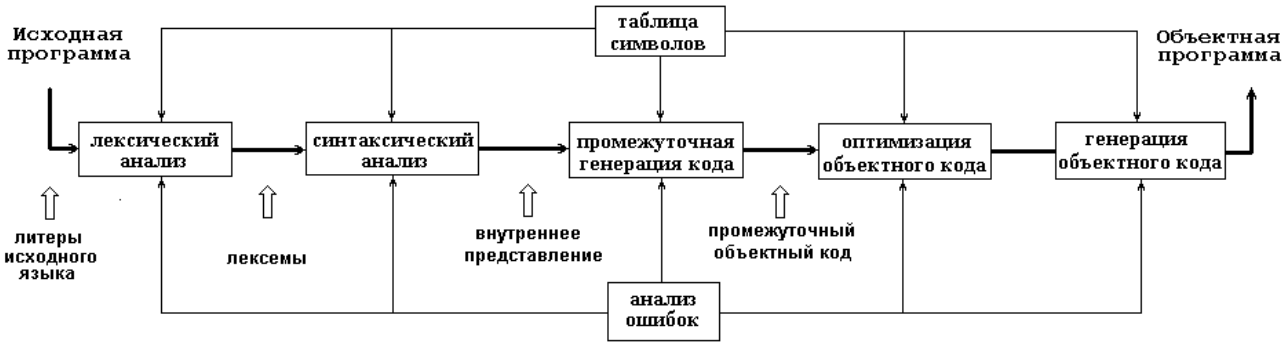
Компилятор — программа, выполняющая *компиляцию*.

Компиляция — трансляция программы, составленной на исходном языке высокого уровня, в эквивалентную программу на низкоуровневом языке, близком машинному коду. Входной информацией для компилятора (исходный код) является описание алгоритма или программа на объектно-ориентированном языке, а на выходе компилятора — эквивалентное описание алгоритма на машинно-ориентированном языке (объектный код).

Процесс компиляции состоит из следующих этапов:

1. Лексический анализ. На этом этапе последовательность символов исходного файла преобразуется в последовательность лексем.
2. Синтаксический (грамматический) анализ. Последовательность лексем преобразуется в дерево разбора.
3. Семантический анализ. Дерево разбора обрабатывается с целью установления его семантики (смысла) — например, привязка идентификаторов к их декларациям, типам, проверка совместимости, определение типов выражений и т. д. Результат обычно называется «промежуточным представлением/кодом», и может быть дополненным деревом разбора, новым деревом, абстрактным набором команд или чем-то ещё, удобным для дальнейшей обработки.
4. Оптимизация.

   Выполняется удаление излишних конструкций и упрощение кода с сохранением его смысла. Оптимизация может быть на разных уровнях и этапах — например, над промежуточным кодом или над конечным машинным кодом.
5. Генерация кода. Из промежуточного представления порождается код на целевом языке.



Отладка - это процесс локализации и исправления ошибок в программе.

Отладка, как мы уже говорили, бывает двух видов:

- **Синтаксическая отладка**. Синтаксические ошибки выявляет компилятор, поэтому исправлять их достаточно легко.
- **Семантическая (смысловая) отладка**. Ее время наступает тогда, когда синтаксических ошибок не осталось, но результаты программа выдает неверные. Здесь компилятор сам ничего выявить не сможет, хотя в среде программирования обычно существуют вспомогательные средства отладки, о которых мы еще поговорим.

Отладка программы в любом случае предполагает обдумывание и логическое осмысление всей имеющейся информации об ошибке. При этом используют различные методы:

- ручного тестирования (При обнаружении ошибки необходимо выполнить тестируемую программу вручную, используя тестовый набор, при работе с которыми была обнаружена ошибка).
  - Индукции(Метод основан на тщательном анализе симптомов ошибки, которые могут проявляться как неверные результаты вычислений или как сообщение об ошибке).
  - Дедукции (По методу дедукции вначале формируют множество причин, которые могли бы вызвать данное проявление ошибки. Затем анализируя причины, исключают те, которые противоречат имеющимся данным).
  - обратного прослеживания(Начинают с точки вывода неправильного результата. Для этой точки строится гипотеза о значениях основных переменных, которые могли бы привести к получению имеющегося результата. Далее, исходя из этой гипотезы, делают предложения о

значениях переменных в предыдущей точке. Процесс продолжают, пока не обнаружат причину ошибки).

Использование отладчиков — программ, которые включают в себя пользовательский интерфейс для пошагового выполнения программы: оператор за оператором, функция за функцией, с остановками на некоторых строках исходного кода или при достижении определённого условия.

Использованные источники:

1. ОП 09 Вычислительная техника : методическое пособие / . — Москва : ФГБУ ДПО «Учебно-методический центр по образованию на железнодорожном транспорте», 2019. — 88 с. — Текст : электронный // УМЦ ЖДТ : электронная библиотека. — URL: https://umczdt.ru/books/1251/234200/ (дата обращения 10.10.2023).

2. Трофименко, В.Н. Вычислительная техника и информационные технологии : / В. Н. Трофименко. — Ростов-на-Дону : РГУПС, 2019. — 151 с. — 978-5-88814-885-3. — Текст : электронный // УМЦ ЖДТ : электронная библиотека. — URL: https://umczdt.ru/books/1214/253832/ (дата обращения 10.10.2023).