ПРЕДИСЛОВИЕ

Цифровая схемотехника нового поколения активно используется в различных научных и технических направлениях и широко внедряется в различные промышленные отрасли и сферы экономики.

Перевод современного электронного оборудования на цифровые «рельсы» является актуальной и важной проблемой. Широкое применение цифровой техники резко увеличивает спрос на специалистов соответствующего профиля. Однако подготовка таких специалистов усложняется нехваткой учебно-методической литературы. Это обусловлено тем, что учебники, изданные в 1980-е и 1990-е гг., устаревают и их запасы в библиотеках ограничены.

В основу предлагаемого учебника положен курс лекций, который автор читал в течение ряда лет в Московском колледже информационных технологий (ранее Московский радиоаппаратостроительный техникум).

Учебник ориентирован на дисциплину «Цифровая схемотехника» для специальностей, связанных с изучением основ построения электрических схем на интегральных микросхемах. Изучив данную дисциплину, студенты должны знать схемы, на основе которых создаются цифровые устройства, понимать и в известном смысле «читать» электрические схемы цифровых устройств, уметь самостоятельно синтезировать относительно несложные схемы, ориентироваться в параметрах и характеристиках микросхем различной степени интеграции, владеть соответствующей терминологией, знать необходимые требования к оформлению конструкторской документации.

При написании учебника автор стремился излагать материал по принципу «от простого к сложному», чтобы студенты, освоив основы цифровой схемотехники, могли перейти к изучению более сложных дисциплин, и в первую очередь микропроцессорной техники.

Учебник состоит из семи разделов.

В первом разделе рассматриваются арифметические основы организации вычислительного процесса, теоретический материал здесь подкреплен примерами выполнения вычислительных операций.

Второй раздел посвящен логическим основам цифровой схемотехники, в процессе изучения которых студенты должны освоить проектирование несложных электрических схем цифровой

техники. В разделе также приводятся принципы построения интегральных микросхем на различной технологической базе.

В третьем разделе подробно рассматриваются комбинационные элементы цифровых устройств, развиваются принципы, идеи реализации арифметических основ и синтез на их основе более сложных схем.

В четвертом разделе представлены последовательностные схемы цифровой техники: триггеров, регистров, счетчиков с различными коэффициентами счета. Работа схем наглядно иллюстрируется временными диаграммами.

В пятом разделе рассматривается взаимодействие устройств в вычислительных системах.

Шестой раздел посвящен изучению запоминающих устройств вычислительных систем.

В седьмом разделе рассматривается организация микроЭВМ на основе микропроцессоров.

Раздел I
АРИФМЕТИЧЕСКИЕ ОСНОВЫ ТЕОРИИ ЦИФРОВЫХ УСТРОЙСТВ

Глава 1
СИСТЕМЫ СЧИСЛЕНИЯ

1.1. Основные понятия

Система счисления — это совокупность приемов и правил изображения чисел цифровыми знаками. Системы счисления делятся на позиционные и непозиционные.

Непозиционная система счисления — это система, в которой значение символа не зависит от его положения в числе. Непозиционные системы счисления возникли раньше позиционных систем. Они использовались в древности римлянами, египтянами, славянами и другими народами. Примером непозиционной системы счисления, дошедшей до наших дней, является римская система счисления. На рис. 1.1 число $30_{(10)}$ представлено в виде последовательного ряда из трех символов — **XXX**.

Позиционная система счисления — это система, в которой значение символа зависит от его позиции в ряду цифр, изображающих число, как это показано на рис. 1.2.

Позиционные системы счисления более удобны для вычислительных операций, поэтому они получили более широкое распространение.

Количество различных символов, используемых в позиционной системе счисления для выражения всех чисел в пределах одного разряда, называется ее основанием и обозначается латинской буквой *S*.

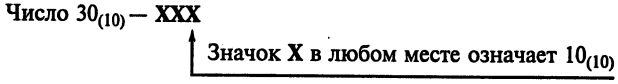


Рис. 1.1

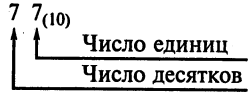


Рис. 1.2

В двоичной системе счисления ($S = 2$) используются два символа 0 и 1, в восьмеричной ($S = 8$) — восемь символов от 0 до 7, в десятичной ($S = 10$) — десять символов от 0 до 9, в шестнадцатеричной ($S = 16$) — шестнадцать символов от 0 до 9 и специально введенные дополнительные символы в виде латинских букв *A*, *B*, *C*, *D*, *E*, *F*.

Числа в таких системах счисления представляются последовательностью ряда цифр, разделенных на две группы: группу разрядов, изображающую целую часть числа, и группу разрядов, изображающую дробную часть числа, которая может быть представлена формулой

$$N = \ldots + a_2 S^2 + a_1 S^1 + a_0 S^0 + a_{-1} S^{-1} + a_{-2} S^{-2} + \ldots$$

Единице каждого разряда присваивается определенный вес S^k, где S— основание системы счисления, а k — номер разряда, равный индексу при символах, изображающих цифры разрядов.

Вне зависимости от рассматриваемой системы счисления последовательность символов подчиняется закономерности, которая определяется порядком (циклом) счета (табл. 1.1).

Контрольные вопросы

1. Какую смысловую нагрузку несет определение основания системы счисления?
2. Дать определение понятия «позиционная система счисления».
3. Может ли быть система счисления пятеричной и семеричной?
4. Какие числа следуют за числом семь в восьмеричной системе счисления, за числом девять в десятичной системе счисления и за числом пятнадцать в шестнадцатеричной системе счисления?
5. Назвать последние числа в двоичной, восьмеричной, десятичной и шестнадцатеричной системах счисления в третьем, пятом, седьмом циклах счета и следующие числа в следующих циклах.
6. Каким системам счисления могут принадлежать следующие числа: 765, 3*A*2, 10101, 1579, 816, 9*E*5, 819?

Контрольные задания

Представить двоичный эквивалент чисел 20, 25, 13, 30, 11, 23, 14, 28 по предложенной схеме.

Дано число 19.

1. Определяем степень 2, в результате чего получаем число, максимально приближенное к 19:

$$2^4 = 16.$$

2. Определяем, что полученное число отличается от 19 на 3 (19 – 16 = 3).
3. Определяем степень 2, в результате чего получаем число, максимально приближенное к 3:

$$2^1 = 2.$$

4. Определяем, что полученное число отличается от 3 на 1 (3 – 2 = 1).

Таблица 1.1

Система счисления	Основание S	Используемые символы	Цикл счета		Последовательность символов
Десятичная	$10_{(10)}$	0, 1, 2, 3, 4, 5, 6, 7, 8, 9	$0_{(10)}$ $1_{(10)}$ $2_{(10)}$ ... $9_{(10)}$ $10_{(10)}$ $11_{(10)}$		0, 1, 2, 3, 4, 5, 6, 7, 8, 9 10, 11, 12, 13, 14, 15, 16, 17, 18, 19 20, 21, 22, 23, 24, 25, 26, 27, 28, 29 ..... 90, 91, 92, 93, 94, 95, 96, 97, 98, 99 100, 101, 102, 103, 104, 105, 106, 107, 108, 109 110, 111, 112, 113, 114, 115, 116, 117, 118, 119 и т.д.
Восьмеричная	$8_{(10)}$	0, 1, 2, 3, 4, 5, 6, 7	$0_{(10)}$ $1_{(10)}$ $2_{(10)}$ ... $7_{(10)}$ $8_{(10)}$	$0_{(8)}$ $1_{(8)}$ $2_{(8)}$ ... $7_{(8)}$ $10_{(8)}$	0, 1, 2, 3, 4, 5, 6, 7 10, 11, 12, 13, 14, 15, 16, 17 20, 21, 22, 23, 24, 25, 26, 27 ..... 70, 71, 72, 73, 74, 75, 76, 77 100, 101, 102, 103, 104, 105, 106,107 и т.д.
Двоичная	$2_{(10)}$	0, 1	$0_{(10)}$ $1_{(10)}$ $2_{(10)}$ $3_{(10)}$ $4_{(10)}$	$0_{(2)}$ $1_{(2)}$ $10_{(2)}$ $11_{(2)}$ $100_{(2)}$	0 0, 0 1 1 0, 1 1 10 0, 10 1 11 0, 11 1 100 0, 100 1 и т.д.

Окончание табл. 1.1

Система счисления	Основание S	Используемые символы	Цикл счета		Последовательность символов
Шестнадцатеричная	$16_{(10)}$	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, *A*, *B*, *C*, *D*, *E*, *F*	$0_{(10)}$	$0_{(16)}$	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, *A*, *B*, *C*, *D*, *E*, *F*
			$1_{(10)}$	$1_{(16)}$	10,11,12,13,14, 15, 16, 17, 18, 19,1*A*, 1*B*,1*C*,1*D*, 1*E*, 1*F*
			...	...	.....
			$10_{(10)}$	$A_{(16)}$	*A*0,*A*1,*A*2,*A*3, *A*4, *A*5, *A*6, *A*7, *A*8, *A*9, *AA*, *AB*, *AC*, *AD*, *AE*, *AF*
			$11_{(10)}$	$B_{(16)}$	*B*0,*B*1,*B*2,*B*3, *B*4, *B*5, *B*6, *B*7, *B*8, *B*9, *BA*, *BB*, *BC*, *BD*, *BE*, *BF*
			...	...	.....
			$15_{(10)}$	$F_{(16)}$	*F*0, *F*1, *F*2, *F*3, *F*4, *F*5, *F*6, *F*7, *F*8, *F*9, *FA*, *FB*, *FC*, *FD*, *FE*, *FF*
			$16_{(10)}$	$10_{(16)}$	100,101,102,103,104,105,106,107,108,109,10*A*,10*B*,10*C*,10*D*,10*E*,10*F*
			$17_{(10)}$	$11_{(16)}$	110,111,112,113,114,115,116,117,118,119,11*A*,11*B*,11*C*,11*D*,11*E*,11*F*
					и т.д.

5. Определяем степень 2, которая в результате дает 1:

$$2^0 = 1.$$

6. Таким образом, число 19 можно представить в виде суммы следующих степеней 2: $1 \cdot 2^4 + 1 \cdot 2^1 + 1 \cdot 2^0$.

При остальных степенях 2 (2^3, 2^2) коэффициенты равны 0.

7. В результате число 19 в двоичном эквиваленте имеет следующий вид:

$$19_{(10)} = 10011_{(2)}.$$

При определенном навыке можно довольно быстро находить эквиваленты двоичных чисел.

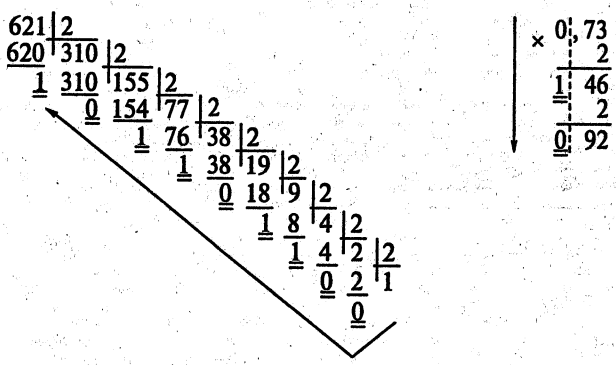
1.2. Правила перевода из одной системы счисления в другую

При переводе чисел из одной системы счисления в другую можно выявить определенную закономерность, которая отражена в табл. 1.2.

Рассмотрим примеры перевода чисел из одной системы счисления в другую.

Пример 1. $621{,}73_{(10)} \rightarrow (2) \rightarrow (8) \rightarrow (2) \rightarrow (16) \rightarrow (10)$.

1) (10)→(2):



$621{,}73_{(10)} = 1001101101{,}10_{(2)}$;

2) (2) → (8):

$00\,1\ 001\ 101\ 101{,}10\,0_{(2)}$

$1\quad 1\quad 5\quad 5\ ,\ 4_{(8)}$;

3) (8) → (2):

$0010\ 0110\ 1101{,}100_{(2)}$;

4) (2) → (16):

$0010\ 0110\ 1101{,}100\,0_{(2)}$

$2\quad 6\quad D\ ,\ 8_{(16)}$;

5) (16) → (10):

$2 \cdot 16^2 + 6 \cdot 16^1 + 13 \cdot 16^0 + 8 \cdot 16^{-1} = 512 + 96 + 13 + 0{,}5 = 621{,}5_{(10)}$.

Таблица 1.2

<table>
<tr><th>Значения чисел</th><th>Перевод</th><th>Правило перевода</th><th>Пример перевода</th><th colspan="4">Примечание</th></tr>
<tr><td rowspan="2">Целые</td><td rowspan="2">(10)→(8)
(10)→(2)
(10)→(16)</td><td rowspan="2">Целое десятичное число необходимо последовательно делить на основание S той системы счисления, в которую оно переводится, до тех пор, пока не получится частное меньше основания</td><td rowspan="2">Исходное число: $25_{(10)}$
1) (10)→(8)
$25_{(10)} \mid 8$
$24 \quad 3$
1
Правило записи
$25_{(10)} \rightarrow 31_{(8)}$
2) (10)→(2)
$25_{(10)} \mid 2$
$24 \quad 12 \mid 2$
$1 \quad 12 \quad 6 \mid 2$
$0 \quad 6 \quad 3 \mid 2$
$0 \quad 2 \quad 1$
1
$25_{(10)} \rightarrow 11001_{(2)}$
3) (10)→(16)
$25_{(10)} \mid 16$
$16 \quad 1$
9
$25_{(10)} \rightarrow 19_{(16)}$</td><td colspan="4">Приведем двоичные и шестнадцатеричные эквиваленты наиболее часто используемых десятичных чисел от 0 до 15</td></tr>
<tr><td>S = 10</td><td>S = 2</td><td>S = 16</td></tr>
<tr><td></td><td></td><td></td><td></td><td>0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15</td><td>0
1
10
11
100
101
110
111
1000
1001
1010
1011
1100
1101
1110
1111</td><td>0
1
2
3
4
5
6
7
8
9
A
B
C
D
E
F</td></tr>
</table>

Дробные	(10)→(8) (10)→(2) (10)→(16)	Цифры дроби надо последовательно умножать на основание той системы счисления, в которую она переводится, до тех пор, пока число не будет вычислено с заданной точностью	Исходное число: $0,83_{(10)}$ 1) (10)→(8) Направление записи результата перевода $0,83 \cdot 8 = 6,64$ $0,64 \cdot 8 = 5,12$ $0,12 \cdot 8 = 0,96$ $0,83_{(10)} \rightarrow 0,650_{(8)}$ 2) (10)→(2) $0,83 \cdot 2 = 1,66$ $0,66 \cdot 2 = 1,32$ $0,32 \cdot 2 = 0,64$ $0,83_{(10)} \rightarrow 0,110_{(2)}$ 3) (10)→(16) $0,83 \cdot 16 = 13,28$ $0,28 \cdot 16 = 4,98 + 8,3$ $0,83_{(10)} \rightarrow 0,D_{(16)}$	В процессе умножения для получения очередного произведения на основание S умножаются только цифры справа от запятой. О результате перевода судят по значению цифр целой части, расположенных слева от запятой

Продолжение табл. 1.2

Значения чисел	Перевод	Правило перевода	Пример перевода	Примечание
Целые и дробные	(8)→(10) (2)→(10) (16)→(10)	Число представляется в виде ряда с основанием той системы счисления, из которой оно переводится	1) $173_{(8)} \rightarrow (10)$ $1 \cdot 8^2 + 7 \cdot 8^1 + 3 \cdot 8^0 = 64 + 56 + 3 =$ $= 123_{(10)}$ 2) $3A5,E_{(16)} \rightarrow (10)$ $3 \cdot 16^2 + 10 \cdot 16^1 + 5 \cdot 16^0 + 14 \cdot 16^{-1} =$ $= 768 + 160 + 5 + 0{,}875 = 933{,}875_{(10)}$ 3) $321{,}7_{(8)} \rightarrow (10)$ $3 \cdot 8^2 + 2 \cdot 8^1 + 1 \cdot 8^0 + 7 \cdot 8^{-1} =$ $= 192 + 16 + 1 + 0{,}875 = 209{,}875_{(10)}$ 4) $101{,}01_{(2)} \rightarrow (10)$ $1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} =$ $= 5{,}25_{(10)}$.	$2^0 = 1$ $8^0 = 1$ $16^0 = 1$ $2^1 = 2$ $8^1 = 8$ $16^1 = 16$ $2^2 = 4$ $8^2 = 64$ $16^2 = 256$ $2^3 = 8$ $8^3 = 512$ $16^3 = 4096$ $2^4 = 16$ $8^4 = 4096$ $2^5 = 32$ $2^6 = 64$ $2^7 = 128$ $2^8 = 256$ $2^9 = 512$ $2^{10} = 1024$ $2^{11} = 2048$ $2^{12} = 4096$
Целые и дробные	(8)→(2)	Восьмеричное число записывается в виде трехразрядного двоичного числа — триады	3 7 2 , $5_{(8)} \rightarrow (2)$ ↓ ↓ ↓ ↓ 011 111 010 , $101_{(2)}$ Триада	—

Целые и дробные	(16)→(2)	Шестнадцатеричное число записывается в виде четырехразрядного двоичного числа — тетрады	$F\ \ 3\ \ 2\ ,\ A_{(16)}\to(2)$ $\downarrow\ \ \downarrow\ \ \downarrow\ \ \ \ \downarrow$ $1111\ \ 0011\ \ 0010\ ,\ 1010_{(2)}$ Тетрада	—
Целые и дробные	(2)→(8)	Триада двоичных чисел записывается восьмеричным эквивалентом	1) $\underline{0}10\ \ 110\ \ 111\ ,\ 100_{(2)}\to(8)$ $\downarrow\ \ \downarrow\ \ \downarrow\ \ \ \ \downarrow$ $2\ \ 6\ \ 7\ ,\ 4_{(8)}$ 2) $\underline{00}1\ \ \underline{101}\ \ \underline{101}\ ,\ \underline{1}\underline{00}_{(2)}\to(8)$ $1\ \ 5\ \ 5\ ,\ 4_{(8)}$	При делении числа на триады в случае необходимости оно дополняется нулями влево и вправо от запятой
Целые и дробные	(2)→(16)	Тетрада двоичного числа записывается шестнадцатеричным эквивалентом	1) $1101\ \ 0001\ _{(2)\to(16)}$ $D\ \ 1\ _{(16)}$ 2) $1001\ _{(2)\to(16)}$ $9\ _{(16)}$ 3) $\underline{00}10\ \underline{1100}\ \underline{1010}\ \underline{0110}\ ,\ \underline{001}\underline{0}_{(2)}$ $2\ \ C\ \ A\ \ 6\ ,\ 2_{(16)}$	Дополнение нулями до тетрады осуществляется влево и вправо от запятой

Окончание табл. 1.2

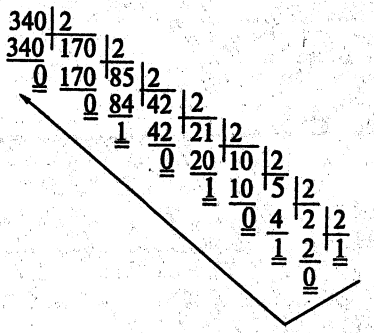
Значения чисел	Перевод	Правило перевода	Пример перевода	Примечание
Целые и дробные	(10)→(2–10)	Каждая цифра десятичного числа записывается в виде его четырехразрядного двоичного эквивалента	$2467{,}39_{(10)\rightarrow(2-10)}$ $\underbrace{0010}_{2}\ \underbrace{0100}_{4}\ \underbrace{0110}_{6}\ \underbrace{0111}_{7}\ ,\ \underbrace{0011}_{3}\ \underbrace{1001}_{9}{}_{(2-10)}$	—
Целые и дробные	(2 – 10)→(10)	Каждые четыре разряда двоично-десятичного числа записываются его десятичным эквивалентом	$\underbrace{0010}_{2}\ \underbrace{0101}_{5}\ \underbrace{0011}_{3}\ ,\ \underbrace{0100}_{4}{}_{(2-10)}{}_{(10)}$	—

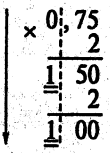
Пример 2. $524{,}6_{(8)} \rightarrow (10) \rightarrow (2) \rightarrow (16) \rightarrow (10) \rightarrow (8)$.

1) (8) → (10):

$5 \cdot 8^2 + 2 \cdot 8^1 + 4 \cdot 8^0 + 6 \cdot 8^{-1} = 320 + 16 + 4 + 0{,}75 = 340{,}75_{(10)}$;

2) (10) → (2):





$101010100{,}11_{(2)}$;

3) (2) → (16):

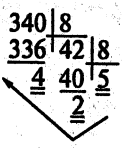
$\underline{000}1\ 0101\ 0100{,}11\underline{00}_{(2)}$

$1\quad 5\quad 4\ ,\quad C_{(16)}$;

4) (16) → (10):

$1 \cdot 16^2 + 5 \cdot 16^1 + 4 \cdot 16^0 + 12 \cdot 16^{-1} = 340{,}75_{(10)}$;

5) (10) → (8):





$524{,}6_{(8)}$.

Контрольные вопросы

1. В чем состоит закономерность перевода целых и дробных чисел из десятичной системы счисления в восьмеричную, двоичную и шестнадцатеричную системы?
2. В чем состоит закономерность перевода чисел из восьмеричной, двоичной и шестнадцатеричной систем счисления в десятичную?
3. Почему восьмеричное число в двоичном эквиваленте изображается триадами, а шестнадцатеричное — тетрадами?
4. Почему для изображения десятичной цифры в двоично-десятичной системе счисления отводится четыре разряда?

Контрольные задания

Перевести исходные числа в другие системы счисления:

1) $349{,}75_{(10)} \rightarrow (2) \rightarrow (8) \rightarrow (16)$;

2) $278{,}3_{(10)} \rightarrow (8) \rightarrow (2) \rightarrow (10)$;
3) $51{,}4_{(10)} \rightarrow (8) \rightarrow (2) \rightarrow (10)$;
4) $69{,}18_{(10)} \rightarrow (8) \rightarrow (2) \rightarrow (16)$;
5) $5C_{(16)} \rightarrow (2) \rightarrow (10)$;
6) $1AB_{(16)} \rightarrow (2) \rightarrow (10)$;
7) $A5{,}7_{(16)} \rightarrow (2) \rightarrow (8) \rightarrow (10) \rightarrow (16)$;
8) $1011{,}01_{(2)} \rightarrow (10) \rightarrow (16) \rightarrow (2)$;
9) $C96{,}E_{(16)} \rightarrow (2) \rightarrow (10) \rightarrow (16)$;
10) $76{,}36_{(10)} \rightarrow (16) \rightarrow (2) \rightarrow (10) \rightarrow (2\text{—}10) \rightarrow (10)$;
11) $0011{,}01_{(2\text{—}10)} \rightarrow (10) \rightarrow (16) \rightarrow (2) \rightarrow (10) \rightarrow (2\text{—}10)$;
12) $31{,}7_{(8)} \rightarrow (10) \rightarrow (2\text{—}10) \rightarrow (10) \rightarrow (16) \rightarrow (2) \rightarrow (8)$.

Глава 2
ФОРМЫ ПРЕДСТАВЛЕНИЯ ЧИСЕЛ

2.1. Форматы данных

Цифровая схемотехника оперирует многоразрядной информацией, представленной словами в двоичной системе счисления. Исторически сложилось так, что в качестве устройств и носителей информации использовались элементы, для которых характерны два состояния (например, электрические реле, элементы индикации, перфокарты, магнитные носители памяти, электронный носитель памяти — триггер). Условно эти состояния приняли за 1 и 0.

Бит — это один разряд слова, который может принимать значение 1 или 0.

Широко используется формат — *байт*, который представляет собой восьмиразрядное слово. Структура формата может быть увеличена и равна 16, 32, 64 и т.д. двоичным разрядам. Таким образом, предыдущий формат отличается от последующего в два раза. Состояние обрабатываемой информации отражается в разрядной сетке, числа в которую могут быть занесены двумя способами: в форме с плавающей запятой и в форме с фиксированной запятой.

2.2. Представление чисел в форме с плавающей запятой

При работе с очень большими или очень малыми числами их представляют в *полулогарифмической форме* (*с плавающей запятой*). Вес числа в этом случае изменяется за счет изменения порядка числа.

Любое число A в форме с плавающей запятой выглядит следующим образом:

$$A = S^{\pm p} \cdot (\pm m),$$

где S — основание системы счисления мантиссы числа — $10_{(2)}$; m — мантисса числа A, причем $|m| < 1$; p — целое число, выражающее порядок числа A, которое показывает, на сколько разрядов вправо ($+p$) или влево ($-p$) необходимо переместить запятую, чтобы разряды мантиссы представляли собой количественную характеристику исходного числа, его целую и дробную части.

Запись такого вида называется полулогарифмической, так как в логарифмической форме представляется не все число, а только его часть $S^{\pm p}$.

Представим число $A = 45_{(10)} = +\,101101_{(2)}$ в форме с плавающей запятой, рассмотрев несколько вариантов представления мантиссы числа в зависимости от изменяемого значения порядка:

$$A = +101101 \cdot 10^{0} = +101{,}101 \cdot 10^{+11} = 1{,}01101 \cdot 10^{+101} =$$
$$= 0{,}101101 \cdot 10^{+110} = 0{,}00101101 \cdot 10^{+1000} = 10110100, \cdot 10^{-10}.$$

Из приведенного примера видно, что, изменяя значение порядка, можно менять вид мантиссы. Запятая как бы «плавает» в изображении числа. Для сохранения в разрядной сетке большего количества значащих цифр и повышения точности вычислений исходные числа и результаты вычислений представляются в *нормализованном виде*, т. е. мантисса числа представляется в виде дроби, в которой после запятой стоит единица старшего значащего разряда:

$$A = 0{,}\underline{1}01101 \cdot 10^{+110}.$$

На рис. 2.1 показан вид разрядной сетки для представления чисел в форме с плавающей запятой.

Представим числа A_1, A_2 в форме с плавающей запятой, используя разные значения порядка:

$$A_1 = -11011{,}10_{(2)}; \qquad A_2 = +0{,}00101_{(2)};$$

$$A_1 = -11011{,}10 \cdot 10^{0} = -0{,}1101110 \cdot 10^{+101} = -1101110, \cdot 10^{-10};$$

$$A_2 = +0{,}00101 \cdot 10^{0} = +0{,}000101 \cdot 10^{+1} = +0{,}101 \cdot 10^{-10} = +10{,}1 \cdot 10^{-100}.$$

Определим исходные числа A_1, A_2 до преобразования их в форму с плавающей запятой:

$$A_1 = -0{,}1101 \cdot 10^{+11}; \qquad A_2 = +0{,}1101 \cdot 10^{-10};$$

$$A_1 = -0{,}1101 \cdot 10^{+11} = -110{,}1 \cdot 10^{0} = -(2^{2} + 2^{1} + 2^{-1}) = -6{,}5_{(10)};$$

$$A_2 = +0{,}1101 \cdot 10^{-10} = +0{,}001101 \cdot 10^{0} = +(2^{-3} + 2^{-4} + 2^{-6}) = +0{,}203_{(10)}.$$

Код знака		Порядок				Мантисса			
мантиссы	порядка	Y_1	Y_2	...	Y_i	X_1	X_2	...	X_j

Рис. 2.1

Контрольные вопросы

1. В каких случаях целесообразно представлять числа в форме с плавающей запятой?
2. Как взаимосвязаны значения порядка и мантиссы числа?
3. Привести примеры чисел, представленных в нормализованном виде.

Контрольные задания

1. В зависимости от значения порядка представить несколько вариантов изображения чисел в форме с плавающей запятой, включая нормализованный вид:

$$A_1 = +100{,}001_{(2)};$$
$$A_2 = -0{,}11001_{(2)};$$
$$A_3 = -11{,}0101_{(2)};$$
$$A_4 = +0{,}01101_{(2)}.$$

2. Представить числа в десятичной системе счисления до преобразования их в форму с плавающей запятой:

$$A_1 = 1{,}11 \cdot 10^{+10};$$
$$A_2 = 0{,}0101 \cdot 10^{+100};$$
$$A_3 = 0{,}101 \cdot 10^{-10};$$
$$A_4 = 1{,}1001 \cdot 10^{-11}.$$

2.3. Представление чисел в форме с фиксированной запятой

Представление чисел в *естественной форме* (*с фиксированной запятой*) значительно проще. Использование такой формы записи стало возможно благодаря созданию микросхем высокой степени интеграции, что позволило увеличить разрядность и таким образом повысить точность вычислений. Все исходные числа представляются как целые. Для приведения исходных чисел к единому виду производится их умножение на общий масштабный коэффициент *M*. Такая операция создает дополнительные трудности, что является недостатком данной формы представления чисел.

Рассмотрим использование масштабных коэффициентов на примерах представления исходных чисел, имеющих как целую, так и дробную части в разрядной сетке, как целых.

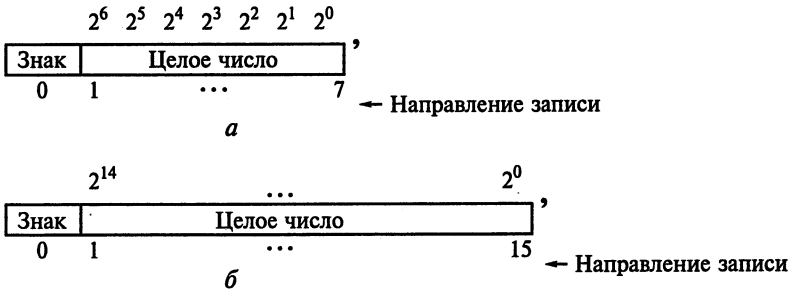


Рис. 2.2

На рис. 2.2, *а* представлен вид восьмиразрядной сетки, а на рис. 2.2, *б* — шестнадцатиразрядной. При любом формате слова нулевой (самый левый) разряд всегда отводится под код знака. Знак «плюс» изображается как 0, а знак «минус» — как 1. Перед заполнением разрядной сетки данными во всех ее разрядах находятся нули. Размещение исходного числа всегда производится от запятой, по ходу определения веса разряда. Если разрядность заносимого числа меньше формата разрядной сетки, то в свободных разрядах остаются предварительно записанные нули. Абсолютное значение располагаемого в разрядной сетке числа называется модулем.

Представим числа A_1, A_2 в восьмиразрядной сетке, когда они являются целыми. Для чисел

$$A_1 = +0{,}1001;$$
$$A_2 = -11{,}01$$

принимаем $M = 10000$, тогда

$$A_1' = A_1 \cdot M = +0{,}1001 \cdot 10000 = +1001 = +\underline{000}\ 1001{,};$$

$$A_2' = A_2 \cdot M = -11{,}01 \cdot 10000 = -110100 = -\underline{0}110100{,}.$$

На рис. 2.3 показан вид разрядной сетки для рассмотренного примера.

Преобразованное число или результат вычислений может превысить разрядность сетки, что в свою очередь приведет к потере значащих цифр и снижению точности. Рассмотрим эту особенность на примере размещения чисел в сетке малой разрядности.

Используя два значения масштабных коэффициентов M_1 и M_2, представим числа A_1 и A_2 как целые в форме с фиксированной запятой и разместим их в восьмиразрядной сетке:

$$A_1 = +1101{,}001;$$
$$A_2 = -10{,}11101.$$

При $M_1 = 1000$ происходит потеря младших разрядов:

| ← Потеря младших разрядов

$$A_1' = A_1 \cdot M_1 = +1101,\ 001 \cdot 1000 \quad = +1101001,\ 00;$$

$$A_2' = A_2 \cdot M_1 = \quad -10,\ 11101 \cdot 1000 = -0010111,\ 01.$$

На рис. 2.4 представлены числа в разрядной сетке при $M_1 = 1000$. Здесь при записи в разрядную сетку произошла потеря младших разрядов числа A_2.

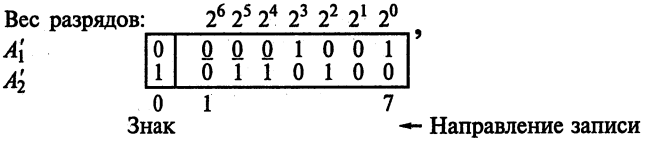


Рис. 2.3

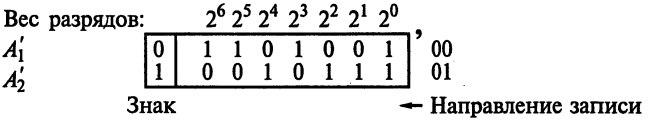


Рис. 2.4

При масштабном коэффициенте $M_2 = 100000$ происходит потеря старших разрядов:

Потеря старших разрядов →|

$$A_1'' = A_1 \cdot M_2 = +1101,\ 001 \cdot 100000 = +11\ 0100100,\ 000;$$

$$A_2'' = A_2 \cdot M_2 = -10,\ 11101 \cdot 100000 = \quad -1011101,\ 00000.$$

На рис. 2.5 представлены числа в разрядной сетке при $M_2 = 100000$. Здесь при записи произошла потеря старших разрядов числа A_1.

Если некоторые числа будут велики или малы по величине, то при занесении их в разрядную сетку значащие разряды могут оказаться за ее пределами. Такое явление, называемое *машинный нуль*, приводит к снижению точности вычислений.

Например, исходные числа, представленные в восьмиразрядной сетке, по абсолютному значению меньше единицы:

$$A_1 = +111{,}01001;$$
$$A_2 = +0{,}0000000101.$$

При $M = 0{,}001$ происходит потеря значащих разрядов числа:

|← Потеря значащих разрядов числа

$$A_1' = A_1 \cdot M = 0{,}1110100\ 1;$$

$$A_2' = A_2 \cdot M = 0{,}0000000\ 000101.$$

На рис. 2.6 число A_2, занесенное в разрядную сетку, содержит нули во всех разрядах.

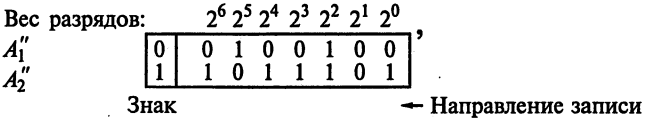


Рис. 2.5

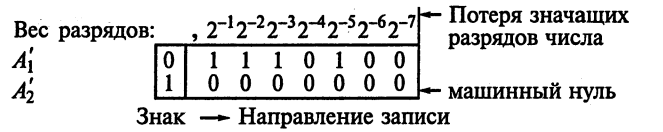


Рис. 2.6

Контрольные вопросы

1. При каких условиях числа целесообразно представлять в форме с фиксированной запятой?
2. Для чего необходимо вводить масштабный коэффициент?
3. В чем заключаются принципы занесения чисел в разрядную сетку в форме с фиксированной запятой?
4. Пояснить, какое действие необходимо выполнить над числами A_1 и A_2 перед их размещением в восьмиразрядной сетке:

$A_1 = +101{,}01;$
$A_2 = -11{,}0001.$

5. Каким должен быть коэффициент при преобразовании исходных чисел A_1, A_2 для обеспечения максимальной точности вычисления?

Контрольные задания

Представить каждую из пар исходных чисел в восьмиразрядной сетке в форме с фиксированной запятой, обеспечивая максимальную точность вычислений:

1) $A_1 = -101{,}01;$ $A_2 = -10{,}1011.$
2) $A_1 = +0{,}0101;$ $A_2 = -0{,}10.$
3) $A_1 = +10011{,}101;$ $A_2 = -110{,}10111.$
4) $A_1 = -10{,}10101;$ $A_2 = +0{,}0001.$

Глава 3
МАШИННЫЕ КОДЫ

3.1. Кодирование положительных чисел в форме с фиксированной запятой

В вычислительных устройствах операции вычитания, деления, умножения сводятся к операциям сложения и сдвига промежуточных результатов влево или вправо в зависимости от выполняемого действия. Это становится возможным, если в операциях будут участвовать не сами числа, а их коды. Существуют три вида кодов: прямой, обратный и дополнительный.

Необходимо помнить, что все числа, которые будут обрабатываться в форме с фиксированной запятой, должны быть целыми, т. е. запятая должна находиться справа от младшего разряда (см. рис. 2.2...2.5). В дальнейшем будем подразумевать эту запятую, но не будем ее показывать. Кроме того, при изображении кодов чисел возникает необходимость отделения знакового нулевого разряда от кода модуля числа. Проще всего отделять код знака от кода числа точкой.

При представлении положительных чисел в форме с фиксированной запятой прямой, дополнительный и обратные коды совпа-

дают по изображению. В знаковой части ставится нуль, который кодирует знак «плюс».

Представим число A в прямом, обратном и дополнительном кодах в восьмиразрядной сетке:

$$A = +1001_{(2)};$$

$$A_{пр} = A_{обр} = A_{доп} = \underline{0}.\ 0001001^{*}.$$

Код знака «плюс»

3.2. Кодирование отрицательных чисел в форме с фиксированной запятой

Коды отрицательных чисел (в отличие от положительных) имеют различный вид, т. е. не совпадают по изображению.

Последовательность и правила кодирования отрицательного числа $A = -1010_{(2)}$, представленного в формате восьмиразрядной сетки, показаны в табл. 3.1.

Таблица 3.1

Вид кода	Правило кодирования	Вид закодированного числа
Прямой	Изображение кода совпадает с изображением числа. В знаковой части ставится **1**	$A_{пр}$ = **1**. 0001010
Обратный	Значение разрядов после точки меняется на обратное — единицы на нули, нули на единицы. Код знака остается без изменения	$A_{обр}$ = **1**. 1110101
Дополнительный	Образуется как обратный с дополнительным прибавлением 1 к младшему разряду	$A_{доп}$ = **1**. 1110101 + 1 **1**. 1110110

3.3. Правило перевода отрицательных чисел из дополнительного и обратного кодов в прямой код

Правило перевода из дополнительного и обратного кодов в прямой аналогично правилу перевода из прямого кода в дополнитель-

* Любой код числа должен иметь вид типа $[A]_{пр}$. В дальнейшем, чтобы не перегружать запись, откажемся от квадратных скобок.

ный и обратный, при этом код знаковой части преобразованию не подвергается. Рассмотрим правило перевода на примере.

Даны дополнительный и обратный коды числа A. Необходимо получить прямой код:

1) $A_{обр} = 1.1110101;$
   $A_{пр} = 1.0001010.$

2) $A_{доп} = 1.1110110;$
   $A_{пр} = 1.0001001$
   $+ \quad 1$
   $1.0001010.$

Таким образом прямые коды совпали.

Контрольные вопросы

1. Почему при вычислениях используются коды чисел, а не сами числа?
2. Чем отличается образование дополнительного кода отрицательного числа от обратного кода?
3. Какой вид кода отрицательного числа максимально приближается к действительному изображению числа? В чем заключается отличие в их изображении?
4. Какие преобразования необходимо выполнить при переводе из обратного и дополнительного кодов отрицательного числа в прямой код?

Контрольные задания

1. Произвести перевод исходных чисел в прямой, обратный и дополнительный коды, используя восьмиразрядный формат:

1) $A_1 = +110111;$
2) $A_2 = -1010;$
3) $A_3 = -1101101;$
4) $A_4 = +10011.$

2. Получить исходное число A, предварительно осуществив перевод данных кодов в прямой код:

1) $A_{обр} = 1.1100110;$
2) $A_{обр} = 0.1011001;$
3) $A_{доп} = 1.1100111;$
4) $A_{доп} = 0.0011001;$
5) $A_{обр} = 1.1101010;$
6) $A_{доп} = 1.0101011;$
7) $A_{обр} = 1.1101011;$
8) $A_{доп} = 0.1011010;$
9) $A_{доп} = 1.0100000;$
10) $A_{обр} = 0.1100110.$

Глава 4
АРИФМЕТИЧЕСКИЕ ДЕЙСТВИЯ НАД ЧИСЛАМИ С ФИКСИРОВАННОЙ ЗАПЯТОЙ

4.1. Сложение положительных чисел

Сложение многоразрядных чисел осуществляется по правилам двоичной арифметики; особенность проявляется при сложении двух единиц. При $S = 10_{(10)}$ сумма двух единиц равна двум, что эквива-

лентно $10_{(2)}$. Таким образом вместо одного разряда образуется два. В этом случае говорят, что при сложении образовалась единица переноса в старший разряд:

$$\begin{array}{rr} 0 & 1 \\ + & + \\ 1 & 1 \\ \hline 1 & 1\ \ 0. \end{array}$$

Рассмотрим сложение двух чисел A_1 и A_2 в формате восьмиразрядной сетки:

$$A_1 = +01010;$$
$$A_2 = +01001.$$

Так как при представлении положительных чисел в форме с фиксированной запятой прямой, дополнительный и обратный коды совпадают по изображению, то вычисление выглядит следующим образом:

$$\begin{array}{r} A_{1\ \text{пр, обр, доп}} = 0.\ 0001010 \\ + \\ A_{2\ \text{пр, обр, доп}} = 0.\ 0001001 \\ \hline (A_1 + A_2)_{\text{пр, обр, доп}} = 0.\ 0010011. \end{array}$$

Иногда результат сложения превышает максимально возможное число и не вписывается в разрядную сетку, а возникающая единица переноса заносится в знаковый разряд. В этом случае знак результата становится противоположен знаку исходных слагаемых. Такое явление называется *переполнением разрядной сетки*. Например, произведем сложение двух чисел A_1 и A_2 в прямом, обратном и дополнительном кодах, разместив их в восьмиразрядной сетке:

$$A_1 = +1010101;$$
$$A_2 = +0111001;$$

$$\begin{array}{r} A_{1\ \text{пр, обр, доп}} = 0.\ 1010101 \\ + \\ A_{2\ \text{пр, обр, доп}} = 0.\ 0111001 \\ \hline (A_1 + A_2)_{\text{пр, обр, доп}} = 1.\ 0001110. \end{array}$$

Знак результата противоположен знаку слагаемых

Для выявления переполнения разрядной сетки производится сложение чисел в модифицированных кодах, когда под знак отводится не один, а два разряда:

$$
\begin{array}{r}
A_1^{\text{м}}{}_{\text{пр, обр, доп}} = 00.\ 1010101 \\
+ \\
A_2^{\text{м}}{}_{\text{пр, обр, доп}} = 00.\ 0111001 \\
\hline
(A_1 + A_2)^{\text{м}}_{\text{пр, обр, доп}} = \underline{01}.\ 0001110.
\end{array}
$$

Знаковые разряды результата сложения отличаются по значению, что и является показателем переполнения разрядной сетки. В дальнейшем без соответствующей коррекции этот результат не может быть использован в вычислениях. Введение нового масштабного коэффициента производится только над *прямыми кодами* чисел.

4.2. Сложение отрицательных чисел или чисел с разными знаками

Если в сложении участвуют отрицательные числа или числа с разными знаками, то операции выполняются в модифицированном обратном или дополнительном кодах. Предпочтение отдается дополнительному коду, так как использование дополнений для отрицательных чисел позволяет все арифметические операции свести либо просто к сложению, либо к серии сложений и сдвигов.

В табл. 4.1 представлены правила и примеры сложения.

Рассмотрим сложение двух чисел A_1 и A_2:

$$A_1 = +1101;$$
$$A_2 = -0101.$$

Произведем сложение чисел A_1 и A_2 в обратном и дополнительном кодах. Полученные результаты переведем в прямой код и сравним их.

Таблица 4.1

Вид кода	Правило	Пример
Модифицированный дополнительный код	Сложение осуществляется по правилу двоичной арифметики. Единица переноса, возникающая в третьем знаковом разряде суммы, отбрасывается	$A_1^{\text{м}}{}_{\text{доп}} = 00.1101$ + $A_2^{\text{м}}{}_{\text{доп}} = 11.1011$ $(A_1 + A_2)^{\text{м}}_{\text{доп}} = \not{1}00.1000$
Модифицированный обратный код	Сложение осуществляется по правилу двоичной арифметики. Единица переноса, возникающая в третьем знаковом разряде суммы, прибавляется к младшему разряду модуля, образуя циклический перенос	$A_1^{\text{м}}{}_{\text{обр}} = 00.1101$ + $A_2^{\text{м}}{}_{\text{обр}} = 11.1010$ $(A_1 + A_2)^{\text{м}}_{\text{обр}} = 100.0111$ $\quad\quad\quad

$$A_1 = -0001001;$$
$$A_2 = -0001110.$$

1) $A_1^{\text{м}}{}_{\text{пр}} = 11.\,0001001;$
$A_2^{\text{м}}{}_{\text{пр}} = 11.\,0001110.$

2) $A_1^{\text{м}}{}_{\text{обр}} = 11.\,1110110$
$+$
$A_2^{\text{м}}{}_{\text{обр}} = 11.\,1110001$

$(A_1 + A_2)^{\text{м}}_{\text{обр}} = 111.\,1100111$

$+1$

$(A_1 + A_2)^{\text{м}}_{\text{обр}} = 11.\,1101000;$
$(A_1 + A_2)^{\text{м}}_{\text{пр}} = 11.\,0010111.$

3) $A_1^{\text{м}}{}_{\text{доп}} = 11.\,1110111$
$+$
$A_2^{\text{м}}{}_{\text{доп}} = 11.\,1110010$

$(A_1 + A_2)^{\text{м}}_{\text{доп}} = 111.\,1101001$

$(A_1 + A_2)^{\text{м}}_{\text{пр}} = 11.\,0010110$
$+ \quad 1$

$11.\,0010111.$

Таким образом результаты совпали.

Контрольные вопросы

1. Привести пример и указать различие в изображении обычного и модифицированного кодов числа.
2. В чем заключаются причины использования модифицированных кодов чисел?
3. Какие действия необходимо выполнить с третьим знаковым разрядом результата при сложении чисел, представленных в обратном и дополнительном кодах?
4. В каких случаях возможно появление переполнения разрядной сетки при сложении в модифицированных кодах?

Контрольные задания

Выполнить сложение пар чисел в прямом, обратном и дополнительном модифицированных кодах, используя восьмиразрядный формат. Полученные результаты перевести в прямой код и сравнить:

1) $A_1 = +110101;$ $A_2 = -000001.$
2) $A_1 = -1010;$ $A_2 = +010101.$
3) $A_1 = +001011;$ $A_2 = -110101.$
4) $A_1 = -010100;$ $A_2 = -1110.$
5) $A_1 = +100101;$ $A_2 = +1101.$
6) $A_1 = -0110;$ $A_2 = +100101.$

4.3. Умножение чисел

В общем случае процесс умножения сводится к отысканию частичных произведений (ЧП) с последующим их сложением.

Произведем умножение двух чисел A и B:

$A = 10101$ — множимое;
$B = 101$ — множитель;

$$
\begin{array}{r}
A = 10101 \\
\times \\
B = 101 \\
\hline
10101 \text{ — ЧП1} \\
+\ 0000 \text{ — ЧП2} \\
10101 \text{ — ЧП3} \\
\hline
1101001 \text{ — произведение.}
\end{array}
$$

Возможны несколько вариантов умножения в зависимости от знака сомножителей:

$$A > 0,\ B > 0;\ A > 0,\ B < 0;\ A < 0,\ B > 0;\ A < 0,\ B < 0.$$

В общих чертах процесс умножения сводится к последовательному анализу разрядов множителя и в зависимости от их значений к получению очередного значения ЧП.

Чтобы понять принципы умножения, рассмотрим первые три варианта.

Вариант 1.

$$\boldsymbol{A > 0;\ B > 0.}$$

Произведем умножение двух чисел:

$$A = +15_{(10)} = +1111_{(2)};\ B = +9_{(10)} = +1001_{(2)}.$$

В табл. 4.2 приведен порядок действий при умножении.

Таблица 4.2

Порядок действий	Вычисления
1. Представление чисел в прямых кодах. Перед умножением необходимо выровнять разрядность сомножителей в сторону числа с большим количеством разрядов	$A_{пр} = 0.1111$ $B_{пр} = 0.1001$ $B_4 \ldots B_0$
2. Умножение кодов чисел, начиная с анализа младшего разряда множителя B_0. Если разряд множителя B_i содержит 1, то множимое A прибавляется к частичному произведению (до начала вычислений исходное ЧП, заполненное нулями, имеет разрядность, равную удвоенной разрядности сомножителей, включая знаковый разряд). После сложения осуществляется сдвиг нового ЧП на один разряд вправо. Если разряд множителя B_i содержит 0, то производится лишь сдвиг ЧП на один разряд вправо	$B_0 = 1$ Исх.ЧП = 00000 00000 $+\quad A = 01111$ ЧП = 01111 00000 ЧП $\rightarrow$ = 00111 10000 $B_1 = 0$ ЧП $\rightarrow$ = 00011 11000

Окончание табл. 4.2

Порядок действий	Вычисления
3. Анализ всех разрядов B, включая знаковый. Конечное значение ЧП получают в прямом коде. При этом необходимо помнить, что самый левый разряд — знаковый	$B_2 = 0$ ЧП = 00001 11100 $\rightarrow$ $B_3 = 1$ ЧП = 00001 11100 + A = 01111 ЧП = 10000 11100 ЧП = 01000 01110 $\rightarrow$ $B_4 = 0$ ЧП = 00100 00111 $\rightarrow$
4. Получение результата	$15 \cdot 9 = +135_{(10)}$ $AB = +10000111_{(2)} =$ $= 2^0 + 2^1 + 2^2 + 2^7 = 135_{(10)}$

Вариант 2.

$$A > 0;\ B < 0.$$

Произведем умножение двух чисел:

$$A = +15_{(10)} = +1111_{(2)};\quad B = -9_{(10)} = -1001_{(2)}.$$

Умножение выполняется по формуле

$$C_{доп} = A_{пр} \cdot (-B_{доп}) + (-A_{доп}).$$

В табл. 4.3 приведен порядок действий при умножении.

Таблица 4.3

Порядок действий	Вычисления
1. Образование необходимых для вычислений кодов чисел	$A_{пр} = 0.1111$ $-B_{доп} = 1.0111$ $-A_{доп} = 1.0001$
2. Умножение $A_{пр} \cdot (-B_{доп})$	1) $B_{0\ доп} = 1$ + ЧП = 00000 00000 $A_{пр} = 01111$ ЧП = 01111 00000 ЧП = 00111 10000 $\rightarrow$ 2) $B_{1\ доп} = 1$ ЧП = 00111 10000 + $A_{пр} = 01111$ ЧП = 10110 10000 ЧП = 01011 01000 $\rightarrow$

Окончание табл. 4.3

Порядок действий	Вычисления
	3) $B_{2\,доп} = 1$ ЧП = 01011 01000 $+$ $A_{пр} = 01111$ ЧП = 11010 01000 ЧП = 01101 00100 $\rightarrow$ 4) $B_{3\,доп} = 0$ ЧП = 00110 10010 5) $B_{4\,доп} = 1$ ЧП = 00110 10010 $+$ $A_{пр} = 01111$ ЧП = 10101 10010 ЧП = 01010 11001 $\rightarrow$
3. Вычисление $C_{доп}$ по формуле $C_{доп} = A_{пр} \cdot (-B_{доп}) + (-A_{доп})$	$A_{пр} \cdot (-B_{доп}) = 01010\ 11001$ $+$ $(-A_{доп}) = 10001$ $C_{доп} = 11011\ 11001$
4. Перевод результата в прямой код. Следует помнить, что самый левый разряд в разрядной сетке — знаковый	$C_{пр} = 1.0100\ 00111$
5. Получение результата	$15 \cdot (-9) = -135_{(10)}$ $(A) \cdot (-B) = -10000111_{(2)} =$ $= -135_{(10)}$.

Вариант 3.

$$A < 0;\ B > 0.$$

Произведем умножение двух чисел:

$$A = -15_{(10)} = -1111_{(2)};\quad B = +9_{(10)} = +1001_{(2)}.$$

Умножение выполняется по формуле

$$C_{доп} = (-A_{доп}) \cdot B_{пр} + (-B_{доп}).$$

В табл. 4.4 приведен порядок действий при умножении.

Таблица 4.4

Порядок действий	Вычисления
1. Образование необходимых для вычислений кодов чисел	$-A_{доп} = 1.0001$ $B_{пр} = 0.1001$ $-B_{доп} = 1.0111$

Окончание табл. 4.4

Порядок действий	Вычисления
2. Умножение $(-A_{доп}) \cdot B_{пр}$	1) $A_{0\,доп} = 1$ ЧП = 00000 00000 + $B_{пр} = 01001$ ЧП = 01001 00000 ЧП = 00100 10000 → 2) $A_{1\,доп} = 0$ ЧП = 00010 01000 → 3) $A_{2\,доп} = 0$ ЧП = 00001 00100 → 4) $A_{3\,доп} = 0$ ЧП = 00000 10010 → 5) $A_{4\,доп} = 1$ ЧП = 00000 10010 + $B_{пр} = 01001$ ЧП = 01001 10010 ЧП = 00100 11001 →
3. Вычисление $C_{доп}$ по формуле $C_{доп} = (-A_{доп}) \cdot B_{пр} + (-B_{доп})$	$(-A_{доп}) \cdot B_{пр} = 00100\ 11001$ + $(-B_{доп}) = 10111$ $C_{доп} = 11011\ 11001$
4. Перевод результата в прямой код. Следует помнить, что самый левый разряд в разрядной сетке — знаковый	$C_{пр} = 1.0100\ 00111$
5. Получение результата	$(-15) \cdot 9 = -135_{(10)}$ $(-A) \cdot B = -10000111_{(2)} =$ $= -135_{(10)}$.

Контрольные задания

Произвести умножение чисел с фиксированной запятой, предусмотрев три варианта сочетания знаков сомножителей:

1) $22 \cdot 8$;
2) $15 \cdot 2$;
3) $17 \cdot 3$;
4) $12 \cdot 7$;
5) $14 \cdot 5$;
6) $16 \cdot 9$;
7) $18 \cdot 6$;
8) $10 \cdot 5$.

4.4. Деление чисел

Обозначим делимое буквой D, делитель — буквой d, промежуточные частичные остатки — ЧО.

В общих чертах процесс деления сводится к сложению ЧО и d, имея при этом в виду, что они различны по знаку, т.е. сложение в кодах отражает процесс вычитания, при котором d представляется либо в прямом, либо в дополнительном кодах. Количество операций вычитания повторяется до тех пор, пока младший разряд делителя не сравняется с исходным младшим разрядом (ЧО).

Цифры частного определяются по следующему правилу: если очередной ЧО, полученный в результате сложения, положителен, то в соответствующий разряд частного заносится 1; если очередной ЧО отрицателен — заносится 0.

Знак последнего остатка должен совпадать со знаком делимого. При их несовпадении производится восстановление остатка. В зависимости от знака остатка к нему прибавляется делитель в прямом или дополнительном коде.

В табл. 4.5 приведен порядок действий при делении двух чисел:

$$D = 25_{(10)} = 11001_{(2)};$$
$$d = 3_{(10)} = 011_{(2)}.$$

Таблица 4.5

Порядок действий	Вычисления	Частное
1. Получение кодов исходных чисел. Предварительно выравниваем разрядность делимого и делителя в прямых кодах, увеличивая разрядность делителя до разрядности делимого	$D_{пр} = 0.11001$ $d_{пр} = 0.11$ $d_{пр} = 0.00011$ $d_{доп} = 0.11101$	—
2. Получение первого ЧО путем вычитания из предыдущего ЧО делителя. Исходным ЧО является делимое D	$D_{пр} = 0.00000\ 011001^*$ $+\ d_{доп} = 1.11101$ $ЧО_1 = 1.11101\ 011001^* < 0$	0
3. Сдвиг ЧО на один разряд влево	$\overleftarrow{ЧО} = 1.11010\ 11001^*0$	—
4. Получение второго ЧО путем вычитания из первого ЧО делителя. Значение делителя (в прямом или дополнительном коде) определяется по частичному остатку до сдвига — он должен иметь противоположный знак	$ЧО = 1.11010\ 11001^*0$ $+\ d_{пр} = 0.00011$ $ЧО_2 = 1.11101\ 11001^*0 < 0$	0.0

Окончание табл. 4.5

Порядок действий	Вычисления	Частное
5. Сдвиг ЧО на один разряд влево	$\overleftarrow{\text{ЧО}} = 1.11011\ 1001^{*}00$	—
6. Получение третьего ЧО и т.д. Вычислительный процесс продолжается до тех пор, пока младший разряд ЧО, отмеченный звездочкой (*), не совпадет с младшим разрядом делителя d	$\text{ЧО} = 1.11011\ 1001^{*}00$ $+\ d_{\text{пр}} = 0.00011$ $\text{ЧО}_3 = 1.11110\ 1001^{*}00 < 0$ $\text{ЧО} = 1.11101\ 001^{*}000$ $+\ \overleftarrow{d_{\text{пр}}} = 0.00011$ $\text{ЧО}_4 = 0.00000\ 001^{*}000 > 0$ $\text{ЧО} = 0.00000\ 01^{*}0000$ $+\ \overleftarrow{d_{\text{пр}}} = 1.11101$ $\text{ЧО}_5 = 1.11101\ 01^{*}0000 < 0$ $\text{ЧО} = 1.11010\ 1^{*}00000$ $+\ \overleftarrow{d_{\text{пр}}} = 0.00011$ $\text{ЧО}_6 = 1.11101\ 1^{*}00000 < 0$ $\text{ЧО} = 1.11011^{*}0\ 00000$ $+\ \overleftarrow{d_{\text{пр}}} = 0.00011$ $\text{ЧО}_7 = 1.11110^{*}\ 000000 < 0$	0.00 0.001 0.0010 0.00100 0.001000
7. Восстановление остатка. Значение разрядов ЧО старшей группы является остатком от деления.. В случае, если знак остатка не совпадает со знаком делителя, необходимо прибавить к нему делитель с противоположным знаком по отношению к последнему остатку. Сдвиг последнего ЧО перед восстановлением остатка не производится	$\text{ЧО}_7 = 1.11110$ $+\ d_{\text{пр}} = 0.00011$ $\text{ЧО} = 0.00001$	—
8. Получение результата	Остаток: $0.00001_{(2)} = +1_{(10)}$	$0.1000_{(2)} = +8_{(10)}$

Контрольные задания

Произвести деление чисел с фиксированной запятой:

1) 15 : 6;
2) 24 : 9;
3) 18 : 7;
4) 13 : 4;
5) 30 : 2;
6) 25 : 8;
7) 13 : 5;
8) 12 : 7.

Глава 5
АРИФМЕТИЧЕСКИЕ ДЕЙСТВИЯ НАД ЧИСЛАМИ С ПЛАВАЮЩЕЙ ЗАПЯТОЙ

5.1. Сложение чисел

При сложении чисел с плавающей запятой результат определяется как сумма мантисс слагаемых с общим для слагаемых порядком.

Если знаки обеих мантисс одинаковы, то они складываются в прямых кодах, если разные — в дополнительном или обратном кодах.

В табл. 5.1 приведен порядок действий при сложении чисел с разными знаками:

$$A_1 = -20_{(10)} = -0{,}10100 \cdot 10^{+101};$$
$$A_2 = +5{,}5_{(10)} = +0{,}1011 \cdot 10^{+11}.$$

Таблица 5.1

Порядок действий	Вычисления
1. Выравнивание порядков чисел в сторону большего	$A_2' = +0{,}0010110 \cdot 10^{101}$
2. Выравнивание разрядности мантисс слагаемых. При выравнивании разрядности нули добавляются в конце мантиссы числа с меньшей разрядностью, при этом величина числа не меняется, так как учитывается его порядком	$A_1 = -0{,}101000 \cdot 10^{101}$
3. Сложение мантисс в дополнительных кодах	$M^{м}_{1\,доп} = 11.011000$ $+\ M^{м}_{2\,доп} = 00.001011$ $(M_1 + M_2)^{м}_{доп} = 11.100011$
4. Перевод мантиссы результата в прямой код	$(M_1 + M_2)^{м}_{пр} = 11{,}011101$
5. Обобщение полученного результата вычислений (результат имеет одинаковый порядок)	$A_1 + A_2 = -0.011101 \cdot 10^{101}$
6. Нормализация результата	$A_1 + A_2 = -0{,}11101 \cdot 10^{100}$
7. Получение результата	$A_1 + A_2 = -1101{,}1_{(2)} = -14{,}5_{(10)}$

Контрольные задания

Произвести сложение чисел, представленных в форме с плавающей запятой:

1) $A_1 = +9{,}25_{(10)} = +0{,}100101 \cdot 10^{100}$;
   $A_2 = -26_{(10)} = -0{,}1101 \cdot 10^{101}$.
2) $A_1 = -21{,}25_{(10)} = -0{,}1010101 \cdot 10^{101}$;
   $A_2 = +15_{(10)} = +0{,}1111 \cdot 10^{100}$.
3) $A_1 = -0{,}110110 \cdot 10^{100}$;
   $A_2 = +0{,}100101 \cdot 10^{101}$.
4) $A_1 = +14{,}5_{(10)}$;
   $A_2 = -18{,}75_{(10)}$.
5) $A_1 = +0{,}10101 \cdot 10^{011}$;
   $A_2 = -0{,}1101 \cdot 10^{010}$.
6) $A_1 = -23{,}5_{(10)}$;
   $A_2 = -5{,}75_{(10)}$.

5.2. Умножение чисел

При умножении чисел с плавающей запятой произведение определяется как произведение мантисс сомножителей, а порядок — как сумма их порядков. При умножении разрядность сомножителей не выравнивается, анализ знакового разряда не производится.

Рассмотрим варианты умножения чисел в зависимости от знаков сомножителей.

Вариант 1.

$$A > 0;\ B > 0.$$

В табл. 5.2 приведен порядок действий при умножении двух чисел:

$$A = 5_{(10)} = 101_{(2)} = 0{,}101 \cdot 10^{11};$$

$$B = 3_{(10)} = 011_{(2)} = 0{,}11 \cdot 10^{10}.$$

Таблица 5.2

Порядок действий	Вычисления
1. Определение порядка результата, равного сумме порядков сомножителей	$p = p_A + p_B = 11 + 10 = 101$
2. Умножение мантисс сомножителей: $M_A = 0.101$; $M_B = 0.11$	$M_{B_0} = 1$ ЧП $= 0000\ 0000$ $+\ A_{\text{пр}} = 0101$ ЧП $= 0101\ 0000$ ЧП $= 0010\ 1000$ $\rightarrow$

Окончание табл. 5.2

Порядок действий	Вычисления
	$M_{B_1} = 1$ ЧП = 0010 1000 + $A_{\text{пр}} = 0101$ ЧП = 0111 1000 ЧП $\rightarrow$ = 0011 1100
3. Обобщение результата	$A \cdot B = 0{,}0111100 \cdot 10^{101}$
4. Нормализация результата	$A \cdot B = 0{,}111100 \cdot 10^{100}$
5. Получение результата	$A \cdot B = +1111_{(2)} = 15_{(10)}$

Вариант 2.

$$A > 0;\ B < 0.$$

Произведем умножение двух чисел:

$$A = +25_{(10)} = +0{,}11001 \cdot 10^{101};$$
$$B = -3_{(10)} = -0{,}11 \cdot 10^{10}.$$

Умножение выполняется по формуле

$$M_{C\,\text{доп}} = M_{A\,\text{пр}} \cdot (-M_{B\,\text{доп}}) + (-M_{A\,\text{доп}}).$$

В табл. 5.3 приведен порядок действий при умножении двух чисел с разными знаками.

Таблица 5.3

Порядок действий	Вычисления
1. Определение порядка результата	$p = p_A + p_B = 101 + 10 = 111$
2. Образование необходимых кодов мантисс чисел	$-M_{B\,\text{доп}} = 1.01$ $-M_{A\,\text{доп}} = 1.00111$ $-M_{A\,\text{пр}} = 1.11001$
3. Умножение мантисс: $M_{A\,\text{пр}} \cdot (-M_{B\,\text{доп}})$	$M_{B_0\text{доп}} = 1$ ЧП = 000000 000000 + $M_{A\,\text{пр}} = 011001$ ЧП = 011001 000000 ЧП $\rightarrow$ = 001100 100000 $M_{B_1\text{доп}} = 0$ ЧП $\rightarrow$ = 000110 010000

Окончание табл. 5.3

Порядок действий	Вычисления
4. Вычисление $M_{C\,доп}$ по формуле $M_{C\,доп} = M_{A\,пр} \cdot (-M_{B\,доп}) + (-M_{A\,доп})$	$M_{A\,пр} \cdot (-M_{B\,доп}) = 000110\ 010000$ $+\ (-M_{A\,доп}) = 100111$ $M_{C\,доп} = 101101\ 010000$
5. Перевод мантиссы результата в прямой код	$M_{C\,пр} = 1.10010\ 101111$ $+\ 1$ $1.10010\ 110000$
6. Получение результата	$A \cdot B = -0{,}10010110000 \cdot 10^{111} =$ $= -1001011_{(2)} = -75_{(10)}$

Вариант 3.

$$A < 0;\ B > 0.$$

Умножение производится аналогично второму варианту (см. табл. 5.3).

Контрольные задания

Произвести умножение чисел в форме с плавающей запятой, предусмотрев три возможных сочетания знаков сомножителей.

1) $13 \cdot 5$;
2) $25 \cdot 4$;
3) $9 \cdot 7$;
4) $16 \cdot 8$;
5) $21 \cdot 6$;
6) $14 \cdot 5$;
7) $8 \cdot 7$;
8) $19 \cdot 5$.

5.3. Деление чисел

При делении чисел, представленных в форме с плавающей запятой, частное определяется как деление делимого D на делитель d, а порядок — как разность порядков делимого и делителя. Деление мантисс чисел производится по тем же правилам, что и деление чисел с фиксированной запятой.

В табл. 5.4 приведен порядок действий при делении двух чисел:

$$D = 17_{(10)} = 10001_{(2)} = 0{,}10001 \cdot 10^{101};$$

$$d = 9_{(10)} = 1001_{(2)} = 0{,}1001 \cdot 10^{100}.$$

Необходимо иметь в виду, что при вычислениях исходное делимое D должно иметь удвоенную разрядность и соответствующий ему порядок:

$$D = 0{,}0000\ 10001 \cdot 10^{1001}.$$

Таблица 5.4

Порядок действий	Вычисления	Частное
1. Определение порядка результата, равного разности порядков делимого и делителя	$p = p_D - p_d = 1001 - 100 = 101$	—
2. Получение кодов мантисс делителя	$M_{d\,\text{пр}} = 0.1001$ $-M_{d\,\text{доп}} = 1.0111$	—
3. Деление мантисс. Вычислительный процесс продолжается до тех пор, пока младший разряд ЧО, отмеченный звездочкой (*), не совпадет с младшим разрядом делителя d	1) $M_{D\,\text{пр}} = 0.0000\ 10001^*$ $+M_{d\,\text{доп}} = 1.0111$ $\text{ЧО}_1 = 1.0111\ 10001^* < 0$ 2) $\text{ЧО} \overleftarrow{} = 0.1111\ 0001^*0$ $+\ M_{d\,\text{пр}} = 0.1001$ $\text{ЧО}_2 = 1.1000\ 0001^*0 < 0$ 3) $\text{ЧО} \overleftarrow{} = 1.0000\ 001^*00$ $+\ M_{d\,\text{пр}} = 0.1001$ $\text{ЧО}_3 = 1.1001\ 001^*00 < 0$ 4) $\text{ЧО} \overleftarrow{} = 1.0010\ 01^*000$ $+\ M_{d\,\text{пр}} = 0.1001$ $\text{ЧО}_4 = 1.1011\ 01^*000 < 0$ 5) $\text{ЧО} \overleftarrow{} = 1.0110\ 1^*0000$ $+\ M_{d\,\text{пр}} = 0.1001$ $\text{ЧО}_5 = 1.1111\ 1^*0000 < 0$ 6) $\text{ЧО} \overleftarrow{} = 1.1111^*\ 00000$ $+\ M_{d\,\text{пр}} = 0.1001$ $\text{ЧО}_6 = \underline{0.1000}\ 00000 > 0$ Остаток	0, 0,0 0,00 0,000 0,0000 0,00001

Таким образом получаем следующий результат:
частное равно $17 : 9 = +0{,}00001 \cdot 10^{101} = 0{,}1 \cdot 10^{1} = +1_{(10)}$;
остаток равен $+1000_{(2)} = +8_{(10)}$.

Контрольные задания

Произвести деление двух чисел, представленных в форме с плавающей запятой:

1) 17 : 5;
2) 22 : 9;
3) 26 : 7;
4) 13 : 6.

Глава 6

АРИФМЕТИЧЕСКИЕ ДЕЙСТВИЯ НАД ЧИСЛАМИ В ДВОИЧНО-ДЕСЯТИЧНОЙ СИСТЕМЕ СЧИСЛЕНИЯ

6.1. Особенности сложения в двоично-десятичной системе счисления

Для изображения цифр в двоично-десятичной системе счисления отводится четыре разряда — *тетрада.*

Рассмотрим два примера.

Пример 1. Произведем сложение двух чисел, в результате которого будет получена тетрада суммы меньше или равная $9_{(10)}$:

$$\begin{array}{r} A = 5_{(10)} = 0101_{(2)\ (2-10)} \\ + \\ B = 4_{(10)} = 0100_{(2)\ (2-10)} \\ \hline (A + B) = 9_{(10)} = 1001_{(2)\ (2-10)}. \end{array}$$

В этом случае код результата как в двоичной, так и двоично-десятичной системе счисления одинаков.

Пример 2. Произведем сложение двух чисел, в результате которого будет получена сумма больше $9_{(10)}$:

$$\begin{array}{r} A = 7_{(10)} = 0111_{(2)\ (2-10)} \\ + \\ B = 8_{(10)} = 1000_{(2)\ (2-10)} \\ \hline (A + B) = 15_{(10)} = 1111_{(2)}. \end{array}$$

Результат суммирования получен в двоичной системе счисления. Для преобразования числа $15_{(10)}$ в двоично-десятичный код необходимо иметь две тетрады — $0001\ 0101_{(2-10)}$. Для получения младшей двоично-десятичной тетрады числа $15_{(10)}$ — числа $5_{(10)}$ из $15_{(10)} - 1111_{(2)}$ необходимо вычесть $10_{(10)} - 1010_{(2)}$. Соответственно в двоичном коде это будет выглядеть следующим образом:

$$1111_{(2)} - 1010_{(2)} = 0101_{(2)}.$$

В вычислительных устройствах операция вычитания заменяется операцией сложения в обратном или дополнительном коде. Дополнительным кодом числа $10_{(10)}$ является число $6_{(10)} = 0110_{(2)\ (2-10)}$.

Таким образом, если результат потетрадного сложения больше 9, необходим перевод из двоичной в двоично-десятичную систему счисления — введение коррекции результата или избытка $6_{(10)}$:

$$\begin{array}{rl} & 15_{(10)} = 1111_{(2)} \quad \text{— результат в двоичной системе счисления} \\ + & 6_{(10)} = 0110_{(2)} \quad \text{— избыток 6} \\ \hline & 1\ \underline{0101}_{(2-10)} \quad \text{— результат в двоично-десятичной системе} \\ & \qquad\qquad\qquad\qquad\quad \text{счисления.} \\ & 1 \quad 5_{(10)} \end{array}$$

Таблица 6.1

Десятичное число	Двоичный поразрядный эквивалент				
	5	4	3	2	1
8		1	0	0	0
9		1	0	0	1
10		1	0	1	0
11		1	0	1	1
12		1	1	0	0
13		1	1	0	1
14		1	1	1	0
15		1	1	1	1
16	1	0	0	0	0
17 и т.д.	1	0	0	0	1

Признаком чисел больше 9 является одновременное наличие единиц в 4-ом и 2-ом или 4-ом и 3-ем разрядах, или для чисел больше $15_{(10)}$ наличие межтетрадного переноса.

В табл. 6.1 отражены характерные особенности чисел больше 9 в двоичной системе счисления.

6.2. Сложение чисел с одинаковыми знаками

Сложение чисел с одинаковыми знаками может быть выполнено двумя способами, которые представлены в табл. 6.2.

Таблица 6.2

Первый способ	Второй способ
Избыток 6 прибавляется к тем тетрадам результата, значения которых больше 9: $A = 7_{(10)} = 0000\ 0111_{(2-10)}$ $+\ B = 18_{(10)} = 0001\ 1000_{(2-10)}$ $A + B = 0001\ 1111_{(2-10)}$ Коррекция (+6) 0110 $A + B = 25_{(10)} = 0010\ 0101_{(2-10)}$	Одно из исходных слагаемых, участвующих в сложении, берется с избытком 6: $A = 7_{(10)} = 0000\ 0111_{(2-10)}$ $+\ (+6) = 0110\ 0110$ $A_{(+6)} = 0110\ 1101$ Затем производится сложение числа с избытком и второго слагаемого: $A_{(+6)} = 0110\ 1101$ $+\ B = 0001\ 1000$ $A + B = 1000\ 0101$ $-$ $\uparrow 1$

Окончание табл. 6.2

Первый способ	Второй способ
	Результат подвергается анализу на наличие межтетрадных переносов. Если межтетрадный перенос присутствует, то вид этой тетрады сохраняется; если отсутствует, то производится коррекция результата, которая сводится к вычитанию из тетрады числа $6_{(10)}$ или прибавлению к ней дополнительного кода $6_{(10)}$ — числа $10_{(10)} = 1010_{(2)}$, при этом межтетрадный перенос не учитывается: $A + B = 1000\ 0101_{(2)}$; $+\ (+6)_{доп} = 0101$; Результат $0010\ 0101_{(2-10)}$ $2\quad 5_{(10)}$

6.3. Сложение чисел с разными знаками

Правила сложения двоично-десятичных чисел с разными знаками аналогичны правилам сложения чисел с одинаковыми знаками. Однако необходимо иметь в виду, что перед сложением отрицательное слагаемое переводится в обратный код.

Окончательный результат сложения зависит от нескольких факторов: знака результата и наличия межтетрадных переносов при сложении.

В зависимости от этих факторов необходимо выполнить определенные действия (табл. 6.3).

Таблица 6.3

Результат сложения отрицательный		Результат сложения положительный	
Результат из обратного кода переводится в прямой	Межтетрадный перенос отсутствует		
	Обратный код II I Тетрада Тетрада – – Прямой код II I	Результат сохраняется в прямом коде. Из тетрады, в которой отсутствует перенос, вычитается избыток 6	Прямой код II I – 1 – 6 II I

Окончание табл. 6.3

Результат сложения отрицательный		Результат сложения положительный	
	Межтетрадный перенос присутствует		
К тетраде, в которой возникает межтетрадный перенос, прибавляется избыток 6. Далее результат переводится в прямой код. Единица переноса, возникающая в старшей тетраде предварительного результата в обратном коде, отбрасывается	Обратный код II I x 1 + 6 Обратный код II I Прямой код II I	Результат сохраняется в прямом коде. Единица переноса из самой старшей тетрады прибавляется к младшему разряду младшей тетрады	Прямой код II I 1 1 1 + II I

Рассмотрим несколько примеров сложения чисел.

Пример 1. $A = -123_{(10)} = -0001\ 0010\ 0011_{(2-10)}$;
$B = +95_{(10)} = +0000\ 1001\ 0101_{(2-10)}$.

$$\begin{array}{r} A_{обр} = 1110\ 1101\ 1100 \\ +\ B_{пр} = 0000\ 1001\ 0101 \\ \hline (A+B)_{обр} = 1111\ 0111\ 0001 \\ \leftarrow\ \ x\ \ x \\ (+6) = \quad 0110\ 0110 \\ \hline (A+B)_{обр} = 1111\ 1101\ 0111 \\ (A+B)_{пр} = 0000\ 0010\ 1000 \end{array}$$

$A + B = -28_{(10)}$.

Пример 2. $A = +123_{(10)} = +0001\ 0010\ 0011_{(2-10)}$;
$B = -95_{(10)} = -0000\ 1001\ 0101_{(2-10)}$.

$$\begin{array}{r} A_{пр} = 0001\ 0010\ 0011 \\ +\ B_{обр} = 1111\ 0110\ 1010 \\ \hline (A+B)_{пр} = 1\ 0000\ 1000\ 1101 \\ x\ \ -\ \ -\ \ 1 \\ \text{+} \end{array}$$

$$
\begin{array}{rl}
(A+B)_{\text{пр}} = & 0000\ 1000\ 1110 \\
+\quad (-6) = & \quad\ 0110\ 0110 \\
\hline
(A+B)_{\text{пр}} = & 0000\ 0010\ 1000 \\
A+B = & +28_{(10)}.
\end{array}
$$

Пример 3. Первый способ:

$$
\begin{array}{rl}
A = 123_{(10)} = & 0001\ 0010\ 0011 \\
+\ B = 95_{(10)} = & 0000\ 1001\ 0101 \\
\hline
(A+B) = & 0001\ \underline{\underline{1011}}\ 1000 \\
+\quad (+6) = & \quad\ 0110 \\
\hline
(A+B) = 218_{(10)} = & 0010\ 0001\ 1000.
\end{array}
$$

Второй способ:

$$
\begin{array}{rl}
A = 123_{(10)} = & 0001\ 0010\ 0011 \\
+\quad (+6) = & 0110\ 0110\ 0110 \\
\hline
A_{(+6)} = & 0111\ 1000\ 1001 \\
+\ B = & 0000\ 1001\ 0101 \\
\hline
A_{(+6)} + B = & 1000\ 0001\ 1110 \\
+ \quad \leftarrow\boxed{-}\ \boxed{\chi}\ \boxed{-} & \\
\quad (-6) \quad 0110\ 0110 & \\
\hline
(A+B) = 218 = & 0010\ 0001\ 1000
\end{array}
$$

или

$$
\begin{array}{rl}
 & 1000\ 0001\ 1110 \\
+ & \\
(+10) & \quad 1010 \qquad 1010 \\
\hline
 & 0010\ 0001\ 1000. \\
 & \leftarrow\boxed{\chi} \qquad \boxed{\chi}
\end{array}
$$

Контрольные вопросы

1. С какой десятичной цифры начинается различие в изображении чисел в двоичной и двоично-десятичной системах счисления? Почему это происходит?
2. Какие условия влияют на необходимость введения коррекции результата при сложении чисел в двоично-десятичной системе счисления? Приведите примеры, когда результаты сложения находятся в диапазоне чисел от $10_{(10)}$ до $15_{(10)}$ и больше $15_{(10)}$.
3. Почему при получении результата сложения больше $15_{(10)}$ (например, $16_{(10)}$) недостаточно анализа разрядов младшей тетрады?
4. Назвать дополнительные коды чисел $10_{(10)}$ и $6_{(10)}$.
5. Какие факторы влияют на использование чисел $6_{(10)}$ и $10_{(10)}$ при коррекции результата сложения чисел?

Контрольные задания

Произвести сложение чисел в двоично-десятичной системе счисления, рассмотрев четыре варианта сочетания знаков слагаемых:

1) 212; 12.
2) 376; 18.
3) 65; 39.
4) 105; 427.
5) 87; 253.

Раздел II
ЛОГИЧЕСКИЕ ОСНОВЫ ЦИФРОВОЙ СХЕМОТЕХНИКИ

Глава 7
ОСНОВНЫЕ ПОНЯТИЯ АЛГЕБРЫ ЛОГИКИ

7.1. Общие сведения

Все схемы вычислительных устройств построены на элементах, выполняющих логические операции. Такие элементы принято называть логическими. Они используются для оценки и решения задач алгебры логики (исчисления высказываний).

Под *высказыванием* понимается любое утверждение, о котором можно сказать, что оно истинно, или что оно ложно. Одно из высказываний принимают за 1, другое за 0.

При конкретной физической реализации логических состояний используются элементы, для которых характерны два устойчивых состояния.

Например, электрическое реле — замкнуто, разомкнуто; элемент индикации — светится, не светится и т.д.

В логике не требуется знания абсолютного значения величины, поэтому физическая величина, подвергаемая логическим преобразованиям, называется *переменной*, или *аргументом*, и представляется как более положительная (*H*) или менее положительная (*L*). Этим двум значениям, называемым *логическими уровнями*, условно присваиваются значения 1 и 0, или наоборот 0 и 1, в зависимости от принятого соглашения.

На рис. 7.1, *а* представлены логические уровни сигналов для положительной логики, а на рис. 7.1, *б* — для отрицательной.

В дальнейшем будут использоваться соглашения положительной логики.

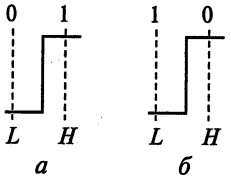


Рис. 7.1



Рис. 7.2

Логические переменные могут подвергаться различным преобразованиям с использованием *логических элементов*. Такие преобразования описываются с помощью переключательных функций, в которых используются различные буквы латинского алфавита: A, B, C, D, X, Y, x_1, x_2, x_3 и т.д.

Функция от входных переменных называется *переключательной*, если она так же, как ее аргументы, принимает два значения: логической единице (лог. 1) или логического нуля (лог. 0).

На рис. 7.2 представлен условный логический элемент, реализующий определенную функцию в зависимости от значений входных переменных.

Любая функция может быть задана двумя способами: в виде формул и в виде таблиц истинности.

Таблица 7.1

Переменные		Выходная функция
A	B	$f(AB)$
0	0	0
0	1	1
1	0	1
1	1	1

В таблице истинности (табл. 7.1) рассматриваются значения функции в зависимости от сочетания переменных.

Число сочетаний переменных определяется как 2^n, где n — число переменных.

В нашем случае возможны четыре сочетания двух переменных.

7.2. Переключательные функции одной и двух переменных

Переключательные функции одного аргумента представлены в табл. 7.2, двух аргументов — в табл. 7.3.

Таблица 7.2

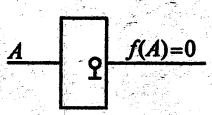
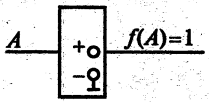
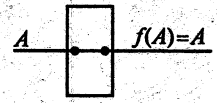
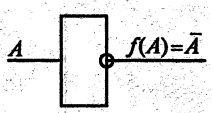
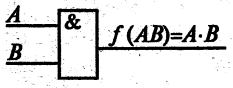
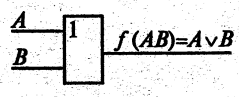
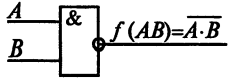
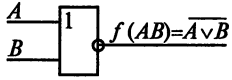
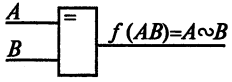
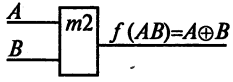
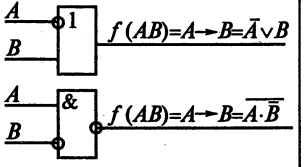
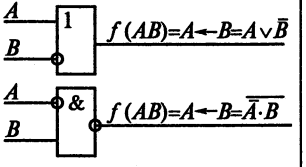
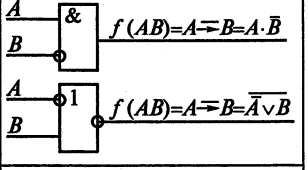
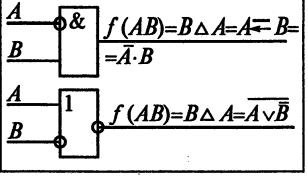
Название функции	Представление функции в виде			Определение функции	Условное обозначение
	формулы	таблицы истинности			
		A	$f(A)$		
Константа 0	$f(A) = 0$	0 1	0 0	При любом значении A выходная функция равна лог. 0	
Константа 1	$f(A) = 1$	0 1	1 1	При любом значении A выходная функция равна лог. 1	
Переменная A	$f(A) = A$	0 1	0 1	Значение функции равно переменной A	
Инверсия A	$f(A) = \overline{A}$	0 1	1 0	Значение функции противоположно (инверсно) входной переменной A	

Таблица 7.3

Название функции	Символика функции	Условная «связка»	Понятие логической функции	Вид функции двух аргументов	Таблица истинности			Условное графическое обозначение (УГО) логического элемента
					A	B	$f(AB)$	
Конъюнкция (логическое умножение)	•; ∧	И	Функция истинна — равна 1, если **и** A **и** B истинны — равны 1. Совпадение 1 на входах элемента приводит к появлению на выходе 1. Достаточно 0 на одном из входов элемента для появления на выходе 0	$f(AB) = A \cdot B = A \wedge B$	0 0 1 1	0 1 0 1	0 0 0 1	
Дизъюнкция (логическое сложение)	+; ∨	ИЛИ	Функция истинна — равна 1, если **или** A, **или** B, **или** A и B истинны — равны 1. Совпадение 0 на входах элемента приводит к появлению на выходе 0. Достаточно 1 на одном из входов элемента для появления на выходе 1	$f(AB) = A \vee B$	0 0 1 1	0 1 0 1	0 1 1 1	

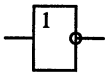
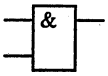
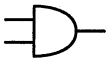
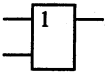
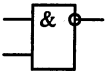
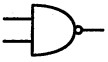
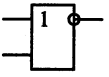
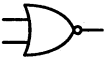
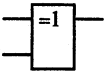
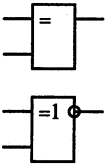
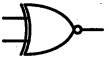
Операция Шеффера (штрих Шеффера). Функция является инверсией функции «Конъюнкция»	/	И-НЕ	Функция ложна — равна 0, если и A и B истинны — равны 1. Совпадение 1 на входах элемента приводит к появлению на выходе 0. Достаточно 0 на одном из входов элемента для появления на выходе 1	$f(AB) = A/B = \overline{A \cdot B}$	0 0 1 1	0 1 0 1	1 1 1 0	
Стрелка Пирса (функция Вебба). Функция является инверсией функции «Дизъюнкция»	↓	ИЛИ-НЕ	Функция ложна — равна 0, если хотя бы одна из переменных истинна или функция истинна, если обе переменные ложны	$f(AB) = A \downarrow B = \overline{A \vee B}$	0 0 1 1	0 1 0 1	1 0 0 0	
Эквивалентность (равнозначность)	∾ ; =	—	Функция истинна — равна 1, когда значения переменных совпадают по изображению	$f(AB) = A \backsim B = \overline{A} \cdot \overline{B} \vee AB$	0 0 1 1	0 1 0 1	1 0 0 1	
Сложение по модулю 2 (исключающее ИЛИ, неравнозначность). Функция является инверсией функции «Эквивалентность»	⊕; = 1; ∀; $m2$	—	Функция истинна — равна 1, если переменные не совпадают по изображению. Реализация функции соответствует правилу двоичного сложения без учета единицы переноса	$f(AB) = A \oplus B = \overline{A}B \vee A\overline{B}$	0 0 1 1	0 1 0 1	0 1 1 0	

Окончание табл. 7.3

Название функции	Символика функции	Условная «связка»	Понятие логической функции	Вид функции двух аргументов	Таблица истинности			Условное графическое обозначение (УГО) логического элемента
					A	B	$f(AB)$	
Импликация (от A к B)	$\rightarrow$	—	Это высказывание ложно только в том случае, когда A истинно, а B ложно, и истинно во всех остальных случаях	$f(AB)=A\rightarrow B=$ $=\overline{A}\vee B=\overline{A\cdot\overline{B}}$	0 0 1 1	0 1 0 1	1 1 0 1	
Импликация (от B к A)	$\leftarrow$	—	Это высказывание ложно только в том случае, когда B истинно, а A ложно, и истинно во всех остальных случаях	$f(AB)=A\leftarrow B=$ $=A\vee\overline{B}=\overline{\overline{A}\cdot B}$	0 0 1 1	0 1 0 1	1 0 1 1	
Запрет по B. Функция является инверсией функции «Импликация» (от A к B)	Δ; $\not\rightarrow$; $\nrightarrow$	—	Функция истинна, когда A истинно, а B ложно, и ложна во всех остальных случаях	$f(AB)=A\not\rightarrow B=$ $=A\cdot\overline{B}=\overline{\overline{A}\vee B}$	0 0 1 1	0 1 0 1	0 0 1 0	
Запрет по A. Функция является инверсией функции «Импликация» (от B к A)	Δ; $\not\leftarrow$; $\nleftarrow$	—	Функция истинна, когда B истинно, а A ложно, и ложна во всех остальных случаях	$f(AB)=B\not\leftarrow A=$ $=\overline{A}\cdot B=\overline{A\vee\overline{B}}$	0 0 1 1	0 1 0 1	0 1 0 0	

В табл. 7.4 представлены зарубежные аналоги условных графических обозначений логических элементов.

Таблица 7.4

Логическая функция	Отечественное	Зарубежное
НЕ		
И		
ИЛИ		
И-НЕ		
ИЛИ-НЕ		
Сложение по модулю 2 (Исключающее ИЛИ)		
Эквивалентность (Исключающее ИЛИ-НЕ)		

Контрольные вопросы

1. Дать определение физической величины, подвергаемой логическим преобразованиям.
2. Каковы различия при представлении величины в положительной и отрицательной логике?
3. Какие действия выполняет логический элемент по отношению к входным переменным?
4. Какими способами может быть задана функция, реализуемая логической схемой?
5. Какие существуют пары взаимно инверсных функций двух аргументов?

Контрольные задания

Составить таблицы для функций по аналогии с приведенным примером (табл. 7.5).

Пример. $f(AB)=\left[\left(A\cdot\overline{B}\right)/\left(\overline{A}\vee B\right)\right]\circ\left[\left(\overline{A\downarrow B}\right)\leftarrow\left(A\oplus\overline{B}\right)\right]$.

Таблица 7.5

A	B	$\overline{A}$	$\overline{B}$	$A\cdot\overline{B}$	$\overline{A}\vee B$	1/2	$\overline{A\downarrow B}$	$A\oplus\overline{B}$	$4\leftarrow 5$	$3\circ 6$
				1	2	3	4	5	6	7
0	0	1	1	0	1	0	0	1	0	1
0	1	1	0	0	1	0	1	0	1	0
1	0	0	1	1	0	0	1	0	1	0
1	1	0	0	0	1	0	1	1	1	0

1) $f(AB)=(\overline{A\vee B})/(A\cdot B)$;

2) $f(AB)=(A\cdot B)\downarrow(A\vee B)$;

3) $f(AB)=(A\circ B)\rightarrow(A\oplus B)$;

4) $f(AB)=(\overline{A})\leftarrow[(A\,\Delta\,B)\vee(B\,\Delta\,A)]$;

5) $f(ABC)=(\overline{A}/C)\circ(B\vee A)$;

6) $f(X_1X_2X_3)=\left[(X_1\oplus\overline{X}_2)\rightarrow(X_2/X_3)\right]\circ(X_1\downarrow X_2)$;

7) $f(X_1X_2X_3)=\left[\overline{X}_2\cdot(\overline{X}_1\wedge X_2)\right]\vee(X_2\cdot\overline{X}_3)$;

8) $f(X_1X_2X_3)=\left[(X_1\vee\overline{X}_3)\cdot(X_2\rightrightarrows X_3)\right]/\left[(\overline{X}_3)\vee(\overline{X_1\downarrow X_2})\right]$.

7.3. Тождества и законы алгебры логики

Для преобразования логических выражений, содержащих одну переменную, пользуются следующими легко доказываемыми *тождествами*:

$$\begin{gathered}A\vee 1=1;\quad A\cdot 1=A;\quad A\vee\overline{A}=1;\\ A\vee 0=A;\quad A\cdot 0=0;\quad A\cdot\overline{A}=0;\\ A\vee A\vee A\vee\ldots\vee A=A;\\ A\cdot A\cdot A\cdot\ldots\cdot A=A.\end{gathered}\qquad(7.1)$$

Для преобразования логических выражений, содержащих несколько переменных, необходимо соблюдать определенный порядок, который представляется в виде законов алгебры логики.

***Переместительный закон (коммутативности)*:**

1) $A\vee B=B\vee A$;
2) $A\cdot B=B\cdot A$.

***Сочетательный закон (ассоциативности)*:**

1) $(C\vee A)\vee B=B\vee(C\vee A)$;
2) $(C\cdot A)\cdot B=B\cdot(C\cdot A)$.

Распределительный закон (дистрибутивности):

$$1)\ (A \vee B) \cdot C = AC \vee BC;$$
$$2)\ A \cdot B \vee C = (A \vee C) \cdot (B \vee C), \qquad (7.2)$$

а для четырех переменных

$$A \cdot B \cdot C \vee D = (A \vee D) \cdot (B \vee D) \cdot (C \vee D).$$

Закон двойного отрицания:

$$\overline{\overline{A}} = A.$$

Закон инверсии или правило де Моргана (двойственности):

$$1)\ \overline{A \vee B} = \overline{A} \cdot \overline{B}; \qquad (7.3)$$
$$2)\ \overline{A \cdot B} = \overline{A} \vee \overline{B}. \qquad (7.4)$$

Используя закон двойного отрицания, можно преобразовать закон инверсии в более удобный вид:

$$A \vee B = \overline{\overline{A} \cdot \overline{B}}; \qquad (7.5)$$
$$A \cdot B = \overline{\overline{A} \vee \overline{B}}. \qquad (7.6)$$

Докажем справедливость равенств (7.3) и (7.4) закона инверсии. На этом примере рассмотрим один из способов доказательства равенств с помощью таблиц истинности. Заполним таблицу для всех возможных сочетаний переменных и сравним значения столбцов для левой и правой частей уравнения. Если значения совпадут, равенство считается доказанным (табл. 7.6).

Таблица 7.6

A	B	$\overline{A}$	$\overline{B}$	$A \cdot B$	$\overline{A \cdot B}$	$A \vee B$	$\overline{A \vee B}$	$\overline{A} \cdot \overline{B}$	$\overline{A} \vee \overline{B}$
0	0	1	1	0	1	0	1	1	1
0	1	1	0	0	1	1	0	0	1
1	0	0	1	0	1	1	0	0	1
1	1	0	0	1	0	1	0	0	0

Совпадение

Табл. 7.7 доказывает справедливость распределительного закона (7.2):

$$A \cdot B \vee C = (A \vee C) \cdot (B \vee C).$$

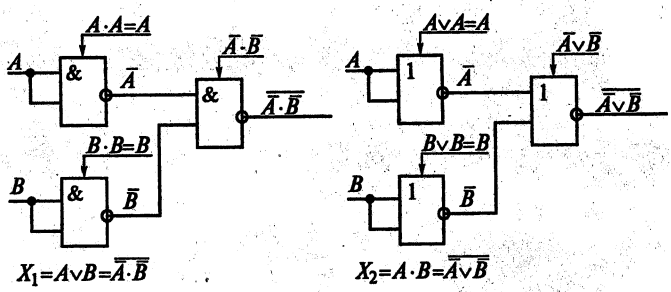
Необходимо отметить широкое практическое использование закона инверсии в тех случаях, когда необходимо реализовать функ-

Таблица 7.7

A	B	C	$\frac{A \cdot B}{D}$	$D \vee C$	$\frac{A \vee C}{E}$	$\frac{B \vee C}{F}$	$E \cdot F$
0	0	0	0	0	0	0	0
0	0	1	0	1	1	1	1
0	1	0	0	0	0	1	0
0	1	1	0	1	1	1	1
1	0	0	0	0	1	0	0
1	0	1	0	1	1	1	1
1	1	0	1	1	1	1	1
1	1	1	1	1	1	1	1

Совпадение

цию сложения на логических элементах, реализующих логическое умножение, и наоборот, — функцию логического умножения на элементах, реализующих логическое сложение. На рис. 7.3 пред-



а *б*

Рис. 7.3

ставлены примеры реализации двух функций с использованием закона инверсии (7.5) и (7.6).

Контрольные вопросы

Указать какие законы и тождества необходимо использовать для упрощения следующих выражений:

1) $X_1 = A \cdot B \cdot C \vee \overline{B}$;

2) $X_2 = \left[\left(\overline{\overline{A \cdot B}}\right) \cdot (A \cdot B)\right] \vee 1$;

3) $X_3 = A \cdot B \cdot \overline{C} \vee (C \cdot 0)$;

4) $X_4 = \left(\overline{A} \vee \overline{B}\right) \cdot \left(C \cdot \overline{C}\right) \cdot \left(A \cdot \overline{A}\right)$.

Контрольные задания

С помощью таблиц, аналогичных табл. 7.6 и 7.7, проверить равенство следующих функций:

1) $(A \cdot B) \vee (\overline{A} \cdot B) \vee (A \cdot \overline{B}) = \overline{\overline{A} \cdot \overline{B}}$;

2) $(\overline{A} \cdot B) \cdot (A \cdot \overline{B}) = (\overline{A} \cdot \overline{B}) \cdot (A \cdot B)$.

7.4. Дизъюнктивные и конъюнктивные нормальные формы представления функций алгебры логики

Как уже отмечалось, функция может быть задана в аналитическом, формульном или табличном виде. При этом для того чтобы задать переключательную функцию, не обязательно задавать все ее значения при всех сочетаниях переменных, а достаточно знать состояния, при которых она, например, равна единице (или нулю), так как для всех остальных состояний переменных значение функции равно нулю (единице).

В формульном виде функция в своей основе имеет набор логических произведений (или сумм) переменных, связанных между собой, как правило, знаками логических сумм (или произведений), например:

$$X = ABC \vee A\overline{B}C \vee \overline{A}\overline{B}\overline{C};$$

Минтерм

Макстерм

$$X = (\overline{A} \vee \overline{B} \vee \overline{C}) \cdot (\overline{A} \vee B \vee \overline{C}) \cdot (A \vee B \vee C).$$

Произведение переменных, в которое каждая из переменных входит только один раз в прямом или инверсном виде, называется *минтермом* (*m*).

Сумма переменных, в которую каждая из переменных входит только один раз в прямом или инверсном виде, называется *макстермом* (*M*).

Так, например, у двух переменных может быть четыре возможных макстерма или минтерма, виды которых представлены в табл. 7.8. Количество переменных, входящих в макстерм или минтерм, называется *рангом*. В рассматриваемом примере используются две переменные, следовательно, ранг равен 2.

Рассмотрим правила перехода от табличной формы записи функции к формульной. Например, функция задана в виде табл. 7.9.

Таблица 7.8

Переменные		Макстермы	Минтермы
A	B	M	m
0	0	$M_0 = A \vee B$	$m_0 = \overline{A} \cdot \overline{B}$
0	1	$M_1 = A \vee \overline{B}$	$m_1 = \overline{A} \cdot B$
1	0	$M_2 = \overline{A} \vee B$	$m_2 = A \cdot \overline{B}$
1	1	$M_3 = \overline{A} \vee \overline{B}$	$m_3 = A \cdot B$

Таблица 7.9

X_0 (M)	A	B	C	X	X_1 (m)
—	0	0	1	1	$\overline{A}\overline{B}C$
—	0	1	0	1	$\overline{A}B\overline{C}$
$A \vee \overline{B} \vee \overline{C}$	0	1	1	0	—
—	1	0	0	1	$A\overline{B}\overline{C}$
$\overline{A} \vee B \vee \overline{C}$	1	0	1	0	—
$\overline{A} \vee \overline{B} \vee C$	1	1	0	0	—
—	1	1	1	1	ABC

Для перехода от табличной формы к формульной руководствуются следующими положениями.

1. Если функция алгебры логики задана таблицей состояний, то из таблицы всегда можно взять логическую сумму — дизъюнкцию всех переменных, для которых функция равна единице. Эта запись будет точно представлять функцию и называться *совершенной дизъюнктивной нормальной формой* (СДНФ) представления функции:

$$X_1 = \overline{A}\overline{B}C \vee \overline{A}B\overline{C} \vee A\overline{B}\overline{C} \vee ABC.$$

Минтерм

СДНФ — это логическая сумма минтермов, при которых значение функции равно единице.

2. Если функция алгебры логики задана таблицей состояний, то из нее всегда можно взять логическое произведение всех макстермов, для которых функция равна нулю. Причем необходимо иметь в виду, что переменные, входящие в макстермы, имеют инверсный вид по отношению к табличным значениям. Эта запись будет точно представлять функцию и называться *совершенной конъюнктивной нормальной формой* (СКНФ) представления функции:

$$X_0 = \left(A \vee \overline{B} \vee \overline{C}\right) \cdot \left(\overline{A} \vee B \vee \overline{C}\right) \cdot \left(\overline{A} \vee \overline{B} \vee C\right).$$

Макстерм

СКНФ — это логическое произведение макстермов, при которых значение функции равно нулю.

Рассмотрим несколько примеров. Вернемся к переключательным функциям двух аргументов, представленным в табл. 7.3. Рассмотрим примеры записи функций в СКНФ и СДНФ. Кроме того, на их примере покажем возможности использования закона инверсии.

Функцию «Операция Шеффера» (И-НЕ) целесообразно записать в более короткой СКНФ:

$$f(AB)_0 = \overline{A} \vee \overline{B} = \overline{A \cdot B}.$$

При этом необходимо помнить, что в СКНФ переменные имеют инверсный вид.

Функцию «Стрелка Пирса» (ИЛИ-НЕ) целесообразно записать в СДНФ:

$$f(AB)_1 = \overline{A} \cdot \overline{B} = \left(\overline{A \vee B}\right).$$

Контрольные вопросы

1. Что представляют собой макстерм и минтерм? Дать определения и показать на примере.
2. Сколько минтермов и макстермов может быть у трех переменных? Перечислить их.
3. Что такое СДНФ и СКНФ?
4. Назовите СКНФ и СДНФ для функций двух аргументов «Эквивалентность» и «Сложение по модулю 2».
5. В какой форме целесообразно записать функции «Импликация» и «Запрет»? Назвать СКНФ и СДНФ для этих функций.

Контрольные задания

Записать в СКНФ и СДНФ функции «Эквивалентность», «Сложение по модулю 2», «Импликация от *A* к *B*», «Запрет по *A*» и преобразовать полученные выражения с использованием закона инверсии.

7.5. Минимизация логических функций с использованием законов и тождеств

Используя ранее рассмотренные законы и тождества алгебры логики, а также приведенные далее операции, можно производить упрощение исходных функций.

1. Операция *поглощения*:

$$A \vee AB = A\overset{1}{\cancel{(1 \vee B)}} = A,$$

где A поглощает AB.

2. Операция *склеивания*:

$$AB \vee A\overline{B} = A\overset{1}{\cancel{(B \vee \overline{B})}} = A; \qquad (7.7)$$

$$\left(\overline{A} \vee B\right)(A \vee B) = \overset{0}{\cancel{\overline{A}A}} \vee B = B. \qquad (7.8)$$

Данная операция применяется в случае, когда имеется пара произведений или сумм, в каждую из которых одна переменная входит в прямом и инверсном виде, а вторая — имеет одинаковый вид. Последняя операция склеивания является сверткой распределительного закона для двух переменных.

Для трех переменных свертка выглядит так:

$$\left(\overline{A} \vee B\right)(A \vee B)(C \vee B) = \overset{0}{\cancel{\overline{A}AC}} \vee B = B.$$

Часто, используя различные способы, можно получить более простые формы представления функций, которые называются *упрощенными* или *минимизированными*. В результате минимизации удается значительно упростить схемную реализацию переключательных функций.

Если переключательная функция, равная 0, ограничивается одним или двумя сочетаниями переменных, то запись ее и дальнейшее упрощение производятся в СКНФ. Дальнейшее увеличение количества макстермов в функции делает минимизацию довольно трудоемким процессом. Рассмотрим пример:

$$X_0 = \left(\overline{A} \vee \overline{B}\right) \cdot \left(\overline{A} \vee B\right) = \overset{\overline{A}}{\cancel{\overline{A}\,\overline{A}}} \vee \overline{A}\overline{B} \vee \overline{A}B \vee \overset{0}{\cancel{\overline{B} \cdot B}} =$$

$$= \overline{A} \vee \overline{A}\overset{1}{\cancel{(\overline{B} \vee B)}} = \overline{A} \vee \overline{A} = \overline{A}.$$

Как правило, переключательная функция, подвергаемая упрощению, представляется в СДНФ. Сущность упрощения сводится к отысканию минтермов, имеющих общие переменные, которые группируются. В результате общие переменные выносятся за скобки, а оставшиеся в скобках переменные упрощаются с использованием законов и тождеств алгебры логики.

Рассмотрим несколько примеров минимизации.

Пример 1. $X = AB\overline{C} \vee \overline{A}B\overline{C} \vee A\overline{B}\overline{C} = B\overline{C}\overset{1}{\cancel{(A \vee \overline{A})}} \vee A\overline{B}\overline{C} =$

$$= B\overline{C} \vee A\overline{B}\overline{C} = \overline{C}\left(B \vee A\overline{B}\right) = \overline{C}\left(B \vee A\right)\overset{1}{\cancel{(B \vee \overline{B})}} = B\overline{C} \vee A\overline{C}.$$

Здесь для преобразования $\left(B \vee A\bar{B}\right)$ используется распределительный закон.

Пример 2. $X = \bar{A}BC\bar{D} \vee ABC\bar{D} \vee AB\bar{C}\bar{D} \vee \bar{A}B\bar{C}\bar{D} =$

$= \bar{A}B\bar{D}\underset{1}{(C \vee \bar{C})} \vee AB\bar{D}\underset{1}{(C \vee \bar{C})} = B\bar{D}\underset{1}{(\bar{A} \vee A)} = B\bar{D}.$

Пример 3. $X = \bar{A}BC \vee A\bar{B}C \vee AB\bar{C} \vee ABC =$

$= BC\underset{1}{(\bar{A} \vee A)} \vee A\bar{B}C \vee AB\bar{C} = BC \vee A\bar{B}C \vee AB\bar{C} =$

$= C\left(B \vee A\bar{B}\right) \vee AB\bar{C} = C(A \vee B)\underset{1}{(B \vee \bar{B})} \vee AB\bar{C} =$

$= AC \vee BC \vee AB\bar{C} = AC \vee B\left(C \vee A\bar{C}\right) =$

$= AC \vee B(C \vee A)\underset{1}{(C \vee \bar{C})} = AC \vee BC \vee AB.$

В третьем примере процесс упрощения можно сделать короче, если воспользоваться тождеством (7.1), добавив в функцию X дважды ABC:

$$X = \bar{A}BC \vee A\bar{B}C \vee AB\bar{C} \vee ABC \vee \boldsymbol{ABC} \vee \boldsymbol{ABC} =$$

$$= BC\underset{1}{(\bar{A} \vee A)} \vee AC\underset{1}{(B \vee \bar{B})} \vee AB\underset{1}{(C \vee \bar{C})} = BC \vee AC \vee AB.$$

В результате упрощения логического выражения в минтермах или в макстермах возможно отсутствие ряда переменных. Такой вид записи называется *дизъюнктивной* или *конъюнктивной нормальной формой* (ДНФ или КНФ). Если необходимо вернуться к исходной форме записи — СДНФ, следует дописать отсутствующие переменные в виде суммы прямого и инверсного значений. Например:

$$X_1 = AB \vee AC \vee BC = AB\left(C \vee \bar{C}\right) \vee A\left(B \vee \bar{B}\right)C \vee \left(A \vee \bar{A}\right)BC =$$

$$= ABC \vee AB\bar{C} \vee ABC \vee A\bar{B}C \vee ABC \vee \bar{A}BC = ABC \vee AB\bar{C} \vee A\bar{B}C \vee \bar{A}BC.$$

Аналогично, если необходимо вернуться к СКНФ, следует дважды записать макстермы, дополнив каждый прямым и инверсным значениями отсутствующей переменной. Например:

$$X_0 = \left(A \vee \bar{B}\right)\left(\bar{B} \vee \bar{C}\right) = \left(A \vee \bar{B} \vee \boldsymbol{C}\right)\left(A \vee \bar{B} \vee \bar{C}\right)\left(A \vee \bar{B} \vee \bar{C}\right)\left(\bar{A} \vee \bar{B} \vee \bar{C}\right) =$$

$$= \left(A \vee \bar{B} \vee C\right)\left(A \vee \bar{B} \vee \bar{C}\right)\left(\bar{A} \vee \bar{B} \vee \bar{C}\right).$$

Контрольные вопросы

1. В чем состоит различие СДНФ и ДНФ, СКНФ и КНФ?
2. К каким функциям можно применять операции поглощения и склеивания?

3. При каких условиях целесообразно применять свертки для трех переменных?

Контрольные задания

1. Упростить следующие выражения:

$$X = \left(A \vee \overline{B}\right) \cdot (A \vee B);$$

$$X = \left(\overline{A} \vee B\right) \cdot (A \vee B);$$

$$X = \left(\overline{A} \vee \overline{B} \vee \overline{C}\right) \cdot \left(A \vee \overline{B} \vee \overline{C}\right);$$

$$X = \left(\overline{A} \vee \overline{B} \vee C\right) \cdot \left(\overline{A} \vee B \vee C\right);$$

$$X = ABC \vee AB\overline{C} \vee A\overline{B}C \vee \overline{A}\overline{B}C;$$

$$X = ABC \vee \overline{A}BC \vee \overline{A}\overline{B}C \vee \overline{A}\overline{B}\overline{C};$$

$$X = AB\overline{C} \vee \overline{A}B\overline{C} \vee A\overline{B}C \vee A\overline{B}\overline{C};$$

$$X = ABC \vee \overline{A}BC \vee \overline{A}\overline{B}C \vee \overline{A}\overline{B}\overline{C};$$

$$X = AB\overline{C} \vee \overline{A}B\overline{C} \vee A\overline{B}\overline{C} \vee \overline{A}\overline{B}\overline{C}.$$

2. Выполнить переход от ДНФ к СДНФ и от КНФ к СКНФ для следующих выражений:

$$X = A\overline{C} \vee \overline{A}B \vee BC;$$

$$X = A\overline{B}C \vee \overline{A}B \vee A\overline{C};$$

$$X = \overline{A}C \vee AB\overline{C} \vee AB;$$

$$X = (A \vee C)\left(\overline{B} \vee C\right);$$

$$X = \left(\overline{B} \vee \overline{C}\right)(A \vee B)\left(\overline{A} \vee C\right);$$

$$X = (A \vee B)\left(\overline{A} \vee \overline{B}\right)\left(B \vee \overline{C}\right).$$

7.6. Минимизация логических функций с использованием карт Карно (диаграмм Вейча)

Рассмотренный способ упрощения логических функций с использованием законов и тождеств алгебры логики бывает очень трудоемким, так как при выполнении операции минимизации необходимо сравнить всевозможные сочетания переменных исходного выражения. Один из наиболее удобных методов минимизации основан на применении карт Карно, которые могут отражать функцию, представленную как в СДНФ, так и СКНФ. Наибольшее применение нашли карты для минимизации СДНФ. Такие карты представляют собой специальную таблицу, разбитую на клетки,

каждая из которых соответствует одному из возможных сочетаний переменных (минтермов). Минтермы соседних клеток отличаются друг от друга лишь одной переменной и могут быть описаны уравнениями (7.7) и (7.8) операции склеивания, поэтому расположение клеток такой таблицы позволяет легко определить склеивающиеся между собой минтермы. На рис. 7.4, *а*...*в* представлены карты Карно для функций, содержащих соответственно две, три и четыре переменных.

Процесс упрощения (минимизации) функции состоит из трех этапов.

1. Представление переключательной функции на карте Карно. Для этого на карте следует записать единицы в клетках, соответствующих заданным сочетаниям переменных, при которых функция равна единице.

2. Объединение (склеивание) минтермов. В общем виде минтермы склеиваются в следующих случаях:

единицы соответствующих минтермов, количество которых кратно двум (два, четыре, восемь), расположены рядом в одном столбце (в одной строке) или образуют квадрат;

единицы соответствующих минтермов расположены в противоположных концах столбца (строки) или по противоположным углам.

Необходимо иметь в виду, что склейки должны охватывать максимальное количество единиц. За счет этого достигается оптимальный вариант минимизации, не требующий последующих дополнительных действий по упрощению переключательной функции.

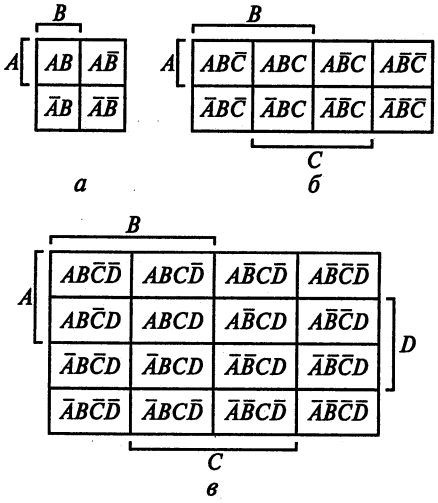
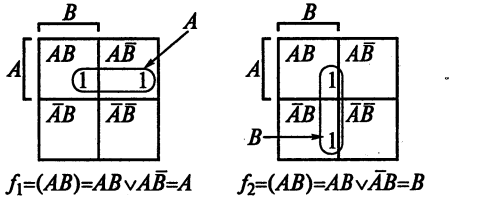
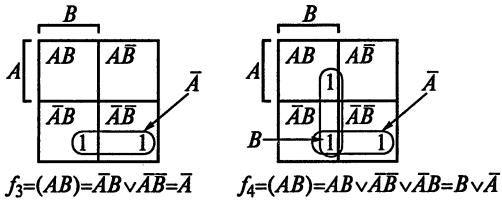


Рис. 7.4

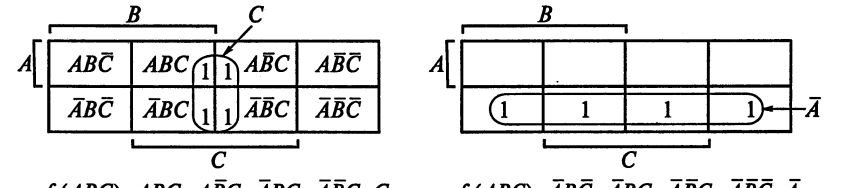


$f_1=(AB)=AB\vee A\bar{B}=A$ $f_2=(AB)=AB\vee\bar{A}B=B$

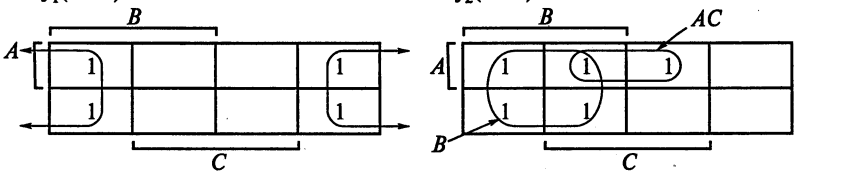


$f_3=(AB)=\bar{A}B\vee\bar{A}\bar{B}=\bar{A}$ $f_4=(AB)=AB\vee\bar{A}B\vee\bar{A}\bar{B}=B\vee\bar{A}$

Рис. 7.5

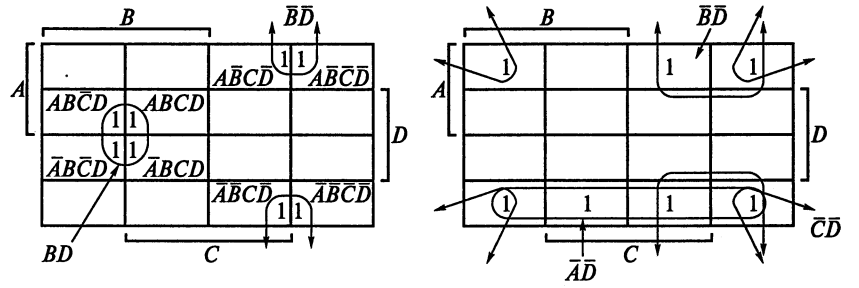


$f_1(ABC)=ABC\vee A\bar{B}C\vee\bar{A}BC\vee\bar{A}\bar{B}C=C$ $f_2(ABC)=\bar{A}B\bar{C}\vee\bar{A}BC\vee\bar{A}\bar{B}C\vee\bar{A}\bar{B}\bar{C}=\bar{A}$



$f_3(ABC)=AB\bar{C}\vee\bar{A}B\bar{C}\vee A\bar{B}\bar{C}\vee\bar{A}\bar{B}\bar{C}=C$ $f_4(ABC)=AB\bar{C}\vee ABC\vee\bar{A}B\bar{C}\vee\bar{A}BC\vee A\bar{B}C=AC\vee B$

Рис. 7.6



$f_1(ABCD)=BD\vee\bar{B}\bar{D}$ $f_2(ABCD)=\bar{A}\bar{D}\vee\bar{B}\bar{D}\vee\bar{C}\bar{D}$

Рис. 7.7

3. Получение результата минимизации. Для минтермов, охваченных оптимальной склейкой, выявляются общие переменные, которые и являются результатом упрощения. На рис. 7.5 и 7.6 показаны примеры упрощения соответственно функций двух и трех переменных.

На рис. 7.7 показан пример упрощения переключательных функций, содержащих четыре переменные:

$$f_1(ABCD) = AB\bar{C}D \vee ABCD \vee \bar{A}B\bar{C}D \vee A\bar{B}C\bar{D} \vee A\bar{B}\bar{C}\bar{D} \vee \bar{A}\bar{B}C\bar{D} \vee \vee \bar{A}\bar{B}\bar{C}\bar{D} \vee \bar{A}BCD;$$

$$f_2(ABCD) = \bar{A}\bar{B}\bar{C}\bar{D} \vee AB\bar{C}\bar{D} \vee A\bar{B}\bar{C}\bar{D} \vee \bar{A}\bar{B}C\bar{D} \vee \bar{A}BC\bar{D} \vee \vee \bar{A}\bar{B}C\bar{D} \vee A\bar{B}C\bar{D}.$$

7.7. Метод испытаний

Метод испытаний, позволяющий исключить в процессе упрощения выражения лишние склейки, заключается в следующем:

последовательно из упрощенной функции исключается один из минтермов и приравнивается единице;

анализируется значение функции без этого минтерма. Если при этом функция равна единице, то исключенный минтерм больше в функцию не вводится.

Упростим переключательную функцию, представив ее на карте Карно (рис. 7.8): $f(ABC) = \bar{A}B\bar{C} \vee \bar{A}\bar{B}\bar{C} \vee A\bar{B}\bar{C} \vee A\bar{B}C = \bar{A}\bar{C} \vee A\bar{B} \vee \vee \bar{B}\bar{C}$.

Проверим наличие лишних минтермов методом испытаний:

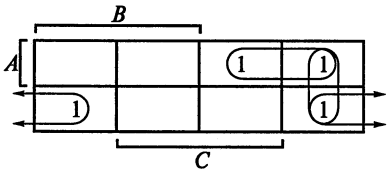
1) $\bar{A}\bar{C} = 1$, следовательно, $\bar{A} = 1$; $\bar{C} = 1$; $A = 0$; $C = 0$.

Тогда $0 \cdot \bar{B} \vee \bar{B} \cdot 1 = \bar{B}$, т.е. функция равна $\bar{B}$, значит минтерм $\bar{A}\bar{C}$ остается;

2) $A\bar{B} = 1$, следовательно, $A = 1$; $\bar{B} = 1$; $\bar{A} = 0$; $B = 0$.

Тогда $0 \cdot \bar{C} \vee 1 \cdot \bar{C} = \bar{C}$, т.е. функция равна $\bar{C}$, значит минтерм $A\bar{B}$ остается;

3) $\bar{B}\bar{C} = 1$, следовательно, $\bar{B} = 1$; $\bar{C} = 1$; $B = 0$; $C = 0$.



$f(ABC)=\bar{A}\bar{C}\vee A\bar{B}\vee\bar{B}\bar{C}$

Рис. 7.8

Тогда $\overline{A} \cdot 1 \vee A \cdot 1 = 1$, т.е. функция равна 1, значит минтерм $\overline{B}\overline{C}$ исключается.

После проведения испытаний окончательно получим функцию вида

$$f(ABC) = \overline{A}\overline{C} \vee A\overline{B}.$$

Контрольные задания

1. Упростить, используя законы и тождества алгебры логики, следующие функции:

1) $X = \overline{A}B\overline{C} \vee \overline{A}BC \vee A\overline{B}\overline{C} \vee AB\overline{C} \vee \overline{A}\overline{B}\overline{C}$;

2) $X = \overline{A}BC \vee AB\overline{C} \vee ABC$;

3) $X = A\overline{B}\overline{C} \vee AB\overline{C} \vee ABC$;

4) $X = \overline{A}BC \vee \overline{A}\overline{B}\overline{C} \vee AB\overline{C} \vee ABC$;

5) $X = AB\overline{C} \vee \overline{A}B\overline{C} \vee A\overline{B}\overline{C} \vee \overline{A}\overline{B}\overline{C} \vee \overline{A}BC \vee \overline{A}\overline{B}C$.

2. Упростить, используя карты Карно, заданные функции для двух переменных:

1) $X = AB \vee A\overline{B} \vee \overline{A}\overline{B}$;

2) $X = \overline{A}\overline{B} \vee \overline{A}B \vee AB$.

3. Упростить, используя карты Карно, функции для трех переменных, представленные в п. 1.

4. Упростить, используя карты Карно, заданные функции для четырех переменных:

1) $X = AB\overline{C}\overline{D} \vee A\overline{B}\overline{C}\overline{D} \vee AB\overline{C}D \vee \overline{A}B\overline{C}D \vee \overline{A}B\overline{C}\overline{D} \vee \overline{A}\overline{B}\overline{C}\overline{D}$;

2) $X = ABCD \vee \overline{A}BCD \vee A\overline{B}CD \vee \overline{A}\overline{B}CD \vee \overline{A}B\overline{C}D \vee \overline{A}\overline{B}\overline{C}D$;

3) $X = ABC\overline{D} \vee A\overline{B}C\overline{D} \vee \overline{A}BC\overline{D} \vee \overline{A}\overline{B}C\overline{D} \vee \overline{A}BC\overline{D} \vee \overline{A}\overline{B}C\overline{D}$;

4) $X = \overline{A}B\overline{C}\overline{D} \vee \overline{A}B\overline{C}D \vee \overline{A}\overline{B}\overline{C}D \vee \overline{A}\overline{B}C\overline{D} \vee A\overline{B}C\overline{D}$.

5. Упростить, используя метод испытаний, заданные функции:

1) $X = \overline{A}BC \vee \overline{A}\overline{B}C \vee AB\overline{C} \vee ABC \vee A\overline{B}\overline{C}$;

2) $X = AB\overline{C} \vee \overline{A}B\overline{C} \vee A\overline{B}\overline{C} \vee A\overline{B}C$.

7.8. Анализ и синтез комбинационных схем

Для реализации простейших функций алгебры логики используются логические элементы. На их основе создаются логические схемы. Элементы в схемах могут быть однотипные или разнотипные. Логические схемы могут иметь n входов и один или m выходов.

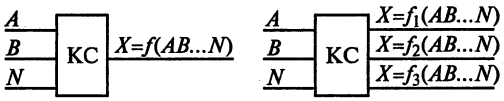


Рис. 7.9

Выходные функции задаются набором (комбинацией) входных сигналов и поэтому называются *комбинационными схемами* (КС). Условное изображение таких КС может иметь вид, представленный на рис. 7.9.

Синтез комбинационных схем включает в себя следующие этапы:

постановка задачи;

составление таблицы истинности, в которой отражена заданная функция при соответствующих сочетаниях переменных;

составление СДНФ (СКНФ) на основе таблицы истинности и упрощение полученного выражения;

построение схемы с учетом оптимального набора элементов.

Прежде чем перейти к созданию комбинационных схем, остановимся на том, как выбрать оптимальный набор элементов, на которых будет создаваться схема, и какие требования необходимо соблюдать при оформлении схем создаваемого комбинационного устройства.

7.9. Правила оформления схем цифровых устройств

Требования к построению схем цифровых устройств изложены в соответствующих стандартах. Мы остановимся лишь на некоторых вопросах, необходимых при синтезе схем.

Схема цифрового устройства представляет собой совокупность условных графических обозначений микросхем, на которых реализовано это устройство, и связей между ними. Линии и условные обозначения элементов должны иметь одинаковую толщину (0,3...0,4 мм). Элементы, используемые в разработке, изображаются на схемах совмещенным или разнесенным способом. Совмещенное обозначение представляет собой функциональный вид всех элементов микросхемы, сконцентрированных в одном месте. Разнесенное обозначение представляет собой отдельные самостоятельные элементы цифровой микросхемы, которые могут находиться в разных местах схемы устройства. Это облегчает работу при составлении схемы и ее чтении. Всем элементам присваивается позиционное обозначение. В позиционном обозначении отражены вид элемента, порядковый номер корпуса в пределах данного вида и номер используемого элемента из корпуса микросхемы.

Позиционные обозначения проставляют рядом с условными графическими обозначениями элементов с правой стороны или над ними. Если используется разнесенное позиционное обозначе-

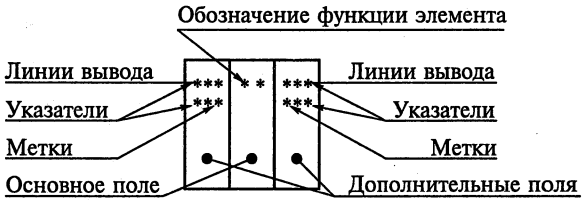


Рис. 7.10

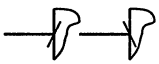
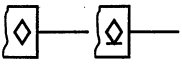
ние, то порядковый номер корпуса состоит из последовательности цифр, разделенных точкой. Первая цифра обозначает номер корпуса, а вторая — номер использованного элемента в корпусе. Например: *DD2.3*, где *DD* — интегральная схема; *2* — номер корпуса микросхемы; *3* — номер элемента в корпусе микросхемы.

Условное графическое обозначение (УГО) элемента имеет форму прямоугольника, к которому подводятся линии выводов. На рис. 7.10 представлен условный вид элемента.

УГО элемента может содержать три поля: одно основное и два дополнительных. Дополнительные поля располагаются справа и слева от основного поля. В основном поле УГО помещают обозначение функции, выполняемой элементом. В дополнительных полях помещают информацию о функциональном назначении выводов.

Выводы элементов делятся на входы, выходы, двунаправленные выводы и выводы, не несущие логической информации. Вхо-

Таблица 7.10

Наименование вывода	Вход	Выход
Прямой статический вход-выход		
Инверсный статический вход-выход		
Прямой динамический вход		—
Вход, не несущий логической информации		—
Выход с тремя состояниями (лог. 1, лог. 0, разомкнуто)	—	
Выход с открытым коллектором	—	

ды элемента изображаются с левой стороны УГО, выходы — с правой.

Размеры УГО определяются по высоте в основном количеством линий выводов и количеством интервалов, а по ширине — наличием дополнительных полей.

Расстояние между линиями выводов должно быть не менее величины C и кратно ей. Расстояние между горизонтальной стороной УГО, границей зоны и линией вывода должно быть не менее величины $C/2$ и кратно ей. При выполнении схем вручную $C = 5$ мм. Размеры УГО по высоте должны быть кратны постоянной величине C. В табл. 7.10 представлены наиболее распространенные обозначения различных видов входов и выходов элементов.

7.10. Функционально полные системы элементов (базис)

Одна из основных задач синтеза заключается в выборе типов элементов, на которых будут реализовываться заданные функции. Поэтому необходимо определить минимальный набор логических элементов (базис), образующих функционально полную систему элементов.

Базис — это функционально полный набор элементов, с помощью которого можно реализовать сколь угодно сложную переключательную функцию. Их может быть несколько. Базис из логических элементов И, ИЛИ, НЕ называется *основным*. Функционально полную систему элементов образует логический элемент И-НЕ или ИЛИ-НЕ, а также один элемент И-ИЛИ-НЕ, УГО которого приведено на рис. 7.11. Сокращение видов логических элементов возможно, если применить закон инверсии.

На рис. 7.12 показана реализация функции «Равнозначность» на различных базисах:

$$X = A \circ B = AB \vee \overline{A}\,\overline{B}.$$

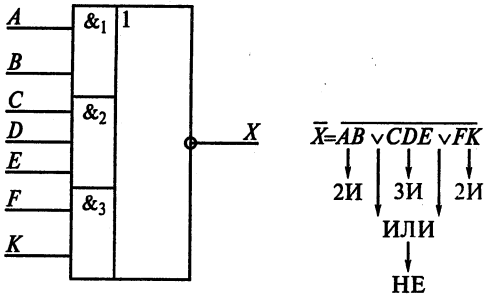


Рис. 7.11

На рис. 7.12, *а* схема построена с использованием основного базиса И, ИЛИ, НЕ. При реализации схемы на базисе одного типа И-НЕ или ИЛИ-НЕ не обойтись без закона инверсии. Для удобства преобразования введем дополнительные обозначения:

$$E = A \cdot B = \overline{\overline{A} \vee \overline{B}}; \quad \overline{E} = \overline{A \cdot B} = \overline{A} \vee \overline{B};$$

$$F = \overline{A} \cdot \overline{B} = \overline{A \vee B}; \quad \overline{F} = \overline{\overline{A} \cdot \overline{B}} = A \vee B.$$

На рис. 7.12, *б* показана схема на базисе И-НЕ. Для построения этой схемы исходная функция должна быть преобразована так, чтобы в ней присутствовали только логические связки И. В этом случае имеем:

$$X = AB \vee \overline{A}\,\overline{B} = E \vee F = \overline{\overline{E} \cdot \overline{F}} = \overline{\overline{(A \cdot B)} \cdot \overline{(\overline{A} \cdot \overline{B})}}.$$

ИЛИ И И

На рис. 7.12, *в* показана схема на базисе ИЛИ-НЕ. Для построения этой схемы исходная функция должна быть преобразована

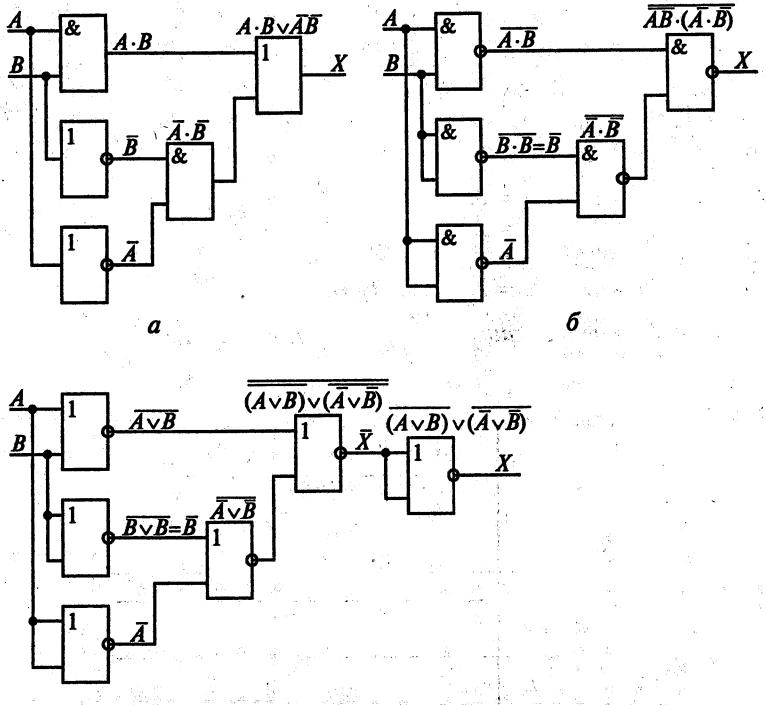


Рис. 7.12

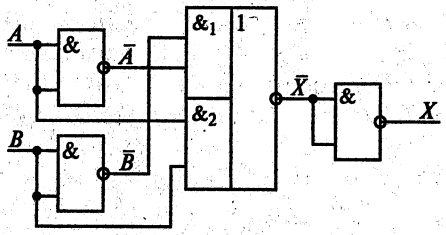


Рис. 7.13

так, чтобы в ней присутствовали только логические связки ИЛИ. В этом случае имеем:

$$X = A \cdot B \vee \bar{A} \cdot \bar{B} = \overline{(\bar{A} \vee \bar{B})} \vee \overline{(A \vee B)}.$$

И И ИЛИ

Использование базиса И-ИЛИ-НЕ целесообразно, когда имеются как прямые, так и инверсные значения переменных. Схема с использованием такого базиса показана на рис. 7.13.

Синтез или построение схемы на элементах вычислительной техники складывается из нескольких этапов.

1. *Постановка задачи.* Построить схему, имеющую три входа *A*, *B*, *C* и один выход *X*, с помощью которой можно реализовать некоторые условия — единичный сигнал на выходе *X* должен появляться только при определенных комбинациях выходных сигналов:

$A = 0, \quad B = 1, \quad C = 0;$
$A = 1, \quad B = 0, \quad C = 1;$
$A = 0, \quad B = 1, \quad C = 1;$
$A = 1, \quad B = 1, \quad C = 1.$

2. *Составление таблицы истинности* (табл. 7.11).

Таблица 7.11

A	*B*	*C*	*X*	
0	0	0	0	
0	0	1	0	
0	1	0	1	1
0	1	1	1	2
1	0	0	0	
1	0	1	1	3
1	1	0	0	
1	1	1	1	4

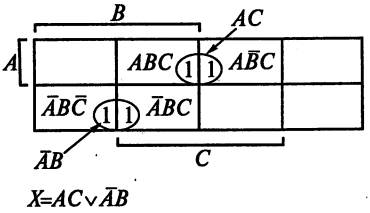


Рис. 7.14

3. *Составление СДНФ и упрощение выражения.* Для этого следует: упростить выражение с использованием карт Карно, как это показано на рис. 7.14, или законов и тождеств:

$$X = \overline{A}B\overline{C} \vee \overline{A}BC \vee A\overline{B}C \vee ABC = \overline{A}B\left(\overset{1}{\cancel{\overline{C} \vee C}}\right) \vee AC\left(\overset{1}{\cancel{\overline{B} \vee B}}\right) = \overline{A}B \vee AC.$$

4. *Выбор оптимального набора логических элементов для реализации функции.* При построении схемы целесообразно придерживаться минимального разброса типов логических элементов и корпусов микросхем. Например, в типовой серии имеется микросхема, в состав которой входит четыре логических элемента И-НЕ. Тогда комбинационная схема, реализующая заданную функцию

$$X = \overline{A}B \vee AC = \overline{\overline{\overline{A}B} \cdot \overline{AC}},$$

будет выглядеть так, как это показано на рис. 7.15.

Для реализации заданной функции X использована одна микросхема *DD1*. Использование других видов логик и соответствующих схем в первом приближении нецелесообразно, так как это приводит к увеличению количества корпусов микросхем. Если использовать один вид логики ИЛИ-НЕ и иметь в виду, что в корпусе микросхемы четыре логических элемента, схема, реализующая функцию

$$X = \overline{A}B \vee AC = \overline{\left(A \vee \overline{B}\right)} \vee \overline{\left(\overline{A} \vee \overline{C}\right)},$$

будет иметь вид, представленный на рис. 7.16.

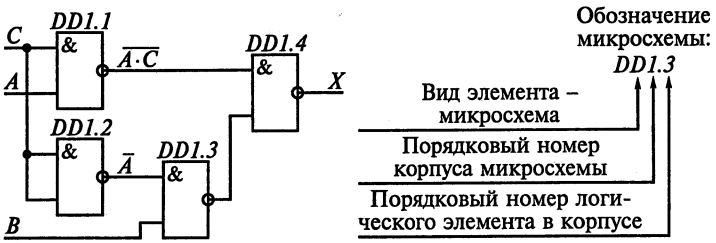


Рис. 7.15

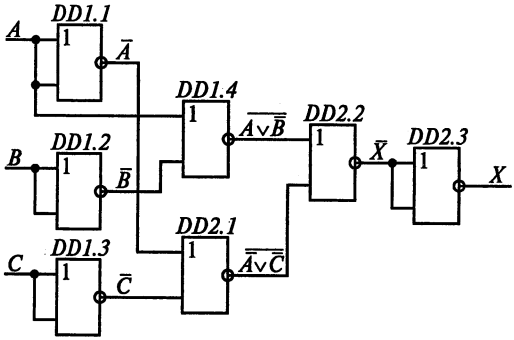


Рис. 7.16

В рассмотренном случае использованы две микросхемы (*DD1* и *DD2*) и увеличено количество логических элементов, что приводит к большой задержке появления сигнала *X* на выходе комбинационной схемы.

Контрольные вопросы

1. Какая схема называется комбинационной?
2. Какова цель анализа и синтеза комбинационной схемы?
3. С помощью какого минимального набора логических элементов можно реализовать любую переключательную функцию?
4. Какие существуют возможные варианты логических схем, образующих базис?
5. Перечислить этапы синтеза комбинационной схемы.

Контрольные задания

1. Используя только элементы И-НЕ и только элементы ИЛИ-НЕ, построить схему, реализующую следующие функции:

«Сложение по модулю 2»;
«Импликация от *A* к *B*»;
«Запрет по *B*»;
«Запрет по *A*».

2. Построить схемы, имеющие три входа *A*, *B*, *C* и один выход *X*, при этом на выходе *X* сигнал, соответствующий лог. 1, должен появиться только при заданных вариантах комбинаций входных сигналов.

Схема 1. $A = 0, B = 1, C = 0;$
$A = 0, B = 1, C = 1;$
$A = 1, B = 0, C = 1;$
$A = 1, B = 1, C = 0;$
$A = 1, B = 1, C = 1.$

Схема 2. $A = 1, B = 0, C = 0;$
$A = 0, B = 0, C = 0;$
$A = 0, B = 1, C = 0;$
$A = 1, B = 1, C = 0;$
$A = 1, B = 1, C = 1.$

Схема 3. $A = 0$, $B = 0$, $C = 1$;
$A = 1$, $B = 0$, $C = 1$;
$A = 1$, $B = 1$, $C = 0$;
$A = 1$, $B = 1$, $C = 1$;
$A = 0$, $B = 1$, $C = 0$.

3. Построить схемы, имеющие четыре входа A, B, C, D и один выход X, при этом на выходе X сигнал, соответствующий лог. 1, должен появиться только при заданных вариантах комбинаций входных сигналов.

Схема 1. $A = 1$, $B = 1$, $C = 0$, $D = 1$;
$A = 1$, $B = 0$, $C = 0$, $D = 1$;
$A = 0$, $B = 1$, $C = 0$, $D = 1$;
$A = 0$, $B = 1$, $C = 0$, $D = 0$;
$A = 0$, $B = 0$, $C = 0$, $D = 1$;
$A = 0$, $B = 0$, $C = 0$, $D = 0$.

Схема 2. $A = 1$, $B = 1$, $C = 0$, $D = 0$;
$A = 0$, $B = 1$, $C = 0$, $D = 0$;
$A = 1$, $B = 0$, $C = 0$, $D = 0$;
$A = 1$, $B = 0$, $C = 0$, $D = 1$;
$A = 0$, $B = 1$, $C = 1$, $D = 0$;
$A = 0$, $B = 0$, $C = 1$, $D = 0$.

Схема 3. $A = 1$, $B = 1$, $C = 0$, $D = 1$;
$A = 1$, $B = 1$, $C = 1$, $D = 0$;
$A = 1$, $B = 0$, $C = 1$, $D = 0$;
$A = 0$, $B = 1$, $C = 1$, $D = 0$;
$A = 1$, $B = 0$, $C = 0$, $D = 1$.

7.11. Особенности работы комбинационных схем. Гонки

При одновременном поступлении сигналов на отдельные участки схемы время реакции на них может быть различным. Это определяется типом элементов на каждом участке, их количеством, быстродействием, что в свою очередь может привести к неправильной реакции схемы логического элемента, на котором эти сигналы сходятся. Такое явление называется *гонками*. Рассмотрим прохождение сигналов в схеме, представленной на рис. 7.17.

При анализе прохождения сигналов будем условно считать, что реакция на выходе логического элемента имеет задержку $t_{зд.\,р}$. Тогда сигналы на выходах схем *DD1* и *DD4* появятся с задержкой распространения $t_{зд.\,р}$ и $3t_{зд.\,р}$ соответственно.

На рис. 7.18 представлены виды диаграммы, поясняющие работу рассматриваемой схемы:

теоретический (см. рис. 7.18, *а*), без учета задержки распространения сигнала по цепям схемы;

практический (см. рис. 7.18, *б*), реальный вид диаграммы, из которой видно, что ожидаемая реакция схемы *DD5* произойдет с

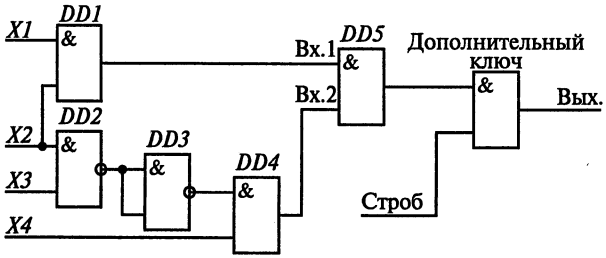


Рис. 7.17

задержкой $4t_{зд.р}$ и будет определяться временем прихода сигнала с элемента *DD4*. До этого момента будет «неправильная» реакция схемы *DD5*, которая называется *информационной помехой* комбинационной схемы.

Для борьбы с информационными помехами, вызванными гонками, наиболее часто используется метод *тактирования* (*стробирования*). В этом случае на логическую схему, используемую в качестве электронного ключа, заводится специальный сигнал — *строб*

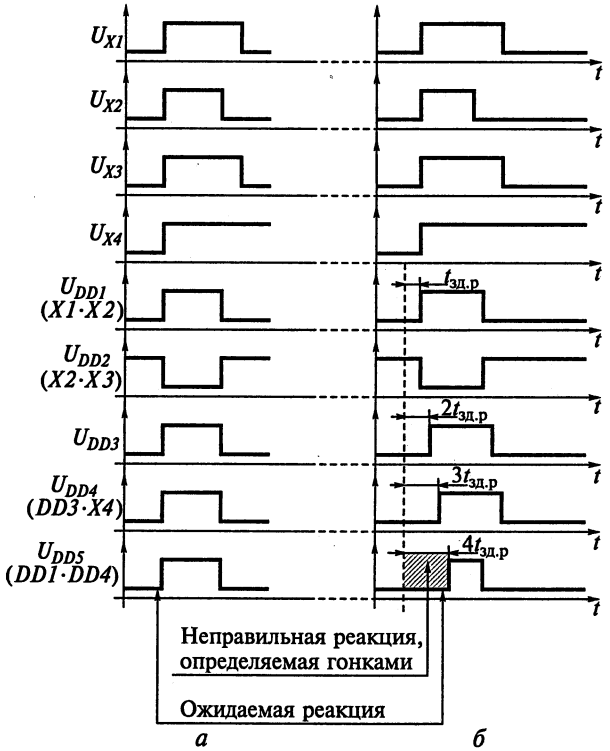


Рис. 7.18

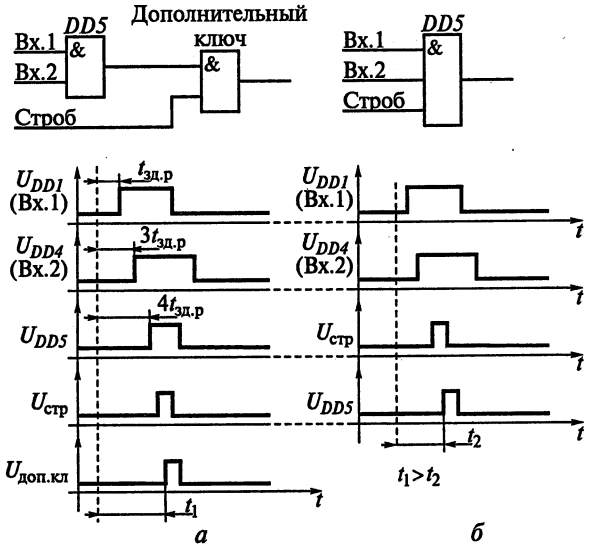


Рис. 7.19

(сигнал разрешения на прохождение сигнала через схему). При этом время поступления и длительность строба выбираются с учетом максимальной задержки формирования основного сигнала и меньше его длительности. Процесс кратковременного и своевременного замыкания электронного ключа называется *стробированием*.

На рис. 7.19 представлены два варианта стробирования выходного каскада ранее рассмотренной схемы:

с использованием дополнительной логической схемы, выполняющей функцию ключа (см. рис. 7.19, *а*);

с использованием дополнительного входа в логической схеме (см. рис. 7.19, *б*).

7.12. Использование логического элемента в качестве ключа

Рассмотрим работу двухвходового электронного ключа, представленного на рис. 7.20, *а*. Обозначим один вход как управляющий, другой как информационный.

Работа электронного ключа полностью соответствует функциям логического элемента И:

достаточно на одном из входов логического элемента иметь лог. 0, чтобы на выходе появился лог. 0, вне зависимости от того, какой сигнал находится на втором входе. Поэтому лог. 0, который определяет значение сигнала на выходе логического элемента, называется ведущим сигналом или активным, определяющим;

наличие лог. 1 на одном из входов не определяет значение выходного сигнала, его значение будет зависеть от сигнала на втором входе.

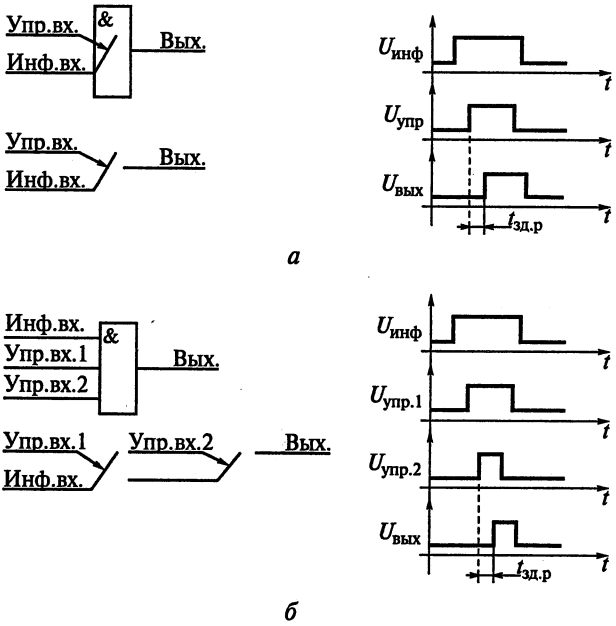


Рис. 7.20

Если на управляющий вход ключа подать лог. 1, то такое состояние будет эквивалентно его «замыканию». Тогда информация, поданная на информационный вход, пройдет на выход схемы. Такое состояние элемента соответствует второму характерному состоянию функции И.

При наличии на управляющем входе лог. 0 входная информация как бы подавляется, и на выходе вне зависимости от ее значения будет лог. 0, что эквивалентно «размыканию» ключа. Такое состояние элемента соответствует первому характерному состоянию функции И.

Если в качестве электронного ключа используется элемент И-НЕ, то на выход будет проходить инверсное значение входного сигнала.

На рис. 7.20, *б* показано использование в качестве ключа многовходового элемента И. Здесь только один из входов выполняет функцию информационного, остальные входы — управляющие. Подразумевается, что количество ключей на пути прохождения информационного сигнала соответствует количеству управляющих сигналов их замыкающих. Тогда информация пройдет только в момент одновременного поступления лог. 1 на управляющие входы, а длительность выходного сигнала будет полностью определяться длительностью самого короткого управляющего импульса — строба.

Контрольные вопросы

1. Какова природа возникновения информационной помехи в комбинационной схеме?
2. Какие существуют способы борьбы с помехой, вызванной эффектом гонки?
3. В чем заключается работа логических элементов, выполняющих функцию электронного ключа? В каких местах комбинационной схемы он, как правило, устанавливается?
4. Каковы возможности использования элементов ИЛИ, ИЛИ-НЕ в качестве электронного ключа?
5. Какие требования предъявляются к временны́м параметрам сигнала «Строб»?

Глава 8
ЦИФРОВЫЕ ИНТЕГРАЛЬНЫЕ МИКРОСХЕМЫ

8.1. Общие сведения о цифровых микросхемах

Цифровые интегральные микросхемы — это электронные устройства, обеспечивающие строительство практически всех узлов и блоков ЭВМ и других цифровых устройств, в которых обрабатываемая информация представлена в виде двоичных чисел. Подавляющее большинство интегральных схем (ИС) представляют собой потенциальные логические элементы (ЛЭ). Характерно наличие связи по постоянному току между входами и выходами ИС. Схемотехническая реализация потенциальных цифровых ИС осуществляется на основе ряда типовых базовых ЛЭ.

Параллельно с развитием цифровой техники шло создание различных серий ИС. Основными тенденциями разработок было стремление к увеличению быстродействия и уменьшению мощности потребления ИС. Эти серии развивались на основе типовых базовых ЛЭ в следующей последовательности: резисторно-транзисторная логика (РТЛ), резистивно-емкостная транзисторная логика (РЕТЛ), диодно-транзисторная логика (ДТЛ), транзисторно-транзисторная логика (ТТЛ), эмиттерно-связанная логика (ЭСЛ), транзисторно-транзисторная логика с диодами Шотки (ТТЛШ), интегральная инжекционная логика ($И^2Л$). Наряду с биполярными схемами очень широкое распространение получили ИС на униполярных МОП-транзисторах, особенно на транзисторах с индуцированными каналами *n*- и *p*-типа (КМОП-транзисторах).

Серия ИС представляет собой комплект микросхем, имеющих единое схемотехническое и конструктивно-технологическое исполнение. В настоящее время наибольшее распространение получили логические элементы ТТЛ, ТТЛШ (в дальнейшем — ЛЭ ТТЛ) и

элементы на КМОП-транзисторах, выпускаемые в виде нескольких серий ИС. Микросхемы этих серий отличаются лучшими электрическими параметрами и позволяют получить более высокий уровень интеграции. Поэтому указанные серии содержат десятки различных по функциональному назначению микросхем, среди которых имеются не только простейшие ЛЭ малой степени интеграции (МИС) типа И-НЕ, ИЛИ-НЕ, содержащие до 10 элементов, но и гораздо более сложные микросхемы: регистры, сумматоры, счетчики и другие, т.е. схемы со средней степенью интеграции (СИС), содержащие до 100 элементов в одном корпусе.

Совершенствование технологий и схемотехники позволило значительно улучшить основной показатель ЛЭ ТТЛ — энергию переключения (произведение средней задержки распространения сигнала $t_{зд.\,р.\,ср}$ на среднюю мощность $P_{пот.\,ср}$, потребляемую в статическом режиме), т.е. уменьшить ее примерно в 20 раз по сравнению со значениями, характерными для первых серий ЛЭ ТТЛ. Это позволило размещать на кристалле микросхемы от 100 до 1000 элементов и получать большие интегральные схемы (БИС). Основные достоинства БИС наилучшим образом реализуются в устройствах, имеющих регулярную структуру. Это в первую очередь относится к запоминающим устройствам ЭВМ.

Еще больший прогресс достигнут в развитии ЛЭ на КМОП-транзисторах, которые по своим параметрам заметно приблизились к параметрам идеального логического элемента: $R_{вх} \to \infty$, $R_{вых} < 100$ Ом, $P_{пот} \to 0$, помехоустойчивость $U_{пом} \approx 0{,}5U_{п}$, где $U_{п}$ — напряжение питания, $t_{зд.\,р.\,ср} \approx 20$ нс. Использование схемотехники на КМОП-транзисторах, отличающихся высокой технологичностью, позволило создать СБИС, содержащие сотни тысяч логических элементов на кристалле и работающие на тактовых частотах ~ 1 ГГц.

Дальнейшее повышение степени интеграции ИС, усложнение их функций связано с более узкой специализацией БИС. Это ограничивает круг потребителей и резко повышает стоимость таких разработок. Одним из выходов в этой ситуации является создание в виде СБИС микропроцессоров, т.е. устройств, управляемых программным способом и осуществляющих обработку цифровой информации и управление, что повышает соответственно универсальность СБИС.

Другим путем использования БИС и СБИС является создание на их основе базовых матричных кристаллов (БМК) — заготовок, содержащих необходимые элементы, как правило, с регулярной структурой, образующей матрицу. Обеспечение необходимых функций в этом случае осуществляется по индивидуальному заказу путем нанесения нескольких слоев металлизации.

Несмотря на достигнутые успехи, СБИС, ИС и СИС остаются основной элементной базой при создании цифровых устройств и

используются и как самостоятельные элементы, и в качестве связующего звена между более сложными схемами. В настоящем разделе рассмотрена схемотехника логических элементов ТТЛ и на КМОП-транзисторах, принципы работы этих элементов, их основные электрические характеристики и параметры, а также особенности использования. У владеющих этими сведениями вырабатывается навык, необходимый для правильного выбора элементной базы при разработке аппаратуры.

8.2. Логические схемы на биполярных транзисторах

Микросхемы транзисторно-транзисторной логики (ТТЛ) находят широкое применение в цифровой технике. В них удачно сочетаются высокие быстродействие, нагрузочная способность, помехоустойчивость, сравнительно низкая потребляемая мощность и невысокая стоимость. В зависимости от требований, предъявляемых к аппаратуре, выпускается несколько серий ТТЛ-микросхем, электрически полностью совместимых друг с другом. Параметрами таких микросхем являются:

время задержки распространения сигнала при включении $t^{0,1}_{\text{зд.р}}$ — интервал времени между входным и выходным импульсами при переходе напряжения на выходе микросхемы от напряжения, соответствующего лог. 1 к напряжению, соответствующему лог. 0, измеренный на уровне 0,5;

время задержки распространения сигнала при выключении $t^{0,1}_{\text{зд.р}}$ — интервал времени между входным и выходным импульсами при переходе напряжения на выходе микросхемы от напряжения, соответствующего лог. 0, к напряжению, соответствующему лог. 1, измеренный на уровне 0,5;

среднее время задержки распространения сигнала $t_{\text{зд. р. ср}}$ — интервал времени, равный полусумме времени задержки распространения сигнала при включении и выключении микросхемы;

потребляемая мощность $P_{\text{пот}}$ — значение мощности, потребляемой микросхемой от источников питания в заданном режиме;

потребляемая мощность в состоянии лог. 1 ($P^1_{\text{пот}}$);

потребляемая мощность в состоянии лог. 0 ($P^0_{\text{пот}}$);

средняя потребляемая мощность $P_{\text{пот. ср}}$ — полусумма мощностей, потребляемых микросхемой от источников питания в двух различных устойчивых состояниях;

энергия переключения $t_{\text{зд. р. ср}} P_{\text{пот. ср}}$ — основной параметр, определяющий качество микросхемы логического элемента (ЛЭ): чем она ниже, тем схема лучше.

По последнему показателю выгодно отличаются серии микросхем КР1531, КР1533 более поздних разработок, что обусловлено использованием диодов Шотки, совершенствованием схемотехники ЛЭ и технологии изготовления.

Таблица 8.1

Серия	$t_{зд.\,р.\,ср}$, нс	$P_{пот.\,ср}$, мВт	$t_{зд.\,р.\,ср}P_{пот.\,ср}$, пДж
К130, К131	11	40	440
К133, К155	18	19	342
К134	65	2	130
ТТЛ с транзисторами Шотки:			
К530, К531	5	32	160
К533, К555	20	4	80
КР1531	4	8	32
КР1533	11,5	2,3	≈26

Серии ТТЛ-микросхем различаются по потребляемой мощности и быстродействию, типовые значения которых приведены в табл. 8.1.

На рис. 8.1 представлена схема базового ЛЭ И-НЕ серии К134. Функция И реализуется в большинстве ТТЛ-серий на многоэмиттерном транзисторе (МЭТ) *VT1*, функция НЕ — на двухтактном инверторе с использованием транзисторов *VT2*... *VT4*. Диод *VD1* служит для увеличения помехоустойчивости ЛЭ при входном напряжении $U^1_{вх}$. Выполнение схемы И на МЭТ с общими для всех эмиттеров базой и коллектором позволяет заметно уменьшить паразитную емкость элемента И и тем самым повысить быстродействие ЛЭ, а также уменьшить площадь ЛЭ на кристалле.

На рис. 8.2 показаны пути протекания токов по цепям схемы для двух характерных состояний сигналов на входах ЛЭ И-НЕ:

наличие на всех входах напряжений, соответствующих лог. 1 ($U^1_{вх}$ = 2,4 В);

наличие на одном из входов напряжения, соответствующего лог. 0 ($U^0_{вх}$ = 0,4 В).

Соответственно в табл. 8.2 даны типовые значения потенциалов на электродах транзисторов *VT1*... *VT4* и формулы для расчета токов при номинальном напряжении питания $U_п$ = 5 В. При этом

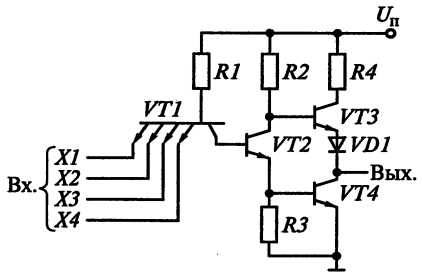


Рис. 8.1

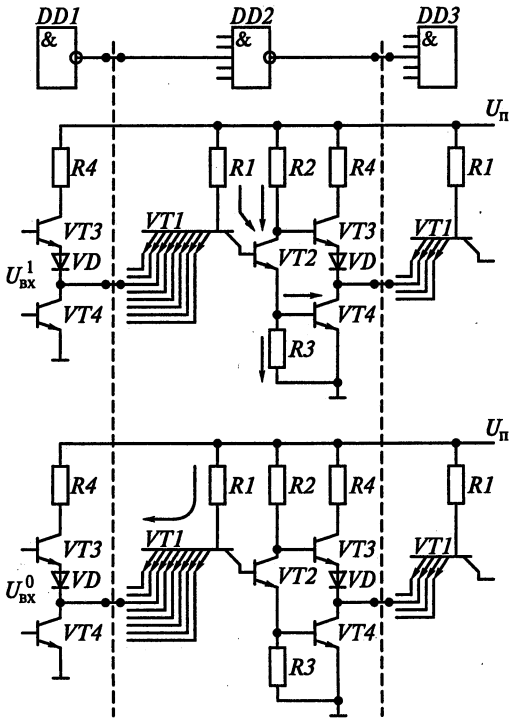


Рис. 8.2

предполагается, что в режиме насыщения транзистора напряжение между коллектором и эмиттером $U_{К-Э} \approx 0{,}1$ В, а напряжение между базой и эмиттером $U_{Б-Э} \approx 0{,}7$ В.

В зависимости от режима работы микросхемы ток эмиттера транзистора *VT1* может являться входным вытекающим (отрицательным) током $I^0_{вх}$ ЛЭ, который для различных серий лежит в диапазоне от −0,2 (серия КР1533) до −2 мА (серия К155), или входным втекающим током $I^1_{вх}$, значение которого значительно меньше (0,02...0,05 мА).

В статическом режиме потребляемая мощность в состоянии лог. 0 $P^0_{пот}$ ЛЭ в несколько раз больше, чем потребляемая мощность в состоянии лог. 1 $P^1_{пот}$. При изменении состояния схемы в момент ее переключения оба выходных транзистора *VT3*, *VT4* оказываются временно открытыми и через них протекает большой сквозной ток, увеличивающий потребляемую микросхемой мощность. Поэтому с увеличением частоты следования импульсов тактовых сигналов $f_т$ эта импульсная составляющая возрастает. Растет также мощность, затрачиваемая на перезарядку емкости нагрузки $C_н$, которая определяется по формуле

$$P_{C_н} = C_н \left(U^1_{вых} - U^0_{вых}\right)^2 0{,}5 f_т.$$

Таблица 8.2

$U_{вх}$	Транзистор			
	VT1	*VT2*	*VT3*	*VT4*
Лог. 0 (0,4 В)	Насыщен; прямое включение $U_Э = 0{,}4$ В $U_Б = 1{,}1$ В $U_К = 0{,}5$ В $I_Э = 3{,}9/R1$ $I_Б = I_Э$ $I_К \approx 0$	Закрыт $U_Э = 0$ В $U_Б = 0{,}5$ В $U_К \approx 5$ В $I_Э \approx 0$ $I_Б \approx 0$ $I_К \approx 0$	Открыт $U_Э \approx 5 - 0{,}7 = 4{,}3$ В $U_Б \approx 5$ В[1] $U_К \approx 5$ В $I_Э = I_н$[3] $I_Б = (U_п - U_Б)/R2$ $I_К \approx I_н$	Закрыт $U_Э = 0$ В $U_Б \approx 0$ В $U_К = U_{вых} =$ $= 3{,}6$ В[2] (лог.1) $I_Э = 0$ $I_Б = 0$ $I_К = 0$
Лог. 1 (2,4 В)	Открыт; инверсное включение $U_Э = 2{,}4$ В $U_Б = 2{,}1$ В $U_К = 1{,}4$ В $I_Э \ll I_Б$[4] $I_Б = (U_п - U_Б)/R1$ $I_К \approx I_Б$	Насыщен; прямое включение $U_Э = 0{,}7$ В $U_Б = 1{,}4$ В $U_К = 0{,}8$ В $I_Э = 4{,}2/R2 + I_Б$ $I_Б = I_{К\ VT1}$ $I_К = 4{,}2/R2$	Закрыт $U_Э \approx 0{,}8$ В $U_Б = 0{,}8$ В $U_К = 5$ В $I_Э = 0$ $I_Б = 0$ $I_К = 0$	Насыщен; прямое включение $U_Э = 0$ В $U_Б = 0{,}7$ В $U_К = 0{,}1$ В (лог. 0) $I_Э = I_н$ $I_Б = I_{Э\ VT2} - 0{,}7/R3$ $I_К = I_н$

[1] Реально потенциал $U_Б$ немного ниже (на величину $I_Б R2$).

[2] В зависимости от нагрузки это напряжение меняется в пределах от 2,4 до 3,6 В.

[3] $I_н$ — ток, вытекающий с выхода микросхемы в нагрузку.

[4] Транзистор *VT1* работает в инверсном включении, когда $U_Э > U_К$, эмиттер и коллектор меняются ролями, а коэффициент передачи тока к эмиттеру очень мал. Поэтому входной, втекающий ток $I^1_{вх}$ микросхемы, равный $I_{Э\ VT1}$, имеет небольшое значение.

При частоте выше 5 МГц потребляемая микросхемой мощность $P_{пот}$ может стать в несколько раз больше $P_{пот.\ ср}$. Для ограничения сквозных токов в цепи коллектора транзистора *VT3* включен резистор *R4*. Остановимся подробнее на специфике построения схем различных серий.

Схема базового ЛЭ серий К130, К131, К133, К155, представленная на рис. 8.3, отличается от ранее рассмотренной схемы. Она дополнена демпфирующими диодами *VD1*... *VD4* для ограничения колебательных процессов на входах ТТЛ, которые могут привести к ложному срабатыванию ЛЭ и появлению ложных импульсов на

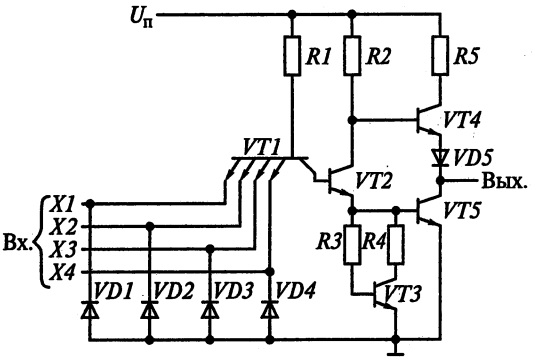


Рис. 8.3

выходе ЛЭ. Диоды открываются только при отрицательном изменении входных напряжений ниже уровня −0,7 В, сглаживая возможные колебательные процессы на входах схемы.

Транзистор *VT3* и резисторы *R3*, *R4* заменяют резистор *R3* в ранее рассмотренной схеме (см. рис. 8.2), что повышает помехоустойчивость данной схемы при напряжении $U^0_{вх}$, соответствующем лог. 0, на входе схемы, вследствие пороговых свойств перехода база—эмиттер транзистора *VT3*.

У базового элемента И-НЕ серий К130, К131 с повышенным быстродействием, представленного на рис. 8.4, верхнее плечо выходного каскада выполнено на транзисторах *VT3*, *VT5* по схеме Дарлингтона, что обеспечивает большой коэффициент усиления в состоянии высокого уровня, повышает нагрузочную способность схемы и ее быстродействие.

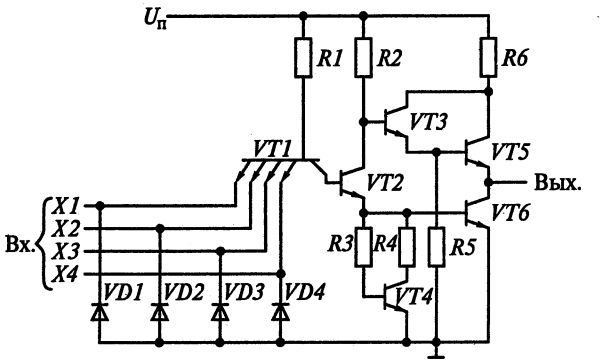


Рис. 8.4

Контрольные вопросы

1. По совокупности каких параметров оценивается качество микросхемы? Как рассчитывается ее показатель качества?

2. Какие типовые значения параметров транзистора в насыщенном состоянии используются для расчета потенциалов в различных точках схемы ЛЭ?

3. Каким образом определяются потенциалы в различных точках схемы?

4. Каково назначение резистора в выходной цепи двухтактного каскада элемента И-НЕ (см. рис. 8.2)?

5. Почему потребляемая мощность у элемента И-НЕ больше при низком уровне выходного напряжения, чем при высоком?

6. Каким образом влияет наличие диода *VD1* в схеме ТТЛ (см. рис. 8.2) на помехоустойчивость схемы?

7. В чем заключается специфика работы транзистора *VT1* при инверсном включении (см. рис. 8.2)? При каком сигнале на входе ЛЭ обеспечивается этот режим?

8. Для чего во входных цепях схемы И-НЕ (см. рис. 8.3) необходимо наличие дополнительных диодов *VD1*... *VD4*?

9. В чем заключаются преимущества схемы на рис. 8.4 с видоизмененным выходным каскадом по сравнению со схемой на рис. 8.2?

8.3. ТТЛ с транзисторами Шотки

Схема элемента И-НЕ серий К530, К531 отличается от схемы И-НЕ серий К130, К131 (см. рис. 8.2) по существу только заменой обычных диодов и транзисторов на диоды и транзисторы Шотки, за счет чего заметно уменьшается значение энергии переключения микросхемы — произведение $t_{зд.\,р.\,ср}P_{пот.\,ср}$. Ток металлополупроводникового выпрямляющего контакта с барьером Шотки обусловлен эмиссией основных носителей из слабо легированного полупроводника n-типа в металл. Благодаря этому время выключения таких диодов очень мало. Кроме того, для отпирания диода Шотки требуется меньшая разность потенциалов, чем для обычных диодов с p—n-переходом (0,2...0,4 В вместо 0,5...0,7 В), причем она может регулироваться технологическим подбором материала металла.

В обычных микросхемах ТТЛ у транзистора в состоянии насыщения коллекторный переход смещен в прямом направлении и инжектирует неосновные носители в базовую область. Это удлиняет процесс выключения транзистора. Диод Шотки подключается параллельно коллекторному переходу транзистора, как это показано на рис. 8.5, и при отпирании транзистора предотвращает смещение его коллекторного перехода в прямом направлении выше

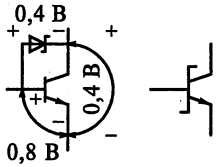


Рис. 8.5

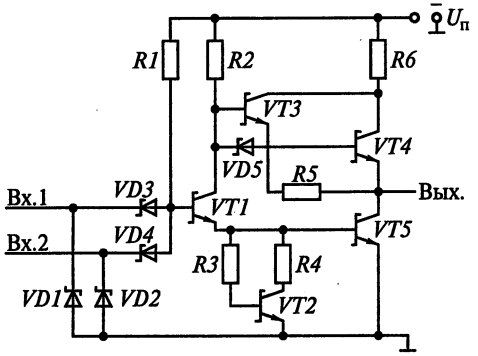


Рис. 8.6

уровня 0,4 В. При этом напряжение между коллектором и эмиттером транзистора Шотки остается достаточно низким, порядка 0,2...0,3 В.

У элемента И-НЕ серии К533, К555 (рис. 8.6) функция И выполняется диодами Шотки, благодаря чему свободные входы могут подключаться к шине питания непосредственно, а не через внешние резисторы, как в случае использования многоэмиттерного транзистора Шотки, которые уменьшают ток $I^1_{вх}$.

Дальнейшее улучшение основного показателя качества элемента И-НЕ ($t_{зд.\,р.\,ср}P_{пот.\,ср}$) было достигнуто благодаря более совершенной технологии: ионная имплантация примесей вместо диффузии и окисная изоляция между транзисторами вместо изоляции *p—n*-переходами привели к значительному снижению внутренних паразитных емкостей микросхем. В результате появились микросхемы серий КР1531 и КР1533, при создании которых была использована более сложная схемотехника, что также позволило уменьшить энергию переключения. На рис. 8.7 представлена схема ЛЭ И-НЕ серии КР1531, содержащая дополнительные диоды *VD3*, *VD4*, *VD7*... *VD12* и транзисторы *VT1*, *VT7*. Реализация функции И осуществляется на диодах *VD1*, *VD2* с *p—n*-переходом и транзисторе *VT1*. Это позволило повысить минимально допустимый уровень помех на входе микросхемы до 0,8 В. Диоды *VD3*, *VD4* ускоряют переключение микросхемы из состояния с низким уровнем выходного сигнала, а диоды *VD7* и *VD8* — из противоположного состояния. Это объясняется тем, что диоды *VD3*, *VD4* сокращают разряд емкости эмиттер—база транзистора *VT1* при отрицательном скачке напряжения на входе микросхемы. Диод *VD7* ускоряет разряд емкости эмиттер—база транзистора *VT6* и, следовательно, сокращает время запирания транзистора *VT5* (и *VT6*). Диод *VD8* обеспечивает с помощью транзистора *VT2* дополнительный ток разряда емкости нагрузки и отпирающий ток в базу транзистора *VT3*, что ускоряет ее разряд до уровня лог. 0.

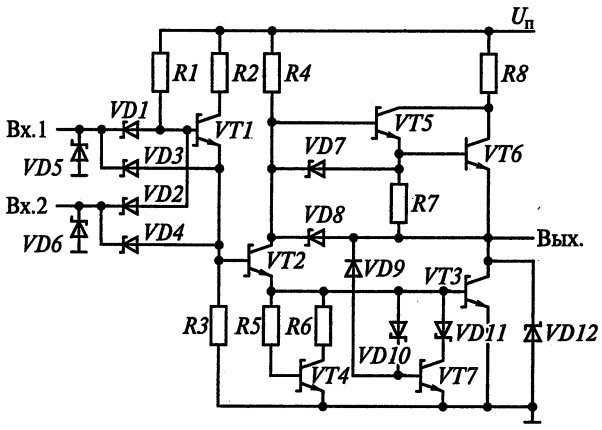


Рис. 8.7

Диоды *VD9... VD11* и транзистор *VT7* обеспечивают на короткий промежуток времени низкий импеданс на базе транзистора *VT3*, ускоряющий запирание при переходе от низкого к высокому уровню выходного напряжения. Кратковременный отпирающий ток в базе транзистора *VT7* создается только в процессе роста напряжения на эмиттере транзистора *VT5* током смещения диода *VD9*. Быстрый разряд его емкости при переключении микросхемы в состояние с низким выходным напряжением происходит через диод *VD7*, транзистор *VT2* и диод *VD10*. Диод *VD11* при высокой частоте переключения предотвращает чрезмерное снижение напряжения на базе транзистора *VT3* (ниже 0,4 В) при его запирании. Диод *VD12* на выходе микросхемы выполняет те же функции, что и демпфирующие диоды *VD5* и *VD6* на ее входе.

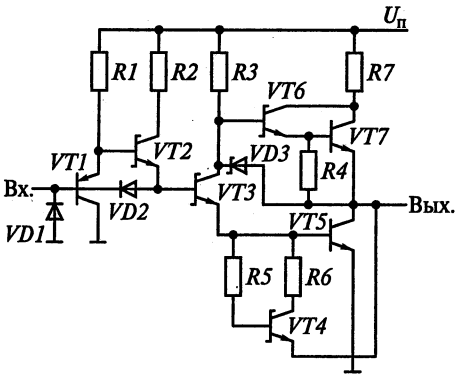


Рис. 8.8

Введение дополнительных элементов в схему И-НЕ (см. рис. 8.7) позволило не только уменьшить задержки $t_{зд.р}^{1,0}$ и $t_{зд.р}^{0,1}$ и значительно ослабить влияние на них емкости нагрузки, но и снизить до минимума динамическую мощность потребления.

Схема элемента И-НЕ маломощной серии КР1533, показанная на рис. 8.8, отличается от схемы, представленной на рис. 8.7, отсутствием диодов *VD7*, *VD9*... *VD11* и транзистора *VT7*, используемых в последней, для повышения быстродействия, а также заменой входных *p—n*-диодов на *p—n—p*-транзисторы *VT1*, *VT2*. Для снижения потребляемой мощности в схеме на рис. 8.8 используются резисторы с бóльшим сопротивлением, чем в схеме на рис. 8.7.

Контрольные вопросы

1. Какие параметры интегральных схем улучшаются с использованием диодов Шотки и почему это происходит?
2. Для чего предназначены диоды *VD3* и *VD4* в схеме И-НЕ (см. рис. 8.6)?
3. Какого показателя качества удалось достичь при разработке ИС серий КР1531, КР1533 в сравнении с аналогичным показателем серии К133? За счет чего он был достигнут?

8.4. Основные электрические параметры и характеристики логических элементов ТТЛ

Основные электрические параметры микросхем ТТЛ всех серий согласованы настолько, что эти микросхемы могут непосредственно работать друг на друга. В табл. 8.3 указаны предельно допустимые режимы эксплуатации микросхем ТТЛ:

максимальное напряжение источника питания $U_{п\,max}$;

максимальное входное напряжение $U_{вх\,max}$;

минимальное входное напряжение $U_{вх\,min}$;

максимальная емкость нагрузки $C_{н\,max}$.

В этих режимах микросхемы сохраняют свою работоспособность, хотя значения их основных электрических параметров могут несколько отличаться от предельно допустимых значений.

Таблица 8.3

Режим	Серия						
	К130 К131	К133 К155	К134	К530 К531	К533 К555	КР1531	КР1533
$U_{п\,max}$, В	6	6	6	5,5	6	6	6
$U_{вх\,max}$, В	5,5	5,5	5,5	5	5,5	5,5	6
$U_{вх\,min}$, В	– 0,4	– 0,4	– 0,4	– 0,4	– 0,4	– 0,4	– 0,5
$C_{н\,max}$, пФ	200	200	200	150	150	150	200

В табл. 8.4 приведены основные электрические параметры микросхем ТТЛ, которые гарантируются в рабочем диапазоне изменения питающего напряжения (для большинства серий $U_п = 4{,}75...5{,}25$ В) и температуры окружающей среды (−0...+70 °С).

Для этих параметров используются следующие обозначения:

входной ток лог. 0 — $I^0_{вх}$;

входной ток лог. 1 — $I^1_{вх}$;

выходное напряжение лог. 1 — $U^1_{вых}$;

выходное напряжение лог. 0 — $U^0_{вых}$;

ток потребления — $I_{пот}$ (значение тока, потребляемого микросхемой от источника питания в заданном режиме);

ток потребления в состоянии лог. 1 — $I^1_{пот}$;

ток потребления в состоянии лог. 0 — $I^0_{пот}$;

частота следования импульсов тактовых сигналов — $f_т$;

коэффициент разветвления по выходу — $K_{раз}$ (число единичных нагрузок, которые можно одновременно подключить к выходу микросхемы).

Единичной нагрузкой является один вход основного логического элемента данной серии микросхем, с помощью которого определяется максимально допустимый ток, равный $K_{раз}I^0_{вх}$, на ее выходе при напряжении, соответствующем лог. 0.

Очевидно, что входное напряжение, которое нужно обеспечить на входе микросхемы, $U^0_{вх}$ ($U^1_{вх}$) ≤ $U^0_{вых\ max}$ (≥ $U^1_{вых\ min}$). При совместной работе микросхем ТТЛ это условие чаще всего выполняется с запасом, так как номинальное, т.е. среднестатистическое значение напряжения $U^0_{вых}$, заметно (на 0,2...0,3 В) ниже указанного в табл. 8.4 максимального значения $U^0_{вых\ max}$, а номинальное значение $U^1_{вых}$ существенно (на 0,7...1 В) выше указанного в табл. 8.4 минимального значения $U^1_{вых\ min}$. При работе микросхемы ТТЛ на микросхемы других серий максимальное количество подключенных микросхем может быть увеличено или уменьшено в зависимости от их параметра $I^0_{вх}$. Поскольку значения задержек распространения $t^{1,0}_{зд.\,р}$ и $t^{0,1}_{зд.\,р}$ устанавливаются при сравнительно небольших емкостях нагрузки и работе микросхемы ТТЛ на частоте, близкой к предельно высокой частоте следования импульсов тактовых сигналов $f_т$, количество подключаемых микросхем может быть ограничено суммарной емкостью монтажа и входов этих микросхем ($C_{вх} \approx 4$ пФ). У микросхем серий КР1531 и КР1533 значения $t^{1,0}_{зд.\,р}$ и $t^{0,1}_{зд.\,р}$ очень слабо зависят от емкости нагрузки.

Помехоустойчивость определяет защищенность микросхем от помех на входах микросхем. Статическая помехоустойчивость определяется помехами, длительность которых больше времени переходных процессов, и характеризуется двумя значениями допустимого уровня помех (для входного сигнала лог. 0 и лог. 1):

Таблица 8.4

Параметр	Серия						
	К130 К131	К133 К155	К134	К530 К531	К533 К555	КР1531	КР1533
$I^0_{вх}$, мА, не более	|–2,3| |–2,0|	|–1,6|	|–0,18|	|–0,2|	|–0,4|	|–0,6|	|–0,1|
$I^1_{вх}$, мА, не более	0,07 0,05	0,04	0,012	0,05	0,02	0,02	0,02
$U^0_{вых\ max}$, В, не более	0,35 0,4	0,4	0,3	0,5	0,5	0,5	0,5
$U^1_{вых\ min}$, В, не менее	2,4	2,4	2,3	2,7	2,7	2,7	2,5
$I^0_{пот}$, мА, не более	11 10	6	0,61	10 9	2,2	2,6	0,75
$I^1_{пот}$, мА, не более	5 4,2	2	0,19	5 4	0,8	0,7	0,21
$K_{раз}$ не более	10	10	10	10	10	20	30
$K_{об}$ не более	8	8	8	8	8	8	8
$t^{1,0}_{зд.\ р}$ (при $C_н$), нс, не более	10 (30 пФ)	15 (15 пФ)	85 (40 пФ)	5 (15 пФ)	20 (15 пФ)	3,8 (15 пФ)	10 (50 пФ)
$t^{0,1}_{зд.\ р}$ (при $C_н$), нс, не более	12 (30 пФ)	22 (15 пФ)	45 (40 пФ)	4,5 (15 пФ)	20 (15 пФ)	3,9 (15 пФ)	11 (50 пФ)
$U_{пом}$, В, не менее	0,4	0,4	0,35	0,5	0,7	0,8	0,8
$f_т$, МГц, не более	60	20	6	100	30	500	200

$$U^0_{пом} = U^0_{вх.\ пор} - U^0_{вых\ max}; \quad (8.1)$$

$$U^1_{пом} = U^1_{вых\ min} - U^1_{вх.\ пор}, \quad (8.2)$$

где $U^0_{вх.\ пор} \approx 0{,}8 \ldots 1{,}2$ В ($U^1_{вх.\ пор} \approx 1{,}6 \ldots 2$ В) — максимальное (минимальное) допустимое пороговое напряжение низкого (высокого) уровня на входе микросхемы.

В табл. 8.4 приведено наименьшее значение $U_{\text{пом}}$. В рабочем диапазоне изменения питающих напряжений и температуры окружающей среды появление положительной помехи $U_{\text{пом}}$ при входном максимальном напряжении $U^0_{\text{вх}} = U^0_{\text{вых max}}$ не может снизить выходное напряжение ниже минимально допустимого напряжения $U^1_{\text{вых min}}$. И наоборот, воздействие отрицательной помехи $U_{\text{пом}}$ на минимальное напряжение на входе $U^1_{\text{вх}} = U^1_{\text{вых min}}$ не может повысить выходное напряжение выше уровня $U^0_{\text{вых max}}$. Реальный запас помехоустойчивости в 2...3 раза больше табличного значения $U_{\text{пом}}$, так как номинальное значение напряжения $U^1_{\text{вых}}$ ($U^0_{\text{вых}}$) заметно выше (ниже) предельно допустимого значения.

Контрольные вопросы

1. Какое количество элементов И-НЕ серии КР1531 может быть подключено к выходу элемента И-НЕ серии КР1533?
2. Какая серия микросхем ТТЛ наиболее выгодна с точки зрения потребляемой мощности для работы на частоте $f_т$ = 20 МГц?
3. Почему недопустима подача на вход элемента ТТЛ напряжения $U^0_{\text{вх. пор}} < U_{\text{вх}} < U^1_{\text{вх. пор}}$?

8.5. Особенности применения микросхем ТТЛ при разработке цифровых устройств

Для обеспечения правильной и надежной работы цифрового устройства на элементах ТТЛ необходимо правильно спроектировать его схему, используя минимально возможное количество различных элементов ТТЛ, в том числе функционального гораздо более сложных, чем элемент И-НЕ. В соответствии с техническим заданием на разрабатываемую схему надо уметь выбрать нужные серии микросхем с учетом их параметров и условий эксплуатации. Кроме того, следует соблюдать основные правила и рекомендации по применению микросхем ТТЛ в цифровых устройствах.

Неиспользуемые логические элементы. В микросхемах ТТЛ, содержащих несколько простейших независимых элементов И-НЕ, И-ИЛИ-НЕ и т. п., с общим питанием могут оказаться элементы, незадействованные в структуре цифрового устройства. Поскольку мощность меньше при выходном напряжении, соответствующем лог. 1, рекомендуется подавать на входы незадействованного элемента соответствующие логические сигналы. Обычно для этого достаточно соединить их с общей (земляной) шиной. При этом выходной сигнал элемента в случае необходимости можно использовать для подачи лог. 1 на входы других элементов.

Неиспользуемые входы элементов ТТЛ. Еще чаще часть входов элементов микросхемы ТТЛ не используется. Как показывает практика, в логических схемах и схемах, управляемых с помощью тактовых импульсов, это может отрицательно сказаться на их поме-

хоустойчивости и быстродействии. На незадействованном входе элемента возрастает уровень динамических (кратковременных) помех, на который влияют паразитные емкости, инерционные процессы в микросхеме, динамические помехи, возникающие при работе соседних элементов. Паразитные емкости этого входа 1,5...3,5 пФ могут оказаться подключенными к базе многоэмиттерного транзистора через отпирающийся переход эмиттер—база при положительном перепаде напряжений на другом используемом входе, увеличивая тем самым значение $t^{1,0}_{\text{зд. р}}$.

Для исключения этих нежелательных эффектов неиспользуемые входы можно по возможности подключить к другим используемым входам этого же элемента. При таком способе подключения следует иметь в виду, что входная емкость элемента увеличивается, а быстродействие снижается. Поскольку непосредственное соединение неиспользуемого входа с шиной питания $U_{\text{п}}$ недопустимо из-за больших входных токов $I^1_{\text{вх}}$, можно использовать подключение неиспользуемых входов к напряжению 2,4...3,6 В. Его можно создать с помощью дополнительного логического элемента, входы которого соединены с общей шиной, или подключением неиспользованных входов к шине питания $U_{\text{п}}$ через резистор с сопротивлением не менее 1 кОм. (Один такой резистор обеспечивает подключение 20 входов ЛЭ.)

Объединение выходов элементов ТТЛ. В элементах ТТЛ со стандартным двухтактным выходным каскадом нельзя объединять выходы, так как это может привести к возникновению на выходе очень больших сквозных токов. Такое объединение возможно только в микросхемах с открытым коллектором на выходе, который может быть подключен к шине питания $U_{\text{п}}$ с помощью внешнего резистора, или в микросхемах с тремя устойчивыми состояниями. Для увеличения нагрузочной способности микросхемы допускается параллельное включение входов и выходов двух элементов из одного корпуса.

Микросхемы с открытым коллектором. Возьмем за основу ранее рассмотренную схему на рис. 8.1. Выходы некоторых микросхем выполнены так, что верхний выходной транзистор *VT3* и относящиеся к нему элементы отсутствуют, т. е. нижний транзистор *VT4* имеет свободный или открытый коллектор, который является выходом ЛЭ. Этот транзистор подключается к источнику питания через внешний резистор или через элемент индикации, реле и т. п. Такие микросхемы (рис. 8.9) используются для управления внешними устройствами, которые могут к тому же работать от других источников питания, имеющих более высокое напряжение (например, элемент К155ЛА11 — до 30 В). При подключении к внешнему резистору элемент выполняет функцию И-НЕ. В условном графическом обозначении элемента с открытым коллектором имеется специальный значок — ромб (или подчеркнутый ромб). Такие элементы приспо-

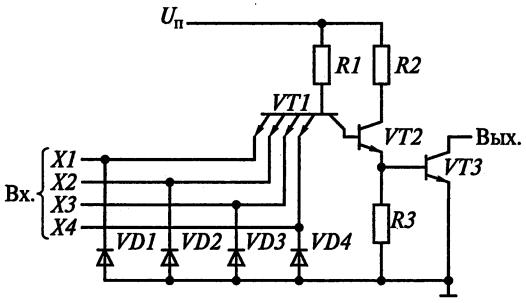


Рис. 8.9

соблены для выполнения логических операций — организации монтажной (проводной) логики. Параллельное объединение нескольких выходов и подсоединение их к общей нагрузке обеспечивают *условное* выполнение функции И-ИЛИ-НЕ (рис. 8.10, *а*). Действительно, при наличии лог. 1 на входах одного из логических элементов И-НЕ на соответствующем выходе появляется лог. 0, которого достаточно, чтобы и на выходе схемы с монтажной логикой был лог. 0 — функция ИЛИ, т.е. при этом как бы отрабатывается функция И-ИЛИ-НЕ (монтажное ИЛИ):

$$F = \overline{AB \vee CD \vee EF}.$$

С другой стороны, на объединенном выходе лог. 1 появится только тогда, когда на выходе каждого логического элемента будут лог. 1. Следовательно, объединение открытых коллекторов на общую нагрузку позволяет рассматривать данную схему как реализующую функцию И, поэтому такую схему еще называют монтажное И. На рис 8.10, *б* показано условное обозначение монтажного ИЛИ (И) в схемах.

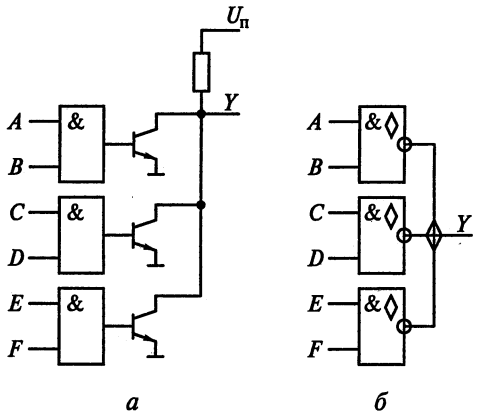


Рис. 8.10

Влияние емкости нагрузки на работу микросхем ТТЛ. Увеличение емкости нагрузки $C_н$ на выходе микросхемы снижает, естественно, ее быстродействие, увеличивая задержку распространения сигнала. Это увеличение не очень велико и составляет примерно 0,03...0,3 нс/пФ для микросхем различных серий. Микросхемы серий КР1531 и КР1533 имеют еще более слабые зависимости значений $t^{1,0}_{зд.\,р}$, $t^{0,1}_{зд.\,р}$ от емкости $C_н$. Однако с увеличением емкости $C_н$ пропорционально растет динамическая мощность $P_{C_н} = C_н \left(U^1_{вых} - U^0_{вых}\right)^2 0,5 f_т$, затрачиваемая на ее перезарядку, которая при высоких частотах $f_т$ может в несколько раз превысить мощность, потребляемую микросхемой в статическом режиме. При этом возрастают по амплитуде и длительности выбросы тока по цепям питания, что увеличивает уровень возникающих при этом помех. Поэтому значения $t^{1,0}_{зд.\,р}$ и $t^{0,1}_{зд.\,р}$ в микросхемах ТТЛ задаются для емкостей нагрузки, в 5...10 раз меньших предельных значений, равных примерно 150...300 пФ.

Фронты входных сигналов. При проектировании цифровых устройств на микросхемах ТТЛ необходимо учитывать критичность последних к длительности фронтов входных сигналов. Когда входной сигнал изменяется медленно, выходное напряжение может начать изменяться до того, как входной сигнал пересечет пороговую зону примерно от 0,8 до 2 В. В этом случае микросхема находится в режиме усиления, и наличие любой паразитной связи (например, через цепь питания) может вызвать генерацию. Следовательно, если входное напряжение микросхемы находится в пороговой зоне 0,8...2 В в течение времени, превышающего задержку сигнала, возникает опасность генерации. Кроме того, увеличение длительности фронтов входного сигнала увеличивает амплитуду и длительность сквозных токов в выходном каскаде микросхемы, что может привести к выходу ее из строя. Предельно допустимые длительности фронтов составляют: 150 нс — для микросхем серий К155 и К555; 100 нс — для микросхем серий К531 и КР1531; 1 мкс — для микросхем серии К1533. Однако для некоторых микросхем ТТЛ допускается существенно большее значение этого параметра, которое оговаривается специально в технической документации. Для предварительного формирования сигналов с крутыми фронтами следует использовать либо микросхемы типа триггера Шмитта, либо микросхемы с открытым коллекторным выходом, некритичные к длительности фронтов входных сигналов.

Рекомендации по обеспечению помехоустойчивости. Минимальная гарантированная статическая помехоустойчивость микросхем ТТЛ в состояниях лог. 1 и лог. 0 составляет 0,4...0,8 В и определяется по формулам (8.1) и (8.2) при следующих значениях:

$$U^1_{вых\,min} \approx 2,4 \text{ В}; \qquad U^1_{вх.\,пор} \approx 2,0 \text{ В};$$

$$U^0_{вх.\,пор} \approx 0,83...1,2 \text{ В}; \qquad U^0_{вх\,max} \approx 0,4 \text{ В}.$$

Однако реальные значения $U^1_{\text{пом}}$, $U^0_{\text{пом}}$ заметно больше, так как обычно $U^1_{\text{вых}} > U^1_{\text{вых min}}$, а $U^0_{\text{вых}} < U^0_{\text{вых max}}$. Динамическая помехоустойчивость растет с уменьшением длительности импульса помехи, однако в общем случае сигналы помехи имеют настолько большую длительность, что решающим фактором является помехоустойчивость по постоянному току.

При наличии мощных внешних или внутренних помех (электромагнитных или электростатических) цифровые устройства на микросхемах ТТЛ следует защищать с помощью металлических экранов, подсоединяемых к общему заземлению низкоомным проводом. Для самих источников таких помех необходимо использовать отдельные шины заземления. Помехи по цепям питания возникают также из-за токовых бросков при переключении микросхем ТТЛ, особенно больших по значению в быстродействующих микросхемах. Эффективным средством защиты от этих помех является включение конденсаторов развязки между шиной питания и общей шиной. Для подавления низкочастотных помех на плату, где может быть расположено порядка 10... 100 корпусов микросхем, устанавливаются электролитические конденсаторы емкостью 2,2... 10 мкФ, для подавления высокочастотных помех — безынерционные керамические или танталовые конденсаторы емкостью 0,047... 0,22 мкФ на каждые 10... 20 корпусов микросхем. Эти высокочастотные конденсаторы необходимо равномерно распределять по площади печатной платы. Сами линии питания должны обладать минимально возможным сопротивлением для протекания высокочастотных токов по общей шине, которая должна состоять из возможно большего числа отдельных проводников. При использовании многослойных плат необходимо предусмотреть отдельные слои для общей шины и шины питания.

Перекрестные помехи являются следствием воздействия электромагнитных полей, которые возникают при переходных процессах в соединительных линиях и оказывают воздействие на другие, близко расположенные к ним линии. Поэтому линии связи, выполненные в виде одиночных проводов, наиболее чувствительных к перекрестным помехам, не рекомендуется выполнять длиной более 25... 30 см. Такого ограничения не имеют линии связи в виде коаксиального кабеля, который не создает перекрестных помех.

Контрольные вопросы

1. Почему входы неиспользуемых логических элементов лучше соединять с общей шиной?
2. Почему выходы ЛЭ ТТЛ нельзя объединять?
3. Для чего используются схемы ТТЛ с открытым коллектором?
4. В чем заключается опасность подачи на вход ЛЭ ТТЛ сигналов с большой длительностью фронтов?

8.6. Логические элементы на КМОП-транзисторах

Логические элементы и целые цифровые узлы на КМОП-транзисторах в виде микросхем с различной степенью интеграции являются, без сомнения, самой перспективной элементной базой для создания цифровых устройств. Это обусловлено целым рядом уникальных достоинств, присущих таким микросхемам. Мощность, потребляемая логическим элементом в статическом режиме, очень мала и определяется только токами утечки. Высокая технологичность их изготовления позволяет на сравнительно небольшой площади кристалла кремния (10...100 мм^2) разместить сотни тысяч и даже миллионы логических элементов.

Большое быстродействие элементов на КМОП-транзисторах в составе БИС и СБИС позволяет последним работать на тактовых частотах, достигающих нескольких сотен мегагерц. У микросхем на КМОП-транзисторах, реализующих простые логические функции (И-НЕ, ИЛИ-НЕ и т. п.) быстродействие ниже ($t_{зд.р} \approx$ $\approx 10...100$ нс) из-за больших внешних емкостей нагрузки и монтажа ($C_н = 15...100$ пФ).

Динамическая мощность в микросхемах на КМОП-транзисторах затрачивается в основном на перезарядку нагрузочных емкостей ($P_{C_н} = C_н U_п^2 0{,}5 f_т$). Это позволяет создавать БИС и СБИС с малой потребляемой мощностью даже при высоких тактовых частотах. Современные микросхемы на КМОП-транзисторах устойчиво работают в диапазоне питающих напряжений от 2 до 15 В. По помехоустойчивости элементы на КМОП-транзисторах существенно превосходят элементы любых других типов (ТТЛ, ЭСЛ и т. д.). Диапазон помехоустойчивости находится в пределах $0{,}3U_п \le U_{пом} \le 0{,}5U_п$. Отличает эти элементы и высокая температурная стабильность их основных электрических параметров и характеристик: в диапазоне от –60 до +125 °С их температурный уход не превышает 10 %.

Большая нагрузочная способность элементов на КМОП-транзисторах ($K_{раз} \approx 50...100$) обусловлена большим входным сопротивлением этих элементов, которое имеет чисто емкостный характер, если пренебречь токами утечки. Полная развязка по постоянному току работающих друг на друга элементов, простая и понятная схемотехника основных цифровых узлов (триггеров, счетчиков, регистров, сумматоров и т. п.) существенно облегчают разработчикам проектирование цифровых устройств на КМОП-транзисторах.

Очень важно, что при использовании элементов на КМОП-транзисторах отсутствует проблема компромисса в выборе между потребляемой мощностью и быстродействием, характерная для всех других типов элементов. Увеличение быстродействия элемента на КМОП-транзисторах путем уменьшения сопротивления этих транзисторов тем или иным способом не изменяет мощность, потребляемую им в статическом режиме ($P_{пот} \approx 0$).

МОП-транзистор, т.е. транзистор со структурой металл—окисел—полупроводник, представляет собой по структуре конденсатор (рис. 8.11), одна из обкладок которого (затвор З) выполнена из проводящего материала (металла, полукристаллического кремния). Вторая обкладка (подложка П) представляет слабо легированный полупроводник *p*- или *n*-типа. Эти обкладки изолированы друг от друга тонким слоем (0,05...0,1 мкм) диэлектрика (обычно окисел SiO_2), выращенного на поверхности полупроводника.

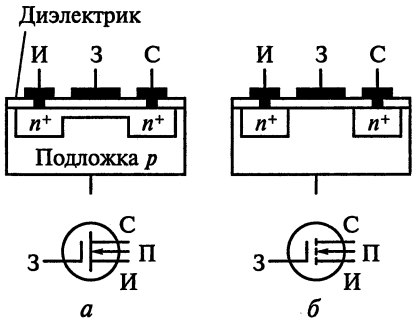


Рис. 8.11

На подложке на небольшом расстоянии друг от друга (L = = 0,5...5 мкм) расположены две области (исток И, сток С) из сильно легированного полупроводника с противоположной материалу подложки проводимостью. Таким образом, между областями исток—подложка и сток—подложка находятся два встречно включенных *p*—*n*-перехода. Поэтому ток между стоком и истоком может протекать (при $U_{С-И} \neq 0$) только в случае наличия между ними проводящего канала на границе диэлектрик—подложка с той же проводимостью, что у стока и истока. Этот канал может быть встроенным (см. рис. 8.11, *а*), т.е. существующим изначально, даже при нулевом значении поперечного электрического поля ($U_{З-П} = 0$)*.

У МОП-транзисторов, используемых в цифровых схемах, канал индуцированного типа (см. рис. 8.11, *б*) возникает (индуцируется) только под действием поперечного электрического поля, при соответствующей полярности приложенного к затвору относительно подложки напряжения.

В дальнейшем будут рассматриваться МОП-транзисторы только с индуцированным каналом. МОП-транзисторы могут быть с каналами *p*- и *n*-типа. Каждый из этих типов транзисторов имеет свои характерные особенности.

Для *МОП-транзисторов с каналом p-типа* характерны области стока и истока *p*-типа, подложка *n*-типа, отпирающее напряжение, приводящее к образованию канала $U_{З-И} < 0$, и рабочий ток, создаваемый только дырками.

Для *МОП-транзисторов с каналом n-типа* характерны области стока и истока *n*-типа, подложка *p*-типа, отпирающее напряже-

* Встроенный канал создается либо легированием поверхности подложки соответствующей примесью, либо благодаря наличию в окисле электрического заряда (обычно положительного).

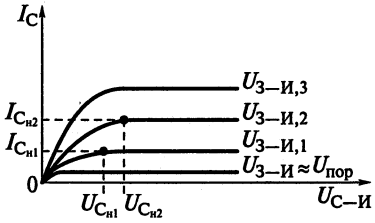


Рис. 8.12

ние, приводящее к образованию канала $U_{З-И} > 0$, и рабочий ток, создаваемый только электронами.

Кроме того, необходимо иметь в виду, что при изображении МОП-транзистора на схемах стрелкой обозначается тип подложки, который противоположен типу канала.

Для МОП-транзисторов характерна полная взаимозаменяемость стока и истока. Истоком считается электрод, из которого вытекают основные носители, у МОП-транзисторов с каналом *p*-типа напряжение между стоком и истоком $U_{С-И} < 0$, а у МОП-транзисторов с каналом *n*-типа $U_{С-И} > 0$.

Работа МОП-транзистора основана на изменении концентрации свободных носителей заряда на поверхности полупроводника под действием электрического поля. Например, у МОП-транзистора с каналом *p*-типа увеличение отрицательного напряжения на затворе приводит к росту концентрации свободных положительных носителей заряда, которая начинает превышать концентрацию электронов. При определенном напряжении между затвором и истоком*, которое называется пороговым ($|U_{пор}| \approx 1...3$ В), тип проводимости в этой области изменяется на противоположный, образуя канал, в результате чего между стоком и истоком возникает ток. Поскольку сопротивление диэлектрика очень велико при любой полярности напряжения $U_{З-И}$ ($I_З \approx 0{,}01$ мкА), МОП-транзистор является полупроводниковым аналогом электронной лампы. Зависимость тока стока $I_С = f(U_{С-И})$ при $|U_{З-И}| > |U_{пор}|$ и малых напряжениях $U_{С-И}$ близка к линейной (рис. 8.12):

$$I_С = C_0 \mu (W/L)(U_{З-И} - U_{пор})\, U_{С-И},$$

где C_0 — удельная емкость диэлектрика ($C_0 = 2 \cdot 10^{-8}$ Ф/см^2 при толщине диэлектрика $d = 0{,}1$ мкм и $\varepsilon_д = 3{,}5$ — оксид SiO_2); μ — подвижность свободных носителей в канале ($\mu_p = 150...220$ см^2/В·с; $\mu_n = 450...650$ см^2/В·с); W — ширина канала; L — длина канала.

При этом сопротивление открытого МОП-транзистора

$$r_{С-И} = U_{С-И} / I_С = \frac{1}{C_0 \mu (W/L)(U_{З-И} - U_{пор})}. \qquad (8.3)$$

Таким образом, сопротивление $r_{С-И}$ МОП-транзистора можно изменять, меняя его геометрические размеры (W, L). Поскольку $\mu_p < \mu_n$, сопротивление $r_{С-И}$ МОП-транзисторов с каналом *n*-типа

* У МОП-транзисторов все напряжения принято отсчитывать относительно истока.

при прочих условиях примерно в три раза меньше, чем МОП-транзисторов с каналом *p*-типа.

С увеличением напряжения $U_{С-И}$ поперечное электрическое поле уменьшается в направлении от истока к стоку и в этом же направлении падает концентрация индуцированных свободных носителей в канале. При этом ток стока растет все медленнее по сравнению с участком характеристики $I_С = f(U_{С-И})$, на котором $|U_{С-И}| \ll U_{З-И}$ (см. рис. 8.12). Наконец при напряжении насыщения $|U_{С-И,\,нас}| = |U_{З-И}| - |U_{пор}|$ канал у стока исчезает, и рост тока стока при дальнейшем увеличении напряжения $U_{С-И}$ практически прекращается. При напряжении $|U_{С-И}| > |U_{С-И,\,нас}|$ слой объемного заряда, создаваемого неподвижными зарядами примеси акцепторов (канал *n*-типа) или доноров (канал *p*-типа), выходит на поверхность в области стока. Поэтому разность напряжений $\Delta U = |U_{С-И} - U_{С-И,\,нас}|$ падает не на канале, а на высокоомном слое объемного заряда. Происходящее с ростом напряжения ΔU незначительное расширение этого слоя в сторону истока несколько уменьшает длину канала и обусловливает небольшой рост тока стока в области насыщения.

В крутой области характеристики $I_С = f(U_{С-И})$ используется удобная для инженерных расчетов аппроксимация:

$$I_С = C_0 \mu (W/L)\left[\left(U_{З-И} - U_{пор}\right)U_{С-И} - 1/2U_{С-И}^2\right]. \qquad (8.4)$$

Эта формула в области насыщения трансформируется подстановкой $U_{С-И} = U_{З-И} - U_{пор}$ в выражение для тока стока насыщения:

$$I_{С\,нас} = 1/2\,C_0 \mu (W/L)\left(U_{З-И} - U_{пор}\right)^2.$$

До сих пор считалось, что напряжение $U_{П-И}$ между подложкой и истоком постоянно (в цифровых схемах обычно $U_{П-И} = 0$). Однако бывают случаи, когда напряжение $U_{П-И}$ может быть различным в диапазоне от нулевого до некоторого максимального значения, при котором запирающее напряжение между подложкой и истоком максимально. Увеличение запирающего напряжения приводит к расширению обедненного слоя закрытого *p*—*n*-перехода между подложкой и стоком, причем это расширение происходит в основном в сторону слабо легированной области подложки. Поскольку при этом возрастает количество дырок (в МОП-транзисторах с каналом *n*-типа) или электронов (в МОП-транзисторах с каналом *p*-типа) в приповерхностном слое, для инверсии проводимости этого слоя и создания в нем проводящего канала необходимо большее отпирающее напряжение на затворе, чем в случае $U_{П-И} = 0$.

Таким образом, у МОП-транзисторов пороговое напряжение $U_{пор}$ зависит от напряжения смещения $U_{П-И}$ и растет с его увели-

чением. Напряжение $U_{пор}$ МОП-транзисторов с каналом *n*-типа больше зависит от напряжения $U_{П-И}$, чем МОП-транзисторов с каналом *p*-типа, так как подложка *p*-типа МОП-транзистора с каналом *n*-типа легируется сильнее для исключения возможности образования встроенного канала под действием имеющегося в окисле неподвижного положительного заряда. Аппроксимация зависимости тока $I_С$ от напряжений $U_{С-И}$ и $U_{С-П}$ несколько отличается от выражения (8.4).

Контрольные вопросы

1. Чем различаются в работе МОП-транзисторы с индуцированным и встроенным каналами?
2. Чем отличаются характеристики $I_С = f(U_{С-И})$ в крутой области и в области насыщения?
3. Как влияют размеры МОП-транзистора (отношение W/L) на сопротивление его канала $r_{С-И}$?

8.7. Принципы построения логических элементов на КМОП-транзисторах

Принцип работы логических элементов на КМОП-транзисторах легко понять на примере простого инвертора, представленного на рис. 8.13, который состоит из двух встречно включенных МОП-транзисторов: *VT1* с каналом *p*-типа и *VT2* с каналом *n*-типа. Их затворы объединены и являются входом инвертора. Исток и подложка у МОП-транзистора с каналом *p*-типа соединяются с источником $U_п$, а у МОП-транзистора с каналом *n*-типа — с общей шиной. Общая точка стоков этих транзисторов служит выходом. При этом их *p*—*n*-переходы исток—подложка, сток—подложка всегда заперты.

Если на вход инвертора подано напряжение низкого уровня $U^0 \approx 0$, то транзистор *VT2* заперт, а *VT1* открыт, так как его напряжение смещения затвор—исток ($U_{З-И}$) больше (по модулю) порогового напряжения $U_{пор\,p} = -(1{,}5...2{,}5)$ В:

$$|U_{З-И}| = U_п > |U_{пор\,p}|.$$

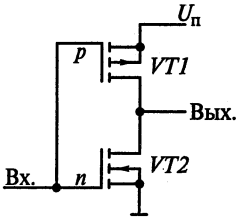


Рис. 8.13

Поскольку входное сопротивление следующего каскада очень велико, то напряжение на выходе инвертора $U^1_{вых} \approx U_п$, а ток стока у транзисторов *VT1* и *VT2* близок к нулю. При высоком уровне входного напряжения $U^1_{вх} \approx U_п$ транзистор *VT1* закрыт, а транзистор *VT2* открыт и выходное напряжение $U^0_{вых} \approx 0$.

У открытого транзистора *VT1* напряжение $U_{С-И} \approx 0$ (при $U^1_{вых} \approx U_п$) так же, как у откры-

того транзистора *VT2* (при $U^0_{\text{вых}} \approx 0$). Таким образом, открытые транзисторы инвертора находятся в крутой области характеристики $I_{\text{С}} = f(U_{\text{С–И}})$ при $U_{\text{С–И}} \approx 0$, и их сопротивления $r_{\text{С–И}\,p}$, $r_{\text{С–И}\,n}$, определяющие выходное сопротивление инвертора в состояниях лог. 1 и лог. 0, могут быть рассчитаны по формуле (8.3).

Желательно, чтобы $r_{\text{С–И}\,p} \approx r_{\text{С–И}\,n}$, так как в этом случае будут равны задержки распространения сигнала инвертора: $t^{1,0}_{\text{зд. р}} \approx t^{0,1}_{\text{зд. р}}$.

Кроме того, при выполнении условий $r_{\text{С–И}\,p} \approx r_{\text{С–И}\,n}$ и $\left|U_{\text{пор}\,p}\right| - U_{\text{пор}\,n} \ll U_{\text{п}}$ начало реального резкого перехода инвертора из одного логического состояния в другое на статической передаточной характеристике $U_{\text{вых}} = f(U_{\text{вх}})$ находится в точке с напряжением $U_{\text{вх}} \approx 0{,}5\ U_{\text{п}}$. Это обеспечивает одинаковую помехоустойчивость инвертора $U_{\text{пом}}$ в обоих логических состояниях, достигающую величины $0{,}5\ U_{\text{п}}$. Поскольку подвижность электронов μ_n в канале *n*-типа в 2,5...3 раза больше подвижности дырок μ_p в канале *p*-типа, для обеспечения условия $r_{\text{С–И}\,p} \approx r_{\text{С–И}\,n}$ отношение *W/L* МОП-транзисторов с каналом *p*-типа в инверторе в 2,5...3 раза больше, чем у МОП-транзисторов с каналом *n*-типа.

В схеме на КМОП-транзисторах практически всегда в паре работают транзисторы с разными видами каналов, затворы которых объединены. В результате в статическом состоянии в одном из транзисторов канал для протекания тока всегда будет отсутствовать. Кроме того, как правило, такие схемы работают на подобные себе схемы, поэтому вытекающий в нагрузку ток будет практически равен нулю, так как под затвором входной цепи микросхемы нагрузки находится диэлектрик. Этим объясняется то, что в статическом состоянии потребляемая мощность микросхем практически равна нулю ($P_{\text{пот}} = 0$), и это один из существенных факторов, позволяющий создавать схемы с высокой степенью интеграции.

Рассмотрим принцип действия логического элемента И-НЕ, схема которого представлена на рис. 8.14.

Логический элемент состоит из двух параллельно включенных МОП-транзисторов с каналом *p*-типа и двух последовательно включенных МОП-транзисторов с каналом *n*-типа. Таким образом, входной сигнал управляет парой МОП-транзисторов с каналами разной проводимости, при этом всегда один из транзисторов будет открыт, а другой закрыт.

При поступлении на оба входа напряжений, соответствующих лог. 1, на выходе логического элемента должно появиться напряжение, соответствующее лог. 0.

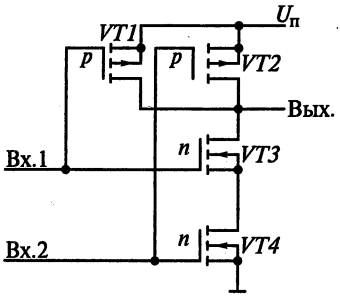


Рис. 8.14

Истоками транзисторов *VT1* и *VT2* являются электроды, подсоединенные к $U_п$, а транзисторов *VT3* и *VT4* — к общей шине. При подаче на оба входа напряжений, соответствующих лог. 1, условия для образования канала между стоком и истоком в транзисторах *VT1* и *VT2* нет, так как нет разности потенциалов между затвором и истоком. В транзисторах *VT3* и *VT4* картина иная. Между затвором и истоком транзистора *VT4* образуется разность потенциалов, что является условием образования в нем канала. Потенциал общей шины, соответствующий лог. 0, по этому каналу передается на сток транзистора *VT4* и исток транзистора *VT3*. Аналогично между затвором и истоком транзистора *VT3* образуется канал, что приводит к передаче через транзистор потенциала лог. 0 с истока на сток и соответственно на выход схемы И-НЕ.

При поступлении хотя бы на один из входов напряжения, соответствующего лог. 0, на выходе должно появиться напряжение, соответствующее лог. 1.

Например, при подаче на Вх. 2 напряжения, соответствующего лог. 0, в одном из параллельно включенных транзисторов (в нашем случае *VT2*) образуется канал, а в соответствующем ему парном транзисторе (*VT4*) канала не будет. Тогда на выход логического элемента через образовавшийся канал передается напряжение $U_п$ или лог. 1.

Подобным же образом работает схема логического элемента ИЛИ-НЕ, представленная на рис. 8.15. Логический элемент состоит из двух последовательно включенных МОП-транзисторов с каналом *p*-типа и двух параллельно включенных МОП-транзисторов с каналом *n*-типа.

При поступлении хотя бы на один из входов напряжения, соответствующего лог. 1, на выходе должно появиться напряжение, соответствующее лог. 0.

Например, при подаче на Вх. 1 напряжения, соответствующего лог. 1, в одном из параллельно включенных транзисторов (в нашем случае *VT4*) образуется канал, а в соответствующем ему парном транзисторе (*VT2*) канала не будет. Тогда на выход логического элемента через образовавшийся канал передается напряжение общей шины или лог. 0.

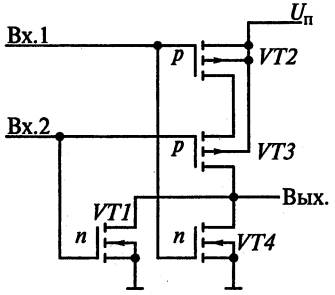


Рис. 8.15

При поступлении на оба входа напряжений, соответствующих лог. 0, на выходе логического элемента должно появиться напряжение, соответствующее лог. 1.

При подаче на оба входа напряжений, соответствующих лог. 0, условия для образования канала между стоком и истоком в транзисторах *VT1* и *VT4* нет,

так как нет разности потенциалов между затвором и истоком. В транзисторах *VT2* и *VT3* картина иная. Между затвором и истоком транзистора *VT2* образуется разность потенциалов, что является условием для образования в нем канала. Потенциал $U_п$, соответствующий лог. 1, по этому каналу передается на сток транзистора *VT2* и исток транзистора *VT3*. Аналогично между затвором и истоком транзистора *VT3* образуется канал, что приводит к передаче через транзистор потенциала лог. 1 с источника $U_п$ на сток и соответственно на выход схемы ИЛИ-НЕ.

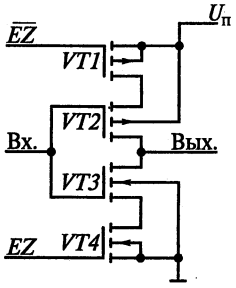


Рис. 8.16

Как видно из схем логических элементов И-НЕ, ИЛИ-НЕ (см. рис. 8.14, 8.15), при параллельном соединении МОП-транзисторов одного типа МОП-транзисторы другого типа необходимо соединять последовательно. В элементе ИЛИ-НЕ МОП-транзисторы с каналом *n*-типа соединяются параллельно, МОП-транзисторы с каналом *p*-типа — последовательно, а в элементе И-НЕ, наоборот. Поскольку при последовательном соединении N транзисторов их эквивалентное сопротивление возрастает в N раз ($r_{С–И\ экв} = Nr_{С–И}$), в микросхеме ИЛИ-НЕ для обеспечения равенства $Nr_{С–И\ p} = r_{С–И\ n}$ отношение W/L МОП-транзисторов с каналом *p*-типа должно быть больше в $(2{,}5 \ldots 3)N$ раза, чем МОП-транзисторов с каналом *n*-типа, так как подвижность носителей в транзисторе с *p*-каналом в 2,5...3 раза меньше, чем в транзисторах с *n*-каналом. Вызванное этим увеличение размеров МОП-транзисторов с *p*-каналом на кристалле ограничивает значение N (обычно $N \le 4$) и коэффициент объединения $K_{об}$ элемента ИЛИ-НЕ.

Следует отметить, что выходное сопротивление элемента ИЛИ-НЕ в состоянии лог. 0 зависит от числа открытых МОП-транзисторов с каналом *n*-типа, т.е. от комбинации входных сигналов. Поэтому у элемента ИЛИ-НЕ $t^{1,0}_{зд.\,р} \ne t^{0,1}_{зд.\,р}$ и $U^1_{пом} \ne U^0_{пом}$*, так как нельзя обеспечить такую же полную симметрию цепей на МОП-транзисторах с каналами *p*- и *n*-типа, как в инверторе.

Все сказанное справедливо и для элемента И-НЕ на КМОП-транзисторах. Однако для этого элемента при $K_{об} = N$ площадь кристалла микросхемы увеличивается в меньшей степени, так как здесь отношение W/L МОП-транзисторов с каналом *n*-типа должно быть больше, чем МОП-транзисторов с каналом *p*-типа всего лишь в $N/(2{,}5 \ldots 3)$ раза.

Широкое применение получили схемы выходного инвертора с тремя устойчивыми состояниями на КМОП-транзисторах (рис. 8.16).

В этом инверторе имеются дополнительные КМОП-транзисторы (*VT1* и *VT4*), управляемые противофазными сигналами *EZ* и $\overline{EZ}$.

* Даже в этом случае $U_{пом\ min} \ge 0{,}3\ U_п$.

При $EZ = 1$ оба транзистора открыты, и инвертор работает как обычно. При $EZ = 0$ транзисторы *VT1* и *VT4* закрыты, и инвертор находится в третьем состоянии, характеризуемом очень большим выходным сопротивлением. Это позволяет легко организовать работу нескольких логических элементов на общую шину (магистраль) передачи информации в микро-ЭВМ.

Контрольные вопросы

1. Почему в ЛЭ на КМОП-транзисторах не используются транзисторы со встроенным каналом?
2. Почему схемы на КМОП-транзисторах имеют высокое входное сопротивление?
3. Какие условия приводят к образованию канала в МОП-транзисторе?
4. В какие моменты времени относительно входных сигналов происходит максимальное и минимальное потребление энергии микросхемой на КМОП-транзисторах?
5. Какие существуют особенности в построении схем на КМОП-транзисторах, приводящие к тому, что в статическом состоянии их потребляемая мощность близка к нулю?

8.8. Физическая структура микросхемы на КМОП-транзисторах

Входное сопротивление микросхемы на КМОП-транзисторах имеет чисто емкостный характер ($C_{вх} < 1...5$ пФ). Сопротивление утечки такого конденсатора очень велико и достигает 10^{12} Ом. Электрическая прочность диэлектрика (окисла) под затвором при обычной его толщине $d = 0{,}05...0{,}1$ мкм не превышает 120...200 В.

Если вход элемента является внешним, он должен быть подсоединен к выводу микросхемы через специальную цепь защиты от статического электричества, которое накапливается из-за стекания статических электрических зарядов при контакте элемента с телом человека или оборудованием. Эти заряды возникают из-за электризации, и их потенциал достигает нескольких киловольт, что при отсутствии цепи защиты может вызвать необратимый пробой диэлектрика под затвором.

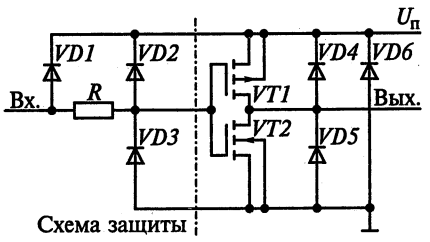


Рис. 8.17

Если вход элемента внутренний и подсоединен к выходу другого элемента на этом кристалле, то необходимость в такой защите отпадает, так как прямой контакт с источником статического электричества в этом случае отсутствует, а вход каждого последующего элемента защищает выходная цепь предыдущего.

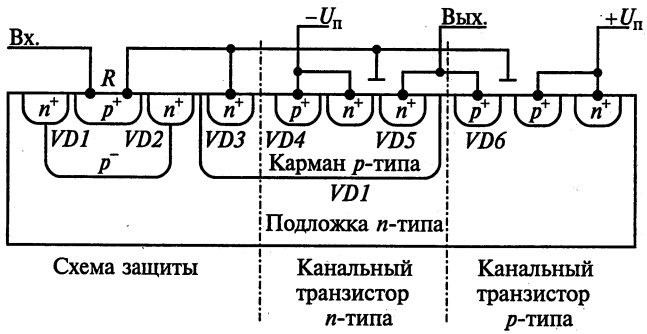


Рис. 8.18

Показанные на схеме инвертора на КМОП-транзисторах (рис. 8.17) диоды *VD4*... *VD6* на его выходе представляют собой не самостоятельные компоненты, а *p*—*n*-переходы, которые обеспечивают взаимную изоляцию областей с разными типами проводимости (рис. 8.18). Диод *VD4* — это *p*—*n*-переход между стоком транзистора *VT1* и его подложкой *n*-типа. Диод *VD5* представляет собой *p*—*n*-переход между подложкой МОП-транзистора с каналом *n*-типа и его стоком. Диод *VD6* образован *p*—*n*-переходом между подложками транзисторов *VT2* и *VT1*. Диоды *VD4* и *VD5* при нормальной работе инвертора закрыты и открываются только в том случае, если напряжение $U_{вх} > U_п + U_D$ или $U_{вх} < -U_D (U_D \approx 0{,}7$ В).

При открывании диода *VD4* или *VD5* вход внутреннего элемента подсоединяется к одной из шин питания, что существенно уменьшает амплитуду напряжения на этом входе.

Для защиты внешнего входа микросхемы используется обычно диодно-резистивная цепь (см. рис. 8.17 и 8.18), которая является неотъемлемой частью микросхемы и изготовляется в одном технологическом цикле. Эта цепь состоит из диффузионного резистора *R* сопротивлением 0,5... 1,5 кОм и диодов *VD1*... *VD3*, которые замыкают повышенные входные напряжения на одну из шин питания. Входной диод представляет собой резистивно-диодную структуру с распределенными параметрами, которая по отношению к шине питания $U_п$ подобна двум диодам. Резистор *R* и входная емкость инвертора образуют *RC*-цепочку, создающую временну́ю задержку 5... 10 нс, необходимую для отпирания диодов *VD1*... *VD3*.

8.9. Основные характеристики и параметры микросхем на КМОП-транзисторах

Такие основные параметры как быстродействие, потребляемая статическая и динамическая мощности и некоторые другие (в пересчете на один логический элемент) КМОП-микросхем с малой

и большой степенью интеграции существенно отличаются. Главная причина такого отличия — значительная разница в емкостях нагрузки: у первых $C_н = 15...50$ пФ, у вторых $C_н = 0{,}3...1$ пФ. Другая важная причина заключается в том, что БИС и СБИС стали выпускаться позже, т.е. с использованием гораздо более совершенной технологии, чем КМОП-микросхемы с малой степенью интеграции.

Значения основных параметров КМОП-микросхем серий К176, К561, К564, КР1561 и КР1564 приведены в табл. 8.5. Как видно из таблицы, достигнутый за последние годы прогресс в КМОП-технологии позволил более чем в 10 раз повысить быстродействие микросхем (средние значения $t_{зд.р}$ для микросхем серий К176, К561, К564 и КР1564). Для внутренних логических элементов КМОП БИС быстродействие выше в 15...50 раз, чем для простых микросхем, так как емкости нагрузки у элементов БИС существенно меньше.

Выходные токи характеризуют фактически сопротивление $r_{С–И}$ МОП-транзисторов в выходном каскаде и определяют быстродействие микросхемы. Выходные токи $I^0_{вых}$, $I^1_{вых}$ зависят от напряжения, поданного на выход микросхемы при их измерении, и сопротивления МОП-транзистора в выходном каскаде и растут с увеличением напряжения питания. Как следует из данных табл. 8.5, у микросхем серии К561 это сопротивление находится в диапазоне от 0,8 до 2 кОм, а у микросхем серии КР1564 — от 60 до 120 Ом. В тех случаях, когда требуется получить больший выходной ток, можно применять параллельное включение нескольких элементов, расположенных на одном кристалле, поскольку они имеют близкие характеристики.

КМОП-микросхемы, как уже отмечалось, потребляют мощность только в динамическом режиме, в моменты переключений. При этом в цепях питания возникают кратковременные импульсы тока, которые ограничены сопротивлением МОП-транзисторов ($r_{С–И} \approx \approx 0{,}1...2$ кОм). Вклад в динамическую потребляемую мощность от сквозных токов заметно меньше, чем от токов, обусловленных периодическим перезарядом внутренних и внешних емкостей. Поэтому динамическую мощность можно достаточно точно определить по формуле

$$P_{C_н} \approx C_н U_п^2 0{,}5 f_т.$$

При $C_н = 50$ пФ и $U_п = 5$ В мощность $P_{C_н} \approx 1{,}3$ МВт на частоте $f_т = 2$ МГц и $P_{C_н} \approx 13$ МВт на частоте $f_т = 20$ МГц.

Таким образом, по потребляемой мощности КМОП-микросхемы становятся соизмеримыми с микросхемами других типов (ТТЛ и ЭСЛ) на частотах $f_т \approx 20$ МГц. Естественно, что в БИС и СБИС преимущества КМОП-элементов по потребляемой мощности сохраняются до частот $f_т \approx 1$ ГГц и выше.

Таблица 8.5

Параметр	Серия				
	К176	К561	К564	КР1561	КР1564
Δt^0, °С	–40...+70	–45...+85	–60...+125	–45...+85	–45...+85
$U_п$, В	9 ± 5 %	3...15	10 ± 10 %	3...15	2...6
$I^0_{вх}$, мкА	≤ 1,0	≤ 1,0 (8, 11)	≤ |-0, 5|	≤ |-0, 3| (9)	≤ 0,1 (4)
$I^1_{вх}$, мкА	≤ 1,0	≤ 1,0 (8, 11)	≤ 1,0	≤ 0,3 (9)	≤ 0,1 (4)
$U^0_{вых}$, В	≤ 0,03 (7, 12)	≤ 0,01 (3, 11)	≤ 0,3 (8)	≤ 0,05 (3, 18) ≤ 0,5 (3, 17) ≤ 1 (8, 19)	≤ 0,26 (2, 14)
$U^1_{вых}$, В	≥ 8,2 (5, 12)	≥ 4,95 (3, 11) ≥ 9,95 (8, 11)	≥ 9,7 (8)	≥ 4,95 (3, 20) ≥ 9 (8, 21)	≥ 3,98 (2, 15)
$I_п$, мкА	≤ 3 (6)	≤ 0,5 (3, 10) ≤ 30 (8, 11)	≤ 6	≤ 1 (3) ≤ 4 (9)	≤ 2 (4)
$I^0_{вых}$, мА		≥ 0,3 (3, 25) ≥ 0,6 (8, 25)		≥ 0,44 (3, 24) ≥ 1,1 (8, 25) ≥ 3 (9, 26)	≥ 4 (2, 10, 23)
$I^1_{вых}$, мА		≥ 0,21 (3, 10, 28) ≥ 0,45 (8, 10, 28)		≥ 0,44 (3, 28) ≥ 1,1 (8, 29) ≥ 3,3 (9, 30)	≥ 4 (2, 10, 27)
$K_{раз}$	≤ 50	≤ 50	≤ 50	≤ 50	≤ 50
$t^{1,0}_{зд.р}$, нс	≤ 200 (10, 13)	≤ 180 (3, 10, 13) ≤ 115 (8, 10, 13)	≤ 110 (10, 13)	≤ 250 (3, 13) ≤ 120 (8, 13) ≤ 90 (9, 13)	≤ 90 (1, 10, 13) ≤ 18 (2, 10, 13) ≤ 15 (4, 10, 13)
$t^{0,1}_{зд.р}$, нс	≤ 200 (10, 13)	≤ 260 (3, 10, 13) ≤ 130 (8, 10, 13)	≤ 160 (10, 13)	≤ 250 (3, 13) ≤ 120 (8, 13) ≤ 90 (9, 13)	≤ 90 (1, 10, 13) ≤ 18 (2, 10, 13) ≤ 15 (4, 10, 13) ≤ 6 (4, 13, 16)
$I_{пот.дин}$, мА $U_{пом}$, В	≥ 2,5	≥ 1 (3) ≥ 2,8 (8)	≥ 2,6	≥ 1,5 (3) ≥ 3 (8)	≥ 2 (4)

Примечание. В скобках указаны номера следующих режимов измерения параметров:

1. $U_п = 2$ В;
2. $U_п = 4, 5$ В;
3. $U_п = 5$ В;
4. $U_п = 6$ В;
5. $U_п = 8,55$ В;
6. $U_п = 9,0$ В;
7. $U_п = 9,45$ В;
8. $U_п = 10$ В;
9. $U_п = 15$ В;
10. $t = +25$ °С;
11. $t = +85$ °С;
12. $R_н = 150$ кОм;
13. $C_н = 50$ пФ;
14. $I^0_{вых} = 4$ мА;
15. $I^1_{вых} = 4$ мА;
16. $f_т = 20$ МГц;
17. $U^1_{вх} = 3,5$ В;
18. $U^1_{вх} = 5$ В;
19. $U^1_{вх} = 7$ В;
20. $U^0_{вх} = 0$;
21. $U^0_{вх} = 3$В;
22. $U^0_{вх} = 4$ В;
23. $U^0_{вых} = 0,26$ В;
24. $U^0_{вых} = 0,4$ В;
25. $U^0_{вых} = 0,5$ В;
26. $U^0_{вых} = 1,5$ В;
27. $U^1_{вых} = 3,98$ В;
28. $U^1_{вых} = 4,5$ В;
29. $U^1_{вых} = 9,5$В;
30. $U^1_{вых} = 13,5$ В.

Большая помехоустойчивость КМОП-микросхем ($U_{пом} \approx 0,5\ U_п$) обеспечивается тем, что входное напряжение, при котором происходит переключение, равно примерно 0,5 $U_п$, а также тем, что падение напряжения на выходных транзисторах очень мало: $U^0_{вых} \approx 0$, $U^1_{вых} \approx U_п$. Учитывая возможный технологический разброс в пара-

метрах отдельных микросхем, заводы-изготовители дают гарантированное значение напряжения переключения с запасом, исходя из значения 0,3 $U_п$.

Контрольные вопросы

1. По каким причинам ограничено максимальное значение коэффициента $K_{раз} \leq 50$?
2. Улучшение какого параметра позволило повысить быстродействие микросхем на КМОП-транзисторах?
3. Как рассчитать значение $I_{пот.\ дин}$ при $C_н$ = 100 пФ, $U_п$ = 10 В и $f_т$ = 10 МГц?
4. Почему помехоустойчивость микросхем на КМОП-транзисторах слабо зависит от температуры в широком диапазоне ее изменения (от –60 до +125 °С)?

8.10. Особенности применения микросхем на КМОП-транзисторах при построении цифровых устройств

Последовательность подачи питающего напряжения и входных сигналов. Поскольку КМОП-структура является четырехслойной, в ней имеются паразитные биполярные транзисторы и возможно возникновение тиристорного эффекта. Как видно из рис. 8.18, паразитный *n—p—n*-транзистор (область истока МОП-транзистора *n*-типа, карман *p*-типа и подложка *n*-типа) и паразитный *p—n—p*-транзистор (область истока МОП-транзистора *p*-типа, подложка *n*-типа и карман *p*-типа) включены так, что образуют тиристор между шинами питания (рис. 8.19).

Сравнительно низкие объемные сопротивления подложки и кармана (R_n и R_p) шунтируют эмиттерные переходы паразитных биполярных транзисторов, поэтому для включения паразитного тиристора требуется достаточно большой ток. При нормальной работе КМОП-инвертора такие токи не возникают. Однако в случае включения тиристора происходит замыкание шины питания и общей шины, и в результате большой мощности, рассеиваемой на микросхеме, возможно ее повреждение. Единственный способ выключения тиристора состоит в отключении питания.

Для исключения нежелательных явлений напряжение питания на КМОП-микросхемы следует всегда подавать раньше подачи любых входных сигналов, а выключение микросхем следует выполнять в обратном порядке.

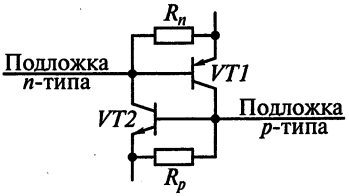


Рис. 8.19

Токоограничивающие резисторы на выходе и входе ЛЭ. Если на выходе КМОП-микросхемы включен конденсатор с достаточно большой емкостью ($C_н$ > 500 пФ), то в моменты переключений выходного ин-

вертора этой микросхемы через его выходные МОП-транзисторы протекают большие токи заряда, что может привести к повреждению прибора. Поэтому для ограничения этого тока между выходом микросхемы и конденсатором следует включить резистор, снижающий этот ток до 1...2 мА. Такой же токоограничивающий резистор необходимо включить на вход микросхемы, если через ее защитную *RC*-цепь возможно даже кратковременное протекание больших токов, создаваемых источником входного сигнала.

Подавление помех в цепи питания. Для подавления помех, возникающих в цепи питания, на плате с микросхемами между шиной питания $U_п$ и общей шиной включают конденсаторный фильтр: электролитический конденсатор емкостью 1...10 мкФ и параллельно ему в зависимости от числа микросхем на плате — несколько керамических конденсаторов емкостью 0,01...0,1 мкФ.

Неиспользуемые входы КМОП-микросхемы. Следует помнить, что входы КМОП-микросхем (в отличие от ТТЛ), недопустимо оставлять свободными. Если какой-нибудь вход окажется неподсоединенным, то на нем могут возникнуть непредсказуемые напряжения за счет наводок и связей через паразитные емкости. Следствием этого может быть не только неверное действие микросхемы, но и ее повреждение. Поэтому свободные входы микросхемы обязательно соединяют с шиной питания $U_п$ или общей шиной в зависимости от функции элемента или объединяют их с другими, задействованными входами.

Правила обращения. Подробные инструкции по обращению с КМОП-транзисторами можно найти в соответствующих справочниках. К основным правилам можно отнести следующие:

в процессе хранения и транспортирования отдельных микросхем выводы их должны быть соединены между собой, например путем обертывания металлической фольгой;

нельзя производить смену микросхем при включенном питании;

при монтаже работающий с микросхемами человек должен быть заземлен с помощью проводящего браслета, соединенного с контуром заземления помещения через резистор 0,5 МОм;

пайку выводов необходимо вести в последовательности — общий, питание, остальные контакты, применяя при этом низковольтный паяльник с заземленным жалом.

Сопряжение микросхем ТТЛ и КМОП. Для оптимизации схемных решений и снижения затрат ЛЭ ТТЛ и на КМОП-транзисторах нередко используются совместно. Выход ЛЭ ТТЛ и вход ЛЭ на КМОП-транзисторах при одинаковом напряжении питания $U_{п1} = U_{п2} = 5$ В могут быть соединены напрямую, если выход ЛЭ ТТЛ не имеет дополнительной нагрузки при выходном напряжении $U^1_{вых}$. Из-за большого входного сопротивления ЛЭ на КМОП-транзис-

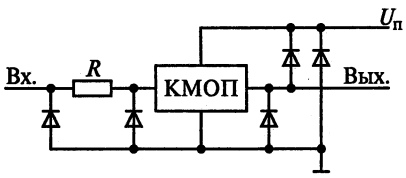


Рис. 8.20

торах напряжение $U^1_{вых}$ достигает значения около 3,5 В. Чтобы улучшить помехоустойчивость ЛЭ желательно между шиной питания $U_п$ и точкой соединения элементов включить резистор с сопротивлением 2...5 кОм, что повышает напряжение $U^1_{вых}$ до 5 В.

Если напряжение питания ЛЭ на КМОП-транзисторах $U_{п2} > U_{п1} = 5$ В, то для увеличения диапазона напряжения на выходе этого элемента до значения $U_{п2}$ можно использовать на его входе микросхему ТТЛ с открытым коллектором и повышенным коллекторным напряжением. Для этой цели служат микросхемы К133ЛН3, К133ЛА11, К155ЛН3, К155ЛА11, открытые коллекторы которых могут быть подсоединены через внешний резистор к источнику питания $U_{п2}$. Также применяют микросхемы К176ПУ5, 564ПУ7, 564ПУ8, которые содержат несколько преобразователей уровня, повышающих диапазон выходного напряжения ЛЭ ТТЛ почти до напряжения $U_{п2}$, причем микросхема 564ПУ7 выполняет эту операцию с инверсией.

Переход от микросхем на КМОП-транзисторах к микросхемам ТТЛ при $U_{п1} = U_{п2} = 5$ В может быть сделан непосредственно, если при выходном напряжении низкого уровня выходное сопротивление $r_{С–И\,п}$ ЛЭ на КМОП-транзисторах настолько мало, что выполняется условие

$$r_{С–И\,п}\, I^0_{вх} < 0{,}4...0{,}6 \text{ В},$$

где $I^0_{вх}$ — входной вытекающий ток ЛЭ ТТЛ.

В микросхемах серии КР1564 ($r_{С–И\,п} \approx 70$ Ом)* это условие выполняется для всех серий микросхем ТТЛ, а в микросхемах серий К561 и КР1561 — только для маломощных серий ТТЛ.

Для целей согласования применяются также микросхемы 564ЛН2 и 564ПУ4, содержащие по шесть одинаковых элементов, обладающих повышенной нагрузочной способностью по току, причем 564ЛН2 инвертирует входные сигналы.

Поскольку входная защитная *RC*-цепочка микросхем 564ЛН2 и 564ПУ4 выполнена по упрощенной схеме (рис. 8.20), на ее вход при напряжении питания $U_{п1} = 5$ В можно подавать повышенное напряжение вплоть до 15 В. Поэтому эти микросхемы могут быть использованы для целей согласования при $U_{п1} > U_{п2} = 5$ В. В этом случае могут быть использованы также микросхемы К176ПУ2 и К176ПУ3, содержащие по шесть одновходовых элементов, которые преобразуют входные напряжения с диапазоном, равным $U_{п1}$,

* См. параметр $I^0_{вых}$ для серии КР1564 в табл. 8.5.

в напряжения, необходимые для входов ЛЭ ТТЛ ($U^0_{вх} < 0{,}4$ В; $U^1_{вх} > 2{,}4$ В), причем микросхема К176ПУ3 выполняет эту операцию с инверсией.

Контрольные вопросы

1. Почему входы незадействованных ЛЭ в микросхеме на КМОП-транзисторах можно подсоединять к любой из шин питания?
2. Какова последовательность включения и выключения аппаратуры на КМОП-транзисторах при наличии внешних источников сигналов?
3. Почему при монтаже КМОП-микросхем необходимо использовать проводящий браслет с заземлением?
4. Какова возможность подключения напрямую входа микросхемы на КМОП-транзисторах с напряжением питания 3,5 В к выходу ТТЛ-микросхемы с напряжением питания 5 В?

Раздел III
КОМБИНАЦИОННЫЕ УСТРОЙСТВА

Глава 9
ДЕШИФРАТОРЫ И ШИФРАТОРЫ

9.1. Общие сведения о дешифраторах

Дешифратор представляет собой комбинационную схему, имеющую, как правило, n входов и $m = 2^n$ выходов.

Дешифратор — это устройство, при подаче определенного кода на вход которого, на выходе возбуждается определенная, соответствующая этому коду выходная шина.

На рис. 9.1 приведены условные графические обозначения различных типов дешифраторов, которые классифицируются по типу входов и выходов:

дешифраторы с прямыми входами и выходами (см. рис. 9.1, *а*);

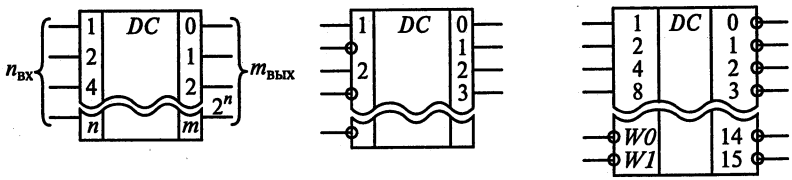
дешифраторы с парафазными входами и прямыми выходами (см. рис. 9.1, *б*);

дешифраторы с прямыми входами, инверсными выходами и наличием входов для синхроимпульсов (см. рис. 9.1, *в*).

На рис. 9.2 показана классификация дешифраторов по способу построения.

Одноступенчатые схемы дешифраторов используются в тех случаях, когда количество разрядов дешифрируемого числа не превышает количества входов логического элемента, используемого для дешифрации.

Дешифрация — это выявление реакции схемы на определенную кодовую комбинацию.



а *б* *в*

Рис. 9.1

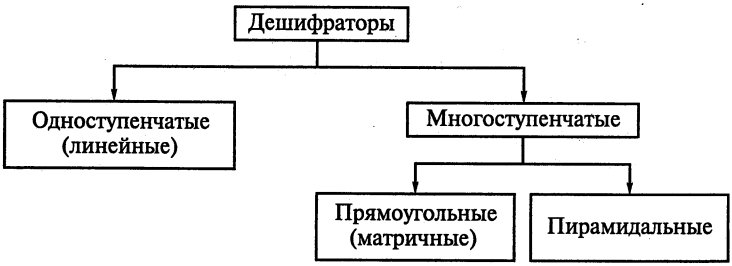


Рис. 9.2

Многоступенчатые схемы дешифраторов применяются в тех случаях, когда количество разрядов дешифрируемого числа превышает количество входов логического элемента, используемого для дешифрации.

9.2. Синтез линейного дешифратора

Чтобы создать схему полного дешифратора, охватывающего все сочетания переменных, необходимо пройти все этапы синтеза комбинационной схемы (КС).

1. *Постановка задачи.* Создать схему, реализующую функцию дешифратора, на два входа *X1*, *X2* и четыре выхода *Y0*, *Y1*, *Y2*, *Y3*. При синтезе схемы использовать логический элемент 2И-НЕ.

2. *Составление таблицы истинности.* Составляется табл. 9.1, в которую входят четыре сочетания переменных и СДНФ для каждого из них.

3. *Построение схемы линейного дешифратора.* На основе функций, представленных в табл. 9.1, и выбранного вида логического элемента строится схема. Следует учитывать, что для построения схемы необходимо иметь как прямые, так и инверсные значения переменных. Схема линейного дешифратора представлена на рис. 9.3.

Процесс дешифрации сводится к следующему. При поступлении одного из сочетаний входного кода необходимо обеспечить «со-

Таблица 9.1

Входы		Выходы				Вид функции (СДНФ)
X2	*X1*	*Y0*	*Y1*	*Y2*	*Y3*	
0	0	1	0	0	0	$Y0 = \overline{X}2 \cdot \overline{X}1$
0	1	0	1	0	0	$Y1 = \overline{X}2 \cdot X1$
1	0	0	0	1	0	$Y2 = X2 \cdot \overline{X}1$
1	1	0	0	0	1	$Y3 = X2 \cdot X1$

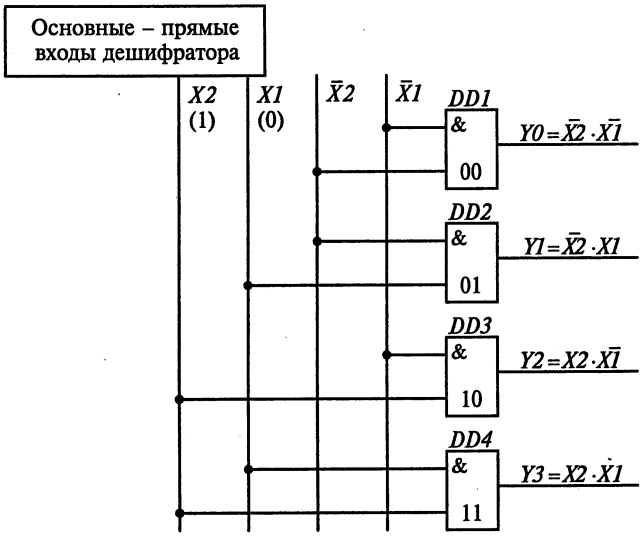


Рис. 9.3

впадение» единиц на входе определенного логического элемента И. Для этого единичные сигналы берутся с прямых или инверсных входов дешифратора, соответствующих данной комбинации. Например, подадим на входы дешифратора *X2*, *X1* лог. 1 и лог. 0 соответственно. Чтобы обеспечить «совпадение» лог. 1 на входах логического элемента *DD3* необходимо взять сигналы с входов *X2* и $\bar{X}1$.

Только тогда на соответствующей этому сочетанию переменных выходной шине *Y2* появится лог. 1.

На остальных выходных шинах *Y0*, *Y1*, *Y3* будут лог. 0, так как хотя бы на одном из входов элементов *DD1*, *DD2*, *DD4* будет присутствовать лог. 0. УГО рассмотренного типа дешифратора приведено на рис. 9.1, *б*.

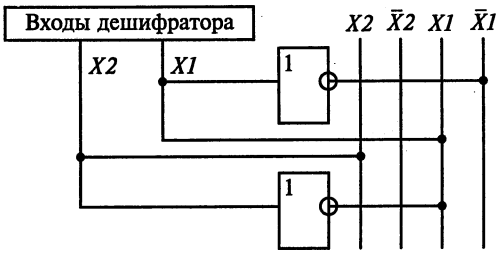


Рис. 9.4

Часто при построении дешифраторов в интегральном исполнении для уменьшения числа входов формирование инверсий входных переменных происходит в самом дешифраторе с использованием дополнительных инверторов. Такое подключение показано на рис. 9.4, а УГО для дешифратора с таким типом входов представлено на рис. 9.1, *а*.

9.3. Схема прямоугольного дешифратора

При построении схемы прямоугольного дешифратора дешифрируемое число разбивается на равные группы так, чтобы количество разрядов в каждой группе соответствовало числу входов логических схем.

Например, если дешифрируемое число четырехразрядное, то при использовании двухвходового логического элемента И образуют две группы по два разряда в каждой.

На рис. 9.5, *а* представлена схема прямоугольного дешифратора в интегральном исполнении.

На рис. 9.5, *б* дана таблица истинности, необходимая при построении дешифратора на четыре входа.

На рис. 9.5, *в* показано построение дешифратора на четыре входа на базе линейных дешифраторов, представленных в условных графических обозначениях.

Рассмотрим принцип организации дешифрации на примере одной из кодовых комбинаций, например 0110.

Разделим число на две группы.

В первую группу войдут второй и первый разряды — *X2*, *X1* (10), во вторую группу войдут соответственно четвертый и третий разряды — *X4*, *X3* (01).

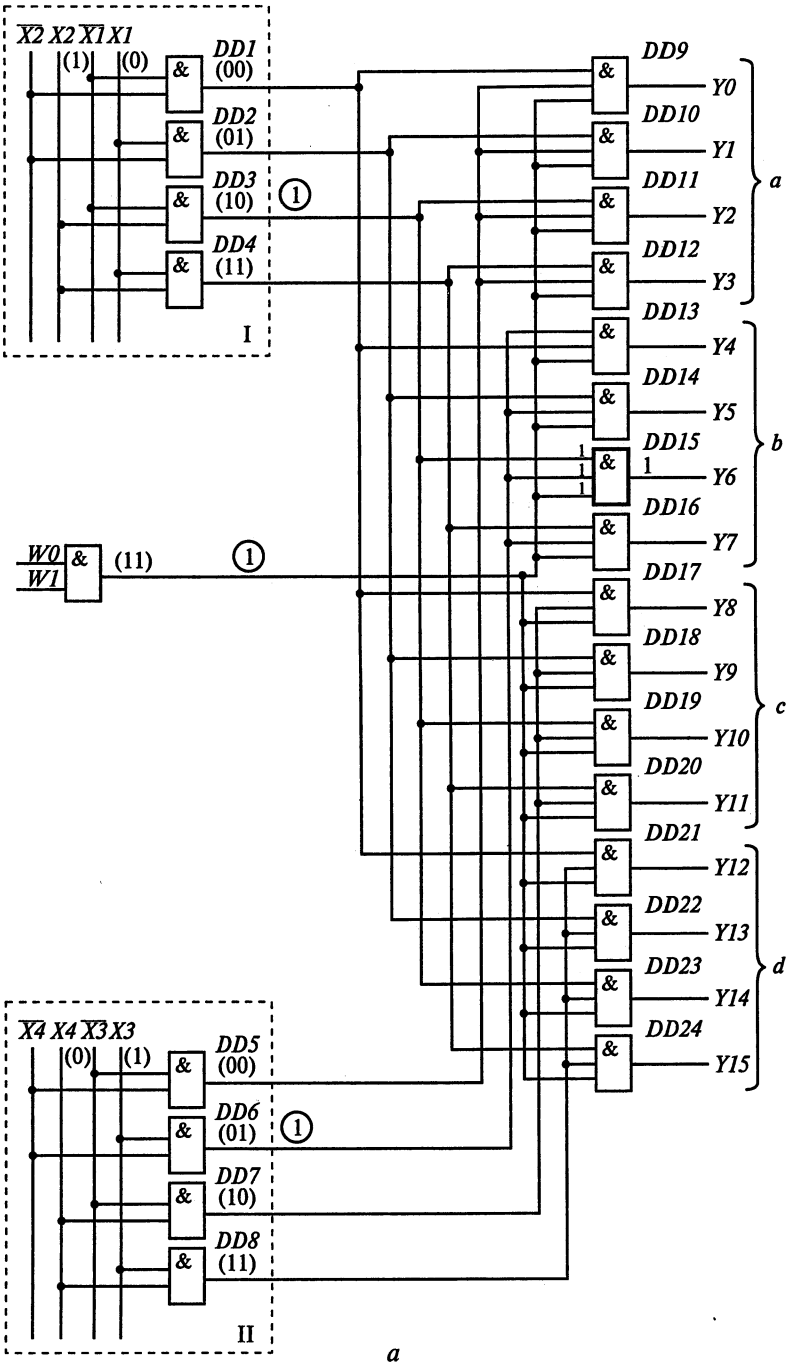
Первая ступень такого дешифратора состоит из двух линейных дешифраторов, в каждом из которых происходит дешифрация соответственно первого разряда, второго и третьего разрядов, четвертого разряда.

В каждой из двух групп найдется по одной схеме, участвующей в дешифрации определенной кодовой комбинации.

В нашем случае это схемы *DD3* и *DD6*.

В результате на выходе этих элементов появятся лог. 1. Во второй ступени дешифрации осуществляется объединение этих лог. 1 на схеме *DD15*, что приводит к полной дешифрации четырехразрядного числа.

На рис. 9.5, *а* представлены входы *W0* и *W1*, которые могут использоваться с различными целями. Основное их назначение — прием сигналов синхронизации, с помощью которых дается разрешение на дешифрацию определенной кодовой комбинации в выходном каскаде схемы.

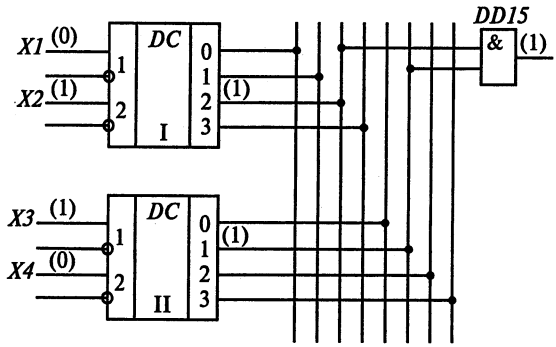

X2 X2 X1 X1
(1)
(0)
DD1
(00)
DD2
(01)
DD3
(10)
DD4
(11)
I
W0
W1
(11)
X4 X4 X3 X3
(0)
(1)
DD5
(00)
DD6
(01)
DD7
(10)
DD8
(11)
II
DD9
DD10
DD11
DD12
DD13
DD14
DD15
DD16
DD17
DD18
DD19
DD20
DD21
DD22
DD23
DD24
Y0
Y1
Y2
Y3
Y4
Y5
Y6
Y7
Y8
Y9
Y10
Y11
Y12
Y13
Y14
Y15
a
b
c
d

a

Входы				Выходы	
X4	*X3*	*X2*	*X1*	*Y*	Группы
0	0	0	0	*Y0*	*a*
0	0	0	1	*Y1*	
0	0	1	0	*Y2*	
0	0	1	1	*Y3*	
0	1	0	0	*Y4*	*b*
0	1	0	1	*Y5*	
0	1	1	0	*Y6*	
0	1	1	1	*Y7*	
1	0	0	0	*Y8*	*c*
1	0	0	1	*Y9*	
1	0	1	0	*Y10*	
1	0	1	1	*Y11*	
1	1	0	0	*Y12*	*d*
1	1	0	1	*Y13*	
1	1	1	0	*Y14*	
1	1	1	1	*Y15*	

б

Линейный дешифратор



в

Рис. 9.5

9.4. Схема пирамидального дешифратора

Пирамидальный дешифратор состоит из нескольких ступеней последовательной дешифрации. При этом в каждой ступени дешифрируется только часть числа. Рассмотрим организацию такого дешифратора на примере дешифрации определенной кодовой комбинации, например 1110.

Дешифрация будет проходить через три последовательные ступени. На первой ступени происходит дешифрация первого и второго разрядов; на второй — дешифрация результата, полученного в первой ступени с прямым и инверсным значениями третьего разряда; на третьей — дешифрация результата, полученного во второй ступени с прямым и инверсным значениями четвертого разряда. На рис. 9.6 дана схема неполного пирамидального дешифратора, представляющая собой одну из четырех его пирамид.

Полная схема дешифратора на четыре разряда будет содержать следующее количество элементов:

I ступень — $2^n = 4$, где $n = 2$ (1, 2-й разряды). Таким образом, I ступень представляет собой ранее рассмотренный линейный дешифратор на два разряда;

II ступень — $2^m = 8$, где $m = 3$ (1, 2, 3-й разряды);

III ступень — $2^k = 16$, где $k = 4$ (1, 2, 3, 4-й разряды).

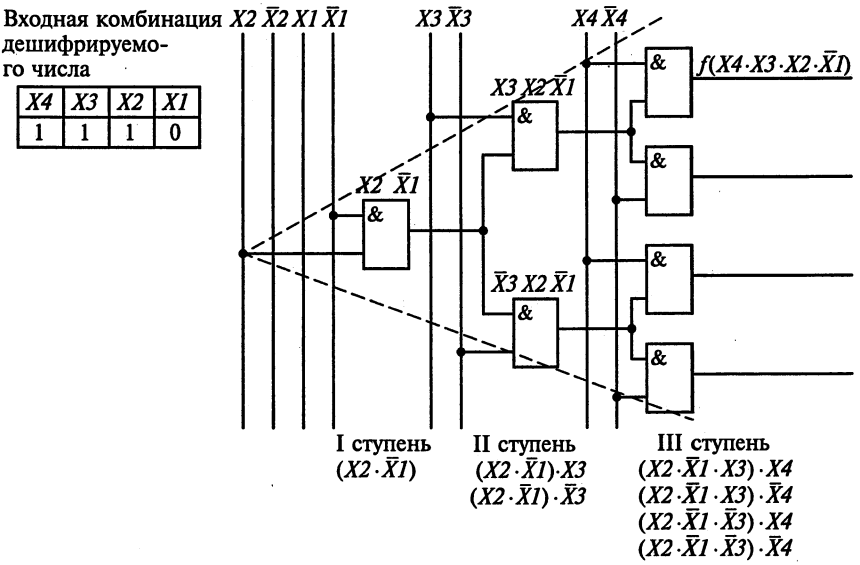


Рис. 9.6

9.5. Сравнительная оценка различных видов дешифраторов. Построение дешифраторов на большое количество входов

Основным критерием оценки комбинационных схем является быстродействие, т.е. время реакции схемы на входные сигналы. Для дешифраторов это время задержки появления сигнала дешифрации на выходе относительно времени поступления сигналов на вход схемы. Безусловно, если разрядность дешифрируемого числа не превышает количества входов логического элемента, целесообразно использовать линейный дешифратор, имеющий минимальное количество ступеней. Реже применяется пирамидальный принцип построения, так как с увеличением разрядности дешифрируемого числа резко возрастает количество ступеней и время дешифрации.

Как уже отмечалось, в схемах дешифраторов, как правило, осуществляется стробирование, разрешающее обращение к схеме дешифратора в определенные интервалы времени, в момент прихода сигналов на входы $\overline{W0}$ и $\overline{W1}$. Кроме того, эти входы используются для построения дешифраторов на большое количество входов. На рис. 9.7 показана схема, в которой использование входов $\overline{W0}$ и $\overline{W1}$ позволило построить схему на пять разрядов дешифрируемого числа.

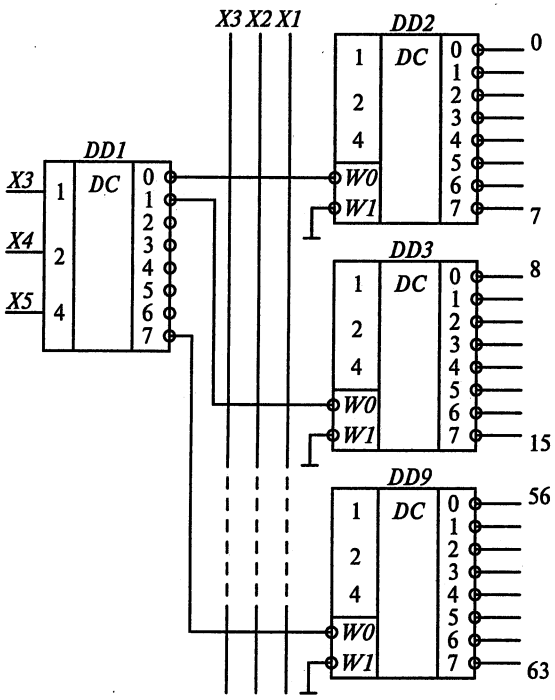


Рис. 9.7

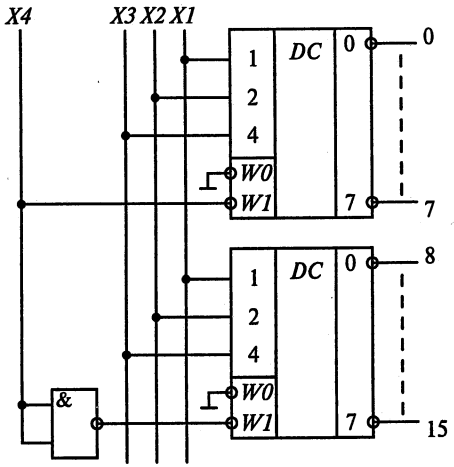


Рис. 9.8

С помощью трех старших разрядов *X5*, *X4*, *X3* кодируется доступ в одну из восьми схем (*DD2*... *DD9*) дешифраторов второй ступени, в каждой из которых в свою очередь происходит дешифрация младших разрядов *X3*, *X2*, *X1* слова.

В случае построения дешифраторов на меньшее количество входов, например на четыре, можно вместо дешифратора *DD1* использовать отдельные логические элементы И-НЕ, как это показано на рис. 9.8.

9.6. Области применения дешифраторов

Дешифраторы широко применяют в устройствах цифровой техники. Чаще всего они используются для формирования управляющих сигналов по определенному коду, в результате чего организуется доступ к другим элементам цифровых устройств.

Так, например, дешифраторы используются в качестве устройств формирования управляющих сигналов, для замыкания ключей, имеющих разные адреса, и передачи через них различной информации или, например, для организации адресного доступа к запоминающим ячейкам памяти вычислительных устройств.

Контрольные вопросы

1. В чем заключается сущность процесса дешифрации?
2. В каких случаях возможно построение одноступенчатых схем дешифраторов?
3. Каково количество выходных шин полного дешифратора при дешифрации трехразрядного числа.
4. Какое сочетание сигналов необходимо обеспечить на входах логического элемента И, отвечающего за дешифрацию кода 01 на входе дешифратора?

5. Каково различие УГО дешифраторов, представленных на рис. 9.1, *а* и *б*?

6. Возможно ли построение восьмиразрядного линейного дешифратора с использованием логических элементов 4И и четырехразрядного линейного дешифратора с использованием логических элементов 8И?

7. Возможно ли построение линейного дешифратора на логических элементах ИЛИ?

8. Какое количество логических элементов 4И должно быть в каждой из групп I ступени прямоугольного дешифратора при дешифрации восьмиразрядного числа?

9. Почему одна из структур дешифратора называется пирамидальной?

10. Определить количество ступеней пирамидального дешифратора на шесть разрядов при использовании логического элемента 2И.

11. Выбрать одну из структур построения восьмиразрядного дешифратора с использованием только логических элементов 8И или 4И. В чем заключается целесообразность такого выбора?

12. Какие дополнительные возможности в построении дешифраторов появляются при использовании входов $\overline{W0}$ и $\overline{W1}$?

13. Привести примеры использования дешифраторов.

Контрольные задания

1. Построить схему линейного дешифратора с прямыми входами и выходами на четыре разряда, используя при этом только логический элемент 4И-НЕ.

2. Построить схему линейного дешифратора с прямыми и инверсными входами и инверсными выходами на четыре разряда, используя логический элемент 4ИЛИ-НЕ.

3. Построить схему неполного прямоугольного дешифратора на восемь разрядов с использованием логического элемента 4И для следующего сочетания входных сигналов:

0001 1111, 0001 1110, 0000 1111, 0000 1110;
0101 1010, 0100 1000, 0101 1000, 0100 1010.

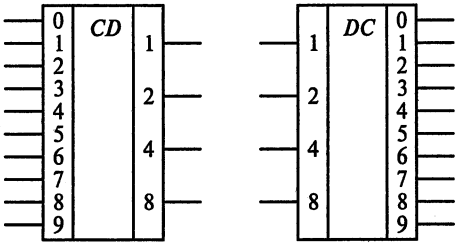
4. Построить схему неполного пирамидального дешифратора с прямыми входами на четыре разряда с использованием логического элемента 2И для следующих сочетаний входных сигналов:

1000, 1100, 0010;
0000, 0010, 0100, 1100.

5. Построить дешифратор на четыре разряда и 32 выхода, используя интегральную схему дешифратора со стробированием по входам $\overline{W0}$, $\overline{W1}$ на три разряда (аналогично схеме на рис. 9.7).

9.7. Шифраторы

Функции шифратора противоположны функциям дешифратора. В дешифраторе кодовой комбинации на входе соответствует возбуждение определенной выходной шины. В шифраторе возбуждению входной шины с определенным номером соответствует появление на выходе двоичной кодовой комбинации, соответствующей



а *б*

Рис. 9.9

этому номеру. Таким образом, это своеобразный преобразователь десятичных чисел (номера входной шины) в двоичный код.

Обозначения шифратора и дешифратора происходят от английских слов *CODER* (*CD*) и *DECODER* (*DC*) соответственно. На рис. 9.9 даны УГО этих устройств.

Произведем синтез шифратора.

1. *Постановка задачи.* Создать схему, реализующую функции шифратора, имеющую восемь входов *Y7... Y0* и три выхода *X1*, *X2*, *X3*, с использованием логических элементов 4ИЛИ.

2. *Составление таблицы истинности* (табл. 9.2).

3. *Составление СДНФ*:

$$X1 = Y1 \vee Y3 \vee Y5 \vee Y7;$$
$$X2 = Y2 \vee Y3 \vee Y6 \vee Y7;$$
$$X4 = Y4 \vee Y5 \vee Y6 \vee Y7.$$

4. *Построение схемы.*

На рис. 9.10 представлена схема шифратора, построенная в соответствии с заданием.

Из рисунка видно, что количество логических элементов, необходимых для построения, соответствует разрядности двоичных кодов. При построении схемы необходимо последовательно анали-

Таблица 9.2

Десятичный код	Входы								Выходы		
	Y0	*Y1*	*Y2*	*Y3*	*Y4*	*Y5*	*Y6*	*Y7*	*X3*	*X2*	*X1*
1	—	1	0	0	0	0	0	0	0	0	1
2	—	0	1	0	0	0	0	0	0	1	0
3	—	0	0	1	0	0	0	0	0	1	1
4	—	0	0	0	1	0	0	0	1	0	0
5	—	0	0	0	0	1	0	0	1	0	1
6	—	0	0	0	0	0	1	0	1	1	0
7	—	0	0	0	0	0	0	1	1	1	1

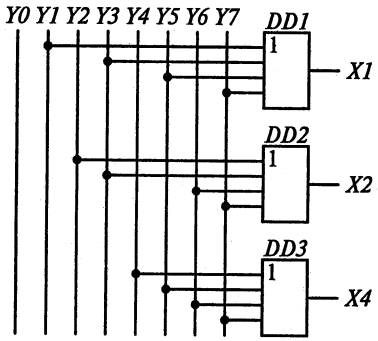


Рис. 9.10

зировать каждую входную комбинацию и единичный сигнал с входной шины заводить на те элементы, с помощью которых получаем выходной двоичный код. Например, при возбуждении пятой входной шины (лог.1 на входе *Y5*) единичный сигнал необходимо подать на входы элементов *DD1* и *DD3*, что обеспечивает появление на выходе шифратора кода $101_{(2)}$ или $5_{(10)}$.

Практический интерес представляет совместное включение дешифратора и шифратора и построение на их основе схемы преобразователя одного кода на входе в заданный код на выходе. Условия преобразования задаются таблицей (например, табл. 9.3).

Далее осуществляется переход от табличной формы записи функций к формульной:

$$Z = \bar{a}_2\bar{a}_1;$$
$$K = \bar{a}_2 a_1 \vee a_2\bar{a}_1;$$
$$J = a_2\bar{a}_1.$$

На рис. 9.11 представлена схема преобразователя, выполненная на логических элементах 2И-НЕ и 2ИЛИ-НЕ.

На рис. 9.12 показана реализация функций при совместном включении схем дешифратора и шифратора.

Таблица 9.3

Входной код		Выходной код		
a_2	a_1	Z	J	K
0	0	1	0	0
0	1	0	0	1
1	0	0	1	1
1	1	0	0	0

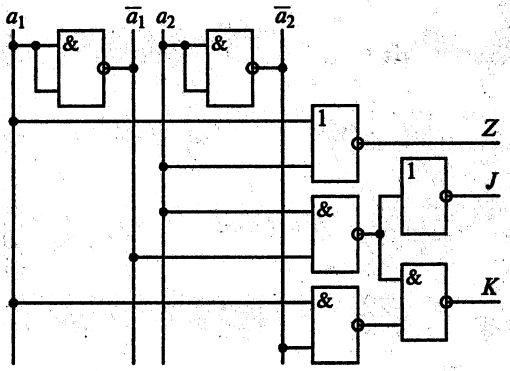


Рис. 9.11

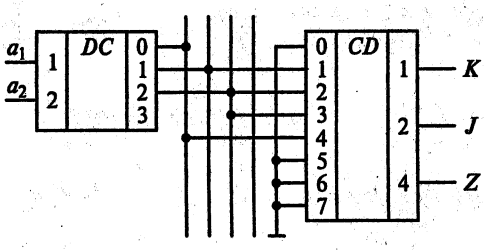


Рис. 9.12

Реализация схемы, приведенной на рис. 9.12, на ПЛМ будет рассмотрена в гл. 12.

Контрольные вопросы

1. Каким должно быть соотношение между количеством входов и выходов в схеме полного шифратора?
2. Возможно ли построение шифратора на логических элементах 4ИЛИ?
3. Почему шифраторы называют преобразователями десятичного кода в двоичный код?

Контрольные задания

1. Построить шифратор для преобразования десятичных чисел 1...15, используя логический элемент И-НЕ с любым количеством входов (указать, какие сигналы необходимо подать на вход шифратора при возбуждении определенной шины).
2. Используя логический элемент ИЛИ построить схему неполного шифратора для следующих выходных кодов:

0110, 0111, 1000, 0101;
0001, 0011, 0100, 0010;
1010, 1001, 1011, 1100.

Глава 10
МУЛЬТИПЛЕКСОРЫ И ДЕМУЛЬТИПЛЕКСОРЫ

10.1. Различные способы построения мультиплексоров

Мультиплексор — это устройство, в котором в зависимости от управляющего кода осуществляется передача информации с одного из нескольких входов на выход. В схемах мультиплексоров используются электронные ключи, замыкание которых производится под действием управляющих сигналов, приходящих, например, с дешифратора. Часто функции дешифратора и ключа объединены в одном элементе И.

На рис. 10.1 представлены различные варианты построения схем мультиплексоров:

на рис. 10.1, *а* — на базе электронных ключей 2И с управлением от внешнего дешифратора. Здесь логический элемент *DD5* реализует функцию ИЛИ, пропуская сигнал с выхода одного из элементов:

$$D = D1 \vee D2 \vee D3 \vee D4;$$

на рис. 10.1, *б* — на базе электронных ключей 3И, в которых совмещены функции ключа и дешифратора. Здесь логический элемент *DD5* реализует функцию ИЛИ, формируя выходной сигнал по закону инверсии:

$$D = \overline{\overline{D1} \cdot \overline{D2} \cdot \overline{D3} \cdot \overline{D4}} = D1 \vee D2 \vee D3 \vee D4;$$

на рис. 10.1, *в* — на базе элемента И-ИЛИ-НЕ. Схема подобного вида практически является аналогом мультиплексора, представленного на рис. 10.1, *а*. Отличие заключается в инверсном значении получаемого сигнала:

$$\overline{D} = \overline{D1 \vee D2 \vee D3 \vee D4}.$$

Схемы мультиплексоров в интегральном исполнении, как правило, имеют стробирующие входы $\overline{S}$. При подаче на этот вход лог. 0 дается разрешение на передачу информации на выход схемы в строго определенное время, определяемое длительностью строба. Кроме того, использование этого входа позволяет строить схемы мультиплексоров на большее число информационных входов.

На рис. 10.2, *а* представлена схема мультиплексора в интегральном исполнении, а на рис. 10.2, *б* — УГО мультиплексора. Сигна-

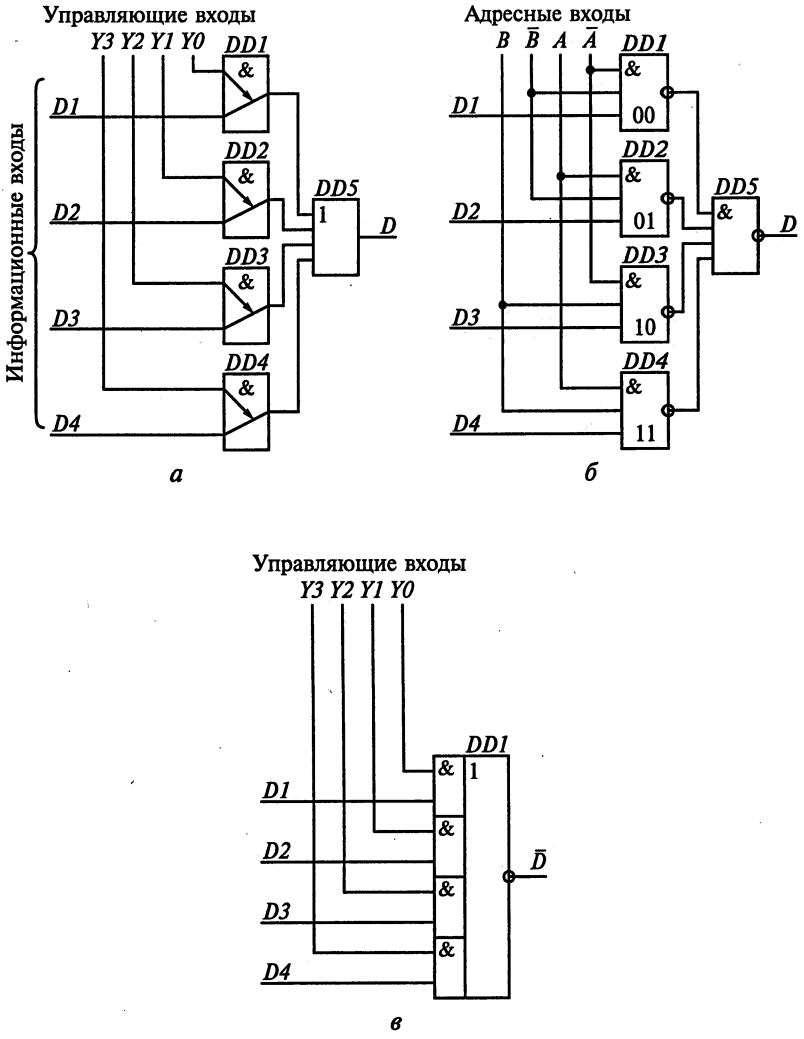


Рис. 10.1

лы проходят через логические элементы мультиплексора и реализуют один из минтермов:

$$DD1 \to S \cdot \left(\overline{A} \cdot \overline{B} \cdot D1\right); \tag{10.1}$$

$$DD2 \to S \cdot \left(A \cdot \overline{B} \cdot D2\right); \tag{10.2}$$

$$DD3 \to S \cdot \left(\overline{A} \cdot B \cdot D3\right); \tag{10.3}$$

$$DD4 \to S \cdot \left(A \cdot B \cdot D4\right). \tag{10.4}$$

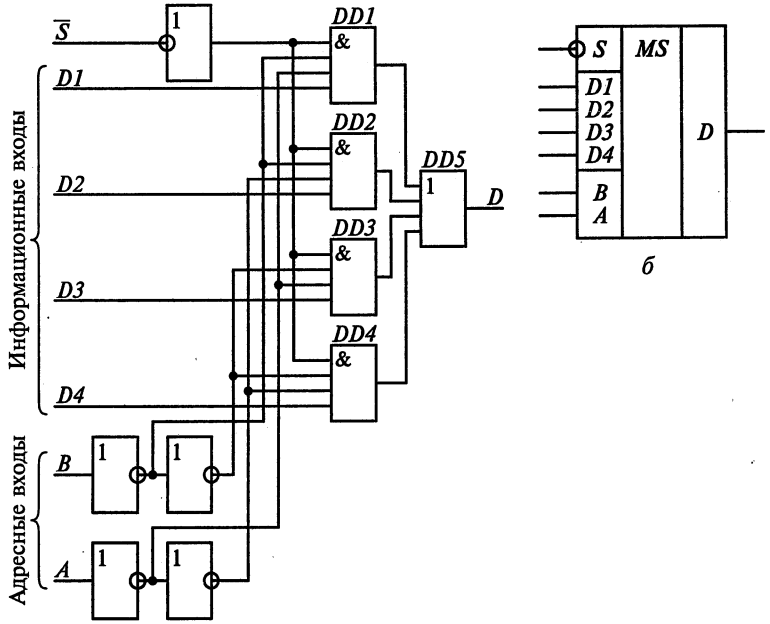


Рис. 10.2

10.2. Универсальность использования мультиплексоров

С помощью мультиплексоров могут быть построены комбинационные схемы, позволяющие реализовать различные функции. При этом число мультиплексоров, используемых при синтезе, может быть меньше, чем при использовании логических элементов. Из ранее рассмотренной схемы мультиплексора в интегральном исполнении (см. рис. 10.2) видно, что каждый из элементов И выполняет логическое умножение между тремя переменными: информацией, адресом со всеми возможными сочетаниями переменных *A* и *B* и стробом.

Рассмотрим пример использования мультиплексора для построения комбинационной схемы, реализующей логическую функцию:

$$f(X1\ X2\ X3) = X1\ X2\ \overline{X3} \vee X1\ \overline{X2}\ X3 \vee \overline{X1}\ X2\ X3 \vee \overline{X1}\ \overline{X2}\ \overline{X3}.$$

Используем адресные входы *A* и *B* для реализации различных сочетаний переменных *X1X2* в соответствии с формулами (10.1)...(10.4). Для обеспечения прямых значений *X3* можно объединить пару любых входов, например *D2*, *D3*, а для получения инверсных значений *X3* воспользоваться дополнительным внешним инвертором и оставшейся парой входов *D1*, *D4*. Тогда по аналогии

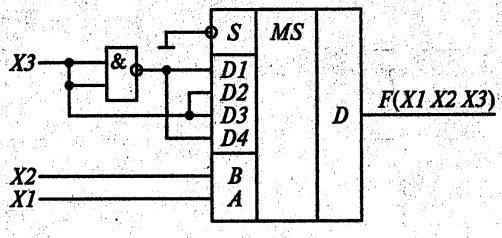


Рис. 10.3

со схемой, представленной на рис. 10.2, будем иметь следующие эквиваленты входов:

$$A - X1;\quad B - X2;\quad D2, D3 - X3;\quad D1, D4 - \overline{X3},$$

т.е. необходимые минтермы будут реализованы с использованием следующих схем:

$$DD1 \rightarrow \overline{X1} \cdot \overline{X2} \cdot \overline{X3};$$
$$DD2 \rightarrow X1 \cdot \overline{X2} \cdot X3;$$
$$DD3 \rightarrow \overline{X1} \cdot X2 \cdot X3;$$
$$DD4 \rightarrow X1 \cdot X2 \cdot \overline{X3}.$$

На рис. 10.3 представлен вариант схемной реализации заданной функции с использованием мультиплексора.

10.3. Мультиплексное дерево

Если количество передаваемых входных сигналов превышает количество входов мультиплексора, применяют несколько схем, объединенных в мультиплексное дерево.

Например, необходимо обеспечить передачу сигнала от любого из шестнадцати источников сигнала *D1*... *D16*. Каждому из источников присваивается четырехразрядный адрес в соответствии с табл. 10.1.

Таблица 10.1

Информационные входы	Адресные входы		Деление на группы	Микросхема
	X4 X3	*X2 X1*		
D1	0 0	0 0	I	*DD1*
D2	0 0	0 1		
D3	0 0	1 0		
D4	0 0	1 1		

Окончание табл. 10.1

Информационные входы	Адресные входы		Деление на группы	Микросхема
	X4 X3	*X2 X1*		
D5	0 1	0 0	II	*DD2*
D6	0 1	0 1		
D7	0 1	1 0		
D8	0 1	1 1		
D9	1 0	0 0	III	*DD3*
D10	1 0	0 1		
D11	1 0	1 0		
D12	1 0	1 1		
D13	1 1	0 0	IV	*DD4*
D14	1 1	0 1		
D15	1 1	1 0		
D16	1 1	1 1		

Адреса делятся на четыре группы так, чтобы они отличались старшими разрядами *X4* и *X3*. На первой ступени мультиплексирования (рис. 10.4) в зависимости от адреса *X2*, *X1* производится выбор одного из четырех сигналов D_i в каждой группе. На второй ступени с помощью старших адресов *X4*, *X3* производится выбор одного из четырех передаваемых сигналов первой ступени.

На рис. 10.5 показан другой способ увеличения возможного количества коммутированных сигналов, при котором используется стробирующий вход $\overline{S}$. В этом случае требуется использование дополнительных схем дешифратора и схемы ИЛИ.

Получение разрешения на коммутацию сигналов одной из четырех групп возможно только при определенном сочетании адреса старших разрядов *X3*, *X4*. На остальные три мультиплексора от дешифратора поступают запрещающие сигналы.

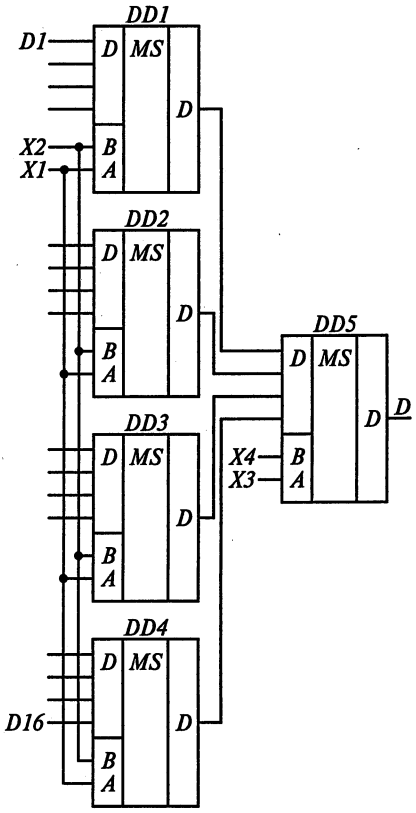


Рис. 10.4

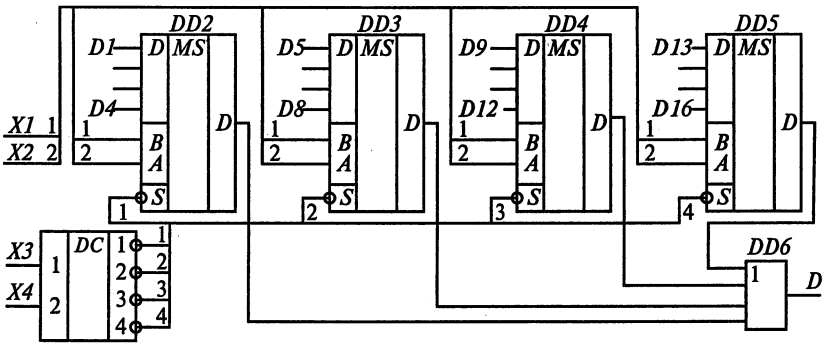


Рис. 10.5

10.4. Демультиплексоры

Функции демультиплексора противоположны функциям мультиплексора. Демультиплексор имеет один информационный вход *D*, несколько выходов и осуществляет коммутацию сигнала на один из выходов схемы, адрес которого установлен на адресных шинах. На рис. 10.6, *а* представлен один из вариантов схемы демультиплексора, а на рис. 10.6, *б* — его условное графическое обозначение. Логические элементы 3И выполняют две функции: дешифра-

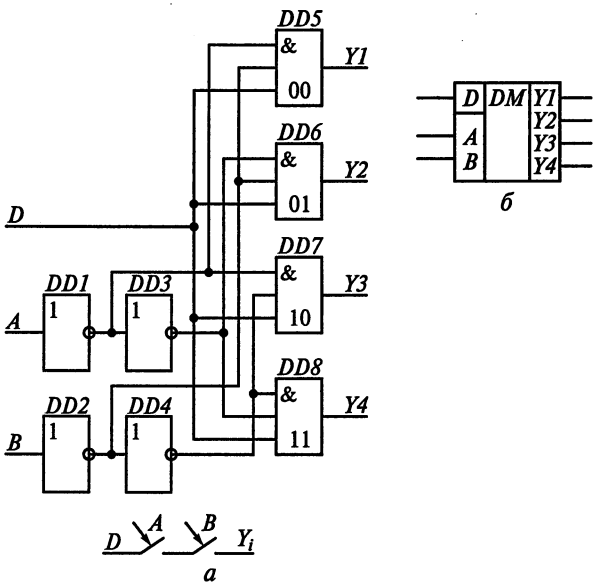


Рис. 10.6



Рис. 10.7

ции (выбор логического элемента в соответствии с адресом) и ключа, обеспечивающего передачу информации по адресу.

На рис. 10.7 показано объединение мультиплексора и демультиплексора, при котором в зависимости от поданных адресов *X1*, *X2* и *X3*, *X4* можно передавать информацию с любого из входов *D1*... *D4* на любой из выходов *Y1*... *Y4*.

Отметим, что из схемы демультиплексора можно получить дешифратор. Для этого необходимо исключить информационный вход *D*, подав на него лог. 1, а адресные входы *A* и *B* использовать в качестве информационных.

Контрольные вопросы

1. Каким должно быть соотношение между количеством информационных и адресных входов в интегральной схеме мультиплексора?
2. Какой закон алгебры логики используется при построении схемы мультиплексора (см. рис. 10.1, *б*)?
3. Каково функциональное назначение входа $\overline{S}$ в схеме мультиплексора? Каким должно быть значение сигнала на этом входе в рабочем режиме мультиплексора?
4. Может ли схема мультиплексора иметь несколько выходов?
5. Через какой логический элемент (*DD1*... *DD4*) произойдет подключение одного из информационных входов к выходу (см. рис. 10.2) при подаче на адресные входы *A* и *B* кодов 01 и 10 соответственно?
6. Какой информационный вход мультиплексора (*D1*... *D4*) подключается к выходу при подаче на адресные входы *A*, *B* кода 11?
7. В каких случаях объединяют схемы в мультиплексное дерево?
8. Какой четырехразрядный адрес (см. схему мультиплексного дерева, представленного на рис. 10.4) обеспечивает передачу информации на выход со следующих входов: *D2*, *D10*, *D6* и *D15*?
9. В чем заключаются различия схем мультиплексора и демультиплексора?
10. Определить необходимый адрес для передачи информации со входа *D* на выходы *Y2*, *Y3*, *Y1* демультиплексора, представленного на рис. 10.6.

Контрольные задания

1. Построить схему, обеспечивающую совместную работу дешифратора и мультиплексора, взяв за основу схему на рис. 10.1, *а*.

2. Построить схему мультиплексора с использованием логических элементов ИЛИ-НЕ.

3. Построить схему, обеспечивающую совместную работу дешифратора с инверсными выходами и мультиплексора на элементах ИЛИ-НЕ.

4. Используя схему мультиплексора (см. рис. 10.2) и логические элементы И-НЕ, построить схемы, реализующие следующие логические функции:

$$f_1(X1X2X3) = \overline{X1}\,X2X3 \vee X1\overline{X2}\,X3 \vee X1X2\overline{X3} \vee \overline{X1}\,\overline{X2}\,\overline{X3};$$

$$f_2(X1X2X3) = X1\overline{X2}\,X3 \vee X1\overline{X2}\,\overline{X3} \vee \overline{X1}\,X2\overline{X3}.$$

Глава 11
СУММАТОРЫ

11.1. Общие сведения

Сумматор — это узел арифметико-логического устройства (АЛУ) ЭВМ, выполняющий операцию суммирования кодов двух чисел.

Классификация сумматоров представлена на рис. 11.1

В зависимости от способа организации суммирования сумматоры подразделяются на комбинационные и накапливающие.

В *комбинационных сумматорах* значение суммы на выходе исчезает со снятием слагаемых, поданных на вход.

В *накапливающих сумматорах* результат суммирования сохраняется после снятия слагаемых со входа. В дальнейшем значение суммы получается как результат сложения предыдущего значения суммы с одним или двумя новыми значениями слагаемых.

По способу обработки многоразрядных чисел сумматоры подразделяются на параллельные и последовательные.

В *параллельных сумматорах* слагаемые всех разрядов вводятся одновременно.

В *последовательных сумматорах* осуществляется поразрядное сложение, начиная с младшего разряда, с запоминанием образовавшегося переноса до момента поступления более старших разрядов слагаемых с последующим их суммированием.

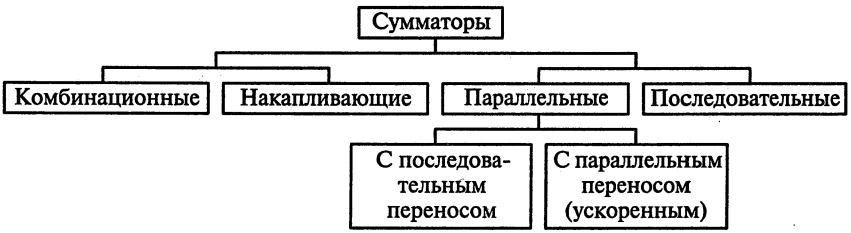


Рис. 11.1

По способу организации переноса различают сумматоры с последовательным и параллельным (ускоренным) переносом.

В *сумматорах с последовательным переносом* сигнал межразрядного переноса в каждом разряде образуется после приема сигнала переноса из предыдущего разряда, в результате чего идет последовательное формирование суммы из разряда в разряд.

В *сумматорах с параллельным переносом* разряды слагаемых обрабатываются одновременно, поэтому быстродействие в них выше, чем в сумматорах с последовательным переносом. (Это достигается благодаря наличию схем ускоренного формирования переносов во всех разрядах одновременно.)

Простейшие схемы сумматоров имеют два или три входа. Схемы, в которых производится сложение двух одноразрядных чисел, называются *полусумматорами*. Схемы, в которых производится сложение двух одноразрядных чисел и третьего слагаемого, в качестве которого, как правило, выступает единица переноса, называются *сумматорами*.

В результате операции сложения чисел на выходе сумматора образуются сигнал суммы S, который формируется как результат реализации функции «Сложение по модулю 2», и сигнал переноса P, который равен единице, если слагаемые равны единице.

11.2. Одноразрядный сумматор на два входа (полусумматор)

При создании схемы полусумматора необходимо пройти этапы синтеза комбинационной схемы.

1. Составим таблицу истинности (табл. 11.1).

Образование суммы S_i осуществляется в соответствии с логической функцией «Сложение по модулю 2»:

$$S_i = A_i \oplus B_i.$$

Таблица 11.1

Исходные данные		Результат сложения	
A_i	B_i	S_i	P_i
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

2. Составим СДНФ:

$$S_i = \overline{A_i} \cdot B_i \vee A_i \cdot \overline{B_i};$$

$$P_i = A_i \cdot B_i.$$

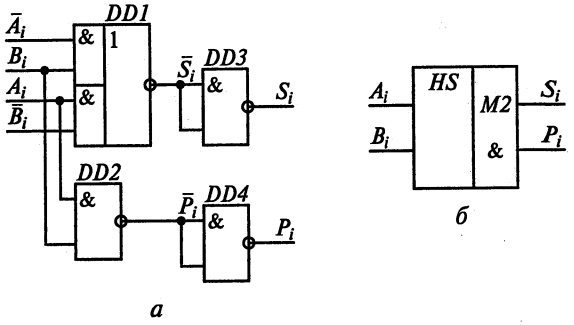


Рис. 11.2

3. Построим схему сумматора.

На рис. 11.2, *а* представлена схема с использованием произвольного набора логических элементов, а на рис. 11.2, *б* — условное графическое обозначение полусумматора.

11.3. Одноразрядный сумматор на три входа

Выполним синтез схемы одноразрядного сумматора на три входа.

1. Составим таблицу истинности (табл. 11.2).

Под P_{i-1} подразумевается третье слагаемое, в качестве которого, как правило, выступает единица переноса из более младшего разряда.

Реальное значение суммы с учетом единицы переноса может быть представлено в следующем виде:

$$S_i = (A_i \oplus B_i) \oplus P_{i-1}. \tag{11.1}$$

2. Составим СДНФ для S_i и P_i и упростим полученные выражения:

$$S_i = \bar{A}_i \bar{B}_i P_{i-1} \vee \bar{A}_i B_i \bar{P}_{i-1} \vee A_i \bar{B}_i \bar{P}_{i-1} \vee A_i B_i P_{i-1};$$

$$\begin{aligned} P_i &= \bar{A}_i B_i P_{i-1} \vee A_i \bar{B}_i P_{i-1} \vee A_i B_i \bar{P}_{i-1} \vee A_i B_i P_{i-1} \vee \boldsymbol{A_i B_i P_{i-1}} \vee \boldsymbol{A_i B_i P_{i-1}} = \\ &= B_i P_{i-1} \underbrace{\left(\bar{A}_i \vee A_i\right)}_{1} \vee A_i P_{i-1} \underbrace{\left(\bar{B}_i \vee B_i\right)}_{1} \vee A_i B_i \underbrace{\left(\bar{P}_{i-1} \vee P_{i-1}\right)}_{1} = \\ &= B_i P_{i-1} \vee A_i P_{i-1} \vee A_i B_i. \end{aligned} \tag{11.2}$$

Следовательно

$$P_i = B_i P_{i-1} \vee A_i P_{i-1} \vee A_i B_i. \tag{11.3}$$

При упрощении P_i, согласно тождеству (7.1), вводится два дополнительных минтерма.

Произведем дальнейшие преобразования минтерм сумм, сгруппировав их определенным образом:

Таблица 11.2

Исходные данные			Результат	
A_i	B_i	P_{i-1}	S_i	P_i
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$S_i = \left(A_i\overline{B}_i \vee \overline{A}_iB_i\right)\cdot\overline{P}_{i-1} \vee \left(\overline{A}_i\overline{B}_i \vee A_iB_i\right)\cdot P_{i-1} =$$
$$= (A_i \oplus B_i)\cdot\overline{P}_{i-1} \vee (A_i \circ\!\!- B_i)\cdot P_{i-1};$$
$$S_i = \left(A_i \oplus B_i\right)\cdot\overline{P}_{i-1} \vee (A_i \circ\!\!- B_i)\cdot P_{i-1}. \qquad (11.4)$$

В соответствии с выражением (11.4) можно сказать, что значение суммы будет равно единице, если хотя бы одно из слагаемых равно единице **и** отсутствует межтетрадный перенос **или** когда слагаемые совпадают по изображению **и** присутствует межтетрадный перенос.

3. На основе выражений (11.2) и (11.3) построим схему сумматора на три входа.

На рис. 11.3, *а* представлена схема одноразрядного сумматора на три входа с использованием произвольного набора элементов, а на рис. 11.3, *б* — его условное графическое обозначение.

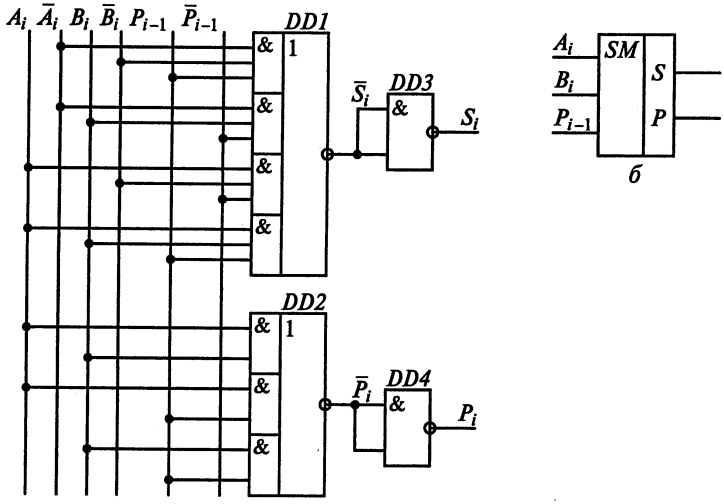
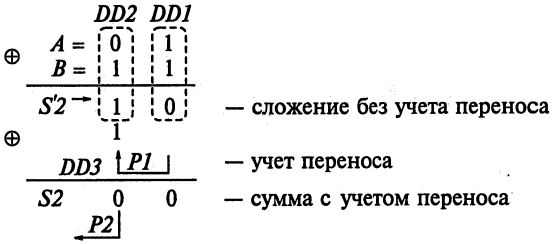


Рис. 11.3

11.4. Организация сумматора на три входа на основе полусумматоров

Для построения сумматора на основе полусумматоров выполним сложение двух двухразрядных чисел A и B:



Образование сумм производится в соответствии с реализацией функции «Сложение по модулю 2»:

$$S1 = A1 \oplus B1;$$

$$S'2 = A2 \oplus B2;$$

$$S2 = S'2 \oplus P1 = (A2 \oplus B2) \oplus P1,$$

что соответствует выражению (11.1).

На рис. 11.4 представлена схема, иллюстрирующая использование полусумматоров при организации сумматора.

В полусумматорах *DD1* и *DD2* осуществляется поразрядное сложение слагаемых A и B, причем значение суммы на выходе схемы *DD2* не учитывает перенос из первого разряда, т.е. $S'2 = (A2 \oplus B2)$.

Для получения реального значения суммы второго разряда $S2$ с учетом единицы переноса $P1$ используется схема полусумматора *DD3*, на вход которого поступает значение суммы второго разряда без учета переноса и перенос из первого разряда:

$$S2 = S'2 \oplus P1.$$

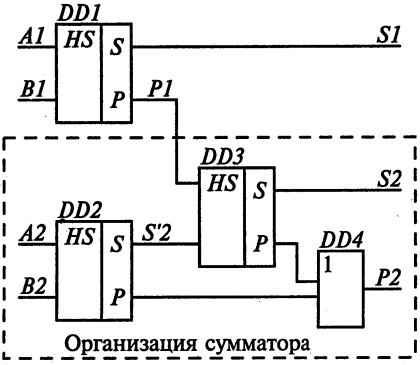


Рис. 11.4

В схеме *DD4* формируется перенос в старший разряд, учитывая возможность его появления в случае, если *A2*, *B2* равны 1 или перенос возникает при получении суммы *S2*.

11.5. Реализация сумматора на интегральных схемах

Простейшим сумматором в интегральном исполнении является одноразрядный сумматор ИМ1. На его основе построены схемы двухразрядного сумматора ИМ2 и четырехразрядного сумматора ИМ3.

Рассмотрим особенности в организации в этом случае суммы S_i. В микросхеме ИМ1 значение суммы S_i связано с использованием переноса P_i, полученного в этом же разряде. На основе таблицы истинности (см. табл. 11.2) можно вывести следующую закономерность.

Сумма S_i равна единице, если **или** A_i, **или** B_i, **или** P_{i-1} равны единице и отсутствует перенос в старший разряд; **или** если все три переменные A_i, B_i, P_{i-1} равны единице, при этом перенос P_i всегда равен единице и в формуле не учитывается:

$$S_i = (A_i \vee B_i \vee P_{i-1}) \cdot \bar{P}_i \vee A_i B_i P_{i-1}.$$

Схема одноразрядного сумматора ИМ1 (рис. 11.5) строится для прямого и инверсного значений суммы:

$$\overline{S_i} = \overline{(A_i \vee B_i \vee P_{i-1}) \cdot \bar{P}_i \vee A_i B_i P_{i-1}}.$$

Кроме того, для построения схемы необходимо инверсное значение переноса, полученного на основе выражения (11.3):

$$\overline{P_i} = \overline{A_i B_i \vee A_i P_{i-1} \vee B_i P_{i-1}}.$$

В схеме для каждого из слагаемых A и B имеется по четыре входа: два прямых и два инверсных. Элементы входной логики объединены цепью «Монтажное И», поэтому сигналы на входы A_i^* и B_i^* должны поступать от устройств с открытыми коллекторами (см. подразд. 8.5). Благодаря усложненным входной и выходной логикам данный одноразрядный сумматор обладает большими функциональными возможностями. Многоразрядные сумматоры на его основе организуются без дополнительных логических элементов и могут работать в положительной и отрицательной логиках.

Прямые и инверсные входы, принадлежащие каждому из слагаемых, одновременно не используются.

Слагаемые *A1* и *B1* подаются на входы A_i и B_i соответственно, а перенос *P0* — на вход P_{i-1}. Полученные значения суммы *S1* снимаются с выхода S_i, а значения переноса *P1* — с выхода $\bar{P}_i$.

На рис. 11.5, *б* представлено условное графическое обозначение одноразрядного сумматора ИМ1.

При работе с прямыми входами на инверсных входах устанавливается уровень логической единицы (или их оставляют свобод-

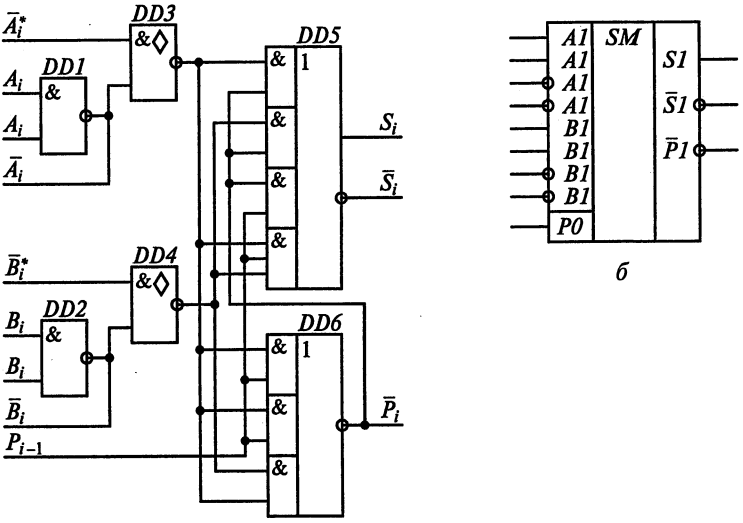


Рис. 11.5

ными). При работе с инверсными входами на прямые входы подается уровень логического нуля.

11.6. Организация многоразрядного параллельного сумматора с последовательным переносом на базе интегральных схем ИМ1, ИМ2, ИМ3

При построении многоразрядного сумматора на базе микросхемы ИМ1 используется свойство полной симметрии сумматора относительно замены входных и выходных сигналов инверсными. На рис. 11.6 представлена схема такого сумматора на три разряда.

На первый разряд сумматора входные слагаемые *A1* и *B1* подаются на прямые входы. Выходной сигнал суммы снимается с прямого выхода *S1*, а сигнал переноса — с единственного инверсного выхода $\overline{P1}$. На второй разряд сумматора входные слагаемые *A2* и *B2* подаются на инверсные входы. В свою очередь на его прямой вход *P0* подается инверсный сигнал переноса с первого разряда. Соответственно выходной прямой сигнал суммы формируется на инверсном выходе $\overline{S1}$ второго разряда, а выходной прямой сигнал переноса — на инверсном выходе $\overline{P1}$. Третий разряд сумматора организуется, как первый, четвертый, как второй и т.д.

Чередование режимов работы одноразрядного сумматора ИМ1 обеспечивает минимальную задержку распространения сигнала переноса из разряда в разряд.

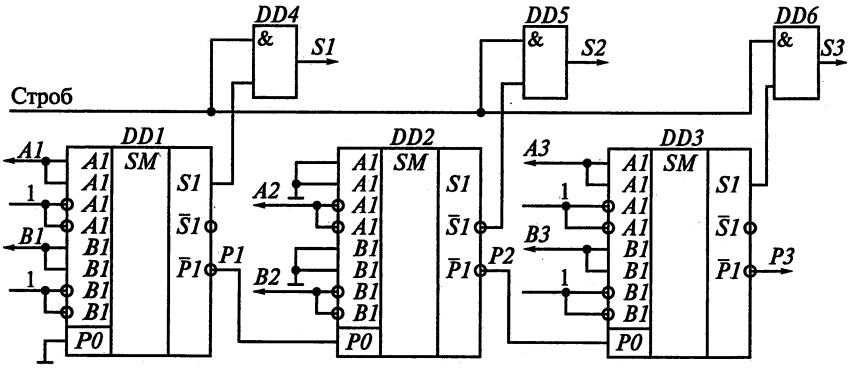


Рис. 11.6

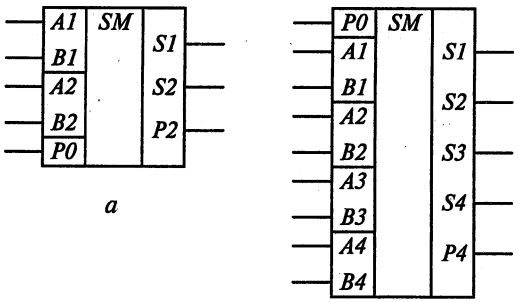
При оценке скорости работы такого сумматора исходят из предположения, что перенос возникает в каждом разряде, и поэтому время суммирования складывается из времени суммирования в каждом разряде T_i и времени распространения переноса из разряда в разряд $t_{зд.р}$. При этом время присутствия слагаемых на входе микросхем ИМ1 всех разрядов $T_{слаг}$ должно быть больше времени суммирования $T_{сум}$, т. е. $T_{слаг} > T_{сум}$. В свою очередь $T_{сум}$ определяется по формуле

$$T_{сум} = T_i + t_{зд.\,р}(n - 1),$$

где n — количество разрядов полного сумматора.

В свою очередь длительность сигнала опроса (строба) результатов суммирования выбирается с учетом максимальной задержки суммирования и должно быть меньше длительности слагаемых.

Микросхема ИМ2 (рис. 11.7, *а*) представляет собой результат объединения двух микросхем ИМ1, соединенных в соответствии с исключаемыми неиспользуемыми инверторами, а микросхема ИМ3 (рис. 11.7, *б*) — двух микросхем ИМ2.



а *б*

Рис. 11.7

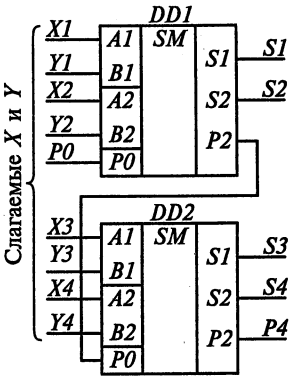


Рис. 11.8

На рис. 11.8 приведен пример организации четырехразрядного сумматора на базе микросхемы ИМ2.

В табл. 11.3 представлены значения задержек распространения переноса для трех видов сумматоров.

При построении четырехразрядного сумматора с использованием трех видов микросхем максимальное время распространения переноса будет следующим:

ИМ1 — $t_{зд.\,р} = 4(12...17)$ нс;

ИМ2 — $t_{зд.\,р} = 2(19...27)$ нс;

ИМ3 — $t_{зд.\,р} = (32...48)$ нс.

Таблица 11.3

Параметр	Микросхемы		
	ИМ1	ИМ2	ИМ3
Задержка распространения переноса $t_{зд.\,р}$, нс	12...17	19...27	32...48

Таким образом, повышение уровня интеграции приводит к снижению паразитных емкостей и улучшению временны́х параметров схемы.

11.7. Использование сумматоров в интегральном исполнении при выполнении различных арифметических операций

Наличие в микросхеме ИМ1 инверсных входов *A1* и *B1* дает возможность использовать сумматор для выполнения операций вычитания. В этом случае возможны два варианта, когда $A > B$ и $A < B$. В первом случае, когда результат вычитания положителен, разность определяется по формуле

$$(A - B)_{пр} = A_{пр} + (B_{обр} + 1),$$

т.е. вычитаемое B представляется в дополнительном коде.

Во втором случае, когда результат вычитания отрицателен, разность определяется по формуле

$$(A - B)_{обр} = -(A_{пр} + B_{обр})_{обр}.$$

Рассмотрим примеры.

Пример 1. $A > B$;

$A = 13$; $A_{пр} = 1101$;

$B = 8$; $B_{пр} = 1000$; $B_{обр} = 0111$.

$(A - B)_{пр} = A_{пр} + (B_{обр} + 1) = 1101 + 0111 + 1 = 0101_{пр(2)} = 5_{(10)}$.

Единицей переноса в старший разряд пренебрегают.

Пример 2. $A < B$;

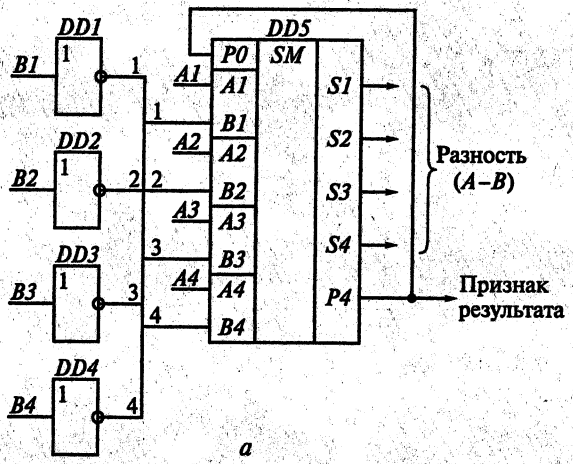
$A = 8$; $A_{пр} = 1000$;

$B = 13$; $B_{пр} = 1101$; $B_{обр} = 0010$.

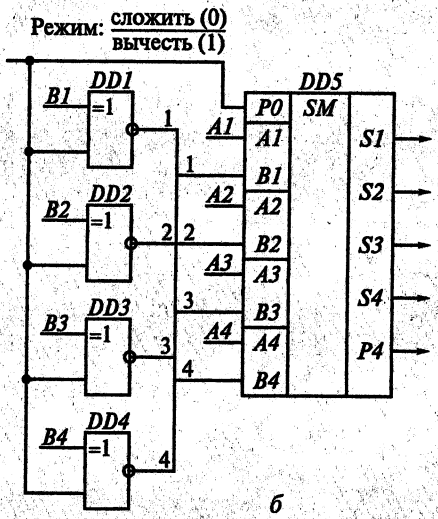
$$(A - B)_{обр} = -(A_{пр} + B_{обр})_{обр} = -(1000 + 0010) = -1010_{обр};$$

$$(A - B)_{пр} = -0101_{(2)} = -5_{(10)}.$$

Единица переноса в старший разряд отсутствует.

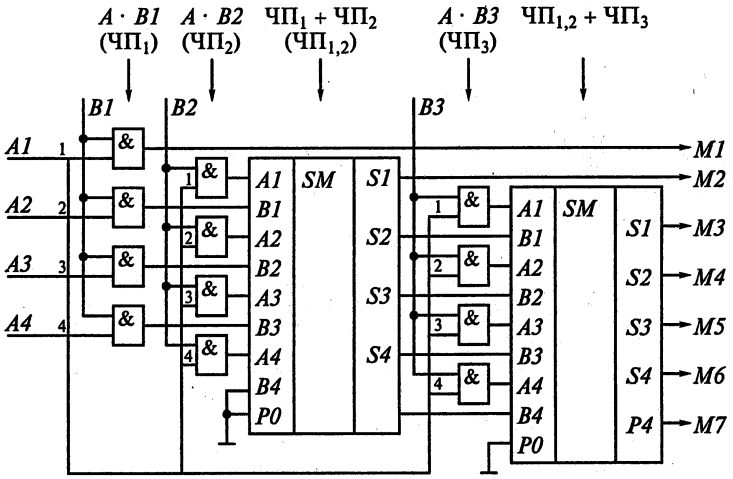


а



б

Рис. 11.9



			× A4	A3 B3	A2 B2	A1 B1	
			A4 B1	A3 B1	A2 B1	A1 B1	— ЧП1
+		A4 B2	A3 B2	A2 B2	A1 B2		— ЧП2
	A4 B3	A3 B3	A2 B3	A1 B3			— ЧП3
M7	M6	M5	M4	M3	M2	M1	

Рис. 11.10

Единица переноса в старший разряд используется в качестве признака знака результата: присутствие единицы означает, что результат положителен; а ее отсутствие, — что результат отрицателен.

Кроме того, наличие переноса в старший разряд используется в качестве единицы для получения дополнительного кода числа B, который будет равен ($B_{обр}$ + 1). В результате на выходах S будет получена разность в прямом коде; отсутствие переноса означает, что результат будет получен в обратном коде, который затем необходимо перевести в прямой. На рис. 11.9, *а* представлена схема для выполнения вычитания четырехразрядных чисел на базе микросхемы ИМ3.

Операции сложения и вычитания можно совместить, если инверторы заменить логическими схемами, выполняющими функцию «Сложение по модулю 2» (рис. 11.9, *б*).

Кроме того, на базе микросхемы ИМ3 можно построить схему для выполнения умножения (рис. 11.10).

11.8. Комбинационный сумматор с ускоренным распространением переноса

В параллельном сумматоре с последовательным переносом при увеличении разрядности числа увеличивается задержка распространения переноса. Для предотвращения этого используются схемы сумматоров с ускоренным распространением переноса. В таких схемах переносы между разрядами отсутствуют. Они формируются отдельно для каждого разряда и одновременно для всех разрядов слагаемых. Проанализируем возможные варианты образования переноса в старший разряд.

Перенос в старший разряд возможен в двух случаях:

когда **и** A_i, **и** B_i равны единице (состояние непосредственно образованного переноса G_i):

$$P_i = G_i = A_i \cdot B_i;$$

когда **или** A_i, **или** B_i равны единице (состояние ожидания переноса T_i) **и** есть перенос из младшего разряда:

$$P_i = T_i \cdot P_{i-1} = (A_i \vee B_i) \cdot P_{i-1}.$$

Рассмотрим различные варианты, когда при сложении двух четырехразрядных чисел могут возникать единицы переноса.

1. Проведем анализ причин возникновения переноса из первого разряда во второй.

Образование переноса P_1 возможно при наличии состояния непосредственно образованного переноса G_1 в первом разряде:

$$P_1 = G_1 = A_1 \cdot B_1.$$

Это состояние можно выразить следующим образом:

Слагаемое	Состояние G_1
A	1
B	1

Образование переноса P_1 возможно при наличии состояния ожидания переноса T_1 в первом разряде:

$$P_1 = T_1 \cdot P_0 = (A_1 \vee B_1) \cdot P_0.$$

Это состояние можно выразить следующим образом:

Слагаемое	Состояние T_1
A	0
B	1

$P_0 = 1$

В общем случае перенос возможен, когда

$$P_1 = G_1 \vee T_1 \cdot P_0 = A_1 \cdot B_1 \vee (A_1 \vee B_1) \cdot P_0. \qquad (11.5)$$

2. Проведем анализ причин появления переноса из второго разряда в третий.

Образование переноса P_2 возможно при наличии состояния G_2 во втором разряде:

$$P_2 = G_2 = A_2 \cdot B_2.$$

Это состояние можно выразить следующим образом:

Слагаемое	Состояние	
	G_2	1-й разряд
A B	1 1	Безразлично

Образование переноса P_2 возможно при наличии состояния ожидания переноса T_2 во втором разряде и состояния G_1 в первом:

$$P_2 = T_2 \cdot G_1 = (A_2 \vee B_2) \cdot (A_1 \cdot B_1).$$

Это состояние можно выразить следующим образом:

Слагаемое	Состояние	
	T_2	G_1
A B	1 0	1 1

Образование переноса P_2 возможно при состояниях ожидания переноса T_2, T_1:

$$P_2 = T_2 \cdot T_1 \cdot P_0 = (A_2 \vee B_2) \cdot (A_1 \vee B_1) \cdot P_0.$$

Это состояние можно выразить следующим образом:

Слагаемое	Состояние	
	T_2	T_1
A B	0 1	1 0

$P_0 = 1$

В общем случае, для того чтобы организовать перенос из второго разряда в третий, необходимо выполнить одно из трех условий:

$$P_2 = G_2 \vee T_2 \cdot G_1 \vee T_2 \cdot T_1 \cdot P_0 = (A_2 \cdot B_2) \vee$$
$$\vee (A_2 \vee B_2) \cdot (A_1 \cdot B_1) \vee (A_2 \vee B_2) \cdot (A_1 \vee B_1) \cdot P_0. \quad (11.6)$$

3. Аналогично можно записать формулы образования переноса из третьего разряда в четвертый и из четвертого в пятый:

$$P_3 = G_3 \vee T_3 \cdot G_2 \vee T_3 \cdot T_2 \cdot G_1 \vee T_3 \cdot T_2 \cdot T_1 \cdot P_0 = (A_3 \cdot B_3) \vee$$
$$\vee (A_3 \vee B_3) \cdot (A_2 \cdot B_2) \vee (A_3 \vee B_3) \cdot (A_2 \vee B_2) \cdot (A_1 \cdot B_1) \vee$$
$$\vee (A_3 \vee B_3) \cdot (A_2 \vee B_2) \cdot (A_1 \vee B_1) \cdot P_0; \quad (11.7)$$

$$P_4 = G_4 \vee T_4 \cdot G_3 \vee T_4 \cdot T_3 \cdot G_2 \vee T_4 \cdot T_3 \cdot T_2 \cdot G_1 \vee T_4 \cdot T_3 \cdot T_2 \cdot T_1 \cdot P_0 =$$
$$= (A_4 \cdot B_4) \vee (A_4 \vee B_4) \cdot (A_3 \cdot B_3) \vee (A_4 \vee B_4) \cdot (A_3 \vee B_3) \cdot (A_2 \cdot B_2) \vee$$
$$\vee (A_4 \vee B_4) \cdot (A_3 \vee B_3) \cdot (A_2 \vee B_2) \cdot (A_1 \cdot B_1) \vee$$
$$\vee (A_4 \vee B_4) \cdot (A_3 \vee B_3) \cdot (A_2 \vee B_2) \cdot (A_1 \vee B_1) \cdot P_0. \quad (11.8)$$

Следовательно, в общем случае значение образования переноса в любом из разрядов подчиняется следующей закономерности:

$$P_i = G_i \vee T_i \cdot G_{i-1} \vee T_i \cdot T_{i-1} \cdot G_{i-2} \vee T_i \cdot T_{i-1} \cdot T_{i-2} \cdot G_{i-3} \vee \ldots$$
$$\ldots \vee T_i \cdot T_{i-1} \cdot T_{i-2} \cdot T_{i-(n-1)} \cdot P_0.$$

11.9. Сумматор с ускоренным переносом на базе интегральной микросхемы ИМ6

В общем случае образование суммы организуется в соответствии с формулой (11.1) и может быть реализовано с помощью схемы, представленной на рис. 11.11.

На схеме *DD1* реализуется функция «Сложение по модулю 2» слагаемых A_i и B_i без учета единицы переноса. В микросхеме ИМ6 логическая функция «Сложение по модулю 2» реализуется на логическом элементе «Запрет», на вход которого поступают инверсные значения логических состояний G_i и T_i (рис. 11.12).

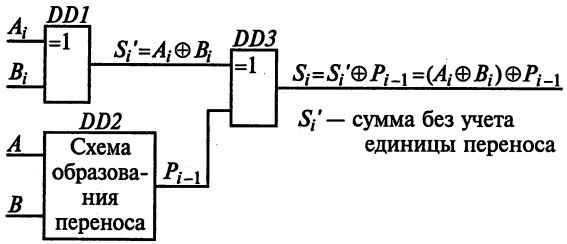


Рис. 11.11

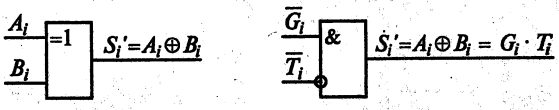


Рис. 11.12

Равнозначность замены одного логического элемента другим можно доказать с помощью таблицы истинности (табл. 11.4).

Схема организации переноса строится на элементах И-ИЛИ-НЕ в соответствии с ранее выведенными формулами (11.5)...(11.8), но в другом виде:

$$P_1 = \underbrace{G_1 \vee T_1 \cdot P_0}_{M} = \overline{\underbrace{\overline{T_1} \vee \overline{G_1} \cdot \overline{P_0}}_{\overline{N}}} = \overline{\left(A_1 \vee B_1\right) \vee \left(A_1 \cdot B_1\right) \cdot \overline{P_0}}. \quad (11.9)$$

Равноценность данных выражений можно доказать с помощью таблицы истинности (табл. 11.5) или аналитически:

$$P_1 = G_1 \vee T_1 \cdot P_0 = \underbrace{\left(A_1 \cdot B_1\right)}_{E} \vee \underbrace{\left(A_1 \vee B_1\right) \cdot P_0}_{F} =$$

$$= \overline{\overline{E} \cdot \overline{F}} = \overline{\left(\overline{A_1 \cdot B_1}\right) \cdot \left(\overline{A_1 \vee B_1}\right) \cdot P_0} = \overline{\left(\overline{A_1 \cdot B_1}\right) \cdot \left[\left(\overline{A_1 \vee B_1}\right) \vee \overline{P_0}\right]} =$$

$$= \overline{\left(\overline{A_1 \cdot B_1}\right) \cdot \overline{P_0} \vee \left(\overline{A_1 \vee B_1}\right) \cdot \left(\overline{A_1 \cdot B_1}\right)} = \overline{\left(\overline{A_1 \cdot B_1}\right) \cdot \overline{P_0} \vee \left[\overline{A_1 B_1} \vee \left(A_1 \vee B_1\right)\right]}.$$

Последнее выражение в квадратных скобках может быть преобразовано следующим образом:

$$A_1 B_1 \vee \left(A_1 \vee B_1\right) = A_1 B_1 \vee A_1 B_1 \vee A_1 \vee B_1 = A_1 \left(\overset{1}{\cancel{B_1 \vee 1}}\right) \vee B_1 \left(\overset{1}{\cancel{1 \vee A_1}}\right) = A_1 \vee B_1.$$

В результате имеем

$$P_1 = \overline{\left(\overline{A_1 \cdot B_1}\right) \cdot \overline{P_0} \vee \left(\overline{A_1 \vee B_1}\right)} = \overline{\overline{G_1} \cdot \overline{P_0} \vee \overline{T_1}}.$$

Таблица 11.4

A_i	B_i	S_i'	G_i	T_i	$\overline{G}_i$	S_i'
		$A_i \oplus B_i$	$A_i B_i$	$A_i \vee B_i$		$\overline{G}_i \cdot T_i$
0	0	0	0	0	1	0
0	1	1	0	1	1	1
1	0	1	0	1	1	1
1	1	0	1	1	0	0

Совпадение

Таблица 11.5

A_1	B_1	P_0	A_1B_1	$A_1 \vee B_1$	$T_1 \cdot P_0$	M	$\overline{T}_1$	$\overline{G}_1$	$\overline{G}_1 \cdot \overline{P}_0$	$\overline{T}_1 \vee \overline{G}_1 \cdot \overline{P}_0$	$\overline{N}$	$\overline{P}_0$
			G_1	T_1						N		
0	0	0	0	0	0	0	1	1	1	1	0	1
0	0	1	0	0	0	0	1	1	0	1	0	0
0	1	0	0	1	0	0	0	1	1	1	0	1
0	1	1	0	1	1	1	0	1	0	0	1	0
1	0	0	0	1	0	0	0	1	1	1	0	1
1	0	1	0	1	1	1	0	1	0	0	1	0
1	1	0	1	1	0	1	0	0	0	0	1	1
1	1	1	1	1	1	1	0	0	0	0	1	0

Совпадение

Таким образом, если сравнить исходное выражение и конечный результат преобразований, получим:

$$P_1 = G_1 \vee T_1P_0 = \overline{\overline{T}_1 \vee \overline{G}_1\,\overline{P}_0}.$$

Аналогично можно переписать остальные логические выражения:

$$P_2 = G_2 \vee T_2G_1 \vee T_2T_1P_0 = \overline{\overline{T}_2 \vee \overline{G}_2\,\overline{T}_1 \vee \overline{G}_2\,\overline{G}_1\,\overline{P}_0};$$

$$P_3 = G_3 \vee T_3G_2 \vee T_3T_2G_1 \vee T_3T_2T_1P_0 = \overline{\overline{T}_3 \vee \overline{G}_3\,\overline{T}_2 \vee \overline{G}_3\,\overline{G}_2\,\overline{T}_1 \vee \overline{G}_3\,\overline{G}_2\,\overline{G}_1\,\overline{P}_0};$$

$$P_4 = G_4 \vee T_4G_3 \vee T_4T_3G_2 \vee T_4T_3T_2G_1 \vee T_4T_3T_2T_1P_0 =$$

$$= \overline{\overline{T}_4 \vee \overline{G}_4\,\overline{T}_3 \vee \overline{G}_4\,\overline{G}_3\,\overline{T}_2 \vee \overline{G}_4\,\overline{G}_3\,\overline{G}_2\,\overline{T}_1 \vee \overline{G}_4\,\overline{G}_3\,\overline{G}_2\,\overline{G}_1\,\overline{P}_0}.$$

Взяв за основу упрощенную схему организации суммы (см. рис. 11.11), рассмотрим принцип формирования сумм S_1 и S_2 (рис. 11.13):

$$S_1 = S_1' \oplus P_0 = \left(\overline{G}_1 \cdot \overline{\overline{T}_1}\right) \oplus P_0; \quad S_2 = S_2' \oplus P_1 = \left(\overline{G}_2 \cdot \overline{\overline{T}_2}\right) \oplus P_1,$$

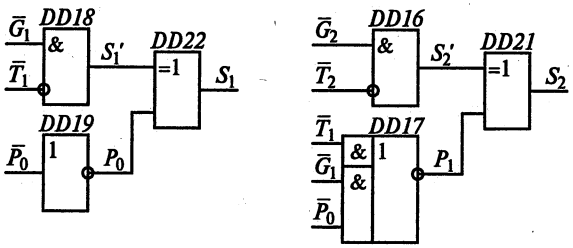
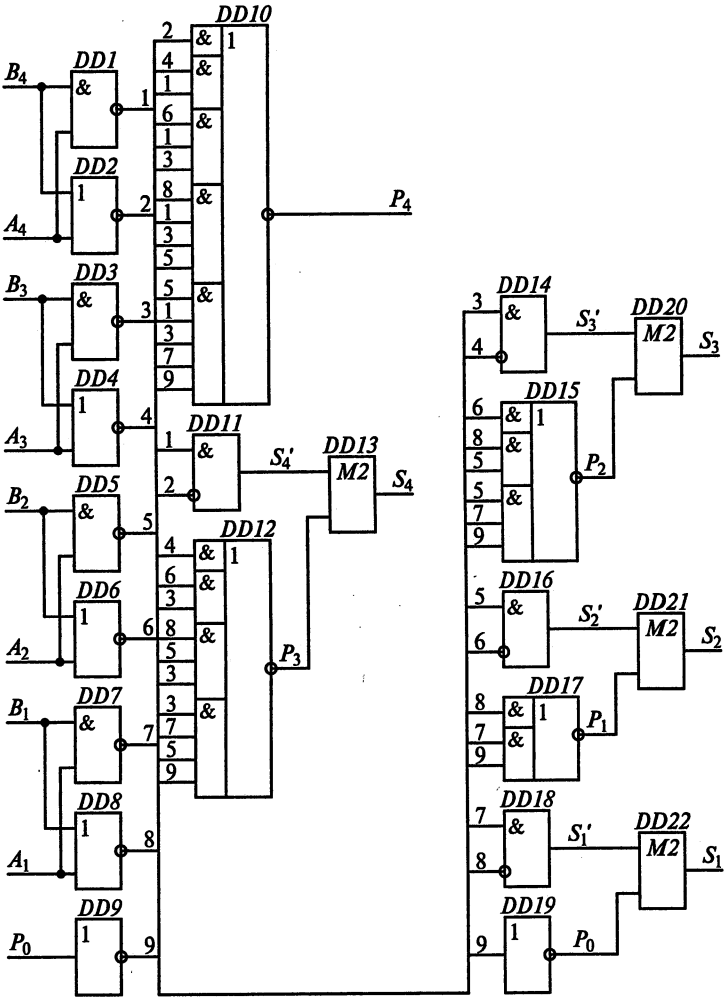


Рис. 11.13

Аналогично строится схема для S_3 (*DD14*, *DD15*, *DD20*) и S_4 (*DD11*, *DD12*, *DD13*), в результате чего получается четырехразрядный сумматор с ускоренным переносом ИМ6 (рис. 11.14).



Узел вспомогательной логики: $\overline{G_i}=\overline{A_i \cdot B_i}$ $\overline{T_i}=\overline{A_i \vee B_i}$	Узел организации переносов P_3 и P_4 и суммы S_4' без учета переноса	Получение суммы S_4 с учетом единицы переноса	Узел организации переносов P_0, P_1, P_2 и сумм S_1', S_2', S_3' без учета переноса	Получение сумм S_1, S_2, S_3 с учетом единицы переноса

Рис. 11.14

11.10. Сумматор последовательного типа

В комбинационных сумматорах последовательного действия поразрядное поступление слагаемых на вход сумматора разнесено по времени и подчинено специальным сигналам — синхроимпульсам (или, как их еще называют, тактовым импульсам). Поэтому время суммирования определяется количеством тактов, в каждом из которых будет происходить сложение одного из разрядов слагаемых. Безусловно, быстродействие таких сумматоров низкое. Они могут быть применены там, где есть ограничения по количеству входных и выходных цепей схемы.

Рассмотрим один из вариантов сумматора последовательного типа, представленный на рис. 11.15.

Схема одноразрядного сумматора последовательного типа содержит:

два электронных ключа *DD1* и *DD2*, через которые происходит передача слагаемых A_i и B_i только в момент действия синхроимпульса (СИ);

сумматор *DD3*;

элемент памяти *DD4*, представляющий собой двухтактный *D*-триггер;

выходной электронный ключ *DD5*.

Так как триггеры будут рассматриваться позже, то элемент памяти можно представить в виде двух последовательно соединенных ячеек, в первую из которых запись информации производится с приходом СИ, а перезапись во вторую ячейку — по окончании действия СИ. Этому моменту соответствует появление переноса P_{i-1} на выходе триггера и на входе сумматора *DD3* в качестве третьего слагаемого для дальнейшего поразрядного сложения.

Рассмотрим сложение двух трехразрядных чисел *A* и *B*:

$$\begin{array}{rl} & A = 101 \\ + & B = 001 \\ \hline & S = 110 \\ & P \quad 1. \end{array}$$

В исходном состоянии элемент памяти очищен и на выходе, а следовательно, и на входе P_{i-1} сумматора 0.

Первый такт. Поданные на вход схемы слагаемые $A_1 = 1$ и $B_1 = 1$ поступят на входы сумматора *DD3* только с приходом первого СИ, когда входные ключи *DD1* и *DD2* замкнутся. В результате получатся значение суммы $S_1 = 0$ и значение переноса из первого разряда во второй $P_1 = 1$. Значение переноса фиксируется в первой ячейке элемента памяти, так как с приходом СИ она становится доступна. Полученное значение суммы поступит на выходной электронный ключ *DD5* и окажется на выходе схемы только

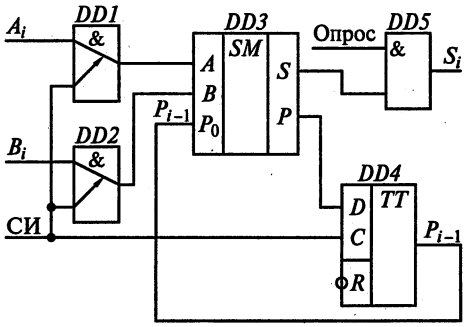


Рис. 11.15

в момент действия сигнала «Опрос». Время прихода этого сигнала выбирается с учетом максимальной задержки суммирования, а длительность должна быть меньше длительности слагаемых, т.е. длительности СИ.

По окончании действия СИ значение переноса $P_1 = 1$ переписывается во вторую ячейку памяти и оказывается на входе P_{i-1} сумматора. Слагаемые A_1 и B_1 на входе схемы к этому времени будут отсутствовать, так как действие СИ уже закончилось.

Второй такт. Перед приходом второго СИ на входе P_{i-1} сумматора находится единица. Поданные на вход схемы слагаемые $A_2 = 0$ и $B_2 = 0$ поступят на вход сумматора только с приходом второго СИ. В результате получатся значение суммы $S_2 = 1$ и значение переноса $P_2 = 0$, который зафиксируется в первой ячейке элемента памяти. В момент действия сигнала «Опрос» полученное значение суммы через замкнутый выходной ключ окажется на выходе схемы.

По окончании действия СИ значение $P_2 = 0$ поступает на вход P_{i-1} сумматора, а действие слагаемых A_2 и B_2 к тому времени заканчивается.

Третий такт. Аналогичным образом происходит формирование S_3 и P_3.

11.11. Накапливающий сумматор

Накапливающий сумматор (рис. 11.16) также относится к комбинационным сумматорам последовательного действия. В отличие от предыдущей схемы здесь с помощью многоразрядного элемента памяти сохраняется не перенос, а полученное поразрядное значение суммы, которое затем используется в следующем такте сложения в качестве второго слагаемого. Такие сумматоры называют *аккумуляторами.*

Рассматриваемая схема производит сложение четырехразрядных слагаемых A и B и состоит из входных ключей; многоразрядного

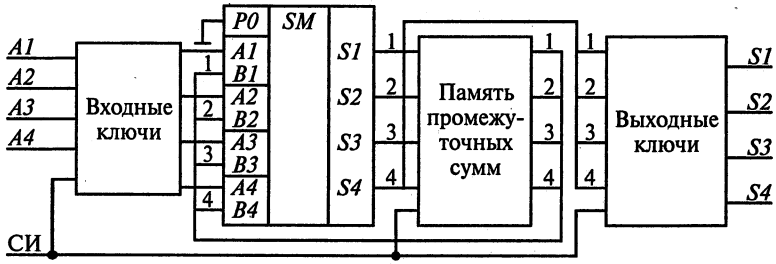


Рис. 11.16

сумматора; четырех двухтактных ячеек памяти, образующих две последовательные ступени, и выходных ключей.

В исходном состоянии многоразрядная память очищена и во всех разрядах содержит нули, которые поступают в качестве исходного слагаемого *B* на вход сумматора.

С приходом первого СИ поданное четырехразрядное число *A* поступает на входы сумматора, где и происходит его сложение с исходным слагаемым *B*, равным 0. Под действием СИ полученное значение суммы фиксируется в первой ступени памяти и проходит через выходные ключи на выход схемы.

По окончании действия СИ значение суммы переписывается во вторую ступень памяти и автоматически поступает на вход сумматора. Действие слагаемого *A* к этому времени заканчивается.

Аналогичным образом происходит формирование очередной суммы в следующих тактах работы схемы.

Схема накапливающего сумматора, где в качестве схемы памяти для хранения промежуточных сумм используется регистр, более подробно рассматривается в гл. 14.

11.12. Двоично-десятичный сумматор

При изучении сложения двоично-десятичных чисел отмечалась необходимость коррекции результата с момента, когда начинается различие в изображении двоичных и двоично-десятичных чисел, т.е. с числа 9.

Признаком необходимости введения коррекции (прибавление числа $6_{(10)}$ — $0110_{(2)}$) является:

одновременное наличие единиц в четвертом и втором разрядах;

одновременное наличие единиц в четвертом и третьем разрядах;

наличие межтетрадного переноса для чисел больше $15_{(10)}$.

Схема двоично-десятичного сумматора представлена на рис. 11.17.

Схема состоит из двух комбинационных сумматоров: сумматора исходных слагаемых *DD1*, где получается значение потетрадной суммы в двоичном коде, и корректирующего сумматора *DD5*, преобра-

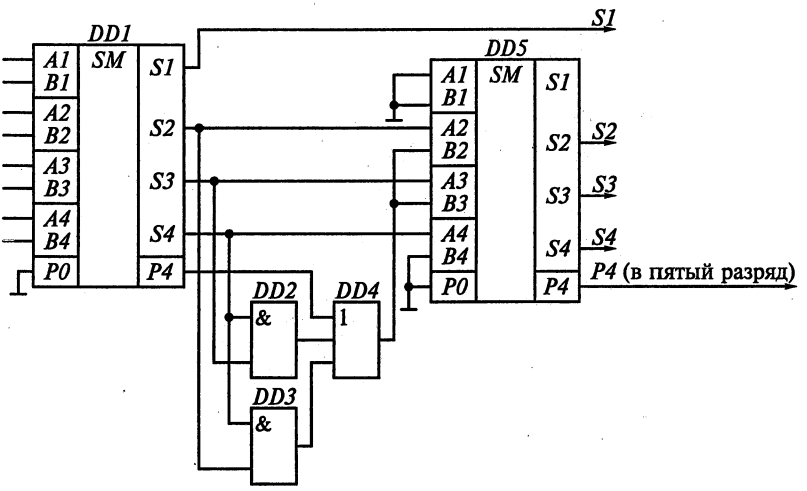


Рис. 11.17

зующего в случае необходимости результат в двоично-десятичный код, а также узла анализа результата суммирования и организации коррекции *DD2*, *DD3*, *DD4*. В случае необходимости коррекции на выходе одной из этих схем появляется единица, которая поступает на входы *B2* и *B3* сумматора *DD5*, образуя тем самым необходимый избыток $6_{(10)} - 0110_{(2)}$, который складывается там с первоначальным значением двоичного значения суммы. В результате на выходе получается значение суммы в двоично-десятичном коде.

Контрольные вопросы

1. В чем заключаются различия построения схем одноразрядных полусумматоров и сумматоров?
2. Назвать функцию алгебры логики, в соответствии с которой происходит формирование значения суммы в сумматорах.
3. Привести вид формулы образования суммы с учетом переноса.
4. Какова возможность реализации одноразрядного сумматора с использованием других типов логических элементов?
5. Определить возможность использования сумматора ИМ1 для сложения чисел $A = 10$ и $B = 11$.
6. Сколько микросхем необходимо при сложении четырехразрядных чисел с использованием сумматоров ИМ1, ИМ2 и ИМ3?
7. От чего в основном зависит быстродействие параллельного сумматора с последовательным переносом?
8. Какие требования предъявляются к временны́м параметрам стробирующего сигнала?
9. Каково быстродействие сумматоров для сложения восьмиразрядных чисел с использованием микросхем ИМ1, ИМ2, ИМ3?

10. Для чего анализируются возможные варианты образования переноса в старший разряд при организации сумматора с ускоренным распространением переноса?

11. Какими понятиями оперируют при выводе формул образования переносов в сумматоре с ускоренным распространением переноса?

12. В чем состоит принципиальное различие построения схем параллельного сумматора с последовательным переносом и сумматора с ускоренным распространением переноса?

13. Каковы различия в построении схем сумматоров последовательного и параллельного действия?

14. Для чего в схеме сумматора последовательного действия устанавливается элемент памяти?

15. В чем заключается специфика работы двухтактного элемента памяти?

16. В чем состоит сущность работы накапливающего сумматора?

17. Какую информацию хранит память накапливающего сумматора?

18. Почему сложение двоично-десятичных чисел осуществляется с использованием двух ступеней суммирования?

19. Совпадение каких сигналов выявляют схемы *DD2* и *DD3* в схеме двоично-десятичного сумматора?

Глава 12
ПРОГРАММИРУЕМЫЕ ЛОГИЧЕСКИЕ СТРУКТУРЫ

12.1. Общие сведения

В последнее время все более широкое распространение получают различные программируемые логические структуры, которые можно разделить на программируемые логические матрицы (ПЛМ), программируемые матрицы логики (ПМЛ) и базовые матричные кристаллы (БМК). Причем ПЛМ и ПМЛ являются наиболее простыми схемами с программируемой структурой. Дальнейшее развитие этого направления привело к разработке БМК, уровень интеграции в которых достиг миллионов вентилей на кристалле. Кроме того, в последние годы появился новый тип логических микросхем — перепрограммируемые логические интегральные схемы (ПЛИС). Эти микросхемы обеспечивают разработчику электронных устройств все преимущества использования стандартной БМК, добавляя при этом гибкость и значительное сокращение времени выхода нового устройства на рынок готовой продукции. Характерными особенностями ПЛИС являются: значительный объем ресурсов (до 10 млн системных вентилей на кристалл); высокая производительность (до 420 МГц); высокая гибкость архитектуры с множеством системных особенностей (внутреннее распределенное и блочное ОЗУ, логика ускоренного переноса, внутренние буферы с третьим состоянием, встроенные блоки умножителей, наличие порядка ста ты-

сяч триггеров и сдвиговых регистров); низкое энергопотребление; возможность использования развитых и недорогих средств проектирования и др.

12.2. Организация программируемой логической матрицы

Программируемая логическая матрица характеризуется простотой получения необходимых функций. Ее основу составляет программируемая матрица, состоящая из элементов, которые соединены между собой вертикальными и горизонтальными шинами. Функции таких схем задаются с помощью перемычек. Однократное программирование ПЛМ в соответствии с необходимой функцией осуществляется либо изготовителем ИС с использованием одного фотошаблона, либо пользователем — электрическим программированием на специальной установке, с помощью которой пережигаются плавкие перемычки. Кроме того, существуют многократно программируемые структуры с ультрафиолетовым стиранием на МОП-транзисторах. Более подробно приемы программирования будут рассмотрены в гл. 19.

Программируемые логические матрицы незаменимы при реализации различных функций алгебры логики. Возможности ПЛМ определяются максимальным количеством минтермов и количеством реализуемых функций.

Для реализации функций типа $F = \overline{A\bar{B}C \vee \bar{A}BC \vee \ldots \vee \bar{A}\bar{B}\bar{C}}$ необходимо: иметь как прямые, так и инверсные значения переменных; выполнить умножение переменных, получив необходимые минтермы; выполнить логическое сложение и при необходимости инверсию полученной функции. Для решения этих задач в ПЛМ предусмотрены отдельные функциональные узлы (рис. 12.1):

узел формирования прямых и инверсных значений переменных;

узел из элементов И (конъюнкторов) для организации различных минтермов;

узел из схем ИЛИ (дизъюнкторов) по одной на каждый выход ПЛМ;

узел выходных инверторов.

Рассмотрим однократно программируемую БИС ПЛМ серии К556РТ1, выполненную по ТТЛШ-технологии. Такая микросхема реализует 8 функций от 16 входных переменных. При этом, если логические функции представить в СДНФ, общее число конъюнкций для всех функций не должно превышать 48. Эта микросхема содержит:

узел формирования прямых и инверсных значений переменных;

узел из 48 конъюнкторов для организации 48 минтермов;

узел из 8 дизъюнкторов для реализации 8 функций;

узел выходных инверторов.

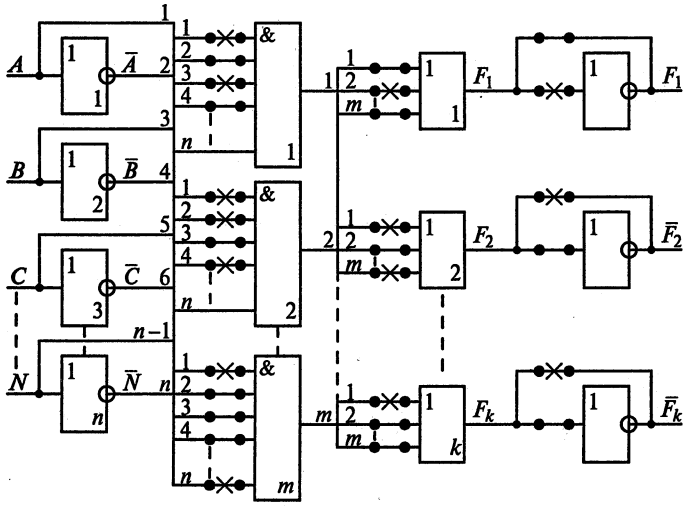


Рис. 12.1

Конъюнкторы (рис. 12.2, *а*) представляют собой 48 диодно-резистивных элементов И, в каждом из которых с помощью пережигаемых перемычек создается определенная комбинация переменных.

Дизъюнкторы (рис. 12.2, *б*) представляют собой эмиттерные повторители, в эмиттерных цепях которых находятся плавкие перемычки. Если минтерм в данную функцию не входит, то перемычка у соответствующего этой функции транзистора пережигается.

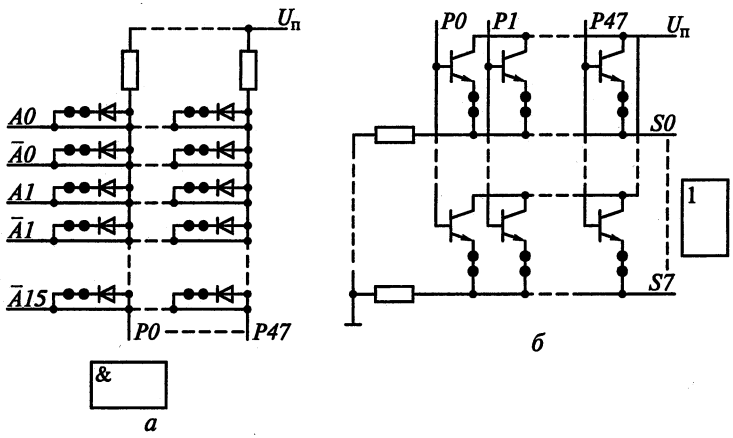


Рис. 12.2

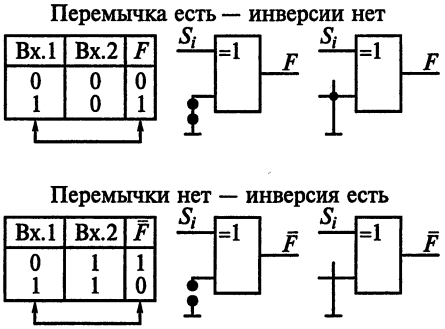


Рис. 12.3

Узел выходных инверторов (рис. 12.3) представляет собой совокупность схем, реализующих функцию «Исключающее ИЛИ» («Сложение по модулю 2»).

Следует иметь в виду, что отсутствие перемычки на входе логического элемента эквивалентно поступлению на его вход лог. 1.

В общем виде схема ПЛМ и ее условное графическое обозначение представлены на рис. 12.4 и 12.5.

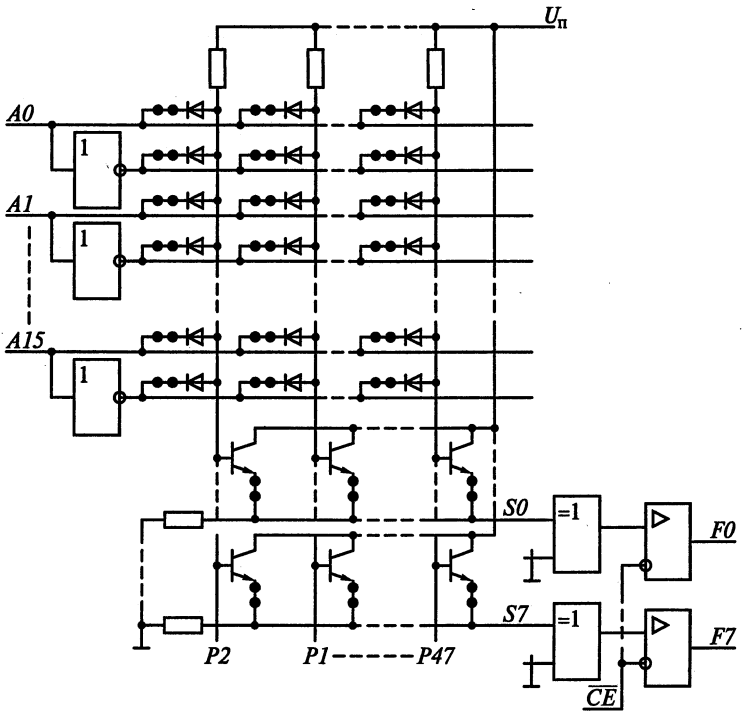


Рис. 12.4

На практике используется упрощенное представление схемы ПЛМ (рис. 12.6), где вертикальные и горизонтальные линии образуют места пересечений, с помощью которых задаются определенные комбинации переменных, входящих в минтерм (матрица *A1*). Таким образом, вертикальных линий должно быть столько, сколько логических функций И должна реализовать схема. Наличие перемычки в местах пересечения вертикальных и горизонтальных линий отмечается точкой. В исходном состоянии точки присутствуют во всех узлах, образуя всегда произведение, равное 0.

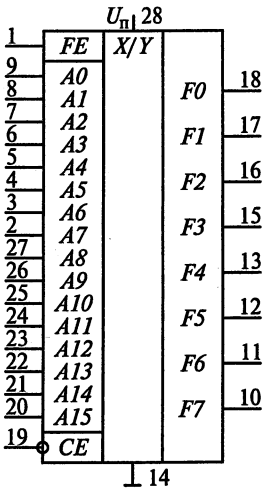


Рис. 12.5

Например, $\left(A \cdot \overline{A}\right) \vee \left(B \cdot \overline{B}\right) \vee \left(C \cdot \overline{C}\right) = 0$.

Матрица *A2* представляет собой неполный шифратор на четыре выхода. Каждая горизонтальная линия в *A2* соответствует одному из входов элемента ИЛИ, точки на пересечении линий — входным сигналам, поступающим с вертикальных линий.

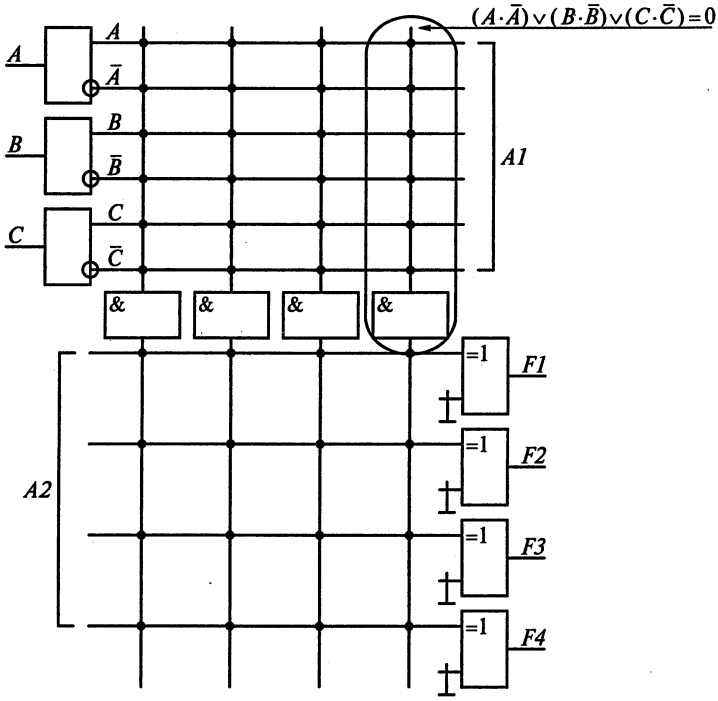


Рис. 12.6

Рассмотрим несколько примеров организации комбинационных схем на основе ПЛМ. На рис. 12.7 выполнена реализация отдельной функции:

$$F = \overline{C} \cdot \left(\overline{A} \vee B\right);$$

$$F = \overline{C} \cdot \left(\overline{A} \vee B\right) = C \vee \overline{\left(\overline{A} \vee B\right)} = \overline{C \vee A \cdot \overline{B}}.$$

На рис. 12.8 приведен пример построения дешифратора.

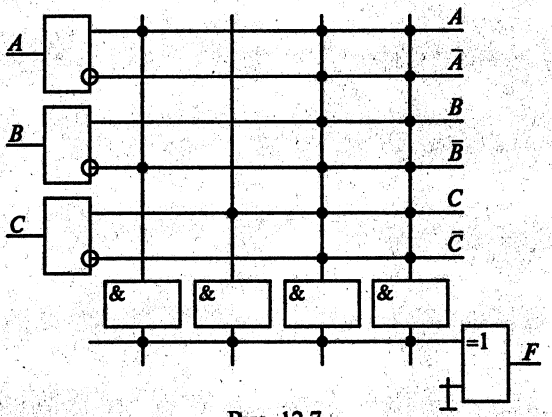
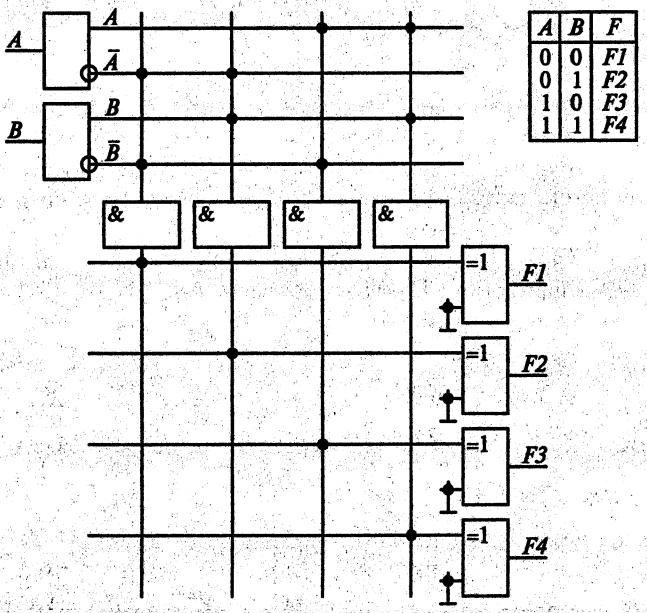


Рис. 12.7



A	B	F
0	0	F1
0	1	F2
1	0	F3
1	1	F4

Рис. 12.8

Контрольные вопросы

1. В чем состоит универсальность ПЛМ?
2. Где может быть использована ПЛМ?
3. Почему логический элемент «Исключающее ИЛИ» можно использовать в качестве инвертора?
4. Каково назначение сигнала $\overline{CE}$, который воздействует на выходной ключ — усилитель ПЛМ микросхемы К556РТ1?

Контрольные задания

Реализовать на ПЛМ следующие функции:

1) $F1 = ABC \vee \overline{A}\overline{B}C \vee A\overline{B}\overline{C}$;

2) $F1 = ABC \vee \overline{A}\overline{B} \vee \overline{C}$, $F2 = AC \vee \overline{B}$, $F3 = A\overline{B}\overline{C} \vee BC \vee \overline{A}$;

3) $F1 = \overline{A\overline{B}\overline{C} \vee \overline{A}B \vee C}$;

4) $F1 = \overline{(A \vee \overline{B}) \cdot (\overline{A} \vee \overline{C})}$;

5) $F1 = (A \vee \overline{B}) \cdot (\overline{A} \vee B)$.

12.3. Программируемые матрицы логики

По сравнению с ПЛМ в программируемых матрицах логики (ПМЛ) программируются только минтермы, т. е. произведения переменных для СДНФ. Элементы ИЛИ, выполняющие функции дизъюнкции, зафиксированы и имеют, как правило, семь-восемь входов. В ПМЛ предусмотрена возможность введения обратных связей с выходов на входы внутри корпуса микросхемы. Это значительно упрощает построение схем в тех случаях, когда количество входов микросхемы ограничено, а программируемые минтермы повторяются в формируемых функциях, при этом существенно упрощается решение задачи программирования и практически не снижаются логические возможности микросхемы. Рассмотрим пример организации ПЛМ при создании на ее основе схемы шифратора. Построим шифратор на ПЛМ в соответствии с табл. 12.1.

Для четырех функций *Y1*, *Y2*, *Y3*, *Y4* необходимо реализовать СДНФ типа:

$$\overline{Y4} = \overline{X0}\,X1\,\overline{X2}\,\overline{X3}\,\overline{X4}\,\overline{X5}\,\overline{X6}\,\overline{X7}\,\overline{X8}\,\overline{X9} \vee \overline{X0}\,\overline{X1}\,\overline{X2}\,X3\,\overline{X4}\,\overline{X5}\,\overline{X6}\,\overline{X7}\,\overline{X8}\,\overline{X9} \vee$$

$$\vee \overline{X0}\,\overline{X1}\,\overline{X2}\,\overline{X3}\,\overline{X4}\,X5\,\overline{X6}\,\overline{X7}\,\overline{X8}\,\overline{X9} \vee \overline{X0}\,\overline{X1}\,\overline{X2}\,\overline{X3}\,\overline{X4}\,\overline{X5}\,\overline{X6}\,X7\,\overline{X8}\,\overline{X9} \vee$$

$$\vee \overline{X0}\,\overline{X1}\,\overline{X2}\,\overline{X3}\,\overline{X4}\,\overline{X5}\,\overline{X6}\,\overline{X7}\,\overline{X8}\,\overline{X9}.$$

Аналогично описываются остальные функции.

Из анализа таблицы видно, что некоторые минтермы входят в несколько функций. Так, минтерм $\overline{X0}\,\overline{X1}\,\overline{X2}\,X3\,\overline{X4}\,\overline{X5}\,\overline{X6}\,\overline{X7}\,\overline{X8}\,\overline{X9}$ входит в функции *Y3* и *Y4*. При реализации шифратора на ПЛМ

Таблица 12.1

Входы										Выходы			
X0	*X1*	*X2*	*X3*	*X4*	*X5*	*X6*	*X7*	*X8*	*X9*	*Y1*	*Y2*	*Y3*	*Y4*
1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	0	0	0	1	0	0
0	0	0	0	0	1	0	0	0	0	0	1	0	1
0	0	0	0	0	0	1	0	0	0	0	1	1	0
0	0	0	0	0	0	0	1	0	0	0	1	1	1
0	0	0	0	0	0	0	0	1	0	1	0	0	0
0	0	0	0	0	0	0	0	0	1	1	0	0	1

каждая функция формируется индивидуально от других и соответственно программирование минтермов будет повторяться. ПЛМ, реализующая шифратор, показана на рис. 12.9.

Из схемы ПЛМ и табл. 12.1 видно, что функция *Y4* состоит из минтермов (назовем их повторяющимися), которые входят и в функции *Y1*, *Y2*, *Y3*. Кроме того, в эти функции входят минтермы, которых в *Y4* нет (назовем их независимыми).

Теперь построим шифратор на ПМЛ (рис. 12.10). При построении схемы введем ограничение на число минтермов (не более трех), связанных с каждым из выходных элементов ИЛИ (не более десяти). В формировании *Y3* участвуют четыре минтерма, в которых *X2*, *X3*, *X6*, *X7* равны 1. Реализация функции *Y3* из четырех минтермов функционально невыполнима, так как по условию в промежуточных и выходных функциях могут участвовать только три минтерма, состоящие из 10 переменных. Поэтому в первую очередь сформируем группу минтермов, которые повторяются в функции *Y4*, обозначив полученную промежуточную функцию *Z3*. Эти минтермы содержат $X3 = 1$ и $X7 = 1$. Тогда выходная функция *Y3* примет следующий вид:

$$Z3 = \bar{X}0\,\bar{X}1\,\bar{X}2\,X3\,\bar{X}4\,\bar{X}5\,\bar{X}6\,\bar{X}7\,\bar{X}8\,\bar{X}9 \vee \bar{X}0\,\bar{X}1\,\bar{X}2\,\bar{X}3\,\bar{X}4\,\bar{X}5\,\bar{X}6\,X7\,\bar{X}8\,\bar{X}9;$$

$$Y3 = \bar{X}0\,\bar{X}1\,X2\,\bar{X}3\,\bar{X}4\,\bar{X}5\,\bar{X}6\,\bar{X}7\,\bar{X}8\,\bar{X}9 \vee \bar{X}0\,\bar{X}1\,\bar{X}2\,X3\,\bar{X}4\,\bar{X}5\,\bar{X}6\,\bar{X}7\,\bar{X}8\,\bar{X}9 \vee$$

$$\vee\, \bar{X}0\,\bar{X}1\,\bar{X}2\,\bar{X}3\,\bar{X}4\,\bar{X}5\,X6\,\bar{X}7\,\bar{X}8\,\bar{X}9 \vee \bar{X}0\,\bar{X}1\,\bar{X}2\,\bar{X}3\,\bar{X}4\,\bar{X}5\,\bar{X}6\,X7\,\bar{X}8\,\bar{X}9 =$$

$$= \bar{X}0\,\bar{X}1\,X2\,\bar{X}3\,\bar{X}4\,\bar{X}5\,\bar{X}6\,\bar{X}7\,\bar{X}8\,\bar{X}9 \vee \bar{X}0\,\bar{X}1\,\bar{X}2\,\bar{X}3\,\bar{X}4\,\bar{X}5\,X6\,\bar{X}7\,\bar{X}8\,\bar{X}9 \vee Z3.$$

Аналогично в формировании *Y2* участвуют четыре минтерма, в которых *X4*, *X5*, *X6*, *X7* равны 1. Реализация функции *Y2* из четырех минтермов также функционально невыполнима. Поэто-

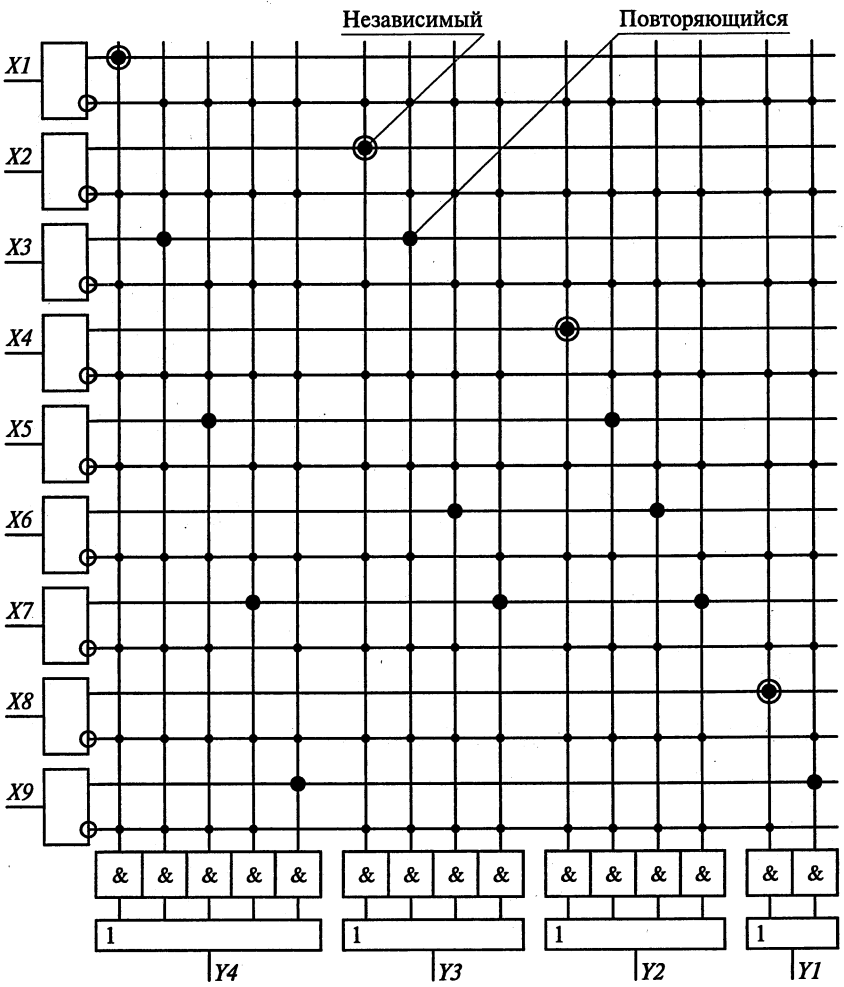


Рис. 12.9

му в первую очередь сформируем группу минтермов, которые повторяются в функции *Y4*, обозначив их *Z2*. Эти минтермы содержат $X5 = 1$ и $X7 = 1$. Тогда выходная функция *Y2* примет следующий вид:

$$Z2 = \bar{X}0\,\bar{X}1\,\bar{X}2\,\bar{X}3\,\bar{X}4\,X5\,\bar{X}6\,\bar{X}7\,\bar{X}8\,\bar{X}9 \vee \bar{X}0\,\bar{X}1\,\bar{X}2\,\bar{X}3\,\bar{X}4\,\bar{X}5\,\bar{X}6\,X7\,\bar{X}8\,\bar{X}9;$$

$$Y2 = \bar{X}0\,\bar{X}1\,\bar{X}2\,\bar{X}3\,X4\,\bar{X}5\,\bar{X}6\,\bar{X}7\,\bar{X}8\,\bar{X}9 \vee \bar{X}0\,\bar{X}1\,\bar{X}2\,\bar{X}3\,\bar{X}4\,X5\,\bar{X}6\,\bar{X}7\,\bar{X}8\,\bar{X}9 \vee$$

$$\vee\, \bar{X}0\,\bar{X}1\,\bar{X}2\,\bar{X}3\,\bar{X}4\,\bar{X}5\,X6\,\bar{X}7\,\bar{X}8\,\bar{X}9 \vee \bar{X}0\,\bar{X}1\,\bar{X}2\,\bar{X}3\,\bar{X}4\,\bar{X}5\,\bar{X}6\,X7\,\bar{X}8\,\bar{X}9 =$$

$$= \bar{X}0\,\bar{X}1\,\bar{X}2\,\bar{X}3\,X4\,\bar{X}5\,\bar{X}6\,\bar{X}7\,\bar{X}8\,\bar{X}9 \vee \bar{X}0\,\bar{X}1\,\bar{X}2\,\bar{X}3\,\bar{X}4\,\bar{X}5\,X6\,\bar{X}7\,\bar{X}8\,\bar{X}9 \vee Z2.$$

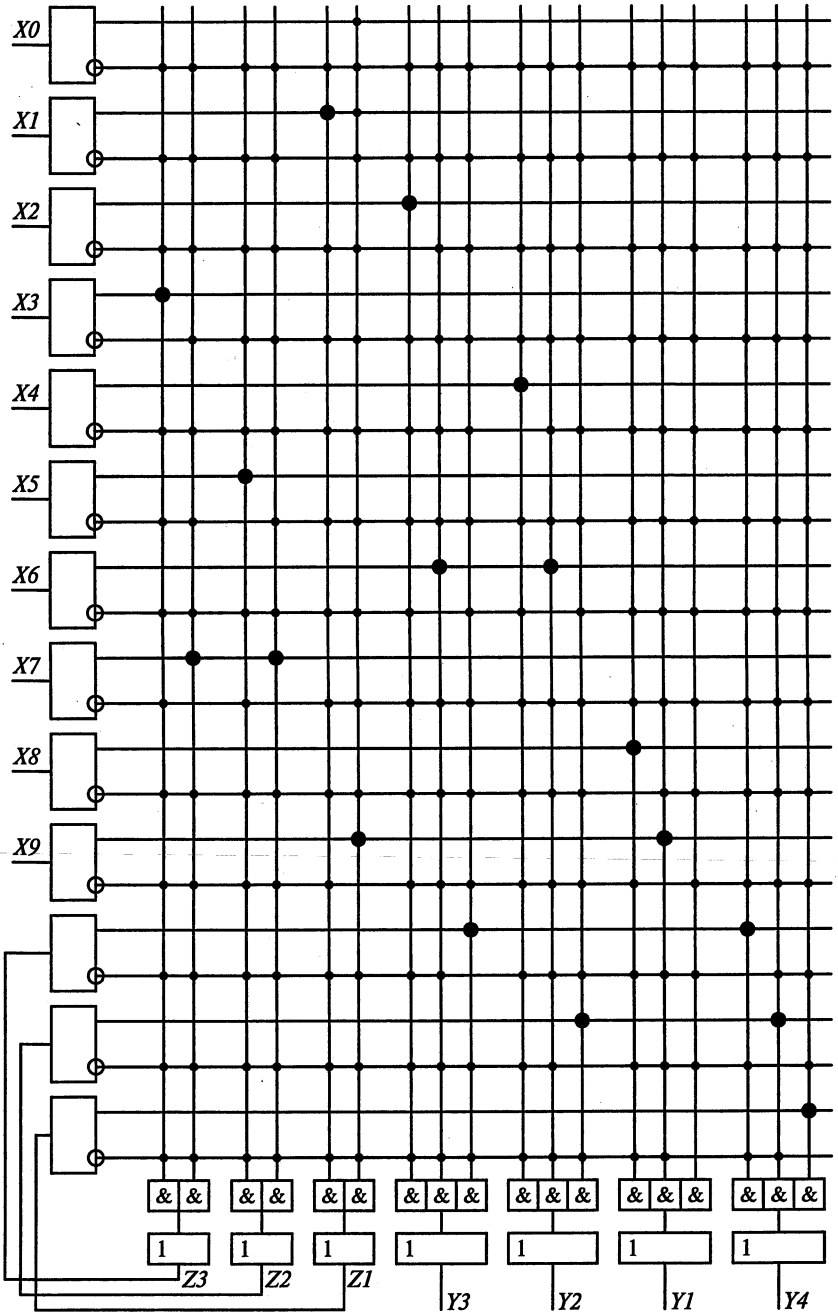


Рис. 12.10

Реализуем рассмотренные функции на ПМЛ. В первую очередь сформируем промежуточные функции *Z3*, *Z2*, *Z1*. Во вторую очередь организуем независимые минтермы, содержащие *X2* = 1, *X6* =1 и *X3* = 1, *X6* = 1, а затем добавим к ним *Z3* и *Z2* соответственно.

В формировании *Y4* участвуют пять минтермов, в которых *X1*, *X3*, *X5*, *X7*, *X9* равны 1. Реализация функции *Y4* из пяти минтермов также функционально невыполнима. Промежуточные функции *Z3* и *Z2* содержат три из пяти минтермов, причем *X7* входит как в *Z3*, так и в *Z2*. Объединение двух минтермов, в которых *X1* и *X9* равны 1, происходит через промежуточную функцию *Z1*. Тогда выходная функция *Y4* примет следующий вид:

$$Z1 = \bar{X}0\,X1\,\bar{X}2\,\bar{X}3\,\bar{X}4\,\bar{X}5\,\bar{X}6\,\bar{X}7\,\bar{X}8\,\bar{X}9 \vee \bar{X}0\,\bar{X}1\,\bar{X}2\,\bar{X}3\,\bar{X}4\,\bar{X}5\,\bar{X}6\,\bar{X}7\,\bar{X}8\,X9;$$

$$Y4 = \bar{X}0\,X1\,\bar{X}2\,\bar{X}3\,\bar{X}4\,\bar{X}5\,\bar{X}6\,\bar{X}7\,\bar{X}8\,\bar{X}9 \vee \bar{X}0\,\bar{X}1\,\bar{X}2\,X3\,\bar{X}4\,\bar{X}5\,\bar{X}6\,\bar{X}7\,\bar{X}8\,\bar{X}9 \vee$$

$$\vee \bar{X}0\,\bar{X}1\,\bar{X}2\,\bar{X}3\,\bar{X}4\,X5\,\bar{X}6\,\bar{X}7\,\bar{X}8\,\bar{X}9 \vee \bar{X}0\,\bar{X}1\,\bar{X}2\,\bar{X}3\,\bar{X}4\,\bar{X}5\,\bar{X}6\,X7\,\bar{X}8\,\bar{X}9 \vee$$

$$\vee \bar{X}0\,\bar{X}1\,\bar{X}2\,\bar{X}3\,\bar{X}4\,\bar{X}5\,\bar{X}6\,\bar{X}7\,\bar{X}8\,X9 = Z3 \vee Z2 \vee Z1.$$

Так как в формировании *Y1* участвуют только два минтерма, то их программирование организовано так же, как в ПЛМ:

$$Y1 = \bar{X}0\,\bar{X}1\,\bar{X}2\,\bar{X}3\,\bar{X}4\,\bar{X}5\,\bar{X}6\,\bar{X}7\,X8\,\bar{X}9 \vee \bar{X}0\,\bar{X}1\,\bar{X}2\,\bar{X}3\,\bar{X}4\,\bar{X}5\,\bar{X}6\,\bar{X}7\,\bar{X}8\,X9.$$

Контрольные вопросы

1. В каких случаях целесообразно использовать ПМЛ?
2. В каких случаях целесообразно формировать промежуточные функции?

Раздел IV
ПОСЛЕДОВАТЕЛЬНОСТНЫЕ УСТРОЙСТВА

Глава 13
ТРИГГЕРЫ

13.1. Общие сведения

Основным элементом памяти, используемым для хранения единицы информации, является триггер. Как правило, схема триггера помимо элемента памяти (ЭП), хранящего бит информации (рис. 13.1, *а*), имеет дополнительные схемы управления — ключи, с помощью которых создаются определенные условия для записи и хранения информации (рис. 13.1, *б* и *в*) в ЭП.

Различные виды триггеров имеют различные входы, но всегда два выхода: прямой выход Q и инверсный выход $\overline{Q}$. Триггер может иметь два устойчивых состояния, которые оцениваются значением сигнала на его прямом выходе Q. Одно из них принято за лог. 1, другое — за лог. 0.

Входы триггера подразделяются на информационные — R, S, J, K, D и управляющие T, V, C. Как правило, тип триггера определяется по виду основных информационных входов. В табл. 13.1 представлены некоторые виды триггеров и дана информация о функциональном назначении их входов. Классификация триггеров приведена на рис. 13.2.

По способу записи триггеры подразделяются на синхронные и асинхронные. В асинхронных триггерах элемент памяти доступен

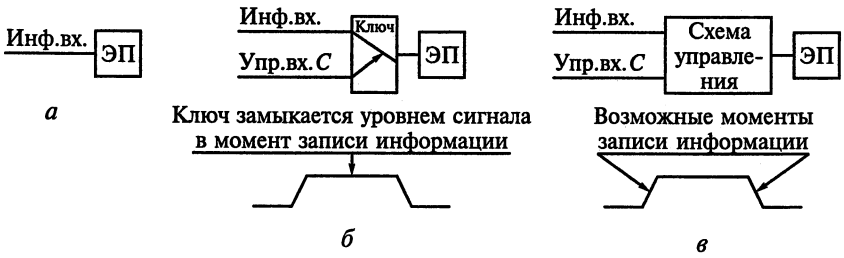
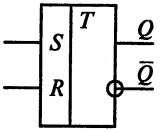
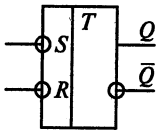
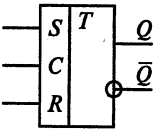
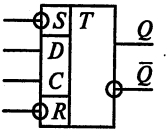
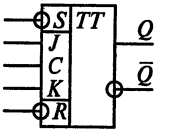
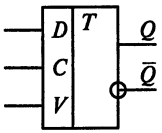
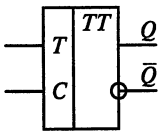


Рис. 13.1

Таблица 13.1

Тип триггера	УГО	Обозначение входов	Функциональное назначение входов
Асинхронный *RS*-триггер с прямыми входами (активным сигналом является лог. 1)		*R* (reset) — сброс	Наличие активного сигнала на входе *R* (при наличии инверсного сигнала на входе *S*) приводит к установке триггера в нуль ($Q = 0$)
Синхронный *RS*-триггер с инверсными входами (активным сигналом является лог. 0)		*S* (set) — установка	Наличие активного сигнала на входе *S* (при наличии инверсного сигнала на входе *R*) приводит к установке триггера в единицу ($Q = 1$)
Синхронный *RS*-триггер		*C* (clock) — часы, синхронизация	Триггер доступен для записи информации только в момент действия синхроимпульса
D-триггер		*D* (delay) — задержка	Информация, поданная на вход *D*, пройдет в элемент памяти через входной ключ с задержкой, определяемой временем прихода управляющего сигнала на вход *C*. $\bar{S}$, $\bar{R}$ — асинхронные входы предварительной установки
JK-триггер		*J* (jerk) — внезапное включение *K* (kill) — внезапное отключение	Информация, поданная на вход *J* (при наличии инверсного сигнала на входе *K*), будет записана в элемент памяти с приходом управляющего сигнала на вход *C*

Окончание табл. 13.1

Тип триггера	УГО	Обозначение входов	Функциональное назначение входов
DV-триггер		*V* (valve) — клапан	Обращение к триггеру возможно только при наличии разрешающего уровня сигнала на управляющем входе *V*
T-триггер		*T* (toggle) — переключатель	Управляющий вход *T* используется для организации счетного режима, когда с приходом синхроимпульса на вход *C* триггер меняет свое состояние на противоположное, реализуя функцию «Сложение по модулю 2»

для записи информации всегда (см. рис. 13.1, *а*). В синхронных триггерах доступ в ЭП возможен только при наличии управляющего сигнала на входе *C*, который воздействует на ключ доступа или схему управления ЭП (см. рис. 13.1, *б*, *в*).

Реакция триггера на управляющий сигнал может быть различна. При статическом управлении запись информации осуществляется с появлением определенного уровня сигнала на входе *C* (см. рис. 13.1, *б*). При динамическом управлении запись информации происходит в момент изменения состояния управляющего сигнала. В отличие от триггеров, управляемых уровнями сигналов, запись информации

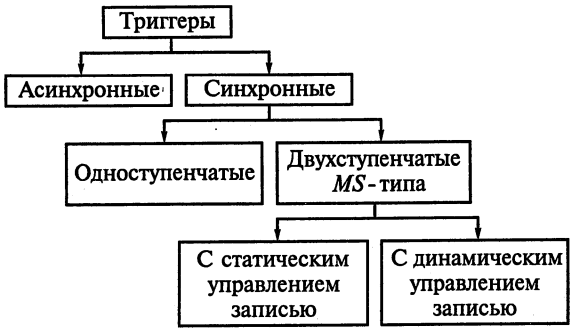


Рис. 13.2

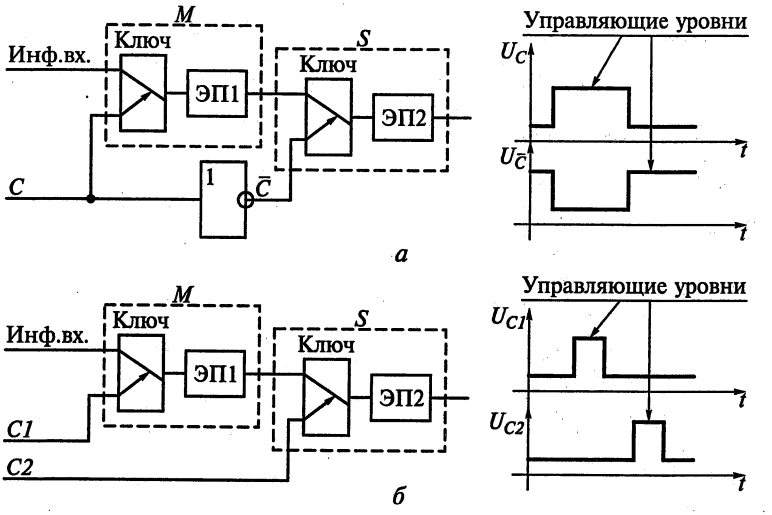


Рис. 13.3

в них никогда не произойдет при постоянном уровне сигнала на входе *C*. Запись становится возможна лишь на коротких участках сигналов вблизи активного фронта или активного среза.

Одноступенчатые триггеры, как правило, представляют собой сочетание ключа доступа к ЭП и самого элемента памяти.

Двухступенчатые триггеры (рис. 13.3) состоят из двух одноступенчатых триггеров. Такая схема называется структурой *MS*-типа. Признаком таких триггеров в УГО являются две буквы *TT*. Первая ступень (*master* — хозяин) — ведущая, она используется для приема информации. Вторая ступень (*slave* — исполнитель) — ведомая, используется для записи и хранения. Управлять той и другой ступенью можно с помощью одного или двух управляющих синхроимпульсов. В первом случае схемы относятся к однотактным триггерам (см. рис. 13.3, *а*), во втором — к двухтактным (см. рис. 13.3, *б*).

Контрольные вопросы

1. Для чего предназначены триггеры?
2. Как можно разделить входы триггеров по их функциональному назначению?
3. Какие существуют устойчивые состояния триггера и в какой точке схемы происходит оценка его состояния?
4. В чем заключаются различия в работе схем триггеров, имеющих только информационный вход, и триггеров, имеющих информационный и управляющий входы?
5. В какие моменты относительно синхроимпульса возможна запись информации в триггер?
6. В чем заключаются различия в схемах *RS*-триггеров с прямыми и инверсными входами?

7. Для какой цели используются входы R и S?

8. При наличии на каком входе активного сигнала триггер установится в нулевое состояние? На каком выводе триггера можно проконтролировать это состояние?

9. В чем заключается сущность режима записи в RS-триггер с прямыми входами лог. 1 и лог. 0?

10. В чем заключается сущность режима записи в RS-триггер с инверсными входами лог. 1 и лог. 0?

11. Провести анализ состояния RS-триггера с прямыми входами, когда на входах R и S находится сочетание сигналов 0 и 1 соответственно.

12. Провести анализ состояния RS-триггера с инверсными входами, когда на входах R и S находится сочетание сигналов 0 и 1 соответственно.

13. В какие моменты времени происходит запись информации в асинхронный и синхронный триггеры?

14. Чем различается работа триггеров с статическим и динамическим управлением?

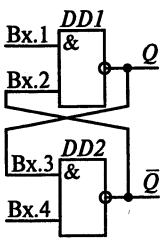
15. Чем различается работа одноступенчатых и двухступенчатых схем триггеров?

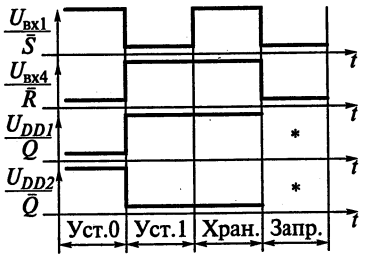
13.2. Асинхронные RS-триггеры

***RS*-триггер с инверсными входами.** Данный тип триггера относится к простейшим элементам памяти.

На рис. 13.4 представлены схема триггера и диаграмма его работы. В таблице состояний (табл. 13.2) отражены состояния триггера при разных сочетаниях входных сигналов. Условное графическое обозначение см. в табл. 13.1.

Триггер построен на базе логического элемента И-НЕ, для которого активным сигналом, определяющим его состояние на выходе, является лог. 0. Схему образуют два логических элемента И-НЕ, охваченных обратными связями, за счет чего обеспечивается сохранение одного из двух устойчивых состояний триггера.





И-НЕ
$\overline{A \wedge B} = \overline{A \cdot B}$

A	1	0	1	0
B	1	1	0	0
F	0	1	1	1

Рис. 13.4

Таблица 13.2

Вых. Q_t	Вх.4 R_t	Вх.1 S_t	Вых. Q_{t+1}	Состояние
0	1	0	1	Установка 1
1	1	1	1	Хранение 1
1	1	0	1	Подтверждение 1
1	0	1	0	Установка 0
0	1	1	0	Хранение 0
0	0	1	0	Подтверждение 0
0	0	0	*	Запрещенное
1	0	0	*	

Нарушив принцип синтеза схемы, рассмотрим сначала действие триггера, определив местоположение входов S и R, а затем опишем его работу.

Не обозначая входы триггера, подадим произвольную комбинацию сигналов на входы 1 и 4 (лог. 1 и 0 соответственно).

Для элемента *DD2* достаточно на одном из входов иметь активный сигнал — лог. 0, чтобы сказать, что на его выходе будет лог. 1.

При этом на входах элемента *DD1* будет совпадение единиц, что приведет к появлению на его выходе (выходе Q триггера) лог. 0. О состоянии триггера судят по выходу Q — там будет находиться лог. 0.

Установка триггера в нулевое состояние произошла под воздействием активного сигнала на входе 4. В табл. 13.1 показано функциональное назначение входов триггера и даны их определения в зависимости от наличия активного сигнала. Вход 4, на который был подан активный сигнал (в данном случае лог. 0), устанавливающий триггер в нулевое состояние, обозначается буквой $\bar{R}$, а второй вход — буквой $\bar{S}$ (рис. 13.5).

Запись в триггер лог. 1 осуществляется аналогично. При этом на входы 1 и 4 подаются логические 0 и 1 соответственно.

В режиме хранения на информационные входы $\bar{S}$ и $\bar{R}$ подаются логические единицы, по наличию которых нельзя судить о состояниях на выходах элементов *DD1* и *DD2*. В этом случае активный сигнал как результат записи в предыдущем цикле обращения, присутствует на одном из выходов триггера и через обратную связь поступает на вход 2 или 3, подтверждая предыдущее состояние схемы.

Так как активным сигналом является лог. 0, то входы $\bar{S}$ и $\bar{R}$ имеют инверсное обозначение.

Одновременно подавать два нулевых (активных) сигнала на входы $\bar{S}$ и $\bar{R}$ запрещено, так как это приводит к неопределенному состоянию логических элементов, охваченных обратной связью, и не обеспечивает устойчивого состояния триггера.

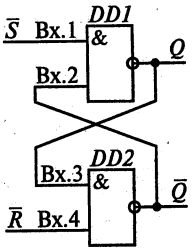


Рис. 13.5

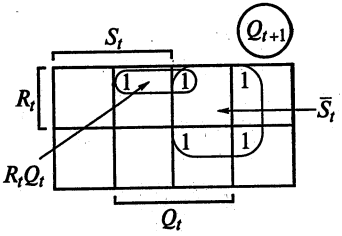


Рис. 13.6

Новое состояние триггера Q_{t+1} зависит от предыдущего состояния Q_t и входных сигналов S_t и R_t. Эта зависимость представлена в таблице состояний (см. табл. 13.2):

$$Q_{t+1} = f(Q_t,\ S_t,\ R_t).$$

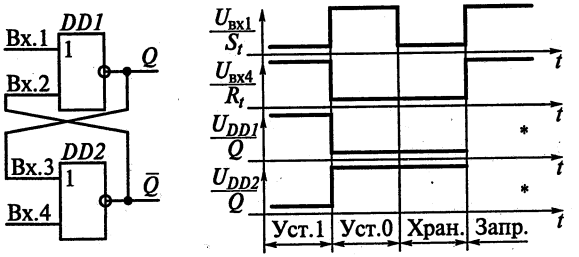
Работу триггера можно описать характеристическим уравнением. Для этого на основе таблицы состояний необходимо построить карту Карно, приравняв клетки, соответствующие неопределенным состояниям, к лог. 1. Если на входах R_t, S_t находятся единицы, то триггер хранит предыдущее состояние; при сочетании сигналов 10 — устанавливается в единичное состояние; при сочетании сигналов 01 — в нулевое (рис. 13.6).

На основании карты Карно можно записать следующее выражение:

$$Q_{t+1} = R_tQ_t \vee \overline{S}_t.$$

***RS*-триггер с прямыми входами.** Триггер такого типа строится на базе логического элемента ИЛИ-НЕ, для которого активным сигналом, определяющим его состояние на выходе, является лог. 1.

На рис. 13.7 представлены схема триггера и диаграмма его работы при различном сочетании входных сигналов. В таблице состояний (табл. 13.3) отражено состояние триггера при разном сочетании входных сигналов.



ИЛИ = НЕ
$A \vee B$

A	0	1	0	1
B	0	0	1	1
F	1	0	0	0

Рис. 13.7

Таблица 13.3

Вых.1 Q_t	Вх.1 R_t	Вх.4 S_t	Вых. Q_{t+1}	Состояние
0	0	1	1	Установка 1
1	0	0	1	Хранение 1
1	0	1	1	Подтверждение 1
1	1	0	0	Установка 0
0	0	0	0	Хранение 0
0	1	0	0	Подтверждение 0
0	1	1	*	Запрещенное
1	1	1	*	

Рассмотрим принцип действия триггера, определив местоположение входов S и R.

Не обозначая входы, подадим произвольную комбинацию сигналов на входы 1 и 4 (лог. 0 и 1 соответственно).

Для элемента *DD2* достаточно на одном из входов иметь активный сигнал — лог. 1, чтобы на его выходе был лог. 0. В свою очередь на входах логического элемента *DD1* будет совпадение нулей, что приведет к появлению на его выходе (выходе Q триггера) лог. 1. О состоянии триггера судят по выходу Q — там будет находиться лог. 1.

Установка триггера в единичное состояние произошла под воздействием активного сигнала на входе 4. В соответствии с табл. 13.1 вход 4, на который подан активный сигнал (в данном случае лог. 1), устанавливающий триггер в единичное состояние, обозначается буквой S, соответственно второй вход обозначается буквой R (рис. 13.8).

Запись в триггер лог. 0 осуществляется аналогично. При этом на входы 1 и 4 подаются лог. 1 и 0 соответственно.

В режиме хранения на информационные входы S и R подаются лог. 0, по наличию которых нельзя судить о состояниях на выходах элементов *DD1* и *DD2*. В этом случае активный сигнал,

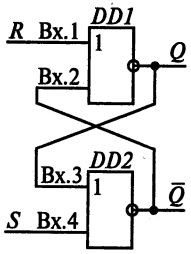


Рис. 13.8

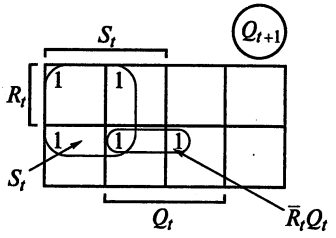


Рис. 13.9

как результат записи в предыдущем цикле обращения, присутствует на одном из выходов триггера и через обратную связь поступает на вход вх. 2 или 3, подтверждая предыдущее состояние схемы.

Одновременно подавать два единичных активных сигнала на входы S и R запрещено, так как это приводит к неопределенному состоянию логических элементов, охваченных обратной связью и не обеспечивает устойчивого состояния триггера.

Описать работу триггера можно характеристическим уравнением. Для этого надо составить карту Карно для Q_{t+1} (рис. 13.9), и на основании этой карты записать следующее выражение:

$$Q_{t+1} = \bar{R}_t Q_t \vee S_t.$$

Контрольные вопросы

1. Почему для *RS*-триггера с инверсными входами активным сигналом является лог. 0?
2. С какого логического элемента *DD1* или *DD2* и почему следует начинать анализ состояния схемы при подаче на входы $\bar{S}$ и $\bar{R}$ триггера лог. 1 и 0 соответственно?
3. Как обеспечивается режим хранения информации в триггере с инверсными входами? За счет чего триггер продолжает хранить ранее записанную информацию?
4. Какое сочетание входных сигналов запрещено? Почему невозможна работа триггера с инверсными входами в таком режиме?
5. Почему для *RS*-триггера с прямыми входами активным сигналом является лог.1?
6. С какого логического элемента *DD1* или *DD2* и почему следует начинать анализ состояния схемы при подаче на входы S и R триггера лог. 0 и 1 соответственно?
7. Как обеспечивается режим хранения информации в триггере с прямыми входами? За счет чего триггер продолжает хранить ранее записанную информацию?
8. Какое сочетание входных сигналов запрещено? Почему невозможна работа триггера с прямыми входами в таком режиме?

13.3. Синхронные триггеры с статическим управлением

Синхронный *RS*-триггер. В синхронном триггере доступ к запоминающему элементу возможен только при наличии управляющего сигнала на входе *C*. На рис. 13.10, *а* представлена логическая структура синхронного *RS*-триггера с статическим управлением, на рис.13.10, *б* — его условное графическое обозначение, а на рис. 13.11 — диаграмма работы.

Триггер состоит из запоминающего элемента, представляющего собой внутренний асинхронный *RS*-триггер с инверсными входами (*DD3*, *DD4*), и входных ключей (*DD1*, *DD2*). Информация,

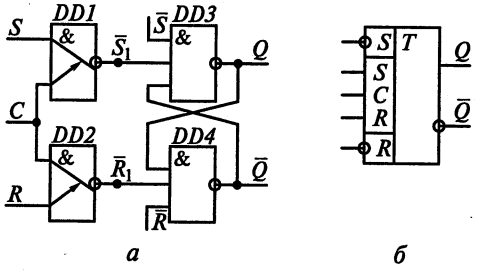


Рис. 13.10

поданная на внешние информационные входы S и R, пройдет к запоминающему элементу, но в инверсном виде, только при наличии разрешающего уровня сигнала на входе C.

Все синхронные триггеры снабжаются парой асинхронных входов $\overline{S}$ и $\overline{R}$, предназначенных для их предварительной установки в одно из устойчивых состояний. В дальнейшем на этих входах поддерживается лог. 1.

Рассмотрим режимы работы синхронного RS-триггера.

1. *Предварительная установка.* Перед началом работы цифровых устройств состояние триггеров может быть различным. Поэтому по специальным входам $\overline{S}$ и $\overline{R}$ осуществляется предварительная установка внутренних RS-триггеров в одно из устойчивых состояний, как правило, нулевое. При этом отсутствие разрешающего уровня сигнала на входе C обеспечивает в точках $\overline{S}_1$ и $\overline{R}_1$ лог. 1, что не мешает установке триггера;

2. *Режим хранения.* По окончании действия сигналов предварительной установки на входы $\overline{S}$ и $\overline{R}$ подаются лог. 1, обеспечивающие сохранение предварительно установленного состояния;

3. *Запись информации* (лог. 1). Записываемая информация подается на внешние прямые входы S и R синхронного RS-триггера (лог. 1 и 0 соответственно). Наличие этих сигналов при отсутствии сигнала на входе C не является достаточным условием для записи информации в элемент памяти, так как входные ключи *DD1* и *DD2* оказываются разомкнутыми (C — лог. 0).

В момент поступления управляющего сигнала на вход C оба ключа

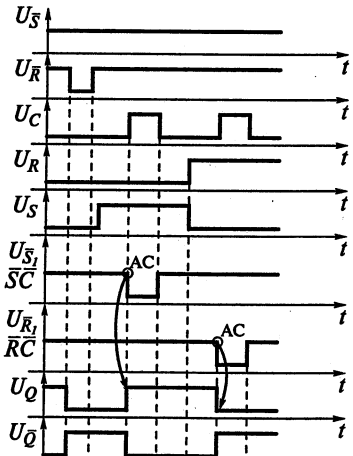


Рис. 13.11

Таблица 13.4

Входы			Выход	Состояние
S_t	R_t	C	Q_{t+1}	
Любое сочетание		0	Q_t	Хранение
1	0	1	1	Запись 1
0	1	1	0	Запись 0
1	1	1	*	Неопределенное

замыкаются, в результате чего на входы внутреннего триггера $\overline{S}_1$ и $\overline{R}_1$ поступает инверсное значение входной информации (лог. 0 и 1 соответственно). Наличие активного сигнала (АС) на входе $\overline{S}_1$ приводит к записи во внутренний *RS*-триггер лог. 1;

4. *Режим хранения.* По окончании действия управляющего сигнала на входе *C* ключи размыкаются, тем самым обеспечивая режим хранения записанной информации для внутреннего *RS*-триггера;

5. *Запись информации* (лог. 0). При записи в триггер лог. 0 рассуждения можно повторить, подав на прямые входы триггера *S* и *R* лог. 0 и 1 соответственно.

Необходимо иметь в виду, что подготовка записываемой информации на входах *S* и *R* должна всегда осуществляться до прихода управляющего сигнала на вход *C*.

В табл. 13.4 приведены состояния триггера при различных сочетаниях входных сигналов.

Основной недостаток такого триггера заключается в том, что при наличии управляющего сигнала и поступлении двух единиц

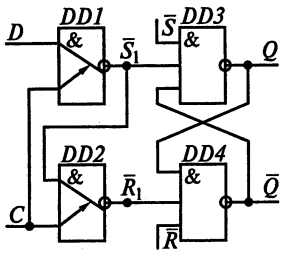


Рис. 13.12

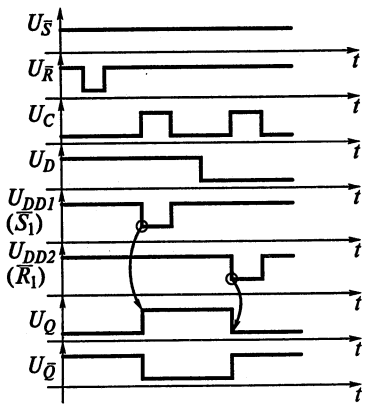


Рис. 13.13

Таблица 13.5

D_t	C_t	Q_t	Q_{t+1}	Состояние
1	0	0	0	Хранение 0
1	1	0	1	Запись 1
1	0	1	1	Хранение 1
1	1	1	1	Подтверждение 1
0	0	1	1	Хранение 1
0	1	1	0	Запись 0
0	0	0	0	Хранение 0
0	1	0	0	Подтверждение 0

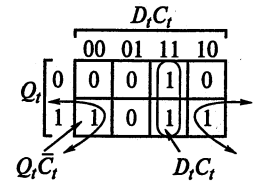
на информационные входы S и R на внутренних входах $\overline{S}_1$ и $\overline{R}_1$ появляются активные сигналы, которые приводят к неопределенному состоянию внутреннего триггера.

Синхронный *D*-триггер. В структуре такого триггера (рис. 13.12) в отличие от синхронного *RS*-триггера имеется один информационный вход D и соответствующий ключ доступа *DD1*. Наличие парафазных сигналов на входах $\overline{S}_1$ и $\overline{R}_1$ внутреннего *RS*-триггера обеспечивается с помощью дополнительного элемента *DD2*. На рис. 13.13 представлена диаграмма работы синхронного *D*-триггера.

Рассмотрим принцип действия этого триггера. Предварительно элемент памяти — внутренний *RS*-триггер (*DD3* и *DD4*) устанавливается в нулевое состояние и сохраняет его до момента замыкания ключа *DD1*. В момент поступления на вход C управляющего сигнала ключ доступа замыкается, обеспечивая поступление инверсного значения сигнала D на вход $\overline{S}_1$ внутреннего *RS*-триггера. Парафазное значение $\overline{R}_1$ получается с помощью ключа-инвертора *DD2*. Наличие активного сигнала на входе $\overline{S}_1$ приводит к записи во внутренний триггер лог. 1. По окончании управляющего сигнала C ключи-инверторы *DD1* и *DD2* размыкаются, обеспечивая *RS*-триггеру режим хранения записанной информации.

При записи в триггер лог. 0 рассуждения можно повторить.

В табл. 13.5 приведены состояния триггера при различных сочетаниях информационных сигналов. На основе таблицы состояний работа триггера может быть представлена картой Карно (рис. 13.14).



$Q_{t+1} = D_tC_t \vee Q_t\bar{C}_t$

Рис. 13.14

Контрольные вопросы

1. Для чего служат входные ключи в синхронных триггерах?
2. В какой момент времени относительно синхроимпульса осуществляется подключение внешних информационных входов синхронных триггеров к внутренним входам элемента памяти?
3. Какие сигналы необходимо подать на входы предварительной установки $\overline{S}$ и $\overline{R}$ для записи в синхронный триггер лог. 0?
4. В каком режиме находится синхронный триггер в промежутках между записью информации? Как этот режим обеспечивается?
5. Какие виды внешних информационных входов (прямые или инверсные) используются в синхронных триггерах?
6. В какой момент времени появляется записываемая информация на выходе синхронного триггера?
7. Какие функции выполняют ключи-инверторы в схеме синхронного *D*-триггера?

13.4. Двухступенчатые триггеры *MS*-типа с статическим управлением

Принцип организации триггеров *MS*-типа рассмотрен в подразд. 13.1. Особенности их работы заключаются в том, что запись информации в первую ступень триггера *M* производится с приходом управляющего сигнала на вход *C*, а перезапись во вторую ступень *S* осуществляется по окончании действия управляющего сигнала.

***RS*-триггер.** Как было отмечено ранее, в одноступенчатой схеме синхронного *RS*-триггера подготовка записываемой информации на входах *R* и *S* должна осуществляться до прихода управляющего сигнала на вход *C*. В двухступенчатой схеме триггера смена входной информации может происходить и во время действия управляющего сигнала, так как перезапись информации из первой ступени во вторую происходит по окончании его действия.

Рассмотрим принцип построения двухступенчатого *RS*-триггера. На рис. 13.15, *а* представлена его логическая структура, а на рис. 13.15, *б* — условное графическое обозначение.

Первая ступень *M* представляет собой синхронный *RS*-триггер, рассмотренный ранее. Отличие в работе второй ступени *S* заключается в том, что ключи доступа к ЭП (*DD6* и *DD7*) управляются инверсным значением входного сигнала $\overline{C}$.

Проведем поэтапный анализ работы элементов схемы данного триггера.

1. Анализ состояния входных ключей до, в момент и по окончании действия синхроимпульса в зависимости от сочетания сигналов *SC* (*DD2*) и *RC* (*DD3*). Сформированные сигналы являются сигналами воздействия $\overline{S}_1$ и $\overline{R}_1$ для внутреннего *RS*-триггера.

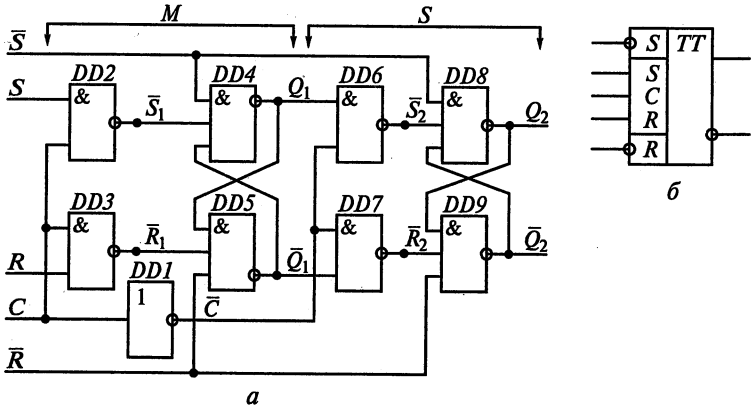


Рис. 13.15

2. Анализ сочетания сигналов $\bar{S}_1$ и $\bar{R}_1$ до, в момент и по окончании действия синхроимпульса и определение состояния триггера первой ступени в эти интервалы времени.

3. Анализ состояния ключей второй ступени до, в момент и по окончании действия синхроимпульса в зависимости от сочетания сигналов $Q_1\bar{C}$ (*DD6*) и $\overline{Q_1 C}$ (*DD7*). Сформированные сигналы являются сигналами воздействия $\bar{S}_2$ и $\bar{R}_2$ для внутреннего *RS*-триггера второй ступени.

4. Анализ сочетания сигналов $\bar{S}_2$ и $\bar{R}_2$ до, в момент и по окончании действия синхроимпульса и определение состояния триггера второй ступени в эти интервалы времени.

Для построения диаграммы работы двухступенчатого *RS*-триггера (рис. 13.16) проведем анализ работы каждого элемента схемы до, в момент и по окончании действия синхроимпульса (СИ), поданного на вход *C*.

Проанализируем состояние элементов первой ступени — входных ключей-инверторов *DD2*, *DD3* и асинхронного триггера (*DD4*, *DD5*).

До прихода СИ значение лог. 0 на входе *C* определяет лог. 1 на выходах *DD2*, *DD3* и соответственно на входах $\bar{S}_1$ и $\bar{R}_1$ триггера, тем самым удерживая его в режиме сохранения предыдущего состояния.

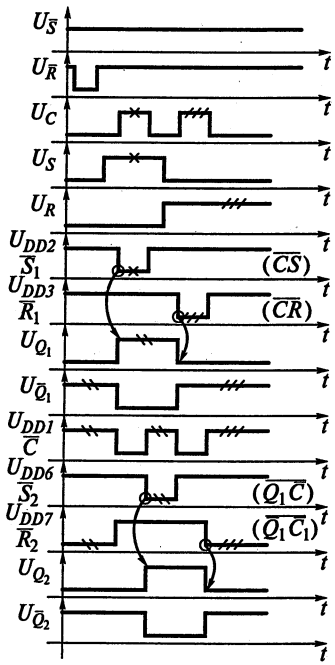


Рис. 13.16

В момент действия СИ (лог. 1) на выходах *DD2* и *DD3* формируются инверсные значения входных сигналов ($\bar{S}_1 = 0$; $\bar{R}_1 = 1$), и под действием активного сигнала на входе $\bar{S}_1$, равного лог. 0, происходит установка триггера в единичное состояние.

По окончании действия СИ значение лог. 0 на входе *C* формирует лог. 1 на входах $\bar{S}_1$ и $\bar{R}_1$, тем самым обеспечивая режим хранения записанной в триггер информации.

Проанализируем состояние элементов второй ступени — ключей-инверторов *DD6*, *DD7* и асинхронного триггера (*DD8*, *DD9*).

До прихода СИ на вход ключей поступает сигнал $\bar{C}$, равный лог. 1, обеспечивая прохождение инвертируемых сигналов с выходов Q_1 и $\bar{Q}_1$ триггера первой ступени на входы $\bar{S}_2$ и $\bar{R}_2$ триггера второй ступени, тем самым подтверждая хранящуюся там информацию.

В момент действия СИ значение сигнала $\bar{C}$ равно лог. 0. Это определяет значение лог. 1 на выходах *DD6*, *DD7* и соответственно на входах $\bar{S}_2$ и $\bar{R}_2$ триггера, тем самым удерживая его в режиме сохранения предыдущего состояния.

По окончании действия СИ на вход ключей поступает сигнал $\bar{C}$, равный лог. 1, обеспечивая прохождение изменившегося в момент действия СИ сочетания сигналов Q_1 и $\bar{Q}_1$, но в инверсной форме. Эти сигналы воздействуют на входы $\bar{S}_2$ и $\bar{R}_2$ триггера второй ступени, в результате чего он переходит в новое единичное состояние.

Необходимо отметить, что, как и в схеме одноступенчатого синхронного триггера, одновременная подача единиц на входы *S*, *R*, *C* вызывает неопределенное состояние рассматриваемого триггера. Этот недостаток устранен в схеме *JK*-триггера.

***JK*-триггер.** На рис. 13.17 представлена логическая структура *JK*-триггера, в табл. 13.1 — его условное графическое обозначение, а на рис. 13.18 — диаграмма работы. На рис. 13.19 показаны пути прохождения сигналов на входы внутренних *RS*-триггеров.

Отличие в построении схемы *JK*-триггера от *RS*-триггера заключается во введении дополнительных связей и подаче сигналов Q_2, $\bar{Q}_2$ на ключи доступа *DD1*, *DD2* и сигналов $\bar{S}_1$, $\bar{R}_1$ на *DD4*, *DD5* соответственно.

Работа триггера описывается таблицей состояний (табл. 13.6).

Рассмотрим особенности работы первой ступени. Информация со входов *J* и *K* при наличии управляющего сигнала на входе *C* проходит только через один из ключей *DD1* или *DD2* в зависимости от состояния *RS*-триггера второй ступени и соответственно наличия разрешающего сигнала с выхода Q_2 или $\bar{Q}_2$ (см. рис. 13.19, *а*). Второй ключ остается разомкнутым, что обеспечивает на его выходе лог. 1. Следовательно, прохождение информации через ключ

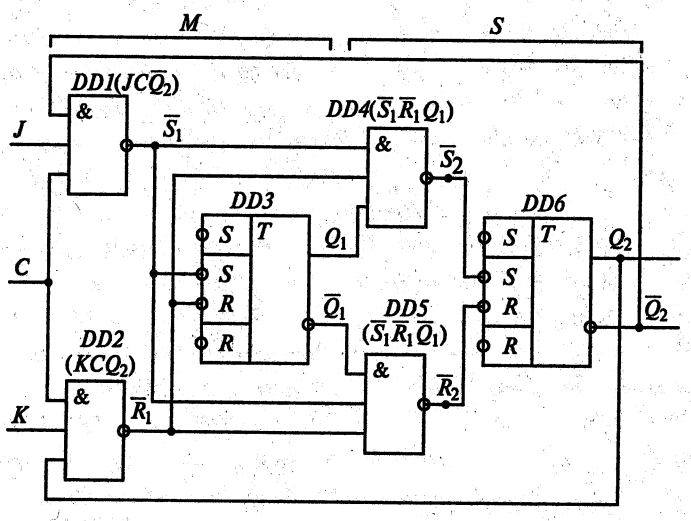
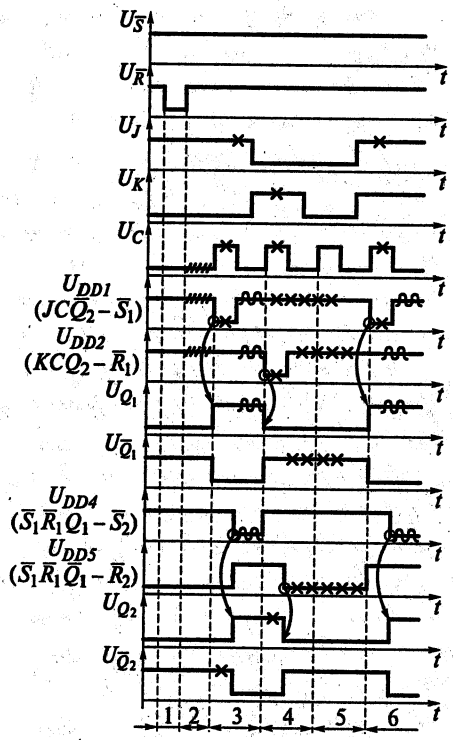


Рис. 13.17



Условные обозначения:

1 — предварительная установка;
2 — хранение;
3 — запись 1;
4 — запись 0;
5 — хранение;
6 — счетный режим

Рис. 13.18

Таблица 13.6

Входы		Выход	Состояние
J_t	K_t	Q_{t+1}	
1	0	1	Запись 1
0	1	0	Запись 0
0	0	Q_t	Хранение
1	1	$\overline{Q}_t$	Счетный режим

зависит от предыдущего состояния схемы. Если записываемая информация имеет противоположное значение относительно предыдущего состояния триггера, то один из ключей откроется, пропуская информационный сигнал воздействия на один из входов $\overline{S}_1$ или $\overline{R}_1$ внутреннего триггера первой ступени. Если на входы J и K подаются логические нули, то ключи доступа остаются разомкнутыми, а внутренние RS-триггеры продолжают хранить информацию. При поступлении на входы J и K единиц под воздействием управляющего сигнала C внутренний RS-триггер автоматически меняет свое состояние на противоположное. Такой режим работы называется счетным. В этом случае функциональное назначение входов ключей изменяется — J, K и C становятся управляющими (см. рис. 13.19, *б*), а с третьего информационного входа осуществляется передача информации с выхода Q_2 на вход $\overline{R}_1$ и с выхода $\overline{Q}_2$ на вход $\overline{S}_1$, в результате чего триггер первой ступени меняет свое состояние на противоположное.

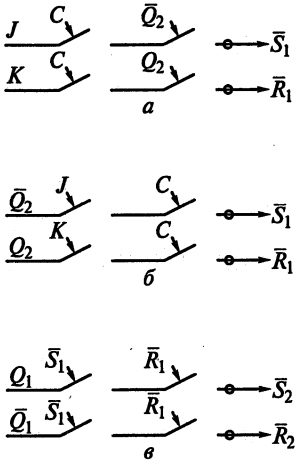


Рис. 13.19

Рассмотрим особенности работы второй ступени. Передача и запись информации с выходов Q_1 и $\overline{Q}_1$ RS-триггера первой ступени в RS-триггер второй ступени происходит при наличии управляющих сигналов $\overline{S}_1$ и $\overline{R}_1$ (см. рис. 13.19, *в*) в момент окончания действия управляющего сигнала на входе C.

Закон функционирования JK-триггера можно выразить формулой

$$Q_{t+1} = J_t \overline{Q}_t \vee \overline{K}_t Q_t.$$

Рассмотрим еще один вариант построения схемы JK-триггера (рис. 13.20), диаграмма работы которого представлена на рис. 13.21.

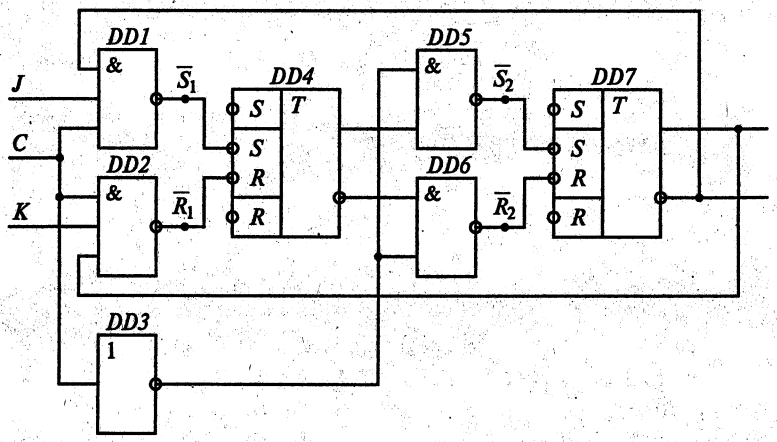


Рис. 13.20

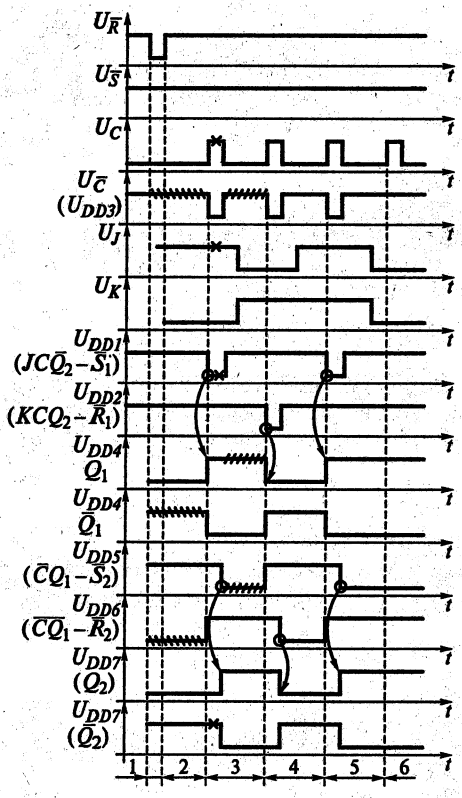


Рис. 13.21

Условные обозначения:

1 — предварительная установка в нулевое состояние;
2 — хранение;
3 — запись 1;
4 — запись 0;
5 — счетный режим;
6 — хранение

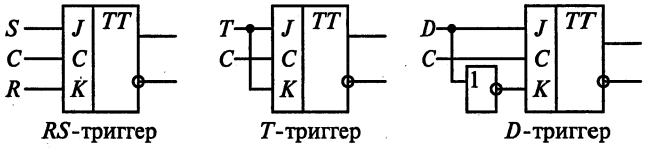


Рис. 13.22

Отличие данной схемы от ранее рассмотренной схемы *JK*-триггера заключается в организации работы только триггера второй ступени. В момент действия синхроимпульса ключи *DD1* и *DD2* передают информацию с входов *J* и *K* на входы $\overline{S}_1$ и $\overline{R}_1$ триггера первой ступени *DD4*. По окончании действия синхроимпульса триггер переходит в режим хранения.

Ключи *DD5* и *DD6*, управляемые инверсным значением синхроимпульса, передают информацию с выходов Q_1 и $\overline{Q}_1$ на входы $\overline{S}_2$ и $\overline{R}_2$ асинхронного триггера второй ступени *DD7* с момента окончания действия синхроимпульса до поступления следующего. В момент прихода синхроимпульса на вход *C* сигнал $\overline{C}$ равен лог. 0. Под действием этого сигнала ключи *DD5* и *DD6* размыкаются, обеспечивая режим хранения ранее записанной информации. По окончании действия синхроимпульса ключи вновь замыкаются, обеспечивая поступление новой информации на входы $\overline{S}_2$ и $\overline{R}_2$.

JK-триггер является универсальным. На его основе в зависимости от способа использования входов можно реализовать схемы двухступенчатых *RS*-, *T*-, *D*-триггеров. На рис. 13.22 представлены различные варианты включения универсального *JK*-триггера.

***T*-триггер.** Этот триггер реализует логическую функцию «Сложение по модулю 2». С приходом каждого импульса его состояние меняется на противоположное, что соответствует сложению без учета единицы переноса. Такой триггер может быть построен на базе двухступенчатых синхронных *RS*-, *D*- или *JK*-триггеров. На рис. 13.23 представлен *T*-триггер на основе *RS*-триггера. Импульсы счета здесь поступают на единственный вход *T*.

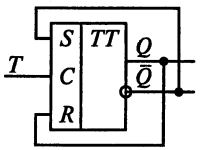


Рис. 13.23

Представленный на рис. 13.22 *T*-триггер на основе *JK*-триггера может работать в двух режимах — счетном и хранения. Эти режимы обеспечиваются подачей на вход *T* лог. 1 и лог. 0 соответственно и аналогичны режимам *JK*-триггера. Импульс счета подается на вход *C*.

Функционирование *T*-триггера описывается уравнением

$$\overline{Q}_{t+1} = Q_t \overline{T}_t \vee \overline{Q}_t T_t.$$

Контрольные вопросы

1. Какую роль выполняет первая и вторая ступени в триггере *MS*-типа?
2. В какие моменты времени по отношению к синхроимпульсу происходит перезапись информации из первой ступени *M* во вторую ступень *S* в двухступенчатом синхронном *RS*-триггере с статическим управлением?
3. Какие сигналы являются синхроимпульсами для триггера второй ступени в *RS*-триггере с статическим управлением?
4. Какая информация поступает на входы $\overline{S}_2$ и $\overline{R}_2$ в момент замыкания и размыкания ключей доступа?
5. При каком сочетании сигналов на входах *J* и *K* с учетом внутреннего состояния триггеров первой и второй ступеней производится запись новой информации?
6. Как работают ключи доступа первой и второй ступеней *JK*-триггера?
7. Какой режим работы *JK*-триггера называется счетным? Как он обеспечивается?
8. Как называется триггер для подсчета единиц информации?
9. На основе каких типов триггеров можно организовать *T*-триггер?
10. Почему *JK*-триггер называется универсальным?

13.5. Синхронные триггеры с динамическим управлением

***RS*-триггер.** Специфика организации триггеров с динамическим управлением была рассмотрена ранее. Особенность их работы заключается в том, что запись информации в триггер никогда не производится под действием управляющего уровня на входе *C*, что предотвращает ложное срабатывание схем. Информация записывается только на коротком интервале в окрестностях фронта синхросигнала, поданного на вход *C*. Среди триггеров с динамическим управлением широкое распространение получила так называемая схема трех триггеров. На рис. 13.24 представлена схема триггера с динамическим управлением и его условное графическое обозначение, а на рис. 13.25 — диаграмма работы.

При построении схем этих триггеров широко используется то, что сигнал, идущий в такой схеме по короткому проводу, всегда обгоняет сигнал, идущий через логический элемент. Это не учитывалось, когда рассматривались принципы действия предыдущих схем триггеров.

В зависимости от сочетания информационных сигналов в формировании сигналов $\overline{S}_1$ и $\overline{R}_1$ для внутреннего триггера участвуют три логических элемента: *DD1*, *DD2*, *DD3* или *DD4*, *DD3*, *DD2*.

Рассмотрим характерные состояния триггера при записи лог. 1, лог. 0, хранении.

1. Подадим на входы $\overline{S}$ и $\overline{R}$ лог. 0 и лог. 1 соответственно. Наличие лог. 0 на входе $\overline{S}$ способствует формированию информационного сигнала *S* для ключа *DD2*, равного лог. 1. Логическая схе-

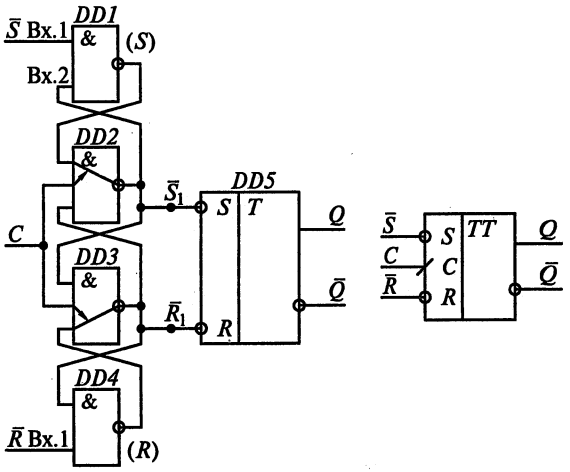
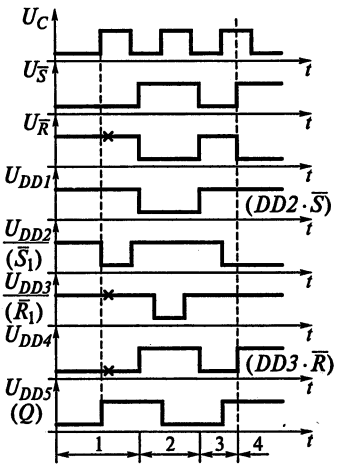


Рис. 13.24

ма *DD4* обеспечивает формирование информационного сигнала *R*, равного лог. 0, так как до прихода синхроимпульса на выходе *DD3* — лог. 1, на входе $\bar{R}$ — лог. 1.



Условные обозначения:

1 — запись и хранение лог. 1;
2 — запись и хранение лог. 0;
3 — запись и хранение лог. 1;
4 — изменение информации в момент действия синхроимпульса

Рис. 13.25

Электронные ключи-инверторы *DD2*, *DD3* замыкаются с приходом синхроимпульса, обеспечивая передачу информационных сигналов *S* и *R* на входы $\bar{S}_1$ и $\bar{R}_1$ внутреннего *RS*-триггера. По окончании действия синхроимпульса на выходе ключей *DD2* и *DD3*, а следовательно, на входах внутреннего *RS*-триггера — лог. 1.

Далее рассматривается поведение внутреннего *RS*-триггера относительно синхроимпульса.

2. Подадим на входы $\bar{S}$ и $\bar{R}$ лог. 1 и лог. 0 соответственно. Наличие лог. 0 на входе $\bar{R}$ способствует формированию информационного сигнала *R*, равного лог. 1. Логическая схема *DD1* обеспечивает формирование информационного сигнала *S*, равного лог. 0.

В момент действия фронта синхроимпульса информационные сигналы передаются на входы $\bar{S}_1$ и $\bar{R}_1$ внутреннего *RS*-триггера.

Далее рассматривается поведение внутреннего *RS*-триггера.

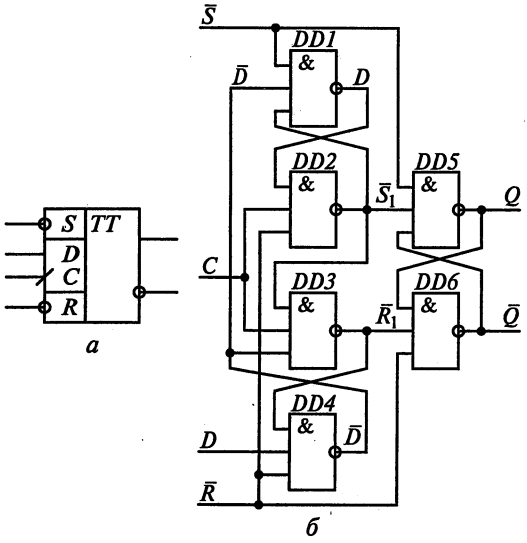


Рис. 13.26

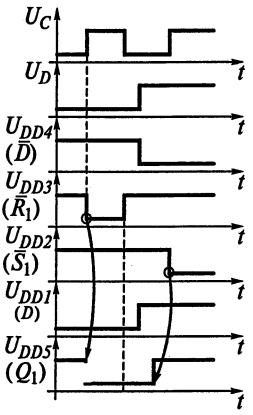


Рис. 13.27

3. Вновь подадим на входы $\bar{S}$ и $\bar{R}$ лог. 0. и лог. 1 соответственно. Рассуждения о происходящих при этом процессах повторяются в соответствии с п. 1.

4. Рассмотрим смену информационных сигналов в момент действия уровня синхроимпульса. Покажем, что этот процесс не влияет на записанную и хранящуюся во внутреннем *RS*-триггере информацию.

До этого по фронту третьего синхроимпульса в схеме формируются сигналы воздействия на внутренний *RS*-триггер: на выходе схемы *DD2* ($\bar{S}_1$) — лог. 0, на выходе схемы *DD3* ($\bar{R}_1$) — лог. 1. Появление лог. 0 на выходе схемы *DD2* влияет на состояние схем *DD1* и *DD3*, т. е. удерживает на их выходах лог. 1.

Изменение информации в рассматриваемый момент на входах $\bar{S}$ (лог. 1) и $\bar{R}$ (лог. 0) при наличии синхроимпульса с уровнем сигнала, равного лог. 1, не влияет на состояние схемы *DD1* (так как на входе 2 лог. 0) и схемы *DD3* (так как наличие лог. 0 на входе $\bar{R}$ схемы *DD4* приводит к появлению лог. 1 на ее выходе). Соответственно состояние схемы *DD3* продолжает определяться сигналом с выхода схемы *DD2*, равным лог. 0, т. е. сигналы в точках $\bar{S}_1$ и $\bar{R}_1$ своего значения не меняют, а внутренний *RS*-триггер продолжает сохранять свое состояние, принятое по фронту третьего синхроимпульса.

***D*-триггер.** Схема *D*-триггера и его УГО представлены соответственно на рис. 13.26, *а* и *б*. Схема этого триггера отличается от ранее рассмотренной схемы *RS*-триггера с динамическим управлением наличием одного информационного входа *D*. Диаграмма работы *D*-триггера приведена на рис. 13.27. Рассмотрим принцип его действия.

Парафазное значение сигналов с входа *D* обеспечивается схемами *DD1* и *DD4*.

Подадим на вход *D* лог. 0. Наличие лог. 0 обеспечивает на выходе схемы *DD4* лог. 1, которая будет являться информационным сигналом для ключа *DD3*.

Логическая схема *DD1* обеспечивает формирование информационного сигнала *D* для *DD2*. До прихода синхроимпульса на выходе *DD2* — лог. 1, на входе $\bar{D}$ с выхода *DD4* — лог. 1, следовательно, на выходе схемы *DD1* будет лог. 0, что подтверждает значение входного сигнала *D*.

Электронные ключи-инверторы *DD2*, *DD3* замыкаются в момент действия фронта синхроимпульса, обеспечивая передачу информационных сигналов *D* и $\bar{D}$ на входы $\bar{S}_1$ и $\bar{R}_1$ внутреннего *RS*-триггера. По окончании действия синхроимпульса на выходе ключей, а следовательно, на входах внутреннего *RS*-триггера — лог. 1.

Далее рассматривается поведение внутреннего *RS*-триггера относительно синхроимпульса.

Затем рассуждения повторяются аналогично тому, как это сделано ранее для *JK*-триггера.

Контрольные вопросы

1. В чем заключаются различия в построении *RS*-триггеров с статическим и динамическим управлением?
2. Каким образом обеспечивается нечувствительность *RS*-триггера с динамическим управлением к входной информации в момент действия синхроимпульса?
3. Через какой ключ (*DD2* или *DD3*) осуществляется прохождение записываемой информации (*S* =1, *R* = 0) в *RS*-триггере с динамическим управлением?
4. С какой целью введены цепи обратной связи, соединяющие схемы *DD2* и *DD3* в *RS*-триггере с динамическим управлением?
5. В чем заключаются различия в построении схем *RS*- и *D*-триггеров с динамическим управлением?
6. С выходов каких элементов схемы осуществляется передача информации через ключи-инверторы *DD2*, *DD3* в *D*-триггере с динамическим управлением?

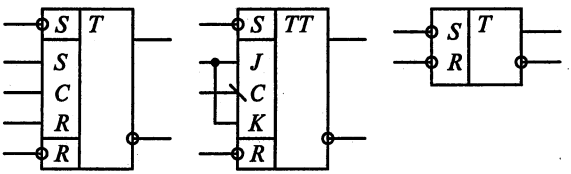


Рис. 13.28

7. Какая характерная особенность в УГО триггеров указывает на динамическое управление ими?

8. Дайте обобщенную характеристику триггерам, УГО которых представлены на рис. 13.28.

Глава 14
РЕГИСТРЫ

14.1. Общие сведения

Регистром называется узел ЭВМ, предназначенный для приема, хранения и передачи информации в другие узлы в процессе выполнения операций в вычислительных устройствах.

Регистр представляет собой совокупность триггеров, число которых соответствует количеству разрядов размещаемой в нем информации. Наличие прямых и инверсных выходов позволяет использовать такое устройство для получения кодов чисел хранимой информации.

По способу приема и передачи информации регистры подразделяются на параллельные, последовательные и последовательно-параллельные (рис. 14.1).

В *параллельных регистрах* прием и передача *n*-разрядного кода производятся параллельно под воздействием управляющего синхросигнала.

В *последовательных регистрах* прием и передача *n*-разрядной информации производятся последовательно разряд за разрядом, при этом внутри регистра возможны однонаправленный сдвиг информации из разряда в разряд влево или вправо и двунаправленный (т. е. реверсивный) сдвиг в зависимости от условий как вправо, так и влево.

В *последовательно-параллельных регистрах* прием и передача *n*-разрядной информации могут производиться в последовательном, параллельном и смешанном режимах. В смешанном режиме регистр выполняет функции преобразователя последовательного входного кода в параллельный выходной код и, наоборот, параллельного входного кода в последовательный выходной код.



Рис. 14.1

14.2. Схемы простейших регистров

Регистр параллельного действия на основе асинхронного *RS*-триггера. Схема простейшего четырехразрядного регистра параллельного действия представлена на рис. 14.2. Она состоит из входных ключей *DD1*, используемых для ввода в регистр четырехразрядной информации; элементов памяти *DD2* — асинхронных *RS*-триггеров; выходных ключей *DD3*, используемых для вывода из регистра хранимой информации.

Регистр может работать в следующих режимах:

предварительная установка (подготовка триггеров регистра к записи информации);

запись информации в триггеры регистра;

хранение ранее записанной информации;

чтение информации из регистра.

Записываемая информация поступает на информационные входы *X1*...*X4*. Вход «Уст. 0» используется при предварительной установке триггеров в нулевое состояние подачей на него лог. 1. Управление режимами работы осуществляется по управляющим входам: «Запись», «Чтение». При отсутствии управляющих сигналов, равных лог. 1, на этих входах находятся лог. 0.

Рассмотрим организацию работы регистра в различных режимах.

1. *Предварительная установка.* В рассматриваемом варианте схемы всегда перед записью информации производится установка триггеров в нулевое исходное состояние. Для этого на вход «Уст. 0» подается лог. 1. При отсутствии управляющего сигнала на входе «Запись» лог. 0 с этого входа воздействует на управляющие входы ключей доступа *DD1*, тем самым обеспечивая поступление лог. 0 на все входы *S* триггеров. Под действием этого сочетания сигналов *RS*-триггеры с прямыми входами устанавливаются в состояние лог. 0.

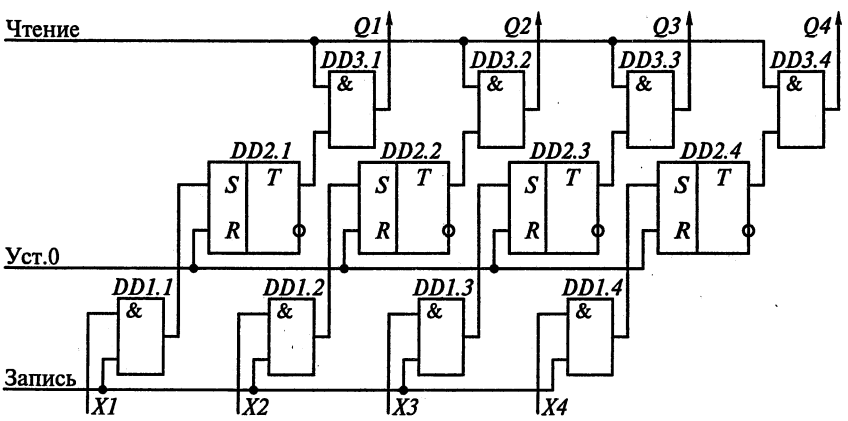


Рис. 14.2

При снятии единичного сигнала с входа «Уст. 0» на все *R* входы *RS*-триггеров поступают лог. 0 и они переходят в режим хранения.

2. *Запись информации по входам параллельного ввода X1, X2, X3, X4.* При записи информации ключи *DD3*, установленные на выходе регистра, должны быть разомкнуты. Это достигается подачей на управляющий вход «Чтение» лог. 0.

Кодовая комбинация, поданная на входы *X1...X4*, пройдет на входы *S* всех триггеров при поступлении управляющего сигнала на вход «Запись», соответствующего лог. 1, который замкнет входные ключи доступа *DD1* к триггерам. Триггеры, на вход *S* которых поступают активные сигналы лог. 1, устанавливаются в единичное состояние. Триггеры, на входы *S* которых поступают лог. 0, остаются в состоянии хранения лог. 0 от предварительной установки. При снятии лог. 1 с входа «Запись» регистр переходит в режим хранения информации.

3. *Чтение информации, хранимой в регистре.* При чтении информации входные ключи *DD1* разомкнуты, а триггеры должны находиться в режиме хранения, что обеспечивается нулевыми сигналами на входах «Уст. 0», «Запись». Для прохождения информации от триггеров на выход регистра надо замкнуть ключи *DD3*. Для этого на управляющий вход «Чтение» подается лог. 1. Для последующего перевода регистра в режим хранения на его управляющие входы подаются лог. 0.

Регистр последовательного действия на основе синхронного *D*-триггера. Простейший регистр последовательного действия представлен на рис. 14.3.

Такой регистр имеет один вход для последовательного ввода информации. Межразрядные связи соединяют выходы триггеров младших разрядов с входами триггеров старших разрядов, что позволяет осуществлять последовательное продвижение информации из разряда в разряд под воздействием управляющих синхроимпульсов (С). Необходимо иметь в виду, что одновременная перезапись информации из одного триггера в другой и далее в последующие триггеры от одного синхроимпульса невозможна, так как это триггеры с динамическим управлением, запись в которые осуществляется только кратковременно по фронту синхроимпульса.

В схеме можно использовать двухступенчатые триггеры, тогда информация на выходе появится только по окончании синхроим-

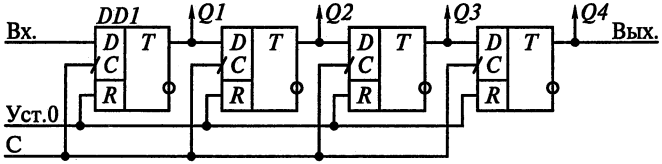


Рис. 14.3

пульса, что исключит перезапись ее в следующий разряд, но быстродействие такой схемы будет ниже.

При объединении выводов входа и выхода образуется схема кольцевого регистра, в котором происходит циркуляция ранее записанной информации.

Регистр последовательного действия уступает регистру параллельного действия в быстродействии и используется только в случаях, когда количество входных и выходных линий связи ограничено.

Контрольные вопросы

1. Какую роль играют регистры при выполнении операций умножения и деления чисел?
2. Какое место занимают регистры при их совместной работе с дешифраторами?
3. Какое место занимают регистры при их совместной работе с мультиплексорами?
4. Какие функции выполняет кольцевой регистр, в который записана единица, если сигналы с его выходов использовались в качестве управляющих для мультиплексора?
5. Сравнить быстродействие регистров параллельного и последовательного действия.
6. Сколько синхроимпульсов необходимо подать при записи байта информации в регистры параллельного и последовательного действия?
7. Сигналы какого уровня необходимо подать на управляющие входы входных и выходных ключей в регистре параллельного действия в режиме хранения?
8. С помощью каких входов регистра производится предварительная установка триггеров в нулевое состояние?
9. Как обеспечивается режим хранения информации в триггерах регистра параллельного действия?
10. Как изменится схема, если в качестве входных электронных ключей в регистре параллельного действия использовать элементы И-НЕ?
11. Где в регистрах параллельного и последовательного действия расположены триггеры старшего и младшего разрядов?
12. В какой последовательности необходимо подавать записываемую информацию в регистр последовательного действия?

14.3. Реверсивные регистры

Логический элемент И-ИЛИ в качестве электронного ключа. Как правило, схемы регистров выполняют более сложные функции, чем рассмотренные ранее. Часто предусматривается возможность их работы как в параллельном, так и последовательном режимах, а также возможность сдвига информации как влево, так и вправо. Когда в схеме совмещены такие функции, то говорят, что она работает в реверсивном режиме. Для организации работы в реверсивном режиме используют логические схемы И-ИЛИ (рис. 14.4). Два внутренних

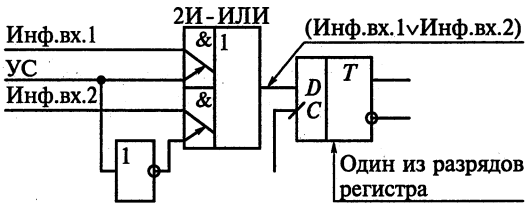


Рис. 14.4

логических элемента И управляются парафазными (противоположными) сигналами, обеспечивая прохождение информации с одного из двух входов.

Рассмотрим упрощенную схему обеспечения сдвига информации в двух направлениях (рис. 14.5). Замыкание ключей, обеспечивающих режим сдвига, осуществляется здесь взаимоинверсными сигналами.

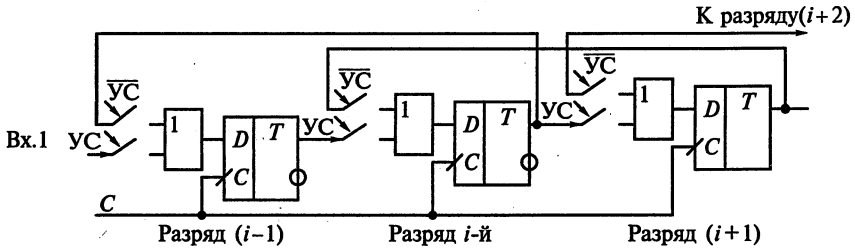


Рис. 14.5

На рис. 14.6 показан пример схемной реализации межразрядных связей с использованием элемента И-ИЛИ, который обеспечивает поступление сигналов с прямых выходов старших разрядов на выходы *D* младших, или наоборот. Фиксация поступающей информации происходит в момент прихода синхроимпульса.

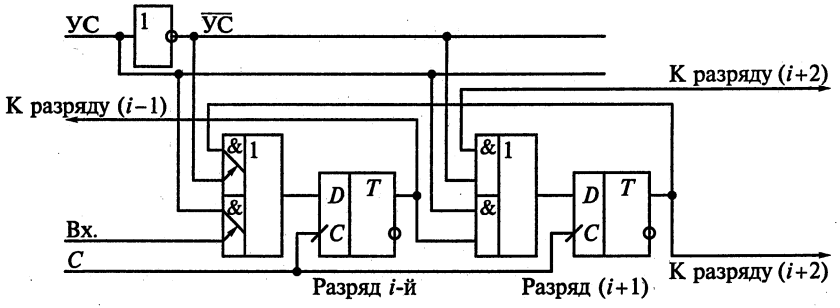


Рис. 14.6

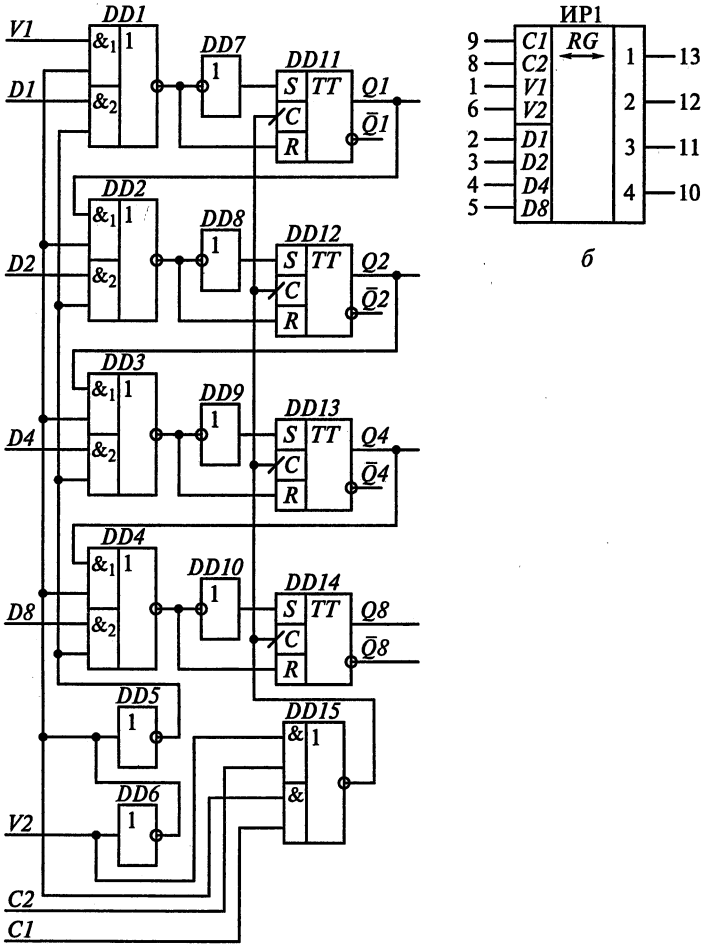


Рис. 14.7

Схема универсального регистра в интегральном исполнении. На рис. 14.7, *а* представлена схема, на рис. 14.7, *б* — условное графическое обозначение универсального регистра ИР1.

Схема может работать в следующих режимах:

параллельная запись;

последовательная запись и сдвиг информации из младших разрядов в старшие;

последовательная запись и сдвиг информации из старших разрядов в младшие;

совмещение двух режимов — параллельной записи и сдвига информации с последующей записью в освобождающиеся разряды новых данных.

Режим работы задается сигналом, поданным на вход *V2*. При параллельной записи и последовательном сдвиге информации от старших разрядов к младшим на вход *V2* подается лог. 1, при последовательном вводе и сдвиге информации от младших разрядов к старшим — лог. 0.

Параллельная запись информации осуществляется по информационным входам *D1*, *D2*, *D4*, *D8*; последовательный ввод информации производится по входу *V1*, начиная со старшего разряда. При параллельной записи синхроимпульс подается на вход *C2*, при последовательном сдвиге — на вход *C1*.

Схемы *DD1*... *DD4* выполняют роль ключей-инверторов, которые пропускают информацию **или** с параллельных входов *D1*, *D2*, *D4*, *D8*, **или** с выходов триггеров младших разрядов на входы старших разрядов при последовательном вводе и сдвиге информации.

Управление замыканием ключей осуществляется через схемы *DD5*, *DD6*. Вводимая в триггеры информация проходит через дополнительные инверторы *DD7*... *DD10*, поступая парафазным кодом на входы *S* и *R* синхронных триггеров, и фиксируется в них с приходом синхроимпульса.

При последовательном сдвиге информации снизу вверх (от старших разрядов к младшим) организация режима ввода информации аналогична параллельной записи. В этом случае используются входы параллельного ввода информации и осуществляются соединения: выхода *Q8* с входом *D4*; *Q4* с *D2*; *Q2* с *D1*. Входом последовательного ввода информации в этом случае является *D8*. При использовании регистра в этом режиме параллельная запись невозможна. Такой режим сдвига обеспечивается подачей на вход *V2* лог. 1, а на *C2* — синхроимпульса.

Управление работой регистра можно осуществлять от одного синхроимпульса, для чего необходимо объединить входы *C1* и *C2*.

Практическое использование регистра ИР1 при построении схемы накапливающего сумматора. Структура и принцип действия накапливающего сумматора рассматривались в подразд. 11.11. В качестве памяти промежуточных сумм в нем используются схемы регистров. На рис. 14.8 представлена схема накапливающего сумматора, которая выполнена на основе полных комбинационных сумматоров (*DD1*, *DD2* — ИМ3; *DD3* — ИМ2) и регистра (*DD4*... *DD6* — ИР1).

На вход *A* сумматоров поступают изменяемые значения внешнего слагаемого, на вход *B* — накопленное в результате предыдущих вычислений значение суммы, хранящееся в регистре.

Перед началом суммирования все регистры памяти устанавливаются в нулевое состояние, для чего можно было бы использовать входы параллельного ввода информации в регистр, но они уже задействованы в схеме, поэтому регистр обнуляется в последовательном режиме путем подачи на вход *V2* и на вход по-

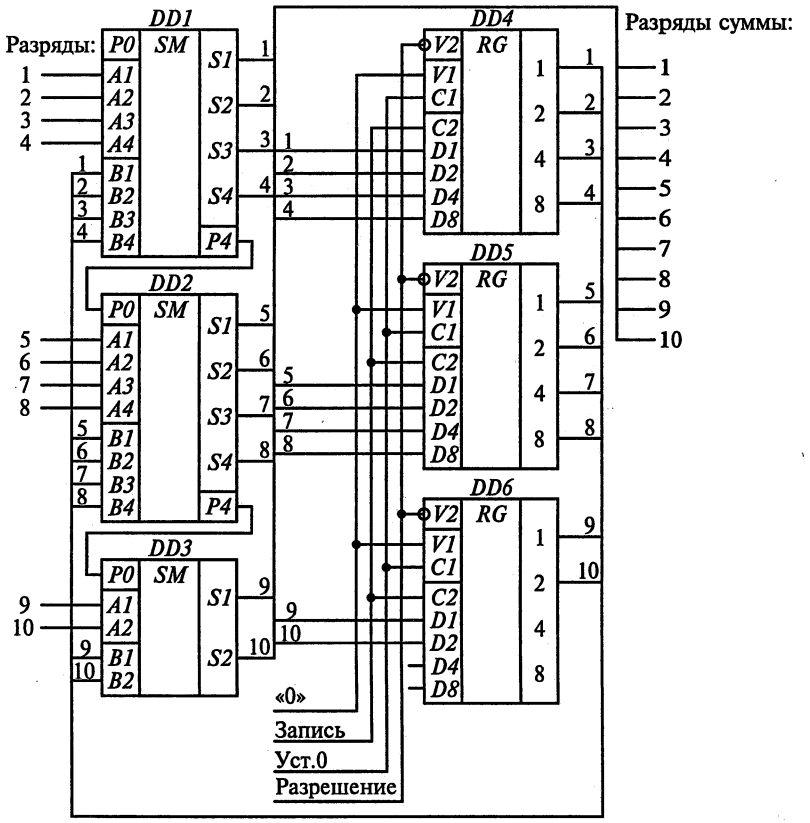


Рис. 14.8

следовательного ввода *V1* лог. 0. Затем, подав четыре синхроимпульса на вход *C1*, осуществляется последовательное обнуление триггеров регистра. Так перед началом суммирования обеспечивается в качестве исходного слагаемого нулевое значение слагаемого *B*. Далее происходит сложение и накопление последующих сумм.

Контрольные вопросы

1. Для чего предназначен логический элемент И-ИЛИ при передаче информации от двух источников информации?
2. Как классифицируются входы универсального регистра по их функциональному назначению?
3. Каково значение управляющих сигналов, с помощью которых обеспечивается параллельный ввод информации в регистр ИР1?
4. Для чего предназначены схемы *DD7*... *DD10* в схеме ИР1?
5. Какие дополнительные соединения необходимо сделать в схеме регистра ИР1, чтобы обеспечить последовательный ввод и сдвиг информа-

ции от старших разрядов к младшим? Какие из выводов микросхемы выполняют функции последовательного ввода и вывода?

6. Какой из разрядов (старший или младший) подается на вход *V1* в первую очередь при вводе и сдвиге информации от младших разрядов в старшие?

7. Какие выводы микросхемы можно использовать при последовательном чтении информации (сдвиге от младших разрядов к старшим)?

Глава 15
СЧЕТЧИКИ

15.1. Общие сведения

Счетчиком называется устройство, предназначенное для подсчета единиц информации, поступающих на его вход. Такое устройство представляет собой набор триггеров, соединенных между собой определенным образом, количество которых определяется разрядностью счетчика.

Возможности счетчика определяются максимальным количеством подсчитываемых им единиц информации и характеризуются коэффициентом пересчета K. Обычно количество единиц информации определяется как $K = 2^n$, где n — количество триггеров (разрядов) в схеме счетчика. Таким образом, из формулы видно, что в зависимости от разрядности счетчика n, в нем может быть осуществлен максимальный подсчет единиц информации кратных 2, т.е. 2, 4, 8, 16 и т.д. Начиная с нулевого состояния, через каждые 2^n счетных импульсов триггеры счетчика будут возвращаться в исходное состояние, после чего процесс счета начинается снова. Если требуется организовать подсчет единиц информации не кратный 2^n, то используют счетчики с произвольным коэффициентом пересчета.

При построении счетчиков нашли широкое применение универсальный *JK*-триггер и построенный на его основе *T*-триггер, работающий в счетном режиме, для которого характерно деление входной частоты на два: $K = 2^n$, т.е. для триггера любого разряда выходная частота будет вдвое меньше частоты на входе. По мере увеличения разрядности n счетчика будет происходить соответствующее деление исходной частоты в 2^1, 2^2, 2^3 и т.д. раз. О количестве подсчитываемых единиц информации судят по состоянию триггеров после прихода одного или серии импульсов счета.

Если необходимо получить каждый i-й импульс счета, используются схемы делителей. Их отличие от схем счетчиков заключается в том, что на выходе устанавливается ключ, который пропускает только определенный счетный импульс. Условия, при которых



Рис. 15.1

это происходит, подготавливаются сигналами с определенных выходов триггеров.

На рис. 15.1 приведена классификация счетчиков.

По направлению счета счетчики делят на суммирующие (с прямым счетом), вычитающие (с обратным счетом) и реверсивные. В суммирующих счетчиках с приходом очередного импульса счета результат увеличивается на единицу, в вычитающих счетчиках — уменьшается на единицу. Реверсивные счетчики могут работать как в режиме сложения, так и вычитания.

По способу организации переноса счетчики делятся на асинхронные, синхронные и комбинированные. При асинхронной организации триггеры всех разрядов соединены последовательно между собой цепями переноса, при синхронной организации импульсы счета поступают на все триггеры одновременно.

Комбинированные счетчики с последовательно-параллельным (групповым) переносом представляют собой определенное число групп, каждая из которых включает в себя несколько триггеров. Группы организованы как синхронные счетчики с параллельным переносом, а объединение этих групп осуществляется последовательно.

15.2. Асинхронные счетчики

Асинхронные счетчики представляют собой цепочку триггеров, последовательно соединенных друг с другом. Импульсами счета для триггеров более старших разрядов являются сигналы с выходов Q или $\overline{Q}$ триггеров младших разрядов.

Асинхронный суммирующий счетчик (прямого счета). Схема асинхронного суммирующего счетчика представлена на рис. 15.2. Временна́я диаграмма, поясняющая принцип действия, приведена на рис. 15.3.

Отметим характерные особенности в организации работы схемы такого счетчика.

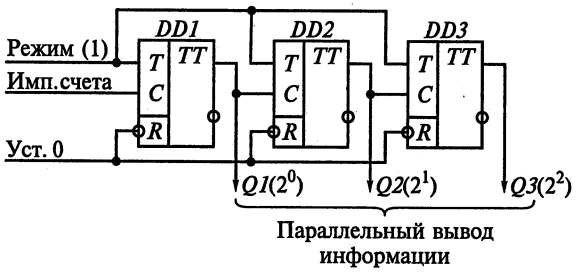


Рис. 15.2

1. *T*-триггеры, используемые в данной схеме, работают в счетном режиме. Для этого на вход «Режим» подается лог. 1.

2. В счетчике импульсами счета для триггеров старших разрядов являются сигналы с выходов Q младших разрядов.

3. Частота срабатывания триггера старшего разряда в два раза меньше частоты срабатывания триггера младшего разряда.

4. По состоянию триггеров после каждого импульса счета судят о количестве поданных единиц информации на вход счетчика. При этом код, снимаемый с их выходов, последовательно увеличивается на единицу.

Стрелками на диаграмме выделены импульсы счета и моменты времени, когда должны сработать триггеры более старших разрядов.

Асинхронный вычитающий счетчик (обратного счета). Схема асинхронного вычитающего счетчика представлена на рис. 15.4. Временна́я диаграмма, поясняющая принцип действия, приведена на рис. 15.5.

Отметим характерные особенности в организации работы схемы такого счетчика.

1. В счетчике импульсами счета для триггеров старших разрядов являются сигналы с выходов $\overline{Q}$ младших разрядов.

2. Анализируя состояние триггеров после каждого импульса счета, нетрудно убедиться, что код, снимаемый с их выходов, последовательно уменьшается на единицу.

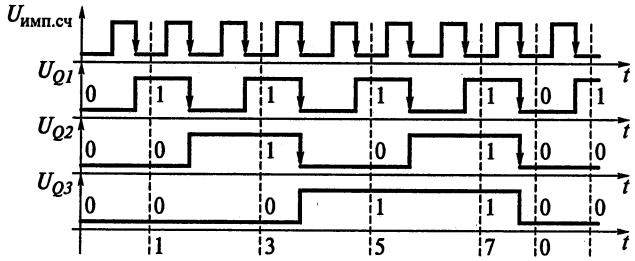


Рис. 15.3

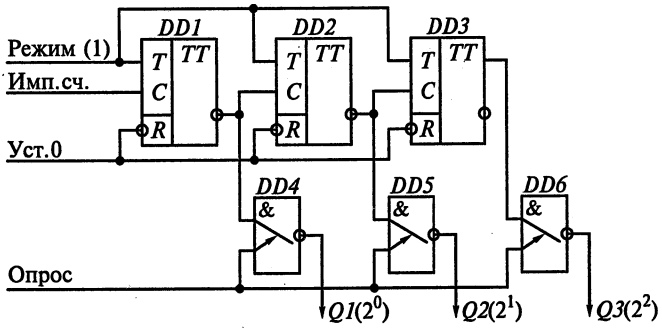


Рис. 15.4

3. Опрос счетчиков производится в строго определенное время с использованием электронных ключей (*DD4*, *DD5*, *DD6*). Поэтому при подаче строба «Опрос» ($t_{опр}$) должны учитываться задержки переключения триггеров всех разрядов.

Счетчик в интегральном исполнении ИЕ5. На рис. 15.6, *а* представлена схема счетчика ИЕ5, а на рис. 15.6, *б* — его условное обозначение.

Микросхема состоит из четырех *JK*-триггеров, работающих в счетном режиме. Самостоятельное включение схемы *DD1* и схем *DD2*, *DD3*, *DD4* позволяет получать счетчики с коэффициентом счета $K_{сч} = 2$ или 8. Для получения $K_{сч} = 16$ необходимо соединить выход *Q1* схемы *DD1* с входом *C2* схемы *DD2*, т. е. образовать ранее рассмотренную схему счетчика сложения.

Предварительная установка триггеров осуществляется через схему *DD5* подачей лог. 1 одновременно на оба входа *R0*. Наличие этой схемы позволяет расширить функциональные возможности счетчика и обеспечить коэффициент пересчета не кратный 2. Организация таких счетчиков рассмотрена далее.

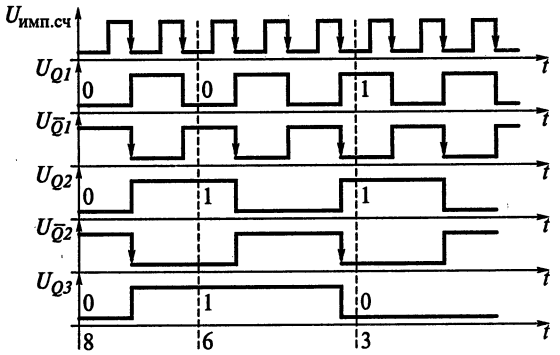


Рис. 15.5

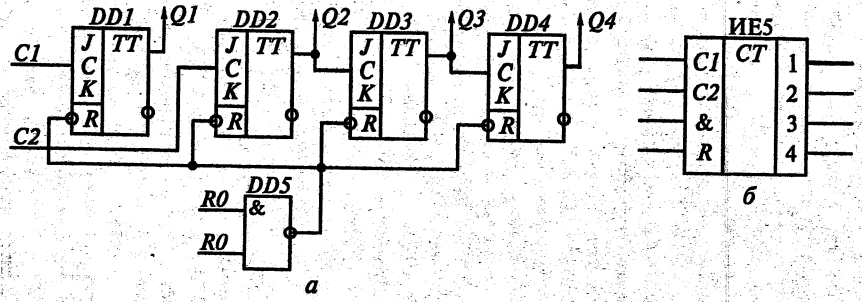


Рис. 15.6

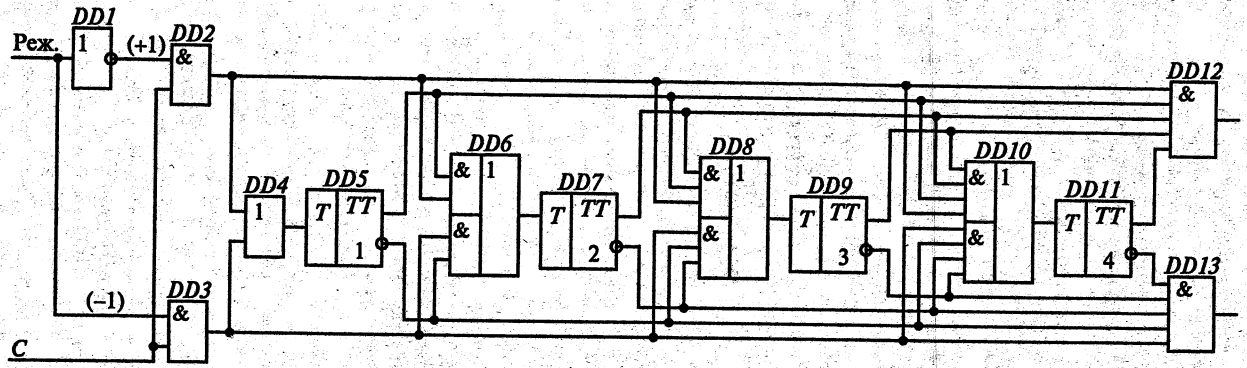


Рис. 15.7

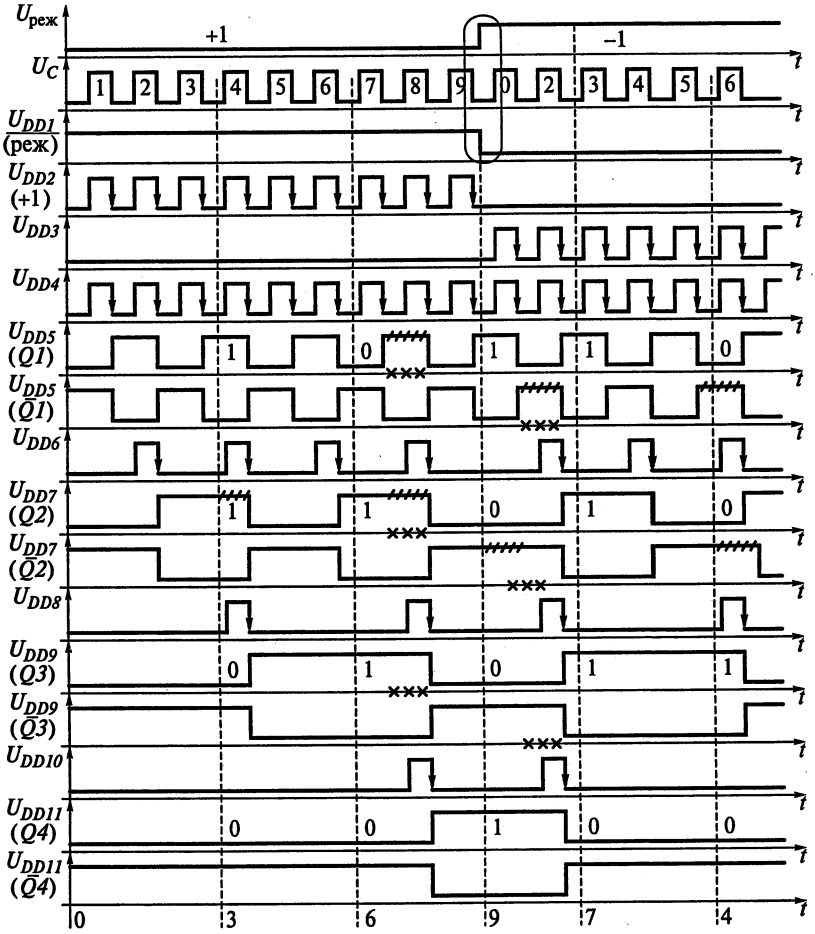


Рис. 15.8

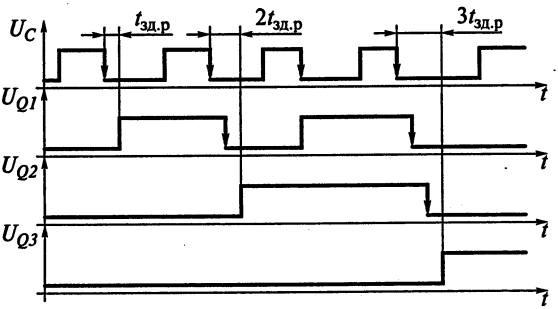


Рис. 15.9

Асинхронный реверсивный счетчик. В таких счетчиках предусмотрена возможность работы в двух режимах: сложения и вычитания.

На рис. 15.7 представлена схема реверсивного счетчика, а на рис. 15.8 — диаграмма его работы. Сигналы с выходов Q и $\bar{Q}$ триггеров управляют прохождением импульсов (+1 или –1) счета от двух источников в зависимости от режима работы.

Управление режимом работы счетчика осуществляется сигналами, поданными на вход «Режим». При сложении на вход подается лог. 0, а при вычитании — лог. 1. Именно эти сигналы управляют прохождением импульсов счета через входные ключи *DD2* или *DD3*, а затем через *DD6*, *DD8*, *DD10* в режимах сложения или вычитания. При этом прохождение импульсов счета в более старшие разряды будет зависеть от сочетания управляющих сигналов Q или $\bar{Q}$ младших разрядов счетчика. Так, например, при сложении импульс счета поступит на вход триггера третьего разряда только при условии, что перед этим триггеры первого и второго разрядов будут находиться в единичном состоянии (число $3_{(10)} — 11_{(2)}$). Аналогично при вычитании импульс счета поступит на вход триггера третьего разряда, только когда триггеры первого и второго разрядов будут находиться в нулевом состоянии.

Основным преимуществом асинхронных счетчиков является простота построения, а основным недостатком — сравнительно низкое быстродействие, которое обусловлено накапливаемой задержкой распространения импульсов счета от разряда к разряду (рис. 15.9).

Основной характеристикой счетчиков является максимальное значение частоты следования импульсов счета:

$$f_{\max} = \frac{1}{nt_{\text{зд. п}} + t_{\text{с}}},$$

где n — число разрядов; $t_{\text{зд. п}}$ — задержка переключения триггера одного разряда; $t_{\text{с}}$ — реакция выходных ключей, с помощью которых ведется опрос счетчика.

Необходимо иметь в виду, что при высокой частоте следования импульсов счета возможно получение ложного результата.

Контрольные вопросы

1. Каким должно быть соотношение между разрядностью счетчика с естественным порядком счета и коэффициентом счета?
2. С каких выводов счетчика снимается информация о количестве подсчитанных импульсов?
3. В какие моменты времени относительно импульсов счета происходит изменение состояния триггеров в счетчике прямого счета?
4. В чем заключается различие в построении счетчиков прямого и обратного счета?
5. В какие моменты времени происходит изменение состояния триггеров первого и остальных разрядов в счетчике обратного счета?

6. В каком режиме работают *T*-триггеры счетчика? Как он обеспечивается?

7. Какие виды триггеров можно использовать при построении счетчиков?

8. Почему в схеме счетчика в интегральном исполнении ИЕ5 на входы *J* и *K* не подаются никакие сигналы? Какому состоянию логического элемента И это соответствует?

9. В чем заключается реверсивный режим работы счетчика и как он обеспечивается?

10. Каков основной недостаток асинхронных счетчиков?

15.3. Синхронные счетчики

По сравнению с асинхронными счетчиками синхронные обладают бóльшим быстродействием. В синхронных счетчиках импульсы счета поступают одновременно на триггеры всех разрядов, но реакция триггеров на них будет разная. При построении таких счетчиков анализируются случаи возможного изменения состояния триггеров старших разрядов в зависимости от состояния младших разрядов. Так, триггер третьего разряда может изменить свое состояние с приходом четвертого импульса счета, только когда перед его приходом триггеры первого и второго разрядов находятся в единичном состоянии (число $3 - 11_{(2)}$). Состояние триггера четвертого разряда меняется с приходом восьмого импульса счета, когда перед его приходом триггеры первого, второго, третьего разрядов находятся в единичном состоянии (число $7 - 111_{(2)}$). К этим моментам времени триггеры третьего и четвертого разрядов необходимо подготовить для работы в счетном режиме, в остальных случаях они должны находиться в режиме хранения. Для этой цели в схему счетчика вводятся дополнительные логические элементы И, на которых отслеживается момент совпадения единиц соответствующих разрядов. Выходной сигнал с них используется для организации счетного режима и режима хранения для *T*-триггера.

Счетчик со сквозным переносом. Схема такого счетчика представлена на рис. 15.10, а диаграмма его работы — на рис. 15.11.

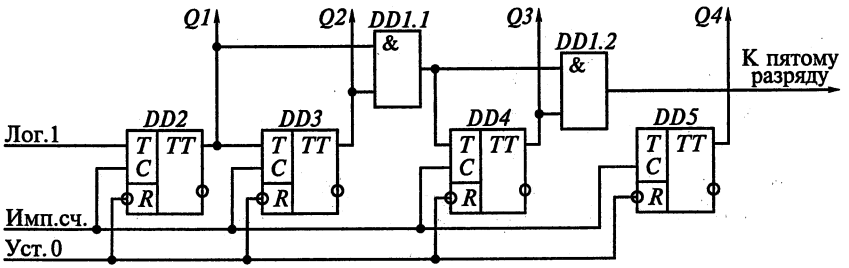


Рис. 15.10

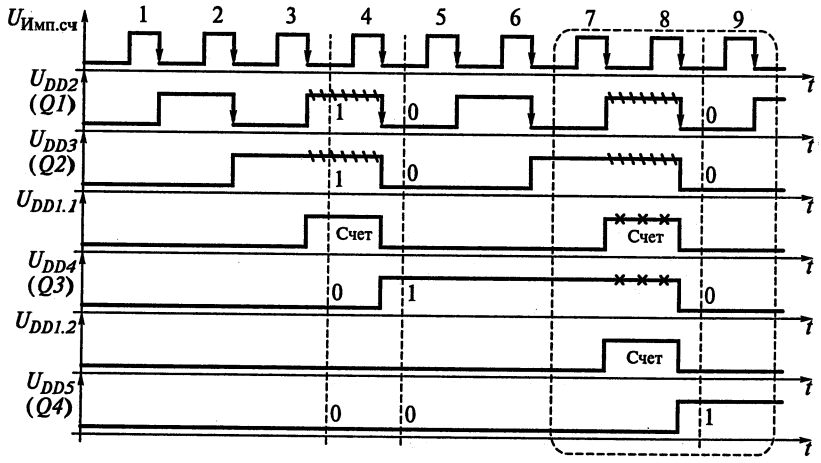


Рис. 15.11

Схема состоит из триггеров и дополнительных логических элементов И (*DD1.1* и *DD1.2*), которые определяют режимы работы *T*-триггеров третьего и четвертого разрядов. С их помощью осуществляется дешифрация первого и второго разрядов; первого, второго с третьим разрядов.

Триггер первого разряда постоянно работает в счетном режиме. Состояние триггера второго разряда зависит от значения сигнала на выходе *Q* триггера первого разряда к моменту прихода импульса счета. Состояние триггеров третьего и четвертого разрядов определяется логическими элементами И.

На рис. 15.12 представлен фрагмент диаграммы работы данного счетчика с учетом временных задержек на логических элементах И и задержек переключения триггеров.

Из графика видно, что формирование управляющих сигналов на выходе схем И идет с нарастанием задержки распространения переноса от одного разряда к другому. Поэтому при увеличении разрядности счетчика эта задержка может превысить время поступления очередного импульса счета. К этому моменту времени определенные входы *T*-триггеров должны быть уже подготовлены к счетному режиму работы. Следовательно, должно выполняться условие

$$f_{max} = \frac{1}{(n-2)t_{л.э} + t_{зд.\,п}},$$

где f_{max} — максимальная частота поступления импульсов счета; n — количество разрядов счетчика; $t_{л.э}$ — задержка реакции логического элемента И; $t_{зд.\,п}$ — задержка переключения триггера первого разряда.

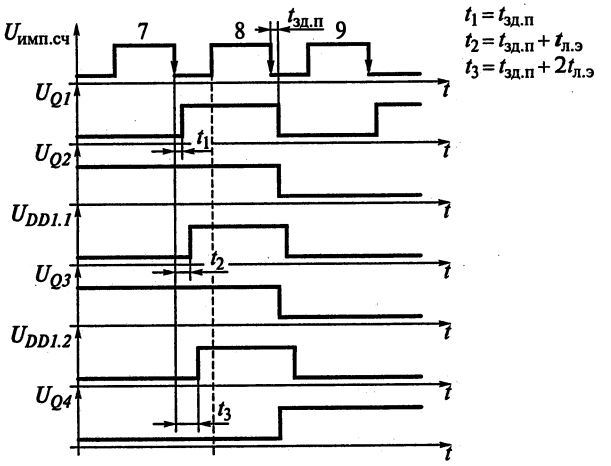


Рис. 15.12

Из диаграммы видно, что каждый последующий логический элемент отслеживает задержки, накопленные предыдущими логическими элементами. Соответственно с увеличением разрядности числа эта величина может быть значительной. Снизить накопление задержек удалось в схеме счетчика с параллельным переносом.

Счетчик с параллельным переносом. Схема этого счетчика представлена на рис. 15.13, а диаграмма его работы — на рис. 15.14. В отличие от счетчика со сквозным переносом в рассматриваемой схеме на логических элементах И (*DD1*, *DD2*, *DD3*) осуществляется дешифрация — выявление возможного состояния триггера старшего разряда путем анализа всех младших разрядов. На входы этих схем соответственно подаются сигналы с выходов: первого, второго разрядов; первого, второго, третьего разрядов; первого, вто-

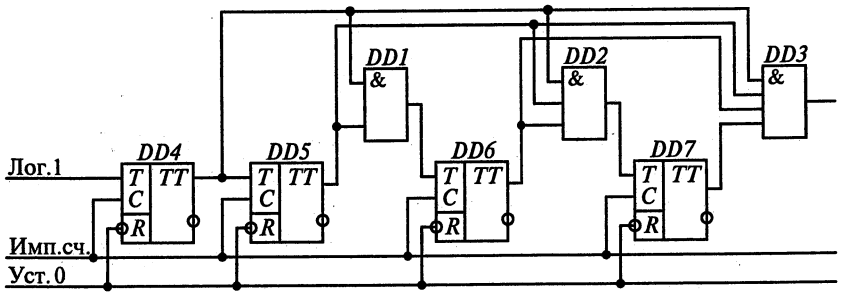


Рис. 15.13

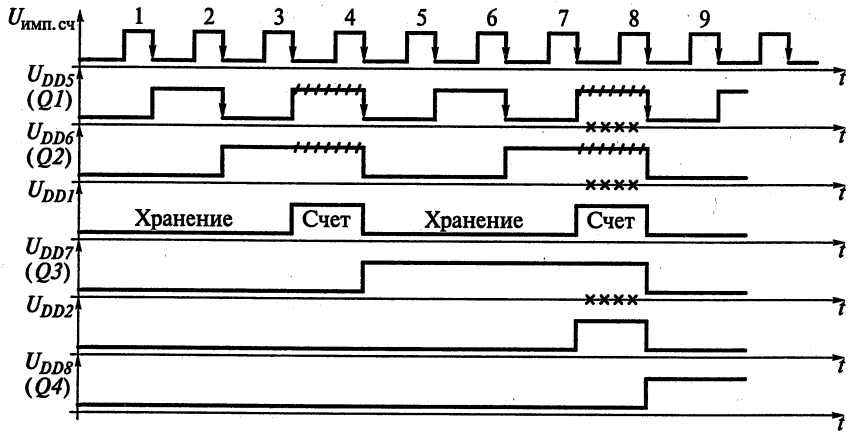


Рис. 15.14

рого, третьего и четвертого разрядов. Из-за отсутствия межразрядных связей между логическими элементами удается снизить задержки реакции логических элементов. В этом случае максимальная частота поступления импульсов счета определяется по формуле

$$f_{max} = \frac{1}{t_{л.э} + t_{зд.\,п}}.$$

Схема счетчика с параллельным переносом может быть упрощена, если вместо отдельных логических элементов И воспользоваться дополнительной логикой *JK*-триггеров. Особенностью *JK*-триггеров в схеме на рис. 5.15 является наличие у них нескольких (1...3) входов *J* и *K*, связанных операцией И.

На рис. 15.16 приведена диаграмма работы такого счетчика. Логические элементы *DD5* и *DD6* в нем используются для формирования внешних, межсхемных переносов.

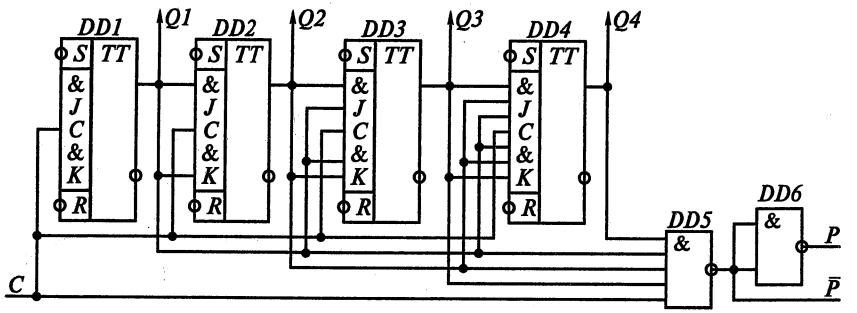


Рис. 15.15

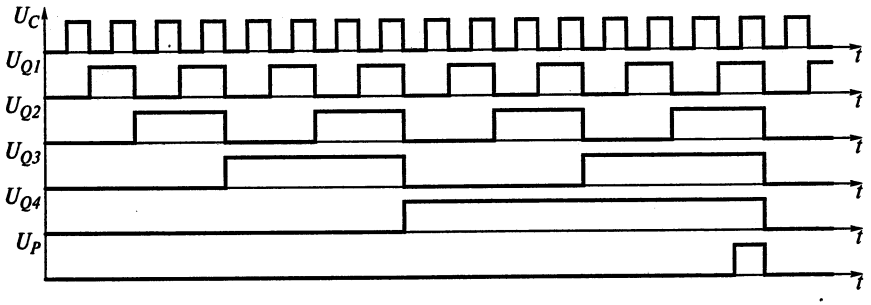


Рис. 15.16

Счетчик с групповым переносом. В рассмотренной схеме счетчика с параллельным переносом на основе *JK*-триггера число входов *J* и *K* триггера ограничено, а следовательно, число разрядов, образующих группу разрядов, такого счетчика также ограничено. Поэтому при создании счетчиков на большее количество разрядов используют дополнительные логические элементы для формирования межгрупповых переносов (*P*). На рис. 15.15 это логические элементы *DD5* и *DD6*. Если такие вспомогательные схемы находятся внутри микросхемы, то дополнительные внешние логические схемы не применяются.

На рис. 15.17 приведен пример организации счетчика с групповым переносом, в котором перенос внутри микросхемы формируется параллельно для всех разрядов, а между микросхемами — последовательно.

Реверсивный счетчик с параллельным переносом. На рис. 15.18 показан пример построения синхронного реверсивного счетчика с параллельным переносом.

Принцип организации реверсивного режима осуществляется с использованием ключей И-ИЛИ-НЕ, с помощью которых на триг-

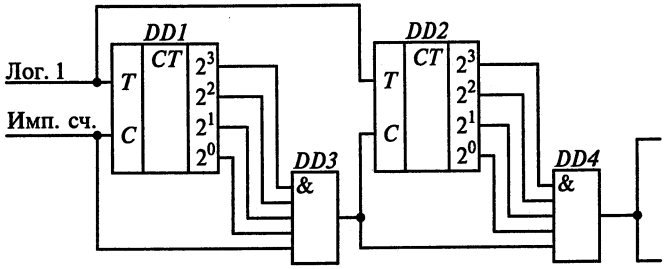


Рис. 15.17

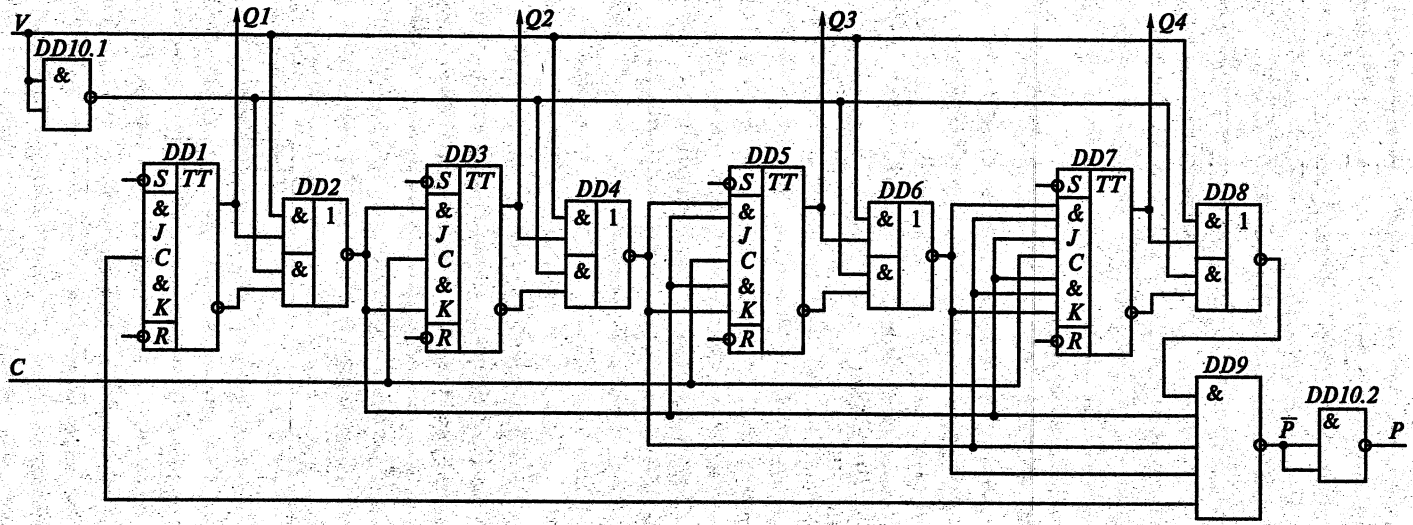

V
DD10.1
&
Q1
Q2
Q3
Q4
DD1
S
TT
&
J
C
&
K
R
DD2
&
1
&
DD3
DD4
DD5
DD6
DD7
DD8
DD9
DD10.2
C
P̄
P

Рис. 15.18

геры старших разрядов поступают сигналы с выходов Q или $\overline{Q}$ всех младших разрядов. Значение сигнала на входе V («Режим») управляет работой этих ключей.

На рис. 15.19 приведен фрагмент диаграммы работы этого счетчика в реверсивном режиме для триггеров первого и второго разрядов.

Схема счетчика в интегральном исполнении. На рис. 15.20 представлена довольно сложная схема счетчика ИЕ7. Остановимся на возможностях схемы. Режим работы будет выбран в зависимости от того, на какой из входов (+1) или (–1) подаются импульсы счета.

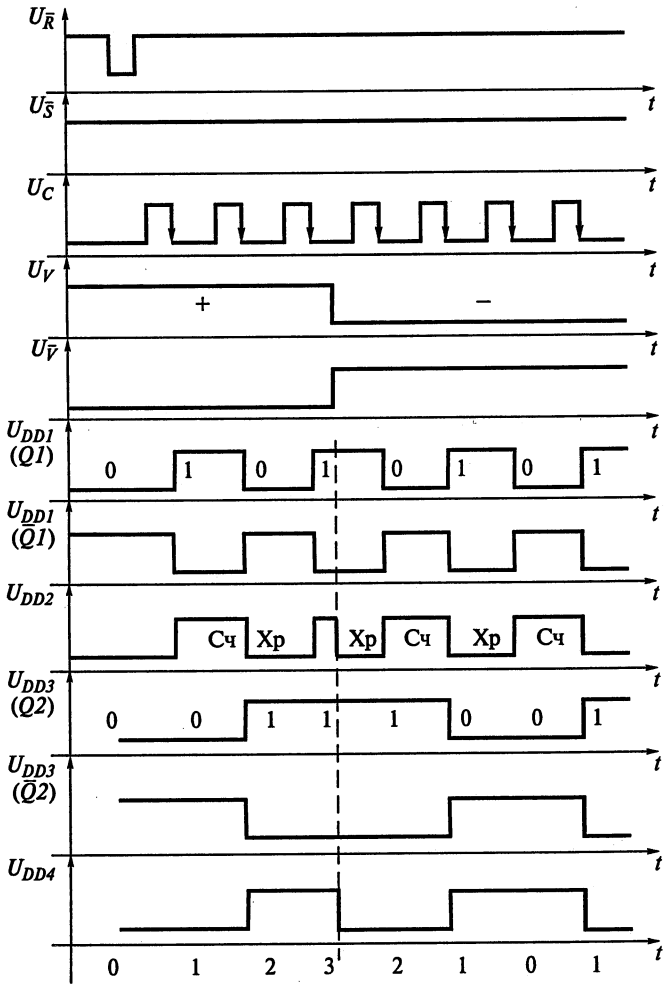


Рис. 15.19

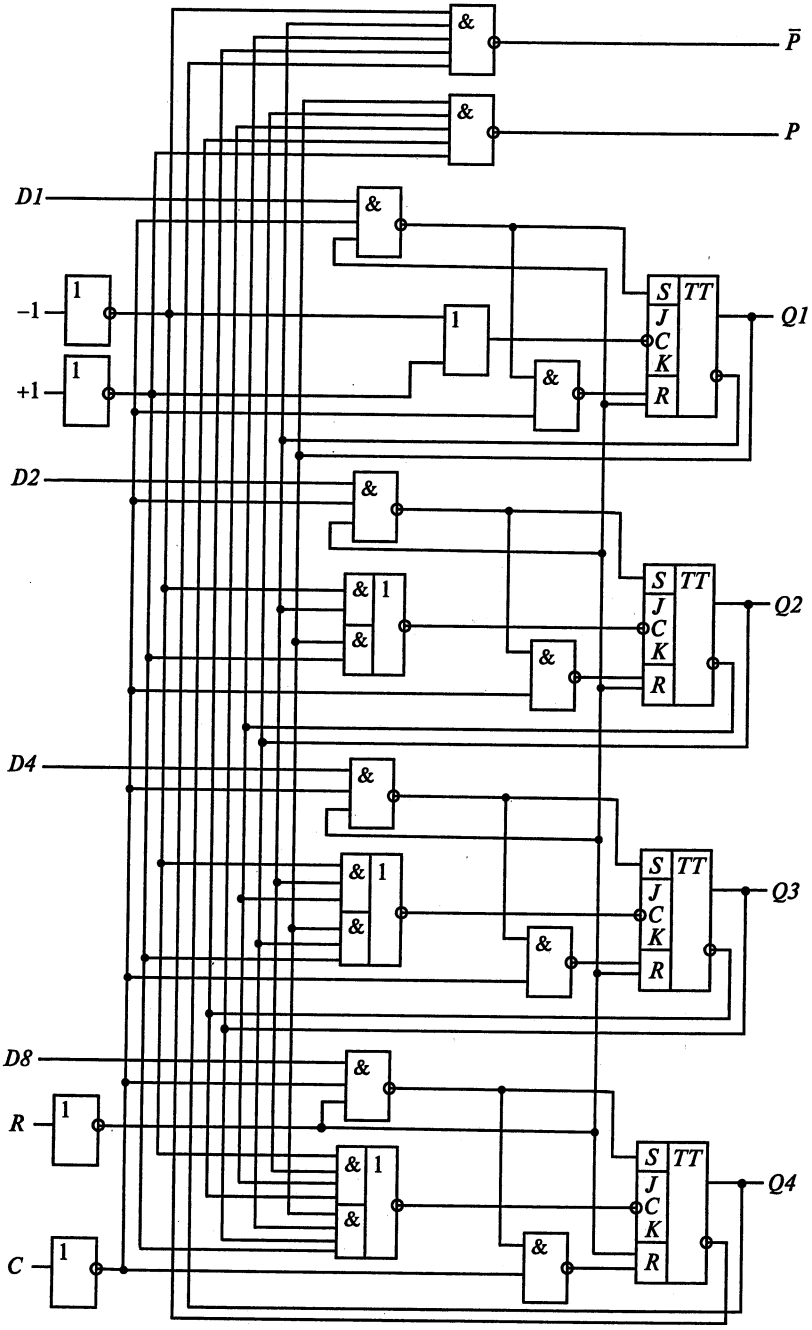


Рис. 15.20

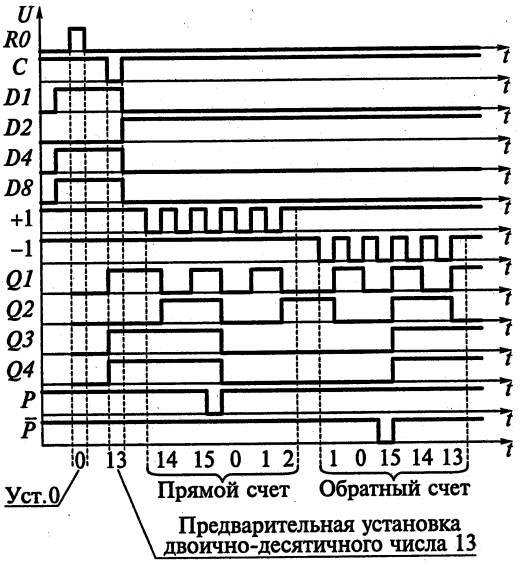


Рис. 15.21

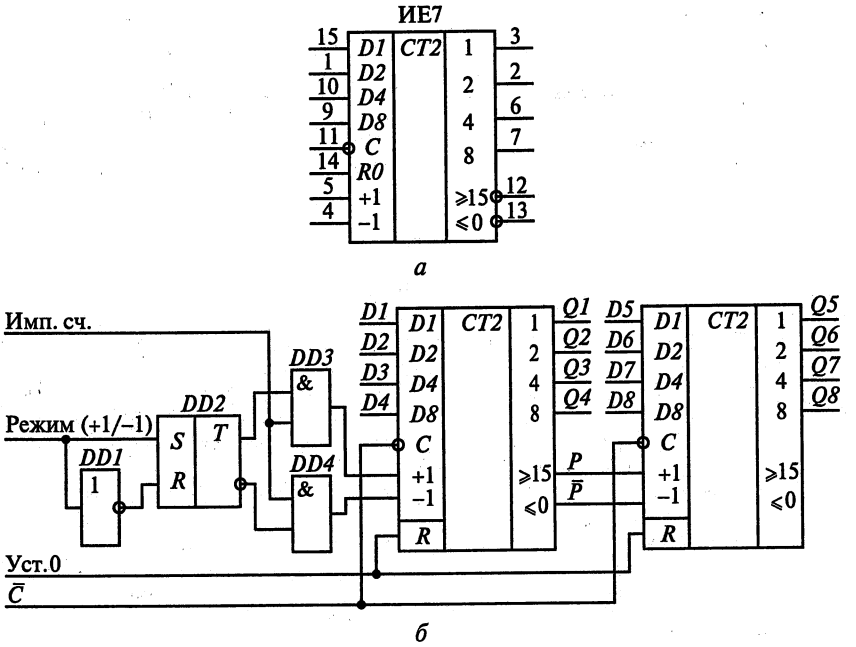


Рис. 15.22

Предварительная установка триггеров счетчика может быть различной. Установка триггеров в нулевое состояние производится по входу *R*. Кроме того, возможна предварительная запись любого числа, ограниченного разрядностью счетчика. Для этой цели используются информационные входы *D1*, *D2*, *D4*, *D8*. В любом режиме предварительной записи необходимо наличие инверсного сигнала $\overline{C}$. Активные парафазные сигналы проходят через схемы 3И-НЕ и 2И-НЕ на входы *R* и *S* *JK*-триггеров.

Реверсивный режим работы счетчика обеспечивается логическими ключами И-ИЛИ-НЕ, на каждый из которых приходят сигналы с выходов Q или $\overline{Q}$ младших разрядов и соответствующие режиму импульсы счета (+1) или (–1). Для построения счетчика с большей разрядностью используются выходы прямого и обратного переноса. С выхода прямого переноса импульсы переноса подаются на вход прямого счета (+1), а импульсы обратного переноса подаются на вход обратного счета (–1) следующей микросхемы. На рис. 15.21 представлена диаграмма работы рассматриваемого счетчика в режимах прямого и обратного счета с предварительной записью числа 13.

На рис. 15.22, *а* дано условное графическое обозначение, а на рис. 15.22, *б* показан пример организации многоразрядного счетчика, предназначенного для работы в реверсивном режиме.

В условном графическом обозначении показаны выходы «≥15», «≤0» для организации счетчиков большей разрядности. При построении счетчиков прямого или обратного счета необходимо объединить выходы «≥15» и «≤0» с входами (+1) и (–1) соответственно.

Контрольные вопросы

1. Пояснить отличие в организации асинхронного и синхронного счетчиков.
2. Чем определяется быстродействие асинхронного и синхронного счетчиков?
3. С какой целью на выходе счетчиков устанавливаются ключи?
4. Какая существует взаимосвязь между частотой поступления импульсов счета и временем поступления импульсов опроса на выходные ключи в синхронных счетчиках?
5. В каких режимах работают триггеры первого и второго разрядов в синхронных триггерах?
6. Какая существует взаимосвязь между задержкой распространения переноса и временем поступления сигнала «Опрос» на выходные ключи в счетчике со сквозным переносом?
7. За счет чего удается повысить быстродействие в синхронных счетчиках с параллельным переносом? Как это влияет на время поступления сигнала «Опрос»?
8. Проанализировать варианты возможного получения ложной информации на выходе синхронных счетчиков со сквозным и параллельным переносом.

9. Какие факторы приводят к необходимости построения счетчиков с последовательными межгрупповыми связями?

10. Какие существуют режимы работы счетчика в реверсивном режиме?

11. Каково функциональное назначение межразрядных ключей И-ИЛИ в синхронном реверсивном счетчике?

12. Пояснить функциональное различие работы межразрядных ключей в асинхронных и синхронных счетчиках.

13. Каково назначение выходов счетчика «≥15» и «≤0»? Какими буквенными символами их можно обозначить?

14. Каким образом в интегральной микросхеме ИЕ7 осуществляется предварительная установка триггеров в нулевое состояние или предварительная запись числа?

15. Какие входы микросхемы ИЕ7 используются для подачи импульсов счета в режимах прямого и обратного счета?

16. Каковы функции дополнительных элементов *DD1*... *DD4* при организации работы многоразрядного счетчика (см. рис. 15.22)?

15.4. Счетчики с произвольным коэффициентом пересчета

В ранее рассмотренных схемах счетчиков коэффициент пересчета K был кратен 2^n, где n — количество разрядов счетчика. В данном подразделе рассматриваются схемы счетчиков, в которых может производиться подсчет произвольного количества единиц информации, не кратный 2^n. На рис. 15.23 приведена классификация счетчиков с произвольным коэффициентом пересчета.

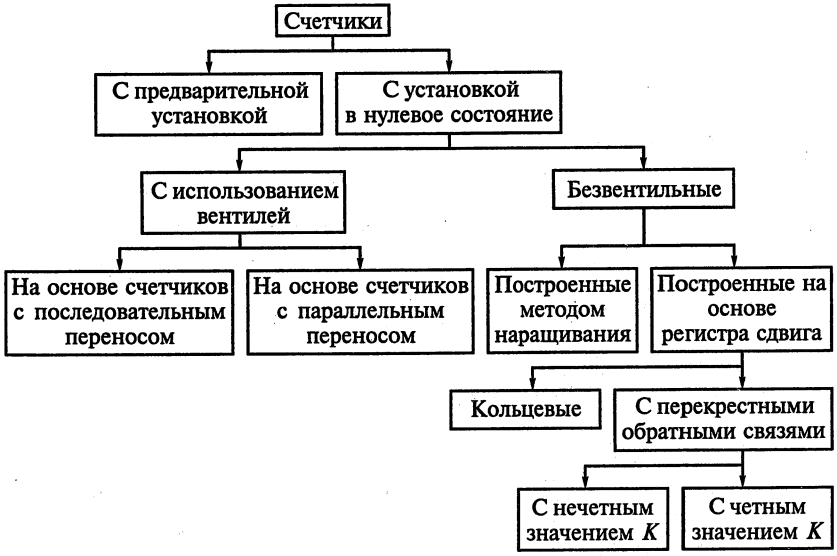


Рис. 15.23

В основном схемы счетчиков строятся таким образом, что по достижении требуемого коэффициента пересчета их триггеры устанавливаются в нулевое состояние. Это может быть достигнуто различными способами. Вопрос состоит в том, как и в какой момент времени исключить лишние состояния в счетчике. Исключение можно выполнить перед началом счета или по его окончанию. В первом случае в исходном состоянии в счетчике формируется число, соответствующее избытку, после чего осуществляется последовательный естественный счет. Во втором случае исключаются последние избыточные состояния, т.е. при определенном $K \neq 2^n$ счетчик устанавливается в нулевое состояние. Такой способ позволяет сохранить естественный порядок счета, начиная с нулевого до необходимого.

Счетчики с произвольным коэффициентом пересчета могут быть построены на основе ранее рассмотренных схем счетчиков с использованием различных способов. Счетчики, построенные на базе синхронных счетчиков со сквозным и параллельным переносом, сохраняют естественный порядок счета, в остальных случаях порядок счета нарушается. Исключение составляют счетчики, построенные методом наращивания на основе синхронного счетчика прямого счета с малым K.

Рассмотрим счетчики на основе регистра сдвига.

Кольцевые счетчики. Простейшим счетчиком на основе регистра сдвига является кольцевой, у которого выходы триггера старшего разряда соединены с входами триггера младшего разряда, образуя кольцо. При этом сдвигаемое в кольце число содержит всего одну единицу. Коэффициент пересчета равен количеству используемых в нем триггеров. Так, для построения декадного счетчика ($K = 10$) необходимо 10 триггеров. Схема строится таким образом, что продвижение единицы можно проследить по выходам Q счетчика. Номер выхода показывает, какой по счету импульс счета поступил на вход. На рис. 15.24 приведена схема счетчика с $K = 5$,

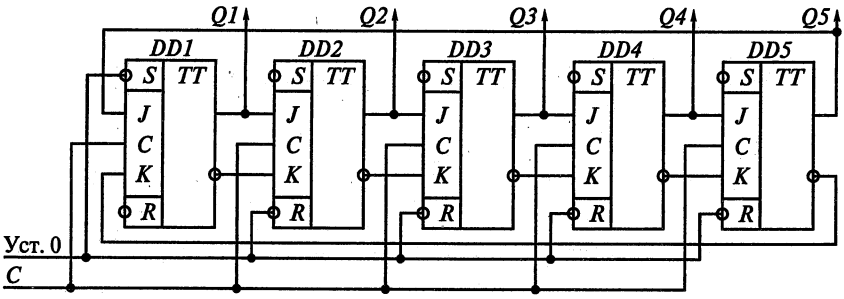


Рис. 15.24

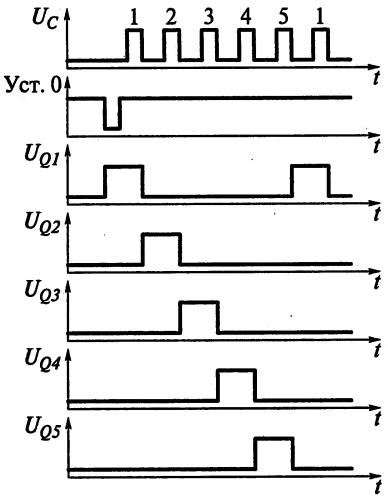


Рис. 15.25

а на рис. 15.25 — диаграмма работы такого счетчика. Схема представляет собой пятиразрядный регистр на *JK*-триггерах. Перед началом счета триггер *DD1* устанавливается в состояние, соответствующее лог. 1, а триггеры *DD2*... *DD5* — в нулевое состояние с помощью сигнала «Установка 0». Далее с приходом импульсов счета лог. 1 продвигается из разряда в разряд.

Кольцевой счетчик очень прост в построении, но не нашел широкого применения из-за большого количества триггеров и сложности обнаружения ошибки в случае сбоя в схеме.

Счетчики с перекрестными обратными связями (безвентильные счетчики Джонсона). В зависимости от организации обратных связей такие счетчики могут иметь как четный, так и нечетный коэффициент счета.

Счетчик Джонсона, имеющий четный коэффициент счета, представляет собой кольцевой регистр сдвига с перекрестными обратными связями, т.е. связями с выходов Q и $\overline{Q}$ триггера старшего разряда на информационные входы K и J (или R и S) триггера младшего разряда соответственно. В основе его работы лежит последовательный сдвиг циркулирующей в регистре информации. Количество триггеров n, используемое в схеме, равно $K/2$. Следовательно, можно организовать счетчики с K, равным 4, 6, 8, 10 и т.д. На рис. 15.26 приведена схема счетчика Джонсона с $K = 6$, а на рис. 15.27 — диаграмма его работы. Схема состоит из трех ($n = K/2$) триггеров, соответствующих трем разрядам регистра сдвига. Обратные связи заводятся с выходов Q и $\overline{Q}$ старшего разряда на входы R и S триггера младшего разряда соответственно. После установки всех триггеров в

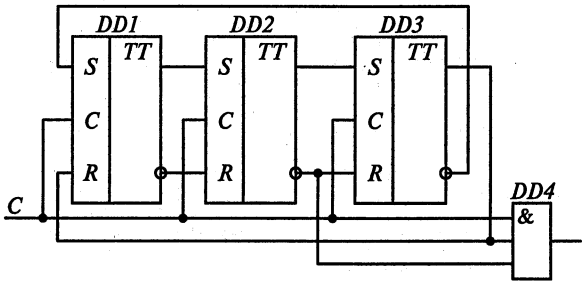


Рис. 15.26

нулевое состояние на входе триггера первого разряда появится лог. 1, которая с каждым импульсом счета будет передаваться следующему триггеру до заполнения всех разрядов единицами. Далее за счет обратных связей в следующем цикле пойдет волна нулей и т.д. Наглядно это видно из таблицы истинности (табл. 15.1). По достижении требуемого значения K с выходов триггеров всех разрядов снимаются нули.

Схема *DD4* используется для выделения каждого шестого импульса. Для организации связей с этой схемой анализируется состояние триггеров перед приходом шестого импульса и с выходов, на-

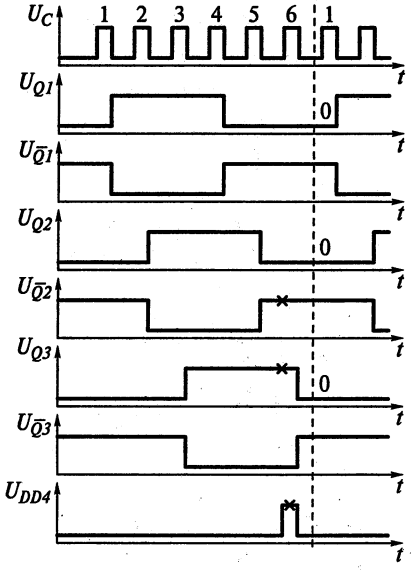


Рис. 15.27

Таблица 15.1

Состояние счетчика	*Q1*	*Q2*	*Q3*
0	0	0	0
1	1	0	0
2	1	1	0
3	1	1	1
4	0	1	1
5	0	0	1

ходящихся в единичном состоянии, заводятся управляющие сигналы на вход логического элемента, обеспечивая тем самым прохождение нужного импульса.

В счетчиках Джонсона с нечетным коэффициентом счета обратные связи заводятся с выходов $\overline{Q}$ триггера старшего разряда и с выхода Q триггера предпоследнего разряда. В этом случае распространение по счетчику лог. 0 начнется после прихода единицы на предпоследний триггер, т.е. на один такт раньше. Коэффициент счета данного счетчика вычисляется как $K = 2(n - 1)$. Используя этот способ построения можно организовать подсчет 3, 5, 7 импульсов и т.д. На рис. 15.28 приведен пример организации подобного счетчика с $K = 5$, а на рис. 15.29 — диаграмма, поясняющая работу схемы. Число триггеров должно обеспечивать количество состояний, которое больше значения $K = 5$. В данном случае используются три триггера, обеспечивающих шесть состояний ($2 \cdot 3 = 6$). Исключение лишнего состояния происходит за счет несимметричной обратной связи, в результате чего перед третьим импульсом счета триггер первого разряда переходит в счетный режим. Таким образом, после третьего импульса ситуация меняется (вместо наличия во всех разрядах единиц в первом разряде появляется нуль), подготавливая на один такт раньше волну нулей (соответственно в следующей нулевой волне на один такт раньше будет подготовлена волна единиц).

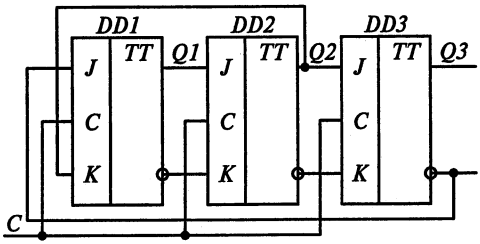


Рис. 15.28

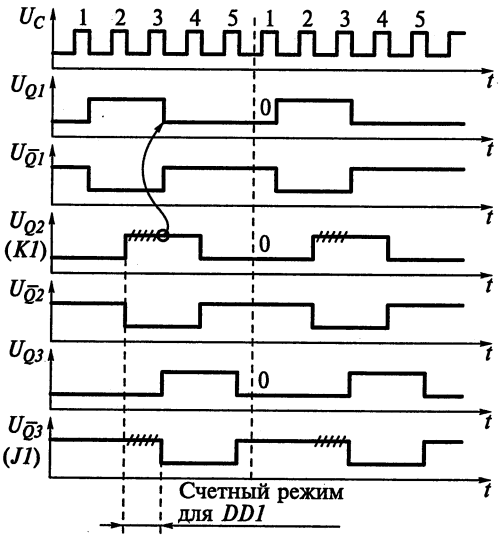


Рис. 15.29

Счетчик на основе регистра с использованием вентилей. Принцип организации сдвига информации, рассмотренный ранее, можно реализовать с использованием вентилей. Пример такого счетчика с $K = 5$ приведен на рис. 15.30, а на рис. 15.31 показана диаграмма, поясняющая работу схемы. Как и в предыдущей схеме, распространение нулей здесь начинается на один такт раньше. Но в данном случае обратные связи заводятся с триггеров двух старших разрядов ($\bar{Q}2$ и $\bar{Q}3$) на дополнительный элемент 2И-НЕ (*DD1*), сигнал с которого подается на вход K младшего разряда и через инвертор на вход J этого же разряда.

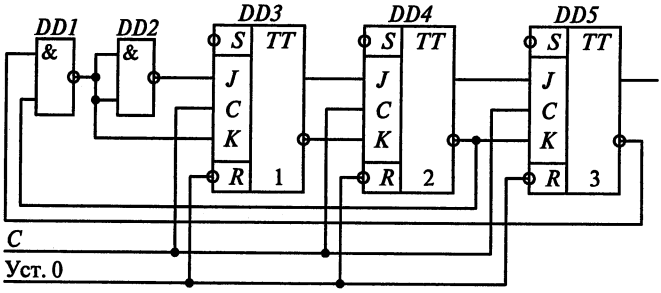


Рис. 15.30

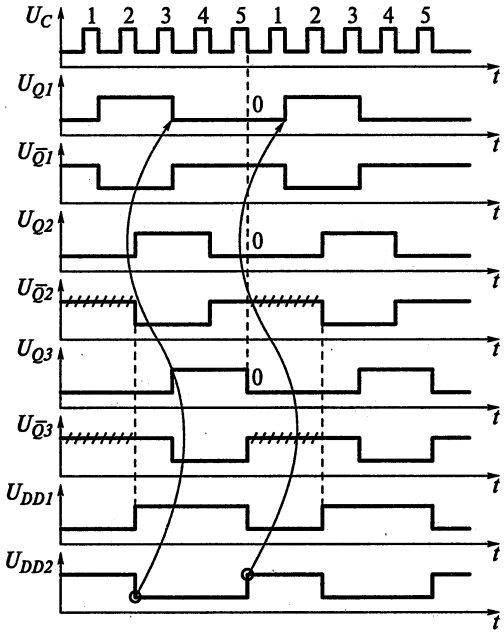


Рис. 15.31

Схемы на основе счетчиков с последовательным переносом в интегральном исполнении. Широкое применение на практике находят схемы, построенные на основе счетчиков с последовательным переносом, в которых введена обратная связь для исключения лишних состояний. Такой счетчик работает в режиме прямого или обратного счета до некоторого состояния, задаваемого коэффициентом K, которое дешифрируется (выделяется) на логическом элементе И, о чем свидетельствует появление сигнала на его выходе, а затем с помощью обратных связей подается на входы $\overline{R}$ триггеров счетчиков. На рис. 15.32 представлена схема организации счетчика с $K = 10$ на основе рассмотренной ранее интегральной схемы ИЕ5, в которой уже предусмотрено наличие логического элемента И. На рис. 15.33 представлена диаграмма работы такого счетчика.

На рис. 15.34 даны условные графические обозначения трех видов счетчиков, а в табл. 15.2 приведены варианты объединения выводов их микросхем для получения требуемого коэффициента пересчета.

Схемы на основе счетчиков с параллельным переносом. В таких счетчиках перед приходом последнего импульса счета, соответствующего заданному коэффициенту счета, выявляются разряды триггеров, которые должны были бы перейти в единичное состояние. Именно на входы J этих триггеров заводится

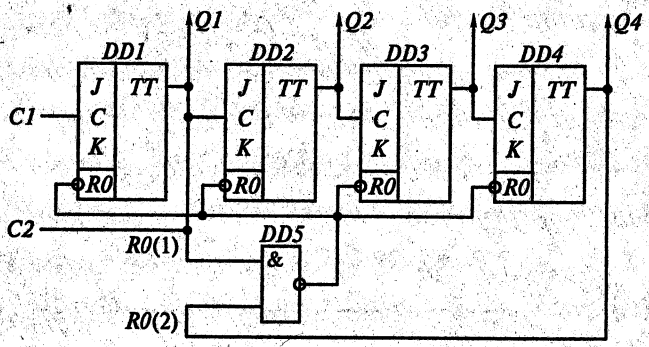


Рис. 15.32

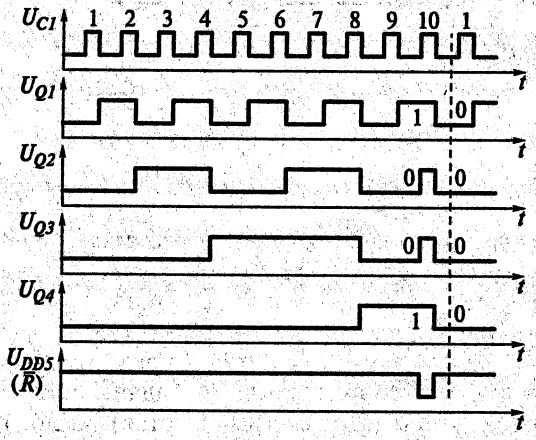


Рис. 15.33

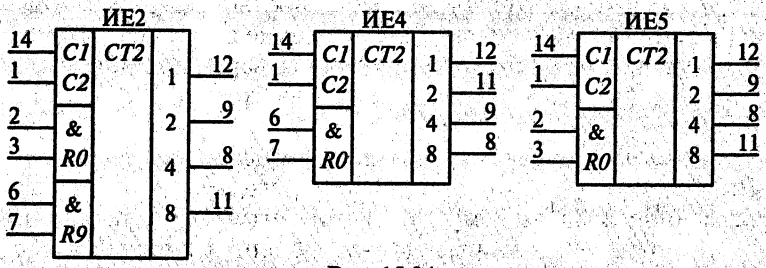


Рис. 15.34

Таблица 15.2

K	ИЕ2			ИЕ4			ИЕ5		
	Вход	Выход	Соединить выводы	Вход	Выход	Соединить выводы	Вход	Выход	Соединить выводы
2	14	12	—	14	12	—	14	12	—
3	1	8	9—2; 8—3	1	9	—	1	8	9—2; 8—3
4	1	8	11—2—3	1	8	11—6; 8—7	1	8	—
5	1	11	—	1	8	9—6; 8—7	1	11	9—2; 11—3
6	14	8	12—1; 9—2; 8—3	1	8	—	1	11	8—2; 11—3
7	14	11	12—1; 9—6; 8—7	14	8	12—1—6; 8—7	—	—	—
8	14	8	12—1; 11—2—3	14	8	12—1; 11—6; 8—7	1	11	—
9	14	11	12—1—2; 11—3	—	—	—	14	11	12—1—2; 11—3
10	14	11	12—1	14	8	12—1; 9—6; 8—7	14	11	12—1; 9—2; 11—3
12	—	—	—	14	8	12—1	14	11	12—1; 8—2; 11—3
16	—	—	—	—	—	—	14	11	12—1

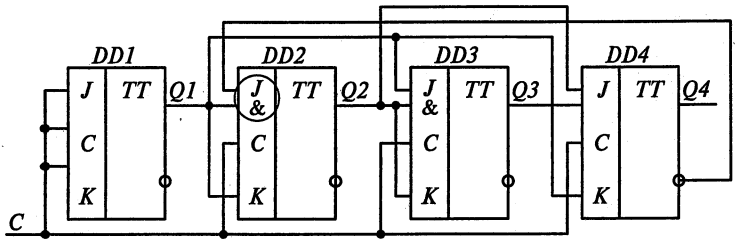


Рис. 15.35

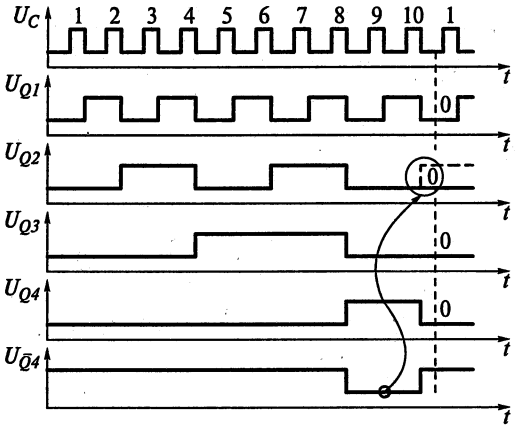


Рис. 15.36

обратная связь и с выхода $\overline{Q}$ старшего разряда. Таким образом осуществляется подготовка их к приходу последнего импульса счета, в результате которого все триггеры счетчика устанавливаются в нулевое состояние, что свидетельствует о достижении требуемого коэффициента счета. После этого подсчет импульсов возобновляется.

На рис. 15.35 приведен пример организации счетчика с $K = 10$, а на рис. 15.36 — диаграмма работы такой схемы.

Безвентильные счетчики, построенные методом наращивания. Среди безвентильных счетчиков нашли широкое применение наращиваемые счетчики, строящиеся на основе базовой схемы, с помощью которой формируется коэффициент счета:

$$K = 2^n + 1.$$

Эта схема может быть использована также в качестве самостоятельного счетчика, коэффициент пересчета которого отличается от $K = 2^n$ на единицу.

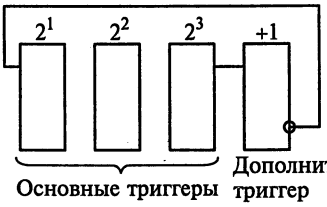


Рис. 15.37

Построение таких схем подчиняется следующим правилам.

Схема состоит из основного счетчика с $K = 2^n$, на выходе которого устанавливается дополнительный триггер, с помощью которого осуществляется добавление единицы (рис. 15.37).

Основной счетчик можно организовать как по схеме асинхронного счетчика сложения, так и по схеме асинхронного счетчика вычитания. В первом случае естественный порядок счета сохраняется, но на таких схемах можно реализовать лишь счетчики с малым коэффициентом пересчета. Для второго случая таких ограничений нет, но при этом порядок счета будет непоследовательным.

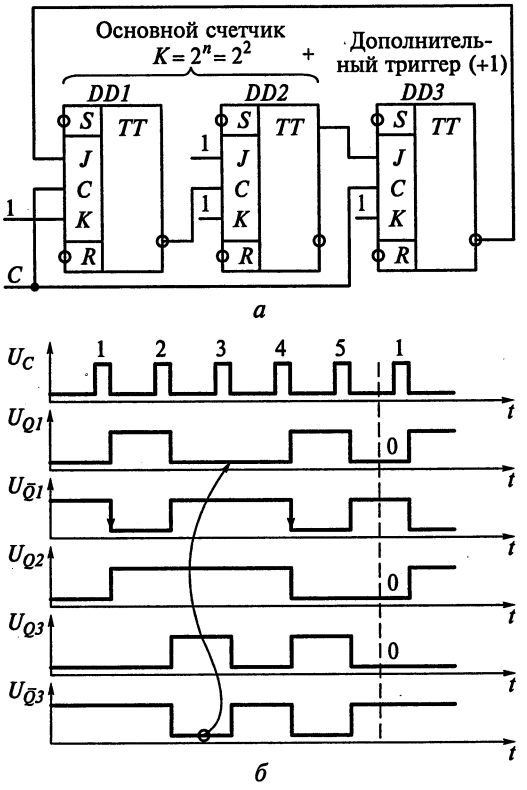


Рис. 15.38

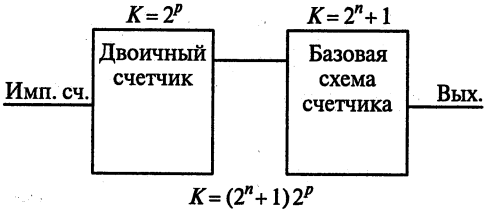


Рис. 15.39

Дополнительный триггер подсоединяется к выходу Q триггера старшего разряда основного счетчика.

С выхода $\overline{Q}$ дополнительного триггера заводится обратная связь на вход J триггера первого разряда.

О достижении необходимого коэффициента судят по одновременному нулевому состоянию всех триггеров.

На рис. 15.38 представлен пример построения счетчиков для $K = 5$.

Для случаев, когда коэффициент счета K не соответствует формуле $(2^n + 1)$, например для чисел 6, 7, 10, 11, 12, 13, 14, 15 и т.д., организация счетчиков сводится к наращиванию базовой схемы с $K = 2^n + 1$. В этом случае коэффициент пересчета определяется по одной из следующих формул:

$$K = (2^n + 1)2^p, \quad \text{для } K = 6,\ 10,\ 12; \tag{15.1}$$

$$K = (2^n + 1)2^p + 1, \quad \text{для } K = 7,\ 11,\ 13; \tag{15.2}$$

$$K = [(2^n + 1)2^p + 1]2^m, \quad \text{для } K = 14,\ 22; \tag{15.3}$$

$$K = [(2^n + 1)2^p + 1]2^m + 1, \quad \text{для } K = 15,\ 23, \tag{15.4}$$

где n — степень для базовой схемы; m, p — степени для наращиваемых схем.

Для счетчиков, организуемых в соответствии с формулой (15.1), наращивание схемы достигается путем добавления к ней двоичного счетчика с $K = 2^p$, где p — число триггеров в схеме с последовательным переносом для режима вычитания (рис. 15.39).

Для малых значений коэффициентов пересчета счетчик можно реализовать по схеме для режима сложения, при этом сохраняется последовательный порядок счета. Рассмотрим организацию счетчика для $K = 6$:

$$K = (2^n + 1)2^p = (2^1 + 1)2^1 = 6.$$

На рис. 15.40, *а*, *б* представлены соответственно схема и диаграмма работы этого счетчика. Такой счетчик представляет собой соединение одного T-триггера *DD1* ($K = 2$) и самостоятельного счетчика ($K = 3$), выполненного на базе элементов *DD2*, *DD3*.

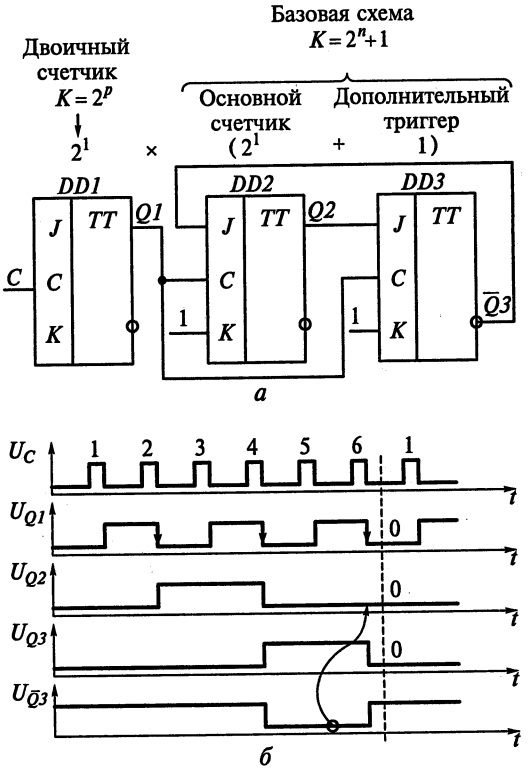


Рис. 15.40

На рис. 15.41, *а*, *б* представлены аналогично схема и диаграмма работы счетчика для режима вычитания.

Рассмотрим организацию счетчика в соответствии с формулой (15.2), упрощенная структура которого показана на рис. 15.42. Первые два фрагмента схемы представляют собой предыдущую схему с дополнительным триггером на выходе, с помощью которого достигается нечетное значение коэффициента счета путем добавления 1 (+ 1).

Таким образом, если к схеме с $K = 6$ подключить дополнительный триггер *DD4*, осуществляющий добавление 1, то можно организовать счетчик ($K = 7$):

$$K = (2^n + 1)2^p + 1 = (2 + 1)2 + 1.$$

На рис. 15.43, *а*, *б* представлены соответственно схема и диаграмма работы такого счетчика.

Построение счетчиков с бóльшим коэффициентом счета представляет собой дальнейшее наращивание рассмотренной схемы.

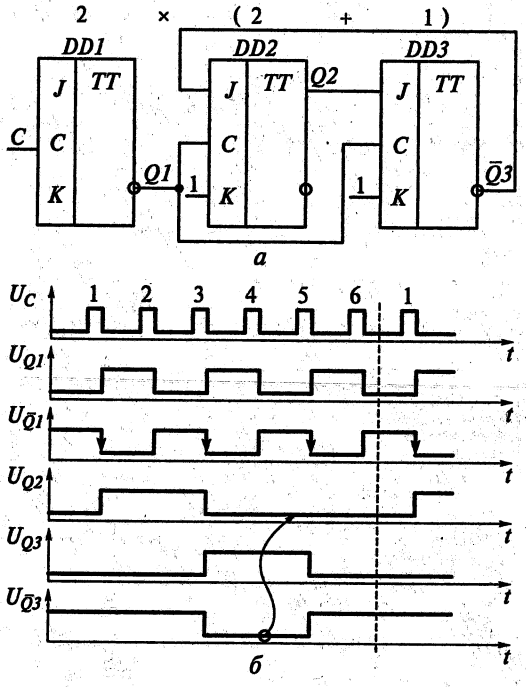


Рис. 15.41

Основными достоинствами таких счетчиков являются возможность наращивания их схем и отсутствие дополнительных логических схем. К недостаткам можно отнести большое число триггеров и сравнительно невысокое быстродействие, которое вызвано тем, что значительная часть триггеров соединяется с использованием последовательного переноса.

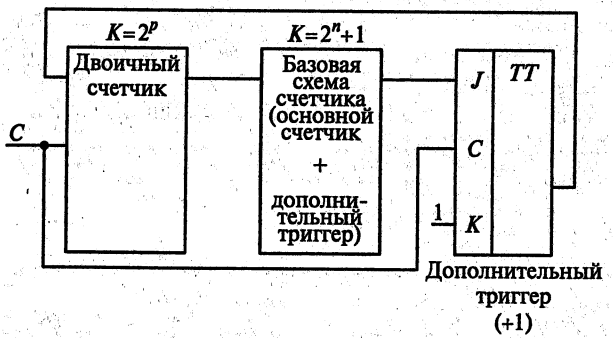
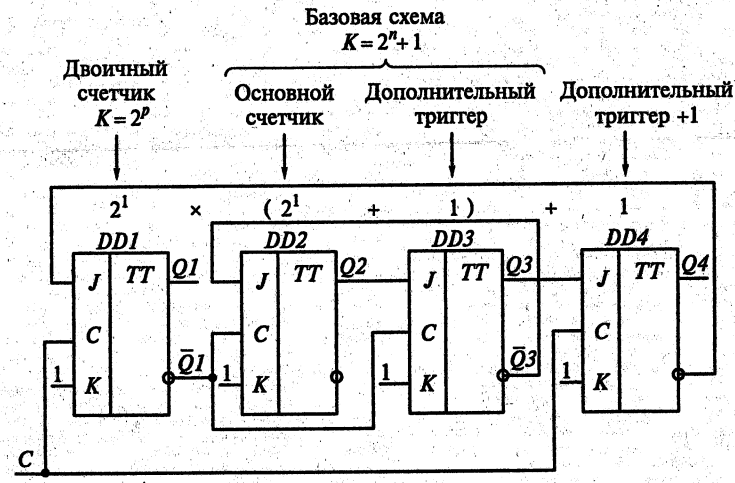
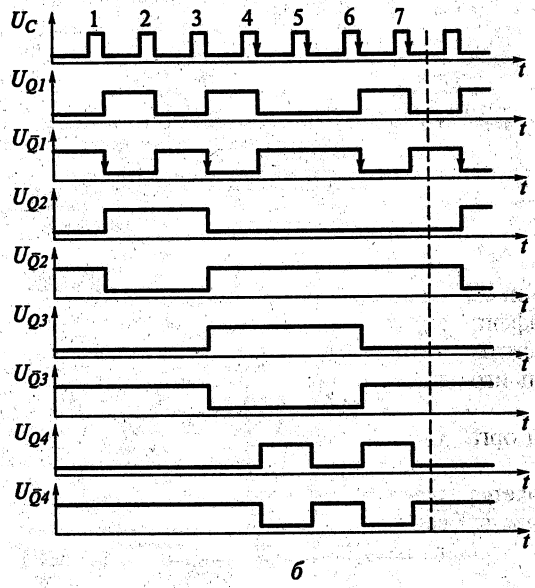


Рис. 15.42



а



б

Рис. 15.43

Контрольные вопросы

1. Как организуется схема кольцевого счетчика?
2. Какова зависимость между разрядностью и коэффициентом деления счетчика на основе регистра сдвига?

3. Какое состояние триггеров кольцевого счетчика говорит о достижении требуемого коэффициента счета?

4. Можно ли на основе кольцевого счетчика построить счетчик с $K = 2^n$?

5. В чем состоит различие в построении кольцевых счетчиков и счетчиков Джонсона?

6. Каким должно быть соответствие между разрядностью и коэффициентом деления в счетчиках Джонсона?

7. Чем отличается построение счетчиков Джонсона с четным и нечетным коэффициентами счета?

8. В каких режимах работают *JK*-триггеры первого, второго и третьего разрядов в счетчике Джонсона с $K = 5$?

9. Можно ли построить счетчик Джонсона с $K = 5$ на базе двухступенчатого *RS*-триггера?

10. Для чего предназначены логические элементы, расположенные в схемах счетчиков с установкой в нулевое состояние?

11. Какой код будет на выходе счетчика с установкой в нулевое состояние по достижении требуемого коэффициента счета и чем определяется разрядность такого счетчика?

12. Каковы возможные режимы работы триггеров счетчика с установкой в нулевое состояние при $K = 7$?

13. В чем заключается сущность принципа организации связей между триггерами и логическими элементами счетчика с установкой в нулевое состояние?

14. Какие существуют возможные коэффициенты счета для счетчика на базе ИС ИЕ2? Чем ограничиваются эти коэффициенты?

15. В чем заключается принцип организации обратных связей в счетчике с параллельным переносом?

16. Каким образом можно заменить входную логику *JK*-триггера логическим элементом в счетчике с параллельным переносом?

17. Каким будет состояние триггеров по достижении требуемого коэффициента в счетчике с параллельным переносом?

18. Какой коэффициент счета может быть получен на основе базовой структуры в безвентильном счетчике?

19. Какую роль играет дополнительный триггер в безвентильном счетчике?

20. Можно ли организовать базовый счетчик на основе синхронного счетчика?

21. Чем отличается состояние триггеров основного счетчика базовой структуры при использовании суммирующих и вычитающих счетчиков?

22. Какие существуют закономерности в построении суммирующих и вычитающих счетчиков?

Раздел V
ПРИНЦИПЫ ПОСТРОЕНИЯ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

Глава 16
СТРУКТУРЫ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

16.1. Классическая структура

Для обеспечения вычислительного процесса вычислительная система должна содержать: устройства по вводу и выводу информации; память для хранения входной, выходной, промежуточной информации, различных программ и данных; операционное устройство (ОУ) или арифметическо-логическое устройство (АЛУ), в котором производится обработка информации; устройство управления (УУ), формирующее управляющие сигналы (УС), с помощью которых осуществляется последовательное выполнение команд программ пользователя. Операционное устройство и управляющее устройство образуют *процессор*.

Существуют два вида взаимосвязи устройств между собой: классическая и магистральная. В классической структуре построения вычислительной системы (рис. 16.1) связи между устройствами индивидуальные, а в магистральной — общие.

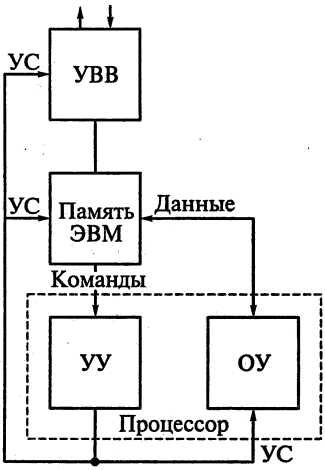


Рис. 16.1

Рассмотрим выполнение вычислительного процесса в вычислительной системе классической структуры. Элементарный неделимый акт обработки информации называется *операцией*, а управляющая информация, вызывающая выполнение этой операции, — *командой*. Каждая команда состоит из двух частей: кода операции, который содержит информацию о выполнении необходимых действий над данными (что надо сделать), и адресной части, содержащей данные или информацию о месте расположения данных, над которыми необходимо произвести действия в соответствии с кодом операции. Последовательность команд, ре-

ализующих требуемый процесс преобразования информации, составляет *программу исходных данных*. Программа вводится в вычислительное устройство через устройства ввода-вывода (УВВ), затем она попадает в память вычислительной системы и лишь после этого начинается ее выполнение. Управление процессом обработки информации в соответствии с командами программы пользователя осуществляется устройством управления, входящим в процессор. В нем происходит дешифрация кода операции каждой команды и в соответствии с этим вырабатываются управляющие сигналы, необходимые для выполнения команды. Под действием управляющих сигналов, воздействующих на нужные участки исполнительных блоков, ОУ получает необходимые данные, выполняет над ними нужные действия, а полученный результат затем размещается в памяти.

Память, содержащая команды, данные, промежуточные и окончательные результаты вычислений, называется *оперативной памятью*. Так как данные, извлекаемые из оперативной памяти, обрабатываются непосредственно процессором, то оперативная память должна иметь большой объем и быстродействие, соизмеримое со скоростью работы процессора. Но помимо выполняемой программы пользователя существуют другие программы, с помощью которых организуется работа системы по вводу и выводу информации. Такие программы находятся в *постоянной памяти* и являются принадлежностью вычислительной системы.

16.2. Магистральная структура

При классической структуре построения достигается максимальное быстродействие вычислительной системы. Это объясняется возможностью параллельной работы устройств по индивидуальным линиям связи, что позволяет практически параллельно производить обработку нескольких программ. По мере совершенствования технологии изготовления интегральных схем появилась возможность создания элементной базы с высокой степенью интеграции (больших интегральных схем — БИС, сверхбольших интегральных схем — СБИС) и интегральных носителей информации большого объема, которые в настоящее время нашли широкое распространение в вычислительных системах на базе микропроцессоров.

Рассмотрим магистральную структуру построения вычислительной системы, представленную на рис. 16.2.

Функции, закрепленные за устройствами вычислительной системы те же, что и в классической структуре, но взаимодействие между ними осуществляется иначе. Каждое устройство имеет выход на шины связи — магистрали, доступ к которым возможен с помощью специальных схем с тремя состояниями (передача лог. 1, лог. 0 и состояние, эквивалентное размыканию ключа, при кото-

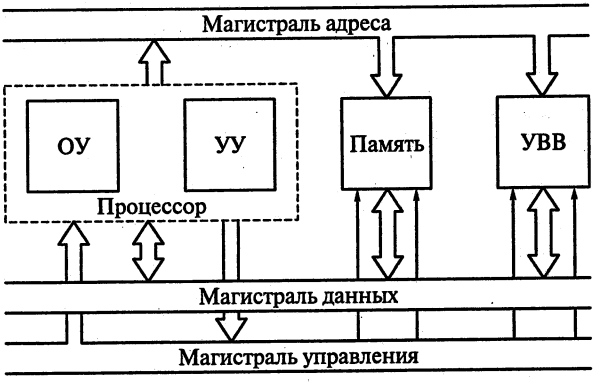


Рис. 16.2

ром возможно отключение от магистрали). Таким образом, к магистрали подключаются устройства, между которыми в данный момент происходит обмен информацией. Под шиной подразумеваются линии связи для передачи сигналов. Группа из n-разрядных линий передачи информации называется шиной или магистралью. В зависимости от вида и направления передаваемой информации за каждой магистралью закрепляется определенное функциональное назначение — передача адреса, данных, управления, причем в зависимости от направления передаваемой информации шины могут быть однонаправленные и двунаправленные. Линии шин адреса и управления являются однонаправленными. Источником сигналов адреса и управления является процессор, а предназначены сигналы для передачи остальным устройствам. Линия шины данных — двунаправленная. Информация может циркулировать от одного устройства к другому в двух направлениях: от устройств ввода в память — при вводе; из памяти в процессор — при выполнении команды; из процессора в память — после выполнения команды; из памяти в устройство вывода — при выводе результатов и т.д. Для этого устройство управления формирует управляющие сигналы доступа к этим устройствам: «Чтение», «Запись», «Ввод на внешние устройства», «Ввод с внешнего устройства».

На основе рассмотренной структуры выделяют два основных подкласса: микропроцессоры, используемые в компьютерах, где требуются большой объем памяти и высокая производительность, и контроллеры (управляющие микрокомпьютеры), используемые в изделиях массового потребления (бытовой технике, игрушках, игровых автоматах, аудио- и видеотехнике), промышленных установках (станках и оборудовании с программным управлением, роботах, и т.д.), измерительной технике, автомобильной промышленности, бизнесе, торговле и т.д.

Глава 17
ОРГАНИЗАЦИЯ УСТРОЙСТВ УПРАВЛЕНИЯ

17.1. Организация микропрограммных устройств управления

Для выполнения программы, состоящей из команд, необходимо сформировать управляющие сигналы, с помощью которых и будет выполнена вся последовательность необходимых операций.

В зависимости от вида команды она может быть выполнена за один или несколько (до 5) машинных циклов. *Машинный цикл* — это время, требуемое для обмена данными по шине данных. В свою очередь машинный цикл разделяется на 3...5 тактов.

Каждой команде соответствует набор микрокоманд, образующих микропрограмму. Выполнение микропрограммы синхронизировано по времени. Микрокоманда содержит информацию о тех действиях, которые необходимо выполнить в течение одного такта. Отдельные единицы в разрядах операционной части микрокоманды являются управляющими сигналами (УС) для выполнения микрооперации (рис. 17.1).

Структурная схема микропрограммного устройства управления (МкУУ) в зависимости от вида микрокоманды и способа формирования управляющих сигналов может быть изменена.

При *горизонтальном микропрограммировании микрокоманд* устройство управления имеет вид, представленный на рис. 17.2.

Команда, предназначенная к выполнению, фиксируется в регистре команд (РК). В определенных разрядах команды находится адрес первой микрокоманды выполняемой микропрограммы, который заносится в регистр адреса (РА). Извлеченная по этому адресу из постоянного запоминающего устройства (ПЗУ) микрокоманда фиксируется в регистре микрокоманд (РгМК). В первом приближении микрокоманду можно разделить на две функциональные части: операционную и адресную. В случае выполнения после-

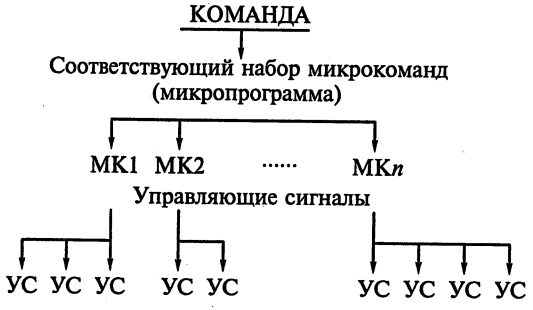


Рис. 17.1

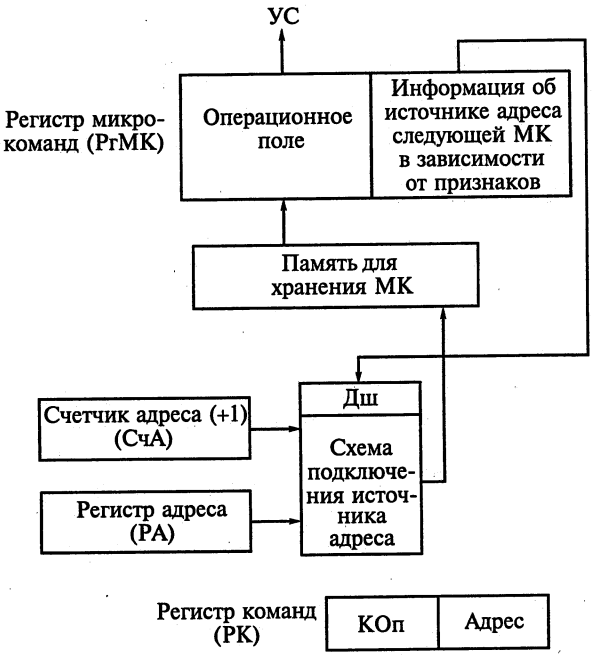


Рис. 17.2

довательности микрокоманд адрес обращения за очередной микрокомандой формируется в счетчике адреса (СчА), путем увеличения предыдущего номера на единицу. В микрокоманде находится информация об источнике адреса следующей микрокоманды, выбор которого осуществляет дешифратор (Дш). Выбор одного из источников адреса осуществляет дешифратор. Разряды операционной части микрокоманды являются управляющими сигналами для выполнения микроопераций. Таким образом, выполнением микропрограммы осуществляется выполнение команды. Последняя микрокоманда микропрограммы разрешает считывание следующей команды в РК, после чего процесс повторяется.

МкУУ характеризуется функциональными возможностями, определяемыми степенью воздействия на определенные точки исполнительных блоков. В данной схеме МкУУ количество УС за один такт ограничено разрядностью (n) операционного поля РгМК.

При *вертикальном микропрограммировании микрокоманд* на выходе РгМК расположен дешифратор микрокоманд (ДшМК). В этом случае количество управляющих сигналов увеличивается до 2^n (рис. 17.3). При этом за один такт возможна выработка только одного УС, что сказывается на скорости выполнения микропрограммы.

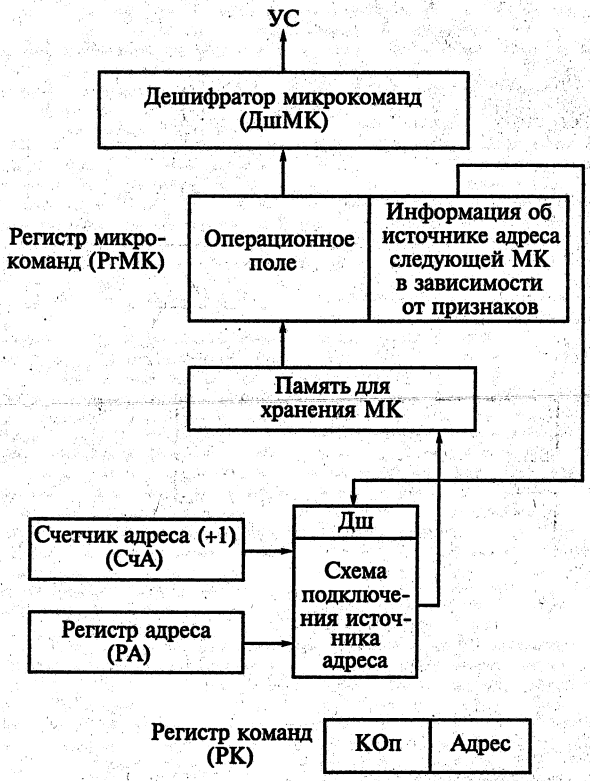


Рис. 17.3

Количество вырабатываемых сигналов за один такт можно повысить, разбив операционную часть микрокоманды на отдельные поля (рис. 17.4).

Коды полей операционной части микрокоманды поступают на дешифраторы полей микрокоманд определенной разрядности. Таким образом, за один такт на выходе каждого из дешифраторов

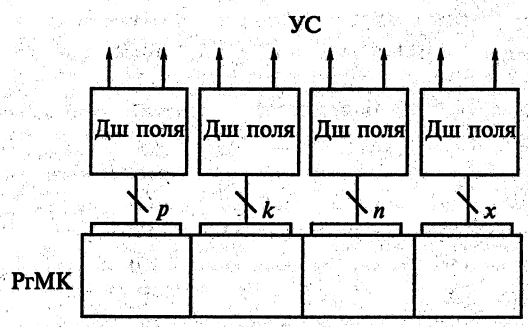


Рис. 17.4

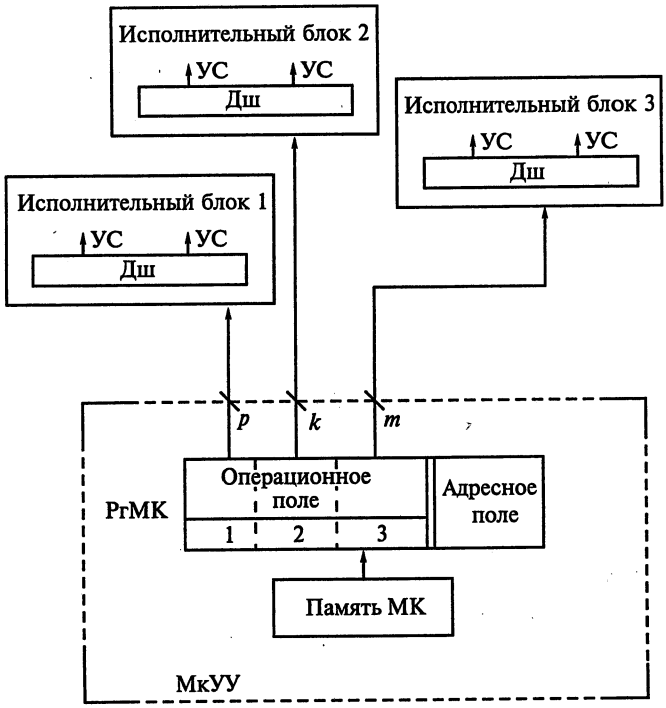


Рис. 17.5

Номер поля	7				6					5				4		
Номер бита	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Назначение бита	*BR3*	*BR2*	*BR1*	*BR0*	*P3*	*P2*	*P1*	*P0*	*MS2*	*I8*	*I7*	*I6*	*MS1*	*I2*	*I1*	*I0*
Определение поля МК	Адрес перехода				Управление следующим адресом				*MS2*	Управление приемником результата			*MS1*	Источник операнда		

Окончание

Номер поля	3				2				1				0			
Номер бита	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Назначение бита	*C0*	*I5*	*I4*	*I3*	*A3*	*A2*	*A1*	*A0*	*B3*	*B2*	*B1*	*B0*	*D3*	*D2*	*D1*	*D0*
Определение поля МК	*C0*	АЛУ			*A*				*B*				*D*			

Условные обозначения: D — входные данные; *B* — адрес РОН по каналу *B*; *A* — адрес РОН по каналу *A*; АЛУ — код управления функцией АЛУ; *C0* — входной перенос в АЛУ; *MS1*, *MS2* — биты управления операциями сдвига; *I0*... *I8* — управление центральной процессорной секцией ВС1; *P0*... *P3* — управление выборкой следующего адреса.

Рис. 17.6

полей появится УС, а их количество будет определяться количеством групп полей.

В зависимости от конструктивного выполнения микропрограммного устройства управления существуют ограничения на количество выводов, а следовательно, и количество управляющих сигналов, что снижает его функциональные возможности.

При *смешанном микропрограммировании* операционная часть микрокоманды делится на отдельные группы разрядов, каждая из которых работает на отдельный дешифратор небольшой разрядности, но при этом дешифраторы не являются частью микропрограммного устройства управления, они выносятся в соответствующие исполнительные блоки (рис. 17.5), снижая тем самым количество выводов МкУУ. При такой структуре удается существенно расширить функциональные возможности процессора.

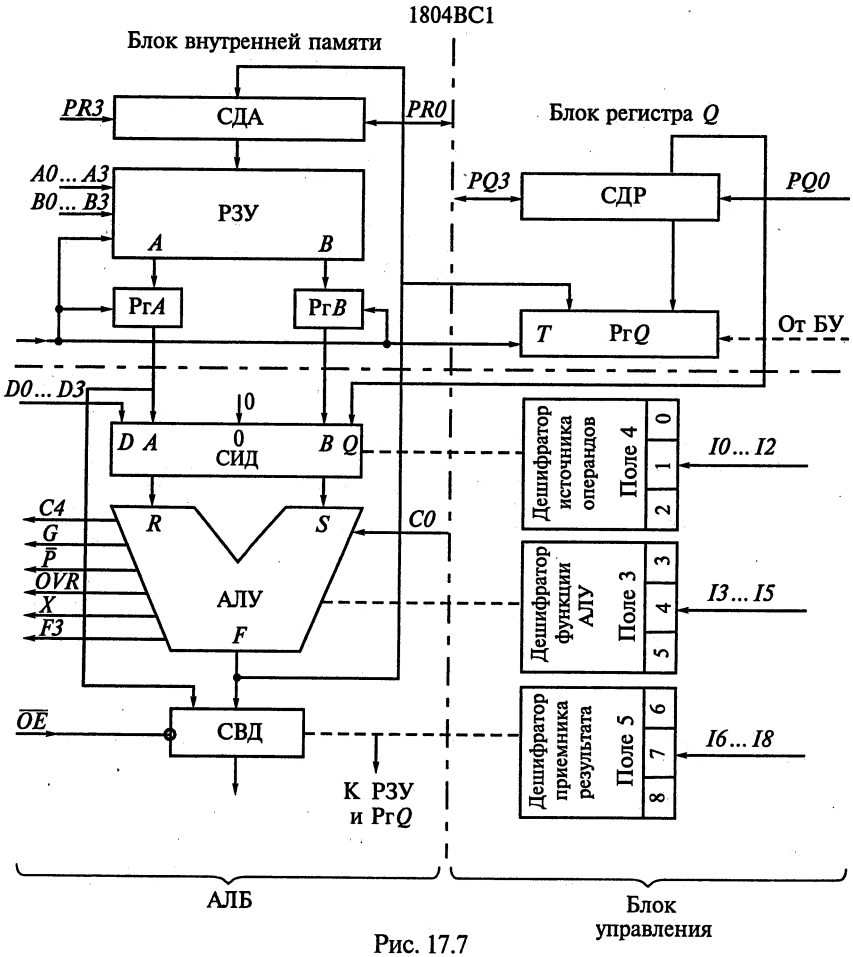
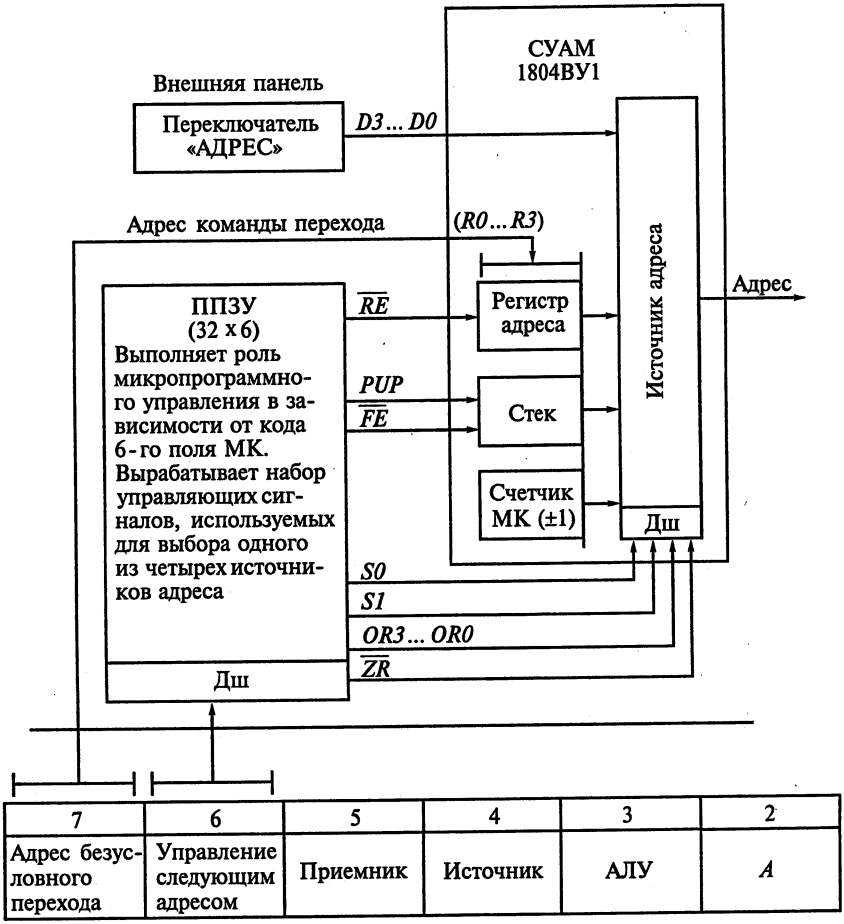


Рис. 17.7



7	6	5	4	3	2
Адрес безусловного перехода	Управление следующим адресом	Приемник	Источник	АЛУ	*A*

Рис. 17.8

На рис. 17.6 приведен пример микрокоманды, используемой при смешанном микропрограммировании, а на рис. 17.7 и 17.8 — соответственно примеры управления арифметико-логическим блоком ВС1 и выбором источника адреса следующей микрокоманды ВУ1 в микроЭВМ на базе микропроцессорной серии 1804.

Контрольные вопросы

1. Для чего предназначены отдельные устройства: устройство ввода-вывода, устройство управления, операционное устройство, память?
2. Возможно ли одновременное функционирование и обмен информацией между УВВ, Памятью, ОУ в классической структуре построения вычислительной системы?
3. Какая взаимосвязь существует между программой, командой, микропрограммой, микрокомандой и микрооперацией?

4. В чем состоит различие между оперативной и постоянной памятью?

5. В чем заключается специфика обмена информацией в магистральной структуре построения вычислительной системы?

6. Возможно ли одновременное функционирование и обмен информацией между УВВ, Памятью, ОУ в магистральной структуре вычислительной системы?

7. Каково функциональное назначение магистралей?

8. Чем отличаются команды пользователя и команды устройства управления: «Чтение», «Запись» и т.д.? Где хранятся эти команды?

9. Для чего предназначены микропрограммные устройства управления?

10. Где хранится адрес первой микрокоманды?

11. Как происходит формирование последующих адресов микрокоманд микропрограммы?

12. На основе какого поля микрокоманды происходит формирование управляющих сигналов?

13. Каковы функциональные возможности УУ при вертикальном и горизонтальном способах программирования микрокоманды для операционного поля размером в байт?

14. Почему при смешанном программировании дешифраторы вынесены в исполнительные блоки?

17.2. Организация устройств управления с жесткой логикой

Рассмотрим организацию управления с использованием устройства, в котором логические схемы соединяются таким образом, чтобы с их помощью происходило формирование временной последовательности определенных управляющих сигналов. Если в микропрограммных устройствах возможно изменение содержимого памяти, а соответственно и изменение последовательностей управляющих сигналов, то в данных устройствах управления это невозможно. В этом случае нужно пересматривать построение схемы формирования управляющих сигналов, т.е. создавать новую схему. Такие схемы называются *устройствами с жесткой логикой управления.*

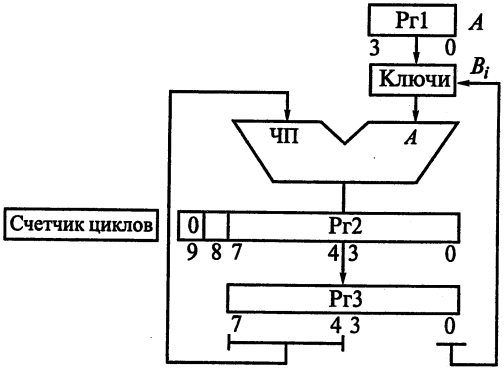


Рис. 17.9

Например, требуется создать устройство формирования управляющих сигналов для операционной схемы, представленной на рис. 17.9, в которой производится перемножение двух четырехразрядных чисел A и B.

Рассмотрим пример умножения двух чисел A и B:

$A = 1100$, $B = 0110$.

			Сч.
			4
Рг3	ЧП	B	
	7...4	3...0	
	0000	0110	
$B_0 = 0$	0000	0011	3
	→		
$B_1 = 1$	0000	0011	
	+		
A	1100		
	1100	0011	
	0110	0001	2
	→		
$B_2 = 1$	0110	0001	
	+		
A	1100		
	10010	0001	
	1001	0000	1
	→		
$B_3 = 0$	0100	1000	0
	→		
	Произведение		

Будем считать, что предварительно в регистры была занесена информация. Множимое $A = 1100$ размещено в четырехразрядном регистре Рг1. Множитель $B = 0110$ размещен в четырех младших разрядах регистра частичных произведений Рг3.

Нулевое, исходное значение частичных произведений (ЧП) размещено в четырех старших разрядах регистра частичных произведений Рг3. Буферный регистр, установленный на выходе сумматора Рг2, обнулен.

Регистр Рг1 постоянно хранит множимое A. В регистре Рг2 временно хранятся промежуточные значения ЧП. Для их записи необходимо обеспечить параллельный режим ввода информации. Затем очередное ЧП должно быть сдвинуто на один разряд (от старшего к младшему). Для сдвига информации можно использовать тот

же регистр, если в нем предусмотрен реверсивный режим работы. В результате выполненных действий новое ЧП переписывается в регистр Рг3. Первоначально в младшие разряды регистра записывается множитель B; значение его младшего разряда используется для управления ключами доступа числа A на вход сумматора. Если очередной анализируемый разряд множителя B равен единице, то ключи доступа замыкаются и множимое A проходит на вход сумматора в качестве слагаемого. Если анализируемый разряд равен нулю, то ключи доступа разомкнуты, т.е. формируется нулевое значение слагаемого. При сдвигах происходит постепенное вытеснение множителя B и заполнение этих разрядов младшими разрядами ЧП.

Так как регистр Рг2 работает в реверсивном режиме, необходимо обеспечить как параллельную запись, так и последовательный сдвиг информации. В регистрах для этой цели используются управляющие входы S. Достаточно иметь один триггер, с помощью которого будут формироваться сигнал $S = 1$ (параллельный режим) и сигнал $S = 0$ (последовательный режим сдвига).

Для определения момента завершения умножения необходим счетчик числа циклов. Предварительно в него записывается число, равное разрядности множителя. Затем по мере получения очередного частичного произведения его содержимое уменьшается на единицу, и по достижении нулевого значения вычисления прекращаются. Содержимое счетчика является признаком X, влияющим на ход решения. Если содержимое счетчика отлично от нуля, то $X = 0$, если оно равно 0, то $X = 1$.

Выявим управляющие сигналы, необходимые при вычислениях.

1. Занесение числа циклов в счетчик циклов — y_1.

2. $B_0 = 0$. Ключи доступа закрываются, формируя на входе сумматора нули. В сумматоре выполняется сложение исходного ЧП с нулями.

3. Для подготовки регистра Рг2 к параллельной записи ЧП триггер, формирующий сигнал $S = 1$, устанавливается в единичное состояние под воздействием управляющего сигнала y_2.

4. Запись ЧП в регистр Рг2 происходит под воздействием управляющего сигнала y_3, подаваемого на вход синхронизации.

5. Для подготовки последовательного сдвига информации в регистре триггер, формирующий сигнал $S = 0$, устанавливается в нулевое состояние под воздействием управляющего сигнала y_4.

6. Выполнение сдвига информации в регистре Рг2 происходит под воздействием управляющего сигнала y_3, подаваемого на вход синхронизации.

7. Так как регистр Рг3, используемый для хранения ЧП, постоянно работает в параллельном режиме, то для записи в него ЧП достаточно подать синхроимпульс, роль которого выполняет управляющий сигнал y_5. Одновременно производится уменьшение

содержимого счетчика циклов на единицу под воздействием управляющего сигнала y_6.

8. $B_1 = 1$. Ключи доступа открываются, пропуская на вход сумматора множимое A.

Далее повторяются пункты 3...7 до тех пор, пока содержимое счетчика не станет равным нулю. При возникновении признака $X = 1$ вычислительный процесс прекращается.

Перечислим управляющие сигналы, необходимые при вычислениях:

y_1 — синхроимпульс для записи числа циклов в счетчик циклов;

y_2 — установка триггера «Режим» в единичное состояние для подготовки параллельного режима записи информации в регистре Рг2 ($S = 1$) — вновь образованного частичного произведения;

y_3 — синхроимпульс, поступающий на вход регистра Рг2 как при последовательном, так и при параллельном режимах работы;

y_4 — установка триггера «Режим» в нулевое состояние для подготовки последовательного сдвига информации в регистре Рг2 ($S = 0$);

y_5 — синхроимпульс, поступающий на вход регистра Рг2 при перезаписи ЧП из Рг2 в Рг3;

y_6 — уменьшение счетчика цикла на 1.

Под воздействием управляющих сигналов в одном такте может выполняться одна или несколько микроопераций, что соответствует одной микрокоманде Y. Приведем микрокоманды и соответствующие им управляющие сигналы:

$$Y_1 : y_1; \quad Y_2 : y_2; \quad Y_3 : y_3; \quad Y_4 : y_4; \quad Y_5 : y_3; \quad Y_6 : y_5, y_6.$$

Синтезируемое устройство представляет собой автомат с памятью, т. е. состоит из двух функциональных частей: памяти и комбинационной схемы формирования управляющих сигналов. *Автоматы* — это схемы, в которых значения выходных сигналов зависят не только от значений входных сигналов в данном такте работы (t), но и от состояния схемы в предыдущем такте ($t - 1$). Такая схема помнит прошлые воздействия, т. е. имеет память, чего комбинационные схемы лишены. Простейшим автоматом с памятью является триггер. В схему автомата (рис. 17.10) входят: память в виде нескольких триггеров, образующих регистр состояния; комбинационный узел (КУ),

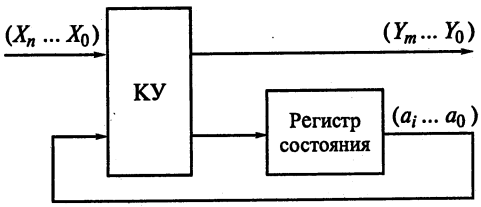


Рис. 17.10

вырабатывающий выходной код, часть которого является формируемыми сигналами для управления вычислениями, а другая часть — сигналами воздействия на входы S и R триггеров регистра состояния внутри самого автомата для работы в следующем такте. Возможности автомата определяются разрядностью регистра — чем больше разрядов, тем больше функциональные воздействия. В качестве входных сигналов выступают признаки X_i, которые влияют на ход вычислительного процесса. Для рассматриваемого примера признаком является значение содержимого счетчика циклов X.

Существуют два способа синтеза управляющих устройств — в форме автоматов Мура и Мили. В построении этих автоматов много общего.

Проектирование управляющего устройства на основе автоматов Мура и Мили осуществляется в несколько этапов.

1. Постановка задачи.
2. Составление алгоритма функционирования в микрокомандах.
3. Разметка алгоритма.
4. Кодирование состояний устройства.
5. Составление графа функционирования.
6. Построение таблицы функционирования комбинационного узла.
7. Запись логических выражений для формируемых сигналов.
8. Построение логической схемы автомата с памятью.

Поскольку задача уже сформулирована переходим ко второму этапу проектирования.

Составление алгоритма функционирования в микрокомандах (Y). Алгоритм определяет порядок вычислительного процесса, на который влияют промежуточные результаты и различные признаки ($X_n \ldots X_0$). Работа управляющего и вычислительного устройств синхронизирована во времени. На любом такте работы состояние управляющего устройства определяется кодом состояния ($a_i \ldots a_0$). Управляющий автомат, находясь в определенном состоянии в такте (t) и получив определенное значение признака или признаков, генерирует управляющие сигналы на выходных шинах ($Y_m \ldots Y_0$) устройства и запоминает состояние до следующего такта ($t+1$). На рис. 17.11, *а*, *б* представлен один и тот же алгоритм операции умножения в микрокомандах соответственно для автоматов Мура и Мили. На первый взгляд они одинаковы, однако отличия заключаются в их разметке, т.е. в определении точек, определяющих состояния устройства.

Разметка алгоритма. При разметке алгоритмов Мура и Мили вопрос состоит в том, в какие моменты времени оценивать состояние устройства, с тем чтобы в следующем такте сформировались новые сигналы. Для автомата Мура это точки a_i выполнения микрокоманды (см. рис. 17.11, *а*), для автомата Мили — это точки после выполнения микрокоманды под воздействием управляющих сигналов (см. рис. 17.11, *б*). Начало и конец схем алгоритмов отмечаются одним и тем же символом — a_0. В теории конечных автома-

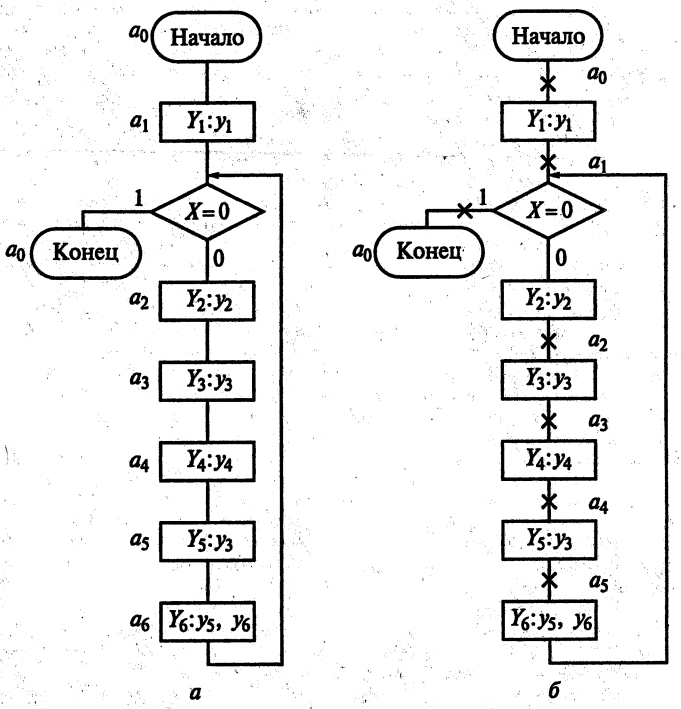


Рис. 17.11

тов доказана эквивалентность автоматов Мили и Мура. Однако автомат Мура, который на практике используется чаще, имеет бóльшее число состояний, чем соответствующий автомат Мили.

Кодирование состояний устройства. Автомат Мура имеет семь состояний ($a_6 \ldots a_0$), автомат Мили — шесть ($a_5 \ldots a_0$). Кодирование этих состояний показано в табл. 17.1.

Составление графа функционирования. Системы взаимодействия элементов создаваемого устройства удобно отображать с помощью

Таблица 17.1

Состояние устройства	Кодовая комбинация		
	Q_3	Q_2	Q_1
a_0	0	0	0
a_1	0	0	1
a_2	0	1	0
a_3	0	1	1
a_4	1	0	0
a_5	1	0	1
a_6	1	1	0

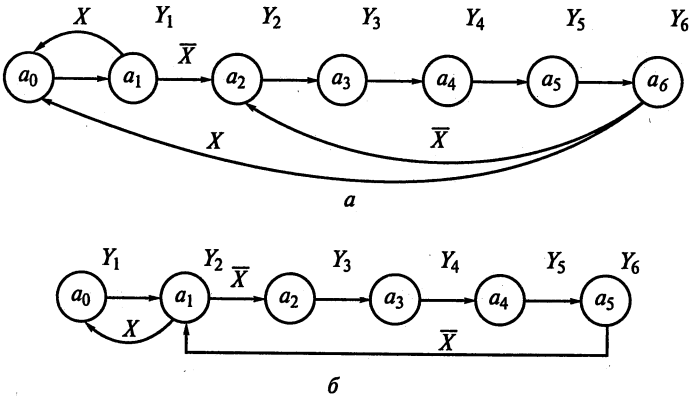


Рис. 17.12

графа. *Граф* — это совокупность вершин и дуг (ребер), представляющих одну или несколько связей между вершинами. Вершины графа соответствуют определенным состояниям устройства (a_i), а дуги и ребра графа отражают взаимосвязи между отдельными схемами устройства в зависимости от результатов выполненной микрокоманды и полученных признаков, т.е. дуги (ребра) показывают, по каким цепям будут сформированы очередные управляющие сигналы. На рис. 17.12, *а* показан граф для автомата Мура, на рис. 17.12, *б* — для автомата Мили.

Таблица 17.2

Текущее состояние				Следующее состояние				Условие перехода	Формируемые сигналы	
Обозначение	Код			Обозначение	Код				Сигналы установки триггеров регистра состояний	Управляющие сигналы микроопераций
	Q_3	Q_2	Q_1		Q_3	Q_2	Q_1			
a_0	0	0	0	a_1	0	0	1	—	*S1*	—
a_1	0	0	1	a_2	0	1	0	$\overline{X}$	*S2, R1*	$Y_1 : y_1$
a_2	0	1	0	a_3	0	1	1	—	*S1*	$Y_2 : y_2$
a_3	0	1	1	a_4	1	0	0	—	*S3, R2, R1*	$Y_3 : y_3$
a_4	1	0	0	a_5	1	0	1	—	*S1*	$Y_4 : y_4$
a_5	1	0	1	a_6	1	1	0	—	*S2, R1*	$Y_5 : y_3$
a_6	1	1	0	a_2	0	1	0	$\overline{X}$	*R3*	$Y_6 : y_5, y_6$
a_1	0	0	1	a_0	0	0	0	X	*R1*	$Y_1 : y_1$
a_6	1	1	0	a_0	0	0	0	X	*R3, R2*	$Y_6 : y_5, y_6$

Таблица 17.3

Текущее состояние				Следующее состояние				Условие перехода	Формируемые сигналы	
Обозначение	Код			Обозначение	Код				Сигналы установки триггеров регистра состояний	Управляющие сигналы микроопераций
	Q_3	Q_2	Q_1		Q_3	Q_2	Q_1			
a_0	0	0	0	a_1	0	0	1	—	*S1*	$Y_1 : y_1$
a_1	0	0	1	a_2	0	1	0	$\overline{X}$	*S2, R1*	$Y_2 : y_2$
a_2	0	1	0	a_3	0	1	1	—	*S1*	$Y_3 : y_3$
a_3	0	1	1	a_4	1	0	0	—	*S3, R2, R1*	$Y_4 : y_4$
a_4	1	0	0	a_5	1	0	1	—	*S1*	$Y_5 : y_3$
a_5	1	0	1	a_1	0	0	1	—	*R3*	$Y_6 : y_5, y_6$
a_1	0	0	1	a_0	0	0	0	*X*	*R1*	—

Иногда в графах показывают дополнительные петли около определенных состояний. Такие петли возникают, когда при поступлении некоторых сигналов автомат не изменяет своего состояния. Петли сильно загромождают чертеж, поэтому их, как правило, не изображают, а лишь подразумевают, полагая что при поступлении любого сигнала, кроме тех, которыми отмечены дуги и стрелки, автомат своего состояния не изменит.

Построение таблицы функционирования комбинационного узла. Состояние устройства в текущий момент времени, условия, необходимые для перехода в следующее состояние, код следующего состояния, сигналы воздействия на триггеры состояния для перехода в новое состояние, а также формируемые управляющие сигналы отражаются в таблице функционирования устройства. Такая таблица может быть выполнена на основе алгоритма или на основе графа функционирования устройства. В табл. 17.2 отражено функционирование устройства формирования управляющих сигналов, построенного для автомата Мура.

В табл. 17.3 отражено функционирование устройства формирования управляющих сигналов для автомата Мили.

Запись логических выражений для формируемых сигналов. С помощью разрабатываемого устройства формируются два вида сигналов: сигналы воздействия на входы R и S триггеров состояния и управляющие сигналы для непосредственного влияния на вычислительный процесс. В табл. 17.2 и 17.3 отмечены входы триггеров, на которых присутствуют активные сигналы, изменяющие состояния *RS*-триггеров, при этом имеется в виду, что на вторые входы этих триггеров подаются сигналы противоположного значения. На основании приведенных

таблиц с помощью логических выражений записываются зависимости формируемых сигналов от текущего состояния и признаков.

Для устройства, построенного как автомат Мура (см. табл. 17.2), логические выражения, отражающие формируемые сигналы, имеют следующий вид:

$S1 = a_0 \vee a_2 \vee a_4$;
$R1 = a_1 \cdot \overline{X} \vee a_3 \vee a_5 \vee a_1 \cdot X = a_1 \vee a_3 \vee a_5$;
$S2 = a_1 \cdot \overline{X} \vee a_5$;
$R2 = a_3 \vee a_6 \cdot X$;
$S3 = a_3$;
$R3 = a_6 \cdot \overline{X} \vee a_6 \cdot X = a_6$;

$y_1 = a_1 \cdot \overline{X} \vee a_1 \cdot X = a_1$;
$y_2 = a_2$;
$y_3 = a_3 \vee a_5$;
$y_4 = a_4$;
$y_5, y_6 = a_6 \cdot \overline{X} \vee a_6 \cdot X = a_6$.

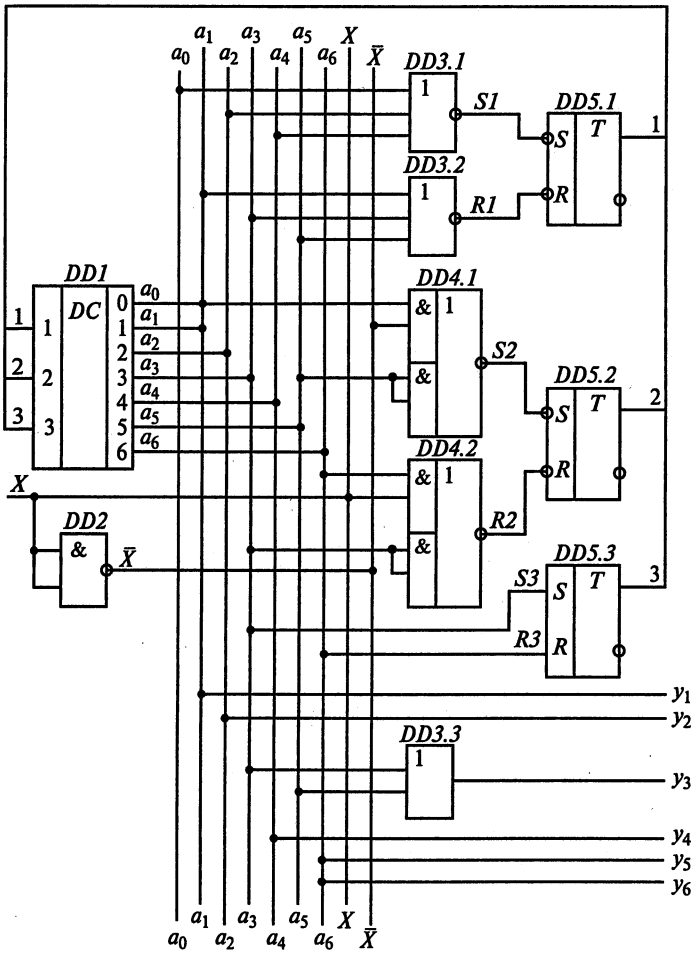


Рис. 17.13

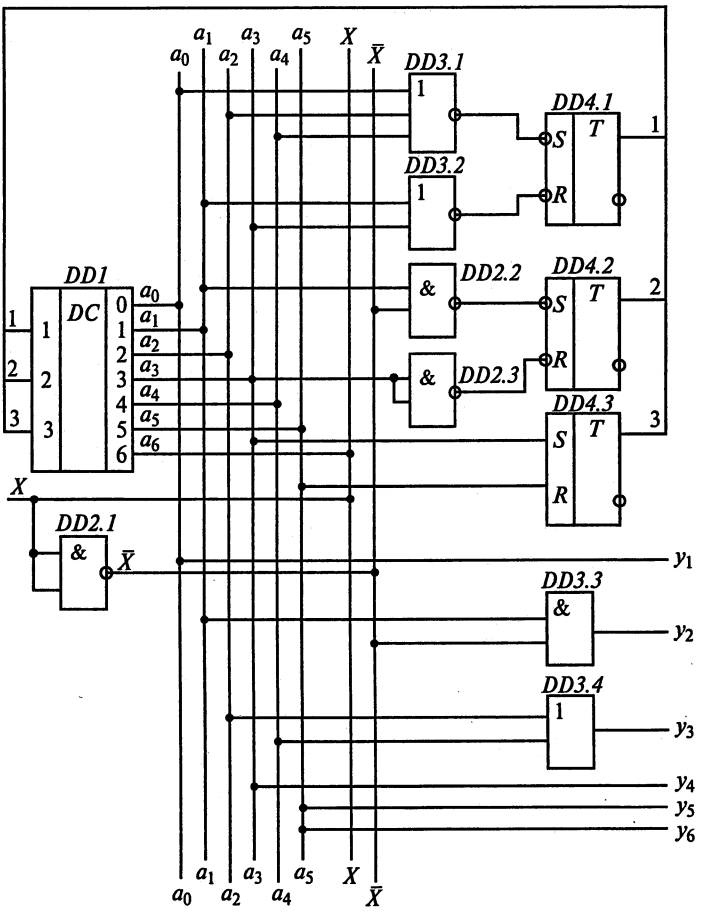


Рис. 17.14

Для устройства, построенного как автомат Мили (см. табл. 17.3), логические выражения, отражающие формируемые сигналы, имеют следующий вид:

$S1 = a_0 \vee \overline{a_2} \vee a_4$;
$R1 = a_1 \cdot \overline{X} \vee a_3 \vee a_1 \cdot X = a_1 \vee a_3$;
$S2 = a_1 \cdot \overline{X}$;
$R2 = a_3$;
$S3 = a_3$;
$R3 = a_5$;

$y_1 = a_0$; $y_2 = a_1 \cdot \overline{X}$;
$y_3 = a_2 \vee a_4$;
$y_4 = a_3$;
$y_5, y_6 = a_5$.

Построение логической схемы автомата с памятью. Далее строится схема разрабатываемого устройства формирования управляющих сигналов или просто устройства управления для одного из

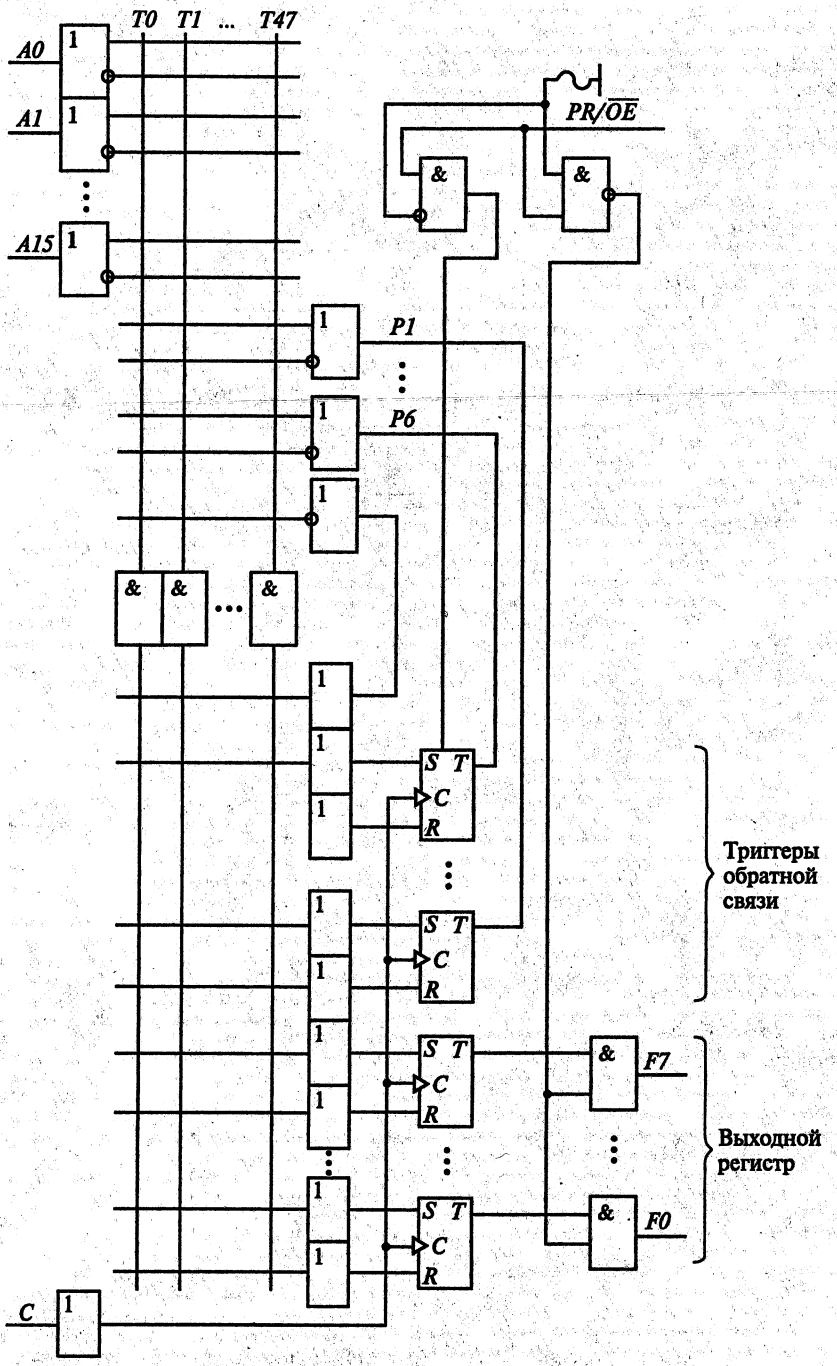

T0
T1
...
T47
A0
A1
A15
PR/OE
P1
P6
S T
C
R
Триггеры обратной связи
F7
F0
Выходной регистр
C

Рис. 17.15

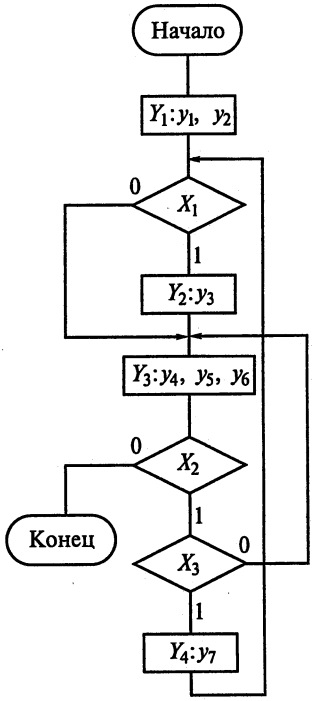


Рис. 17.16

двух способов организации комбинационного узла: на основе автомата Мура (рис. 17.13) или на основе автомата Мили (рис. 17.14). Обе схемы содержат: регистр для хранения кода состояния; дешифратор, с помощью которого в зависимости от кода состояния на входе формируется сигнал a_i на одном их выходов; комбинационные схемы на логических элементах, обеспечивающие в зависимости от возбужденной шины a_i на выходе дешифратора организацию сигналов $\overline{S}_i$ и $\overline{R}_i$, подготавливающих следующие состояния. Кроме того, сигнал с возбужденной шины дешифратора и дополнительные логические элементы участвуют в формировании управляющих сигналов $\overline{y}$.

В настоящее время широко используется организация комбинационных схем на основе ПЛМ. Так, на рис. 17.15 представлена схема управления в интегральном исполнении К556РТ3, на основе которой можно реализовать ранее рассмотренные схемы.

Микросхема имеет 16 входов (*A0*... *A15*), восемь выходов (*F0*... *F7*) и 48 термов (*T0*... *T47*). В состав микросхемы входят шесть триггеров обратной связи и восемь триггеров, образующих восьмиразрядный выходной регистр. Соответственно весь комбинационный узел реализован на ПЛМ.

Контрольное задание

Построить устройство формирования управляющих сигналов, алгоритм работы которого представлен на рис. 17.16.

17.3. Особенности построения устройств управления. Структура построения вычислительной системы на основе БИС секционированного микропроцессора

Рассмотрев два способа организации устройств управления, можно сделать следующие выводы. В аппаратурных устройствах управления, автоматах с памятью каждой операции соответствует свой набор логических схем, вырабатывающих определенные функциональные сигналы для выполнения микроопераций в определенные моменты времени. При таком способе построения устройства управления реализация микроопераций достигается за счет однажды организованных соединений в логических структурах, т. е. жесткой логики управления. В этих вычислительных системах невозможно изменение системы команд ЭВМ после ее изготовления.

При микропрограммной реализации устройства управления в него входит запоминающее устройство (ЗУ). Каждому считанному информационному коду — микрокоманде соответствует микрооперация. Набор микрокоманд и последовательность их реализации обеспечивают выполнение более сложных операций. В зависимости от вида применяемой схемы ЗУ хранимая там информация может заменяться различными способами, поэтому такой способ построения устройства управления называется микропрограммным, или устройством с гибкой (хранимой) логикой управления.

При проектировании микропроцессоров решается задача определения набора команд, выполняемых программным или аппаратурным способом на основе заданной системы микрокоманд. Аппаратурная реализация сложных команд позволяет увеличить быстродействие микропроцессора, но требует значительных ресурсов кристалла интегральной схемы микропроцессора. Программная реализация сложных команд позволяет обеспечивать программирование сложных задач, изменять количество и особенности исполнения сложных команд. Однако скорость исполнения программных команд ниже скорости исполнения аппаратурно реализованных команд.

Практически во всех современных микропроцессорных системах используются сложные развитые системы команд. Их ядро, состоящее из набора универсальных команд, реализуется аппаратурным способом в центральном микропроцессоре. Кроме того, специализированные части наборов системы команд реализуются вспомогательными или периферийными микропроцессорами. Эти расширяющие возможности обработки данных специальные ариф-

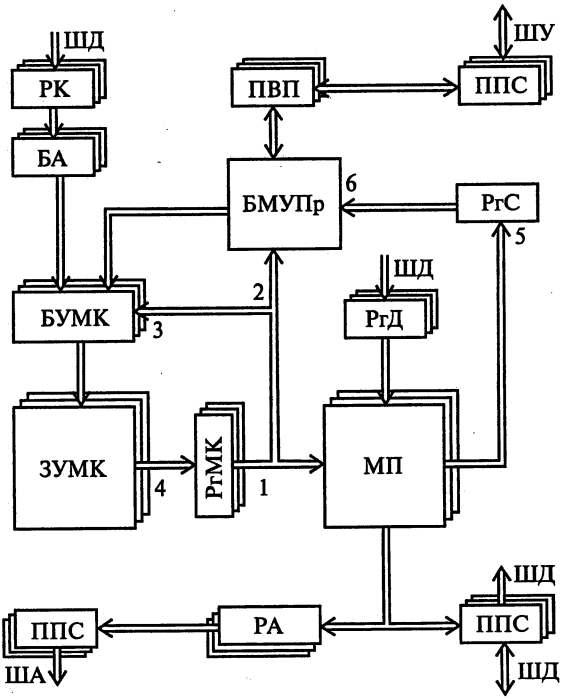


Рис. 17.17

метические или логические микропроцессоры позволяют ускорить выполнение определенных команд и тем самым сократить время выполнения программ. В высокопроизводительных вычислительных системах микропроцессоры представляют собой отдельные секции, каждая из которых предназначена для обработки нескольких разрядов данных с возможностью их наращивания.

На рис. 17.17 представлена обобщенная структура процессора на основе комплекта БИС секционного микропроцессора. Взаимодействие основных блоков здесь осуществляется следующим образом. Секции приема передатчиков сигналов (ППС) позволяют организовать шины данных (ШД), шины адреса (ША) и шины управления (ШУ), связанные соответственно с регистрами данных (РгД), адреса (РА) и команд (РК). Команда извлекается из ОЗУ и по ШД передается в регистр команд РгК. Код команды поступает через блок адресации (БА) и блок управления микрокомандами (БУМК) для преобразования его в первый адрес микрокоманды и обращения к запоминающему устройству микрокоманд (ЗУМК). Затем выполняются несколько микроопераций для поиска данных в оперативном запоминающем устройстве (ОЗУ) и передача их в микропроцессор МП, где происходит выполнение операции.

Наличие регистра микрокоманд (РгМК) позволяет организовать конвейерную работу процессора. В этом случае в цикле, когда РгМК хранит текущую микрокоманду (1), а МП занят ее выполнением, идет подготовка следующей микрокоманды. Для этого блок местного управления (БМУПр) анализирует определенное поле текущей микрокоманды (2) и выдает адрес следующей микрокоманды, по которому происходит ее считывание из ЗУМК (3) в РгМК (4). По результату выполнения операции МП выдает в регистр состояния (РгС) код состояния процессора (5), по значению которого БМУПр принимает решение (6) о целесообразности выполнения следующей микрокоманды, подготовленной в РгМК. Приведенная последовательность из шести тактов при конвейерном методе параллельных действий позволяет сократить время выполнения команд до трех тактов.

17.4. Внутренняя и внешняя память вычислительных систем

Информация, циркулирующая в вычислительной системе, хранится в памяти. На рис. 17.18 представлена схема взаимодействия различных видов памяти, а на рис. 17.19 и 17.20 приведены классификации видов ЗУ, составляющих память ЭВМ.

В зависимости от места нахождения в вычислительной системе память подразделяется на внутреннюю и внешнюю.

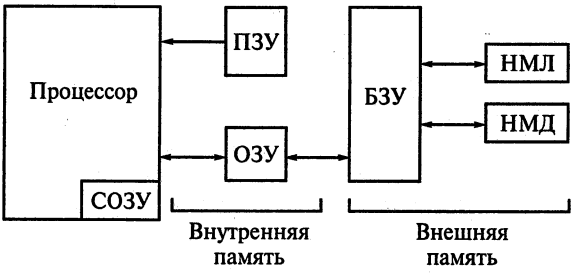


Рис. 17.18

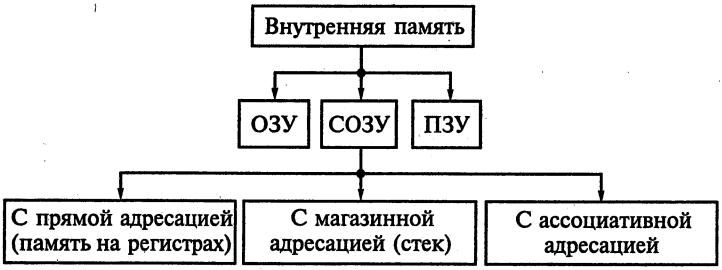


Рис. 17.19

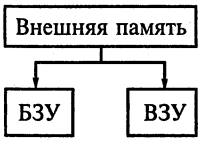


Рис. 17.20

Внутренняя память входит в состав ЭВМ и непосредственно участвует в обмене информацией с процессором в процессе вычислений. Она имеет небольшой объем по сравнению с внешней памятью и высокое быстродействие.

По функциональному назначению внутренняя память подразделяется на оперативные запоминающие устройства (ОЗУ), постоянные запоминающие устройства (ПЗУ) и сверхоперативные запоминающие устройства (СОЗУ).

В ОЗУ хранятся программы, исходные, промежуточные и конечные результаты обработки данных. Информация в ОЗУ может быть записана, считана и сохранена. ОЗУ относятся к запоминающим устройствам с произвольным обращением, т. е. в любой момент времени возможно обращение по любому адресу. При отключении питания информация в ОЗУ разрушается.

В ПЗУ долговременно хранится ранее записанная вне ЭВМ информация: программы длительного пользования, микропрограммы, тестовые программы, подпрограммы, справочные данные. Информация в ПЗУ может быть сохранена даже при отключении питания.

СОЗУ относятся к быстродействующим запоминающим устройствам и являются промежуточным звеном, или буфером, между процессором и относительно медленными ОЗУ. СОЗУ располагается в процессоре, и, являясь его местной памятью, освобождает ОЗУ от временного хранения промежуточной, находящейся в обработке информации.

В зависимости от используемого вида адресации СОЗУ могут иметь различную схемную реализацию: с прямой адресацией, магазинной и ассоциативной.

По способу обращения внутренняя память делится на ЗУ с произвольным обращением (ОЗУ, ПЗУ) и последовательным.

В ЗУ с произвольным обращением допускается любой порядок следования адресов.

В ЗУ с последовательным обращением выборка ячеек памяти возможна только в порядке убывания или возрастания адресов.

По способу хранения информации внутренняя память делится на статические и динамические ЗУ.

В статических ЗУ запоминающий элемент (ЗЭ) представляет собой триггер. В режиме хранения состояние ЗЭ не изменяется, а информация не разрушается.

В динамических ЗУ запоминающий элемент представляет собой конденсатор. В режиме хранения конденсатор постепенно разряжается, поэтому требуется периодическое его восстановление — регенерация.

По технологии изготовления внутренняя память подразделяется на биполярную, инжекционную и на МОП-транзисторах.

Кроме того, к внутренней памяти относятся безадресные ЗУ с ассоциативной выборкой, где поиск информации происходит по признакам хранимой информации.

Внешняя память дополняет и расширяет возможности внутренней памяти.

По функциональному назначению внешняя память подразделяется на буферные (БЗУ) и внешние (ВЗУ) запоминающие устройства.

Буферные ЗУ используются для временного хранения информации при обмене ею между внешними ЗУ и внутренней памятью. В них происходит накопление информации от внешних ЗУ с последующей быстрой передачей ее в быстродействующие ОЗУ.

Внешние ЗУ предназначены для хранения больших массивов информации, не участвующих в данный момент в вычислительном процессе. Они имеют большой объем памяти и низкое быстродействие; организуются на магнитных носителях информации — накопителях на магнитных лентах (НМЛ) и магнитных дисках (НМД).

Контрольные вопросы

1. Чем отличаются режимы работы ОЗУ и ПЗУ?
2. Каким образом производится запись информации в ОЗУ и ПЗУ?
3. Какому виду ЗУ соответствует источник информации, в котором хранятся: таблица умножения, результат только что подсчитанной стоимости покупки, стоимость покупки на выданном чеке, результаты промежуточных и конечных значений при решении простейших примеров в уме, конспект лекций, справочные таблицы, книги из читального зала библиотеки?
4. Чем отличается извлечение информации из памяти с последовательным и произвольным обращением?
5. Сравнить скорость поиска информации в ЗУ с произвольным и последовательным обращением.

Раздел VI
ЗАПОМИНАЮЩИЕ УСТРОЙСТВА

Глава 18
ПОЛУПРОВОДНИКОВЫЕ ЗАПОМИНАЮЩИЕ УСТРОЙСТВА

18.1. Классификация микросхем памяти

Благодаря новым разработкам и совершенствованию технологической полупроводниковой базы удалось создать ЗУ, в которых значительно увеличен объем памяти и повышено быстродействие. Емкость таких СБИС ЗУ может составлять от нескольких миллионов бит до нескольких гигабайт в одном корпусе. На рис. 18.1 приведена классификация микросхем памяти в зависимости от их функционального назначения.

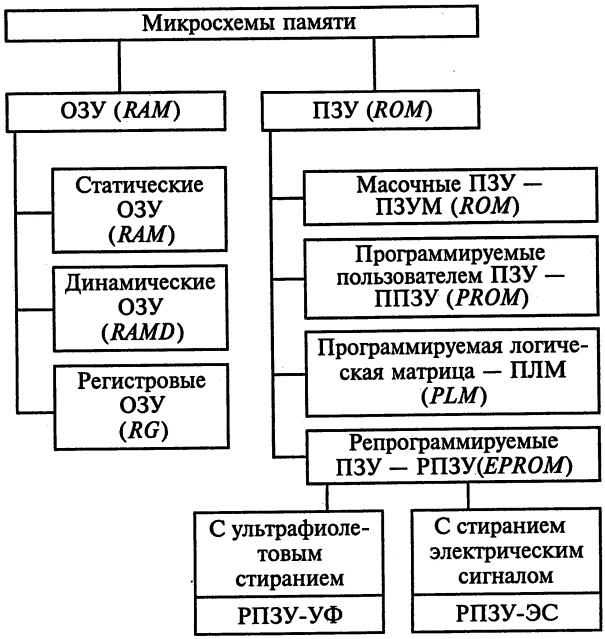


Рис. 18.1

18.2. Режимы работы и характеристики статических БИС ЗУ

Для хранения единицы информации — *бита* используется запоминающий элемент (ЗЭ). Совокупность ЗЭ, используемых для хранения многоразрядных чисел, называется *ячейкой памяти*, а размещаемая в ней информация — *словом*. Длина слов, размещаемых в памяти, может быть различна, но кратна 8. Минимальная длина слова составляет восемь бит или один байт.

Фактически все виды ЗУ работают в трех режимах: запись, хранение, считывание информации.

В режиме записи необходимо:

организовать обращение по адресу ячейки, где будет размещена информация;

подать записываемую информацию;

подать сигнал «Запись».

В режиме хранения обращение по адресу не производится, ячейки ЗУ оказываются недоступными.

В режиме считывания информации необходимо:

организовать обращение по адресу, где хранится информация;

подать сигнал «Чтение».

Организация обращения по адресу оказывается настолько важным параметром, что по затратам времени на поиск информации оценивают производительность ЭВМ. Процедура поиска по адресу определяется значением порядка 100 нс (1нс = 10^{-9} с).

Основными критериями оценки ЗУ являются показатели емкости, быстродействия и потребляемой мощности.

Емкость ЗУ характеризуется общим количеством бит информации, которое может храниться в ЗУ одновременно, а также может задаваться в виде общего объема памяти или в виде слов и их разрядности. Например, общий объем ЗУ составляет 256 бит, или объем ЗУ равен (32 × 8), т. е. ЗУ может хранить 32 слова по восемь разрядов каждое. Зная эти значения, можно определить емкость накопителя:

$$32 \times 8 = 256 \text{ бит.}$$

При определении емкости ЗУ используют более крупные единицы измерения возможных объемов информации, которые представлены в табл. 18.1.

Объем памяти определяется числом, кратным 2^n. Обычно большие размеры памяти характеризуются в этих единицах округленно:

один килобайт (Кбайт) равен 1024 байта;

один мегабайт (Мбайт) равен 1024 Кбайта;

один гигабайт (Гбайт) равен 1024 Мбайта.

Быстродействие ЗУ оценивается по двум параметрам: времени обращения и времени цикла.

Таблица 18.1

Единица измерения	Вес единицы измерения	Содержимое	
		в битах	в байтах
1 Кбит (килобит)	10^3 бит, 2^{10} бит	1024	—
1 Мбит (мегабит)	10^6 бит, 2^{20} бит	1048576	—
1 Кбайт (килобайт)	10^3 байт, 2^{10} байт	—	1024
1 Мбайт (мегабайт)	10^6 байт, 2^{20} байт	—	1048576
1 Гбайт (гигабайт)	10^9 байт, 2^{30} байт	—	107374176

Время обращения — это полный временной цикл записи информации или ее считывания:

$$t_{\text{обр. зп}} = t_A + t_{\text{зп}};$$

$$t_{\text{обр. сч}} = t_A + t_{\text{сч}} + t_{\text{вос}},$$

где $t_{\text{обр. зп}}$, $t_{\text{обр. сч}}$ — полный временной интервал соответственно записи и считывания; t_A — время поиска ячейки памяти по определенному адресу; $t_{\text{зп}}$ — время, затраченное на запись информации в найденную ячейку; $t_{\text{сч}}$ — время, затраченное на считывание информации из найденной ячейки; $t_{\text{вос}}$ — время восстановления считанной информации в случае ее разрушения (по необходимости).

Время цикла ($t_{\text{ц}}$) — это максимальный интервал времени между двумя обращениями. Частота обращения к ЗУ является величиной, обратной времени цикла: $f_{\text{обр}} = 1/t_{\text{ц}}$.

Время выборки ($t_{\text{в}}$). В схемах памяти, как правило, присутствует вход «Выбор микросхемы» — ВМ (*CS*). Сигнал, поданный на этот вход, выполняет роль строба при адресном обращении к ячейке памяти. Часто именно относительно этого сигнала оценивается быстродействие ЗУ.

Потребляемая мощность ЗУ характеризуется средней потребляемой мощностью при максимальной частоте обращения и потребляемой мощностью в режиме хранения. Эти параметры должны быть минимальными. Нередко для БИС ЗУ приводят два значения потребляемой мощности — в режиме обращения и в режиме хранения.

Необходимо иметь в виду, что емкость и потребляемая мощность ЗУ находятся, как правило, в обратной зависимости с быстродействием.

По технологическому исполнению различают БИС ЗУ биполярные, с использованием технологий ЭСЛ, ТТЛ, И^{2}Л, и БИС ЗУ на основе МОП-технологий с использованием структур *n*-МОП, КМОП.

В табл. 18.2 представлены сравнительные данные для ОЗУ на основе различных полупроводниковых технологий.

Таблица 18.2

Тип технологии	Характеристики		
	Время выборки, нс	Плотность размещения, бит/см^3	Энергопотребление, мВт
ЭСЛ	1,4...3	—	16...35
ТТЛ	10...100	—	2...40
ТТЛШ	4...20	До 200	2...16
И^{2}Л	10...100	—	10^{-4}...10^{-3}
КМОП	160	200...300	10^{-5}

Контрольные вопросы

1. В чем заключается сущность процесса обращения к ячейке памяти?
2. Чем отличается организация режимов «Запись» и «Чтение»?
3. Каков общий объем 16-разрядного ЗУ для хранения 32 слов?
4. Сколько бит содержит 1 Кбит и сколько байт — 3 Кбайта?
5. Из каких временных параметров складывается время обращения к ЗУ?
6. Почему ЗУ на КМОП-транзисторах нашли наиболее широкое распространение?
7. Какова взаимосвязь энергопотребления КМОП-структур и плотности размещения транзисторов на подложке микросхемы?

18.3. Организация статических ЗУ

Как правило, все интегральные схемы БИС ЗУ имеют одну из двух структур построения: словарную (однокоординатную) или матричную (двухкоординатную).

Словарная организация БИС ЗУ представлена на рис. 18.2. Такое ЗУ содержит m-разрядных ячеек памяти, состоящих из запоминающих элементов, которые имеют индивидуальные адреса. Кроме того, в его состав входит дешифратор адреса. Чтобы иметь возможность обратиться к одной из ячеек памяти, на вход дешифратора необходимо подать адрес этой ячейки памяти. В результате адресного обращения на выходе дешифратора возбуждается определенная, так называемая адресная шина, наличие сигнала на которой обеспечивает замыкание ключей доступа к ЗЭ выбранной ячейки памяти. Каждый из ЗЭ связан через адресный ключ доступа с соответствующей ему разрядной шиной. При адресном обращении в зависимости от режимов работы «Запись» или «Чтение» ЗЭ через ключи доступа подсоединяются к разрядным шинам. В результате в режиме «Запись» на разрядные шины подается код записываемого слова, а в режиме «Чтение» с разрядных шин считывается хранящаяся в ячейке информация. При хранении информации адресное обращение к ЗУ не производится, поэтому с помо-

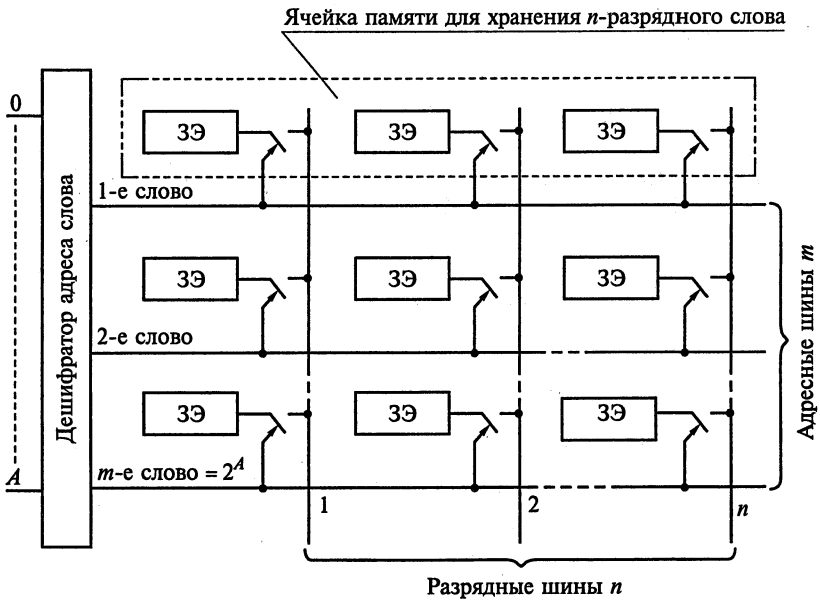


Рис. 18.2

щью разомкнутых ключей доступа ЭЭ оказываются отключенными от разрядных шин.

Из приведенной схемы видно, что один из параметров, характеризующих объем памяти — общее количество слов (ячеек памяти ЗУ), соответствует количеству выходных шин дешифратора адреса, следовательно, разрядность адреса A также косвенно характеризует емкость ЗУ.

Возможное количество слов m определяется по формуле $m = 2^A$, а общий объем памяти как $2^A \times n$, где m — общее количество слов (ячеек памяти); A — разрядность адреса; n — количество разрядов слова.

Например, если адрес пятиразрядный, а количество разрядов равно восьми, то ЗУ может иметь 32 восьмиразрядных запоминающих ячейки памяти, а общий объем памяти составит $m \times n = 2^A \times n = 2^5 \times 8 = 32 \times 8 = 256$ ЭЭ для хранения 256 бит информации.

Матричная (*двухмерная*) *организация БИС ЗУ* представлена на рис. 18.3.

При данной структуре построения ЗУ нагрузка на дешифратор адреса распределена. В результате разрядность каждого из дешифраторов становится в два раза ниже, соответственно и количество выходных шин каждого из дешифраторов снижается.

ЗУ состоит из групп разрядных матриц, количество которых соответствует количеству разрядов хранимых слов. Таким образом, на каждой разрядной матрице находятся ЗЭ одного разряда всех

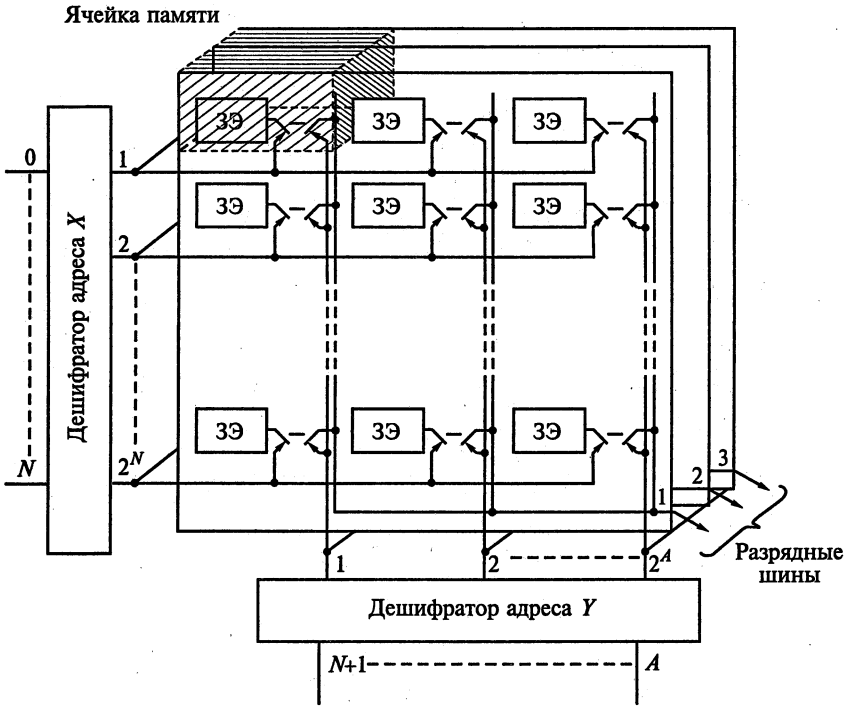


Рис. 18.3

слов, которые объединены с соответствующей этому разряду разрядной шиной. По количеству ЗЭ на одной матрице можно судить о количестве ячеек памяти для хранения информации, а по количеству матриц — о разрядности хранимых слов.

Исходный код адреса разбивается на две группы разрядов, дешифрация которых происходит в дешифраторах адреса X и Y. Адресные шины с выходов дешифраторов являются общими для всех разрядных матриц.

Доступ к каждому ЗЭ организован с помощью двух индивидуальных ключей, замыканием которых управляют сигналы с выходов дешифраторов. Таким образом, в момент обращения по адресу осуществляется доступ к запоминающим элементам, расположенным в одном месте, на всех разрядных матрицах.

Рассмотрим структуру ЗУ для хранения 64 пятиразрядных слов (64×5). Определим необходимое количество матриц и разрядность каждого из дешифраторов.

Количество разрядных матриц равно заданному количеству разрядов, т. е. пяти. Два дешифратора адреса на три разряда каждый по восемь адресных выходных шин (2^3) обеспечивают доступ к 64 ЗЭ на каждой матрице ($8X \times 8Y$).

Контрольные вопросы

1. Какова взаимосвязь между разрядностью адреса и количеством хранимых слов в ЗУ?
2. Какие устройства, установленные на входе дешифратора адреса могут являться источником адреса?
3. Какой элемент можно использовать в качестве адресного ключа доступа?
4. Какой должен быть объем памяти при словарной организации, если код адреса равен 6, а разрядность хранимых чисел равна 16?
5. Сколько ЗЭ должно быть на разрядной плате матричного ЗУ с объемом 1024×8?

Контрольные задания

1. Построить схему ЗУ со словарной организацией объемом 64×5.
2. Построить схему ЗУ с матричной организацией объемом 256×8; 16×4.

18.4. Структурная организация БИС ЗУ

Как правило, БИС ЗУ имеет матричный принцип построения. На рис. 18.4 представлена структурная организация БИС ЗУ.

Схема состоит из следующих типовых узлов:

накопителя, состоящего из ЗЭ;

дешифратора строк (X) — DC_X;

дешифратора столбцов (Y) — DC_Y;

усилителей записи;

усилителей считывания.

В зависимости от типа ЗУ (ОЗУ, ПЗУ, ППЗУ, РПЗУ) некоторые узлы в схеме могут отсутствовать. Например, в БИС ПЗУ отсутствуют усилители записи.

В табл. 18.3 приведены обозначения сигналов, используемых в УГО микросхем.

Имея условное графическое обозначение ЗУ (рис. 18.5), можно оценить структуру построения микросхем.

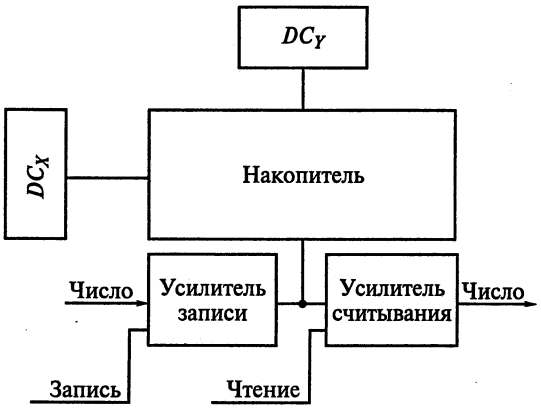


Рис. 18.4

Таблица 18.3

Наименование сигнала	Обозначение
Адрес	*A*
Тактовый сигнал	*C*
Строб адреса столбца	*CAS*
Строб адреса строки	*RAS*
Выбор микросхемы	*CS (V)*
Разрешение	*CE*
Запись	*WR*
Считывание	*RD*
Запись-считывание	*W/R*
Разрешение записи	*WE*
Разрешение по выходу (считывание)	*OE*
Данные (информация)	*D*
Входные данные	*DI*
Выходные данные	*DO*
Регенерация	*REF*
Программирование	*PR*
Стирание	*ER*
Напряжение питания	U_{cc}
Напряжение программирования	U_{PR}
Общий вывод микросхемы	U_{GC} *(0V)*

Из представленного УГО РУ1 (см. рис. 18.5, *а*) понятно, что микросхема имеет матричную структуру, так как ее адрес разделен на две группы *X* и *Y*.

Внутри микросхемы дешифраторы адреса отсутствуют. Микросхема имеет два информационных разрядных входа для записи парафазной информации одного разряда (*W0*, *W1*). Емкость такого ЗУ оценивается как $(A_X \times A_Y) \times 1$ разряд, где A_X и A_Y — коли-

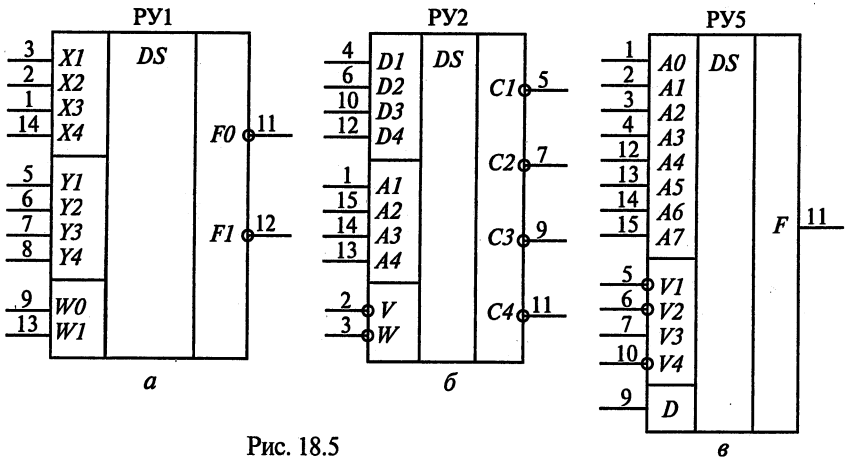


Рис. 18.5

чество адресных шин X и Y, т.е. емкость ЗУ равна $(4 \times 4) \times 1 =$ $= 16$ бит $\times 1$ разряд $= 16$ бит.

Из представленного УГО РУ2 (см. рис. 18.5, *б*) понятно, что дешифратор адреса находится внутри микросхемы. Микросхема имеет четыре информационных входа (*D1*... *D4*) для записи четырехразрядных слов. Емкость такого ЗУ оценивается как

$$(m \times n) = 2^A \times n = 2^4 \times 4 = 16 \times 4 = 64 \text{ бит},$$

где A — количество разрядов адреса; m — количество n-разрядных слов, равное 2^A.

Из представленного УГО РУ5 (см. рис. 18.5, *в*) понятно, что дешифратор адреса находится внутри микросхемы. Микросхема имеет один информационный вход (*D*). Емкость такого ЗУ оценивается как

$$(m \times n) = 2^A \times 1 = 2^8 \times 1 = 256 \times 1 = 256 \text{ бит}.$$

Примеры организации микросхем ЗУ. Рассмотрим принцип построения нескольких микросхем. На рис. 18.6 представлена микросхема РУ1.

Матрица включает в себя 16 ЗЭ на основе триггеров, два усилителя записи и два усилителя считывания. Все 16 выводов *W0* и *W1* — записи лог. 1 и записи лог. 0 и все 16 выводов $\overline{F0}$ и $\overline{F1}$ — считывания лог. 0 и считывания лог. 1 объединены разрядными шинами с соответствующими выходами усилителей записи и входами усили-

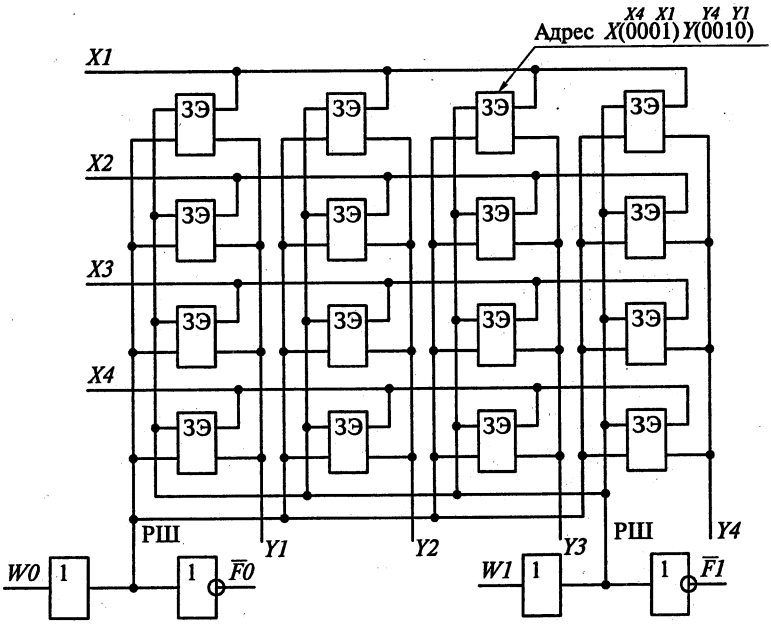


Рис. 18.6

телей считывания. Запись информации в ЗЭ матрицы происходит при адресном обращении путем подачи лог.1 на одну из адресных шин X_i (*X1*... *X4*) и Y_i (*Y1*... *Y4*), на пересечении которых расположен выбранный ЗЭ. На остальных адресных шинах при этом должны присутствовать лог. 0. Запись бита информации в выбранный ЗЭ осуществляется при подаче на входы *W0* и *W1* парафазной информации: при записи нуля — лог. 0 и лог. 1 соответственно; при записи единицы — лог. 1 и лог. 0.

При считывании, как и при записи, необходимо организовать обращение по адресу *X* и *Y*, при этом на входах *W0*, *W1* необходимо обеспечить лог. 0. Считанная информация появится на парафазных, инверсных выходах $\overline{F0}$ и $\overline{F1}$, что отражено в УГО.

Недостатком данной схемы является отсутствие защиты запоминающих элементов в момент переключения адресов и подачи различных числовых комбинаций записываемой информации. При подготовке обращения к ОЗУ необходимо организовать защиту адресных и числовых шин. В эти моменты времени на входах должны присутствовать лог. 0. Это можно обеспечить с помощью дополнительной входной логики *DD1*... *DD3* (рис. 18.7). Тогда при записи доступ к ЗЭ ОЗУ будет возможен только при наличии двух сигналов: «Разрешение записи», «Разрешение адреса». Схему можно упростить (рис. 18.8), если в процессе между двумя обращениями организовать режим полувыборки (табл. 18.4).

При этом в промежутках между обращениями к ОЗУ в процессе изменения информации с помощью управляющего сигнала «Разрешение записи» осуществляется отключение только числовых шин (на числовых шинах лог. 0). При переключении адресов на адресных шинах X_i и Y_i необходимо обеспечить сочетание сигналов 0 и 1 (или наоборот), т. е. состояние аналогичное режиму хранения, при котором на выходах микросхемы ОЗУ — $\overline{F0}$ и $\overline{F1}$ будут лог. 1.

На рис. 18.9 представлена микросхема РУ2, а в табл. 18.5 — режимы ее работы.

Микросхема состоит: из накопителя *DD8*, дешифратора адреса *DD6*, входных информационных ключей *DD2*... *DD5*, усилителя записи *DD7*, выходных усилителей считывания *DD9*... *DD12*.

Рассмотрим работу микросхемы в различных режимах. В микросхеме расположен четырехразрядный дешифратор адреса *DD6*, стробируемый сигналом $\overline{V}$ — «Выбор микросхемы». Сигнал на этом входе разрешает работу с микросхемой ЗУ. Обращение по адресу возможно только при подаче на вход *V* лог. 0. Таким образом, при значении сигнала на входе *V*, равном лог. 1, микросхема недоступна и находится в режиме хранения ранее записанной информации. Кроме того, вход *V* используется для организации ЗУ большого объема. Этот вопрос будет рассмотрен далее.

Управление усилителями записи и считывания происходит инверсными значениями одного сигнала *W* — «Запись/чтение».

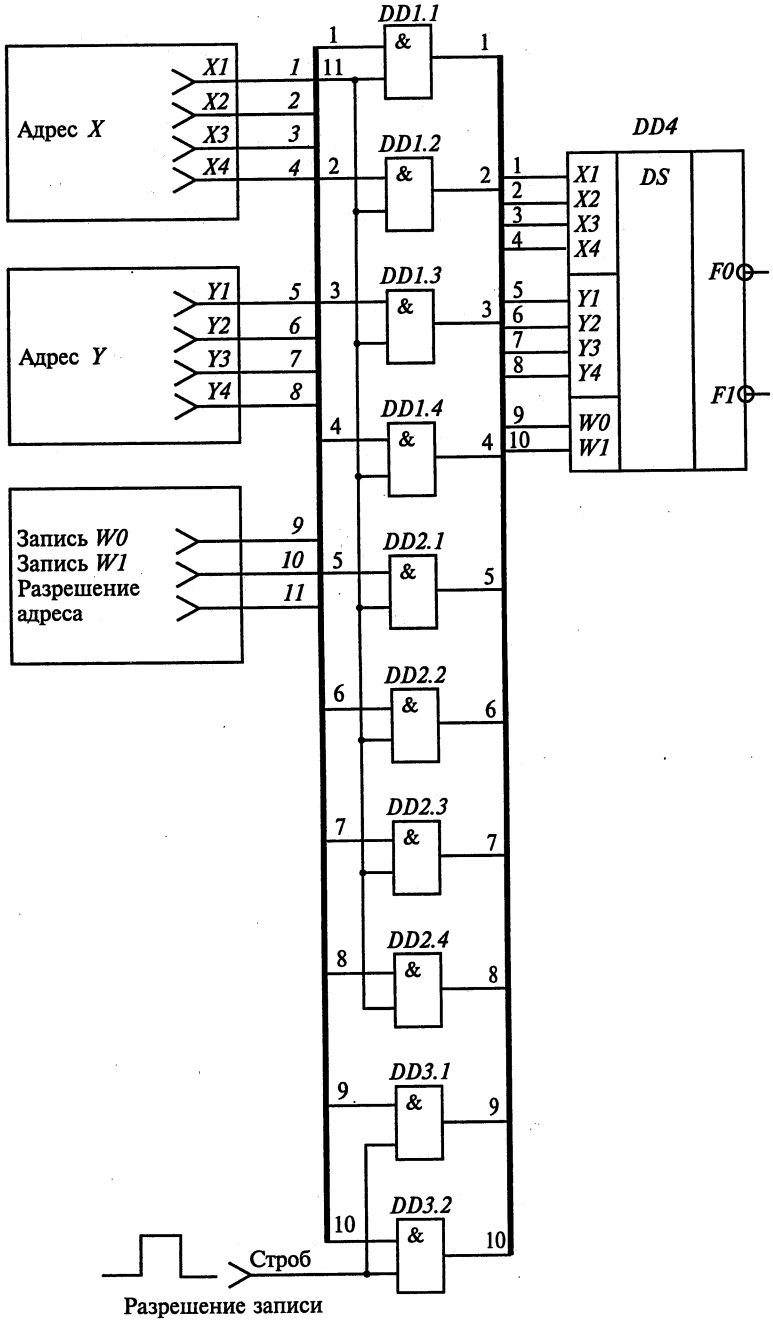

DD1.1
DD1.2
DD1.3
DD1.4
DD2.1
DD2.2
DD2.3
DD2.4
DD3.1
DD3.2
DD4
DS
Адрес X
X1
X2
X3
X4
Адрес Y
Y1
Y2
Y3
Y4
Запись W0
Запись W1
Разрешение адреса
W0
W1
F0
F1
Строб
Разрешение записи

Рис. 18.7

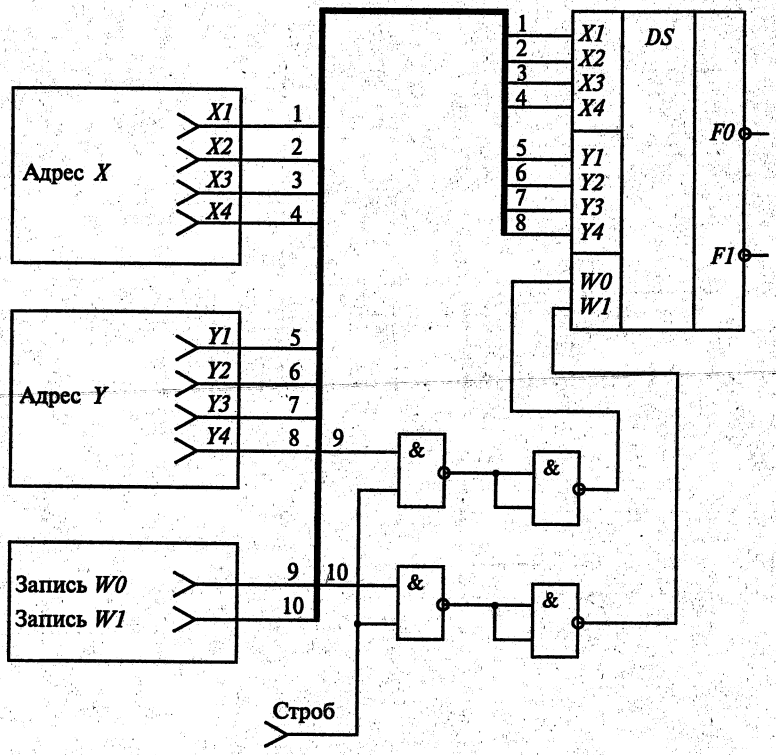


Рис. 18.8

Таблица 18.4

Входные сигналы				Выходные сигналы		Выполняемая операция
Адреса		Числа				
X_i	Y_i	$W1$	$W0$	$\bar{F}1$	$\bar{F}0$	
0	0	0	0	1	1	Режим хранения
1	1	1	0	0	1	Запись 1
1	1	0	1	1	0	Запись 0
1	1	0	0	0	1	Считывание 1
1	1	0	0	1	0	Считывание 0
0	1	*	*	1	1	Режим полувыборки
1	0	*	*	1	1	

* — Значения сигналов безразличны.

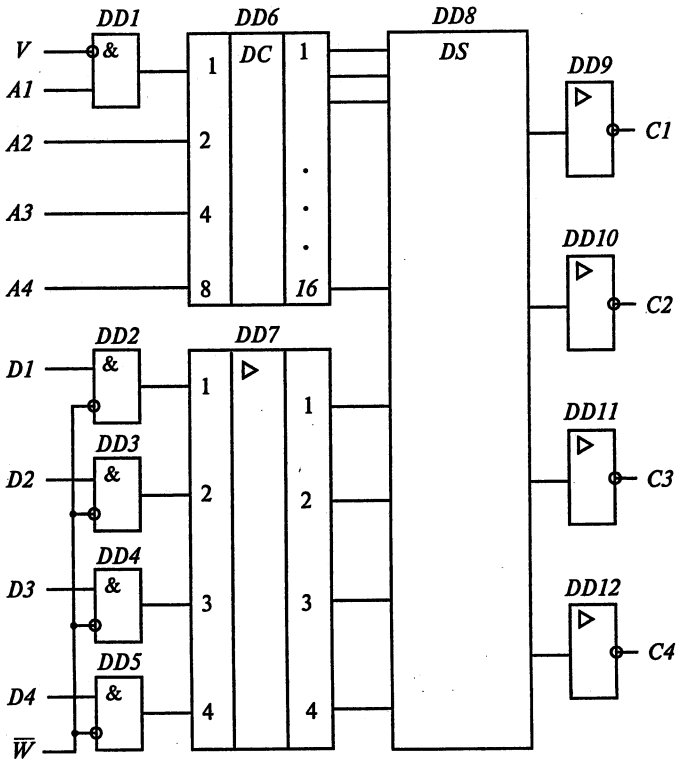


Рис. 18.9

Записываемые данные (*D1*... *D4*) проходят в усилитель записи *DD7* и далее в матрицу памяти *DD8* только при наличии на входе *W* разрешающего сигнала, равного лог. 0. При считывании управляющий сигнал *W* должен иметь значение лог. 1. В этом случае усилители записи становятся недоступны для информации, а из накопителя через усилители считывания происходит передача информации на выход микросхемы.

Таблица 18.5

V	*W*	*D*	Режим работы	*C*
1	1	0	Хранение	1
0	1	0	Считывание	Инверсный код выбранного числа
0	0	1	Запись 1	0
0	0	0	Запись 0	1
1	0	1	Запрет	0
1	0	0	Обращение	1

Контрольные вопросы

1. Для чего предназначены входы *W*, *V*, *WR*, *RD*, *A*, *CS*, *D* запоминающих устройств?
2. Какому типу ЗУ соответствуют обозначения *ROM*, *PROM*, *RAM*?
3. Сколько входов *D* имеет ЗУ емкостью 256 × 1?
4. Для чего предназначены усилители записи и чтения в схеме ЗУ РУ1?
5. Можно ли обратиться в ЗУ РУ1 по следующим адресам: 0110 0100; 0001 0000; 0100 0100?
6. Какие сигналы необходимы для обеспечения записи и считывания информации в микросхеме РУ1?
7. Какие сигналы необходимы для организации режима хранения в микросхеме РУ1?
8. Возможна ли запись информации в ЗУ РУ2 при следующих значениях сигналов на входах микросхемы: $V = 0$, $W = 1$ (см. рис. 18.9)?
9. В каком состоянии может находиться микросхема ЗУ РУ2 при поступлении разрешающего сигнала на вход *V*? Чему равно значение этого сигнала?
10. Под действием каких сигналов происходит считывание информации в ЗУ РУ2?

18.5. Построение разных видов запоминающих элементов ОЗУ

ЗЭ биполярного ОЗУ. Статические запоминающие элементы представляют собой триггеры, различие которых зависит от используемой базовой технологии. На рис. 18.10, *а* показана схема ЗЭ — триггера, выполненного на двух многоэмиттерных биполярных транзисторах (МЭТ) *VT1* и *VT2*, а на рис. 8.10, *б* — диаграмма его работы. Такой ЗЭ помимо разрядного и адресного входов имеет дополнительный вход, на который подается опорное напряжение $U_{оп}$.

Отметим ключевые моменты в работе схемы.

1. Данный ЗЭ может быть использован при словарной организации ЗУ, когда имеется одна адресная шина.
2. Внешние воздействия, влияющие на состояние триггера, подаются на разрядную шину и изменяют свое значение относительно опорного напряжения $U_{оп}$.
3. В режиме «Запись/чтение» на адресной шине должен присутствовать высокий уровень напряжения, соответствующий лог. 1, в режиме «Хранение» — низкий уровень напряжения, соответствующий лог. 0.
4. Открытое состояние транзистора *VT2* и наличие тока в разрядной шине принимается за лог. 1, открытое состояние транзистора *VT1* и отсутствие тока в разрядной шине — за лог. 0.

Режим «Запись». На выбранную адресную шину подается высокий, соответствующий лог.1 (выше $U_{оп}$) уровень напряжения. Режим записи сводится к переключению токов из транзистора *VT1* в *VT2*, или наоборот. Это осуществляется за счет изменения соотношения потенциалов переходов база—эмиттер (Б—Э) одного из транзисторов. На переход Б—Э2 *VT1* подается напряже-

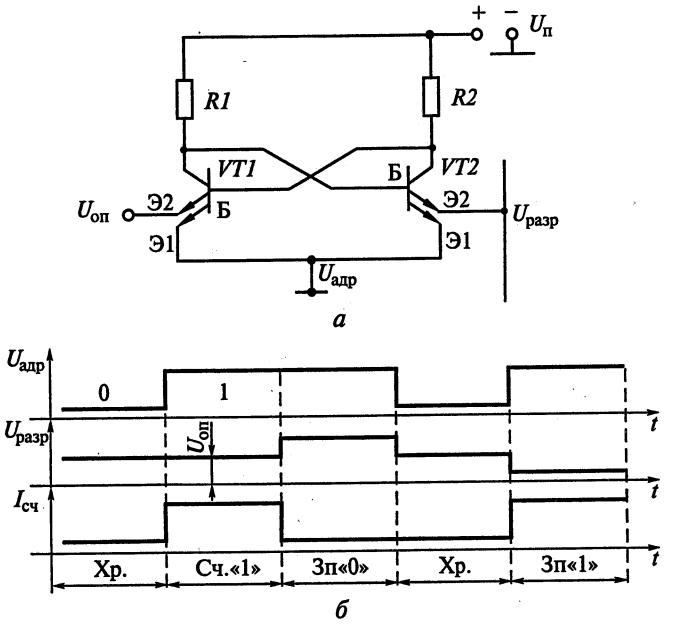


Рис. 18.10

ние $U_{оп}$. Воздействовать на транзистор *VT2* можно, подав на переход Б—Э2 *VT2* напряжение больше или меньше $U_{оп}$. В первом случае транзистор *VT2* откроется больше чем *VT1*. Произойдет переключение токов в *VT2*, в результате чего триггер примет устойчивое состояние, соответствующее лог. 1. Во втором случае произойдет переключение токов в *VT1*, в результате чего произойдет запись лог. 0.

Режим «Хранение». Основной целью режима является сохранение предыдущего состояния. Это достигается выравниванием условий работы переходов Б—Э2 обоих транзисторов, для чего на разрядную шину подается напряжение, равное $U_{оп}$. На адресную шину подается низкий уровень напряжения, соответствующий лог. 0. Таким образом осуществляется равновесие и обеспечивается отсутствие причин, которые привели бы к изменению состояния триггера. Может происходить только переключение тока из цепи Б—Э2 в цепь Б—Э1 одного из транзисторов в зависимости от хранимой информации.

Режим «Считывание». Этот режим сводится к анализу наличия тока в разрядной шине и он аналогичен режиму хранения. При организации процесса необходимо переключить токи из цепи Б—Э1 в цепь Б — Э2 одного из транзисторов в зависимости от хранимой информации. Для этого повышается уровень напряжения на адрес-

ной шине и определяется наличие или отсутствие тока в разрядной шине.

Статические БИС ЗУ (ТТЛ) среднего быстродействия имеют время обращения до сотен наносекунд. Так как в основе ЗЭ на ТТЛ лежит триггер, состоящий из двух инверторов, то в любом из логических состояний происходит потребление мощности, которое практически не зависит от режима работы. Это сказывается на плотности размещения ЗЭ на подложке микросхемы, т.е. на емкости БИС ЗУ.

ЗЭ на КМОП-транзисторах. КМОП-структуры являются доминирующим типом схем при изготовлении БИС ЗУ вследствие сверхмалой потребляемой ими мощности в режиме хранения и высокой помехоустойчивости, а следовательно, и повышенной емкости.

На рис. 18.11 представлены два варианта ЗЭ, которые можно использовать при матричной (см. рис. 18.11, *а*) и словарной (см. рис. 18.11, *б*) организации статических ЗУ.

Рассмотрим ЗЭ, используемый при матричной организации ЗУ.

Отметим ключевые моменты в работе схемы.

1. Запоминающий элемент — триггер построен на МОП-транзисторах с взаимодополняющей структурой *VT1*, *VT2* и снабжен адресными ключами доступа *X* и *Y* — *VT3*, *VT4*.

2. Условием образования канала и передачи информации от истока к стоку между затвором и истоком МОП-транзистора является наличие разности потенциалов.

3. Доступ в триггер при записи и чтении информации осуществляется при подаче на адресные шины *X* и *Y* лог. 1.

В элементе памяти имеется пара информационных разрядных входов *D0* и *D1*, на которые подается парафазным кодом записываемая информация или с которых производится считывание информации.

Режим «*Запись*». При записи информации парафазная информация подается на входы *D0* и *D1* (лог. 0 и лог. 1 соответственно); при адресном обращении на шинах *X* и *Y* — лог.1. Наличие разности потенциалов только на левой паре адресных ключей *VT4′* и *VT3′* приводит к образованию канала в них и передаче информации с входов *D0* в точку $\bar{Q}$, связанную с входами — затворами правого инвертора *VT1″* и *VT2″* триггера. Наличие разности потенциалов только на транзисторе *VT1″* приводит к образованию канала в нем и передаче в точку *Q* и на затворы левой пары транзисторов *VT1′* и *VT2′* лог. 1. Образовавшийся канал в транзисторе *VT2′* левой пары способствует передаче лог. 0 в точку $\bar{Q}$ и на затворы транзисторов *VT1″* и *VT2″*. При изменении значений сигналов на информационных входах окажутся замкнутыми ключи доступа *VT4″*, *VT3″*, и лог. 0 пройдет на затворы транзисторов *VT1′* и *VT2′* триггера.

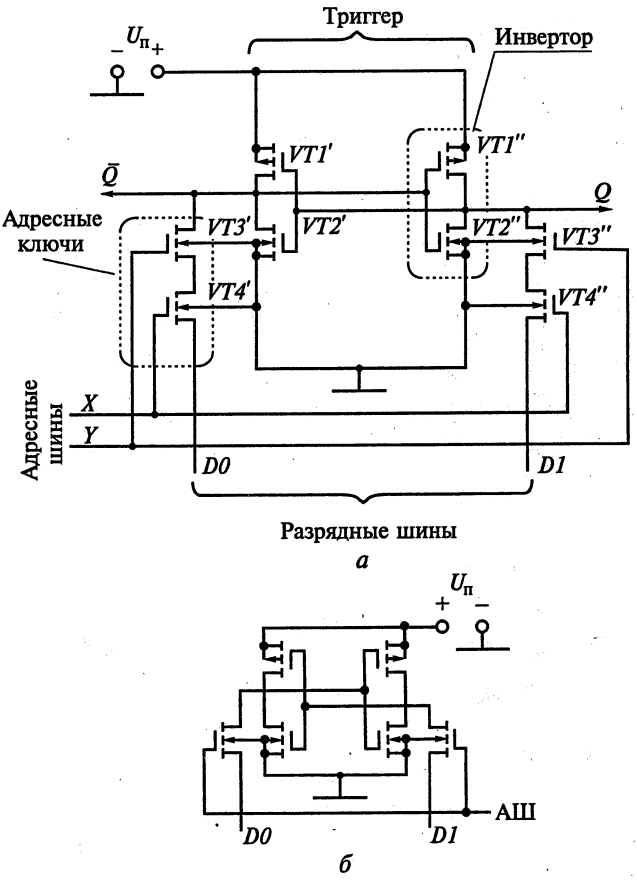


Рис. 18.11

Режим «Хранение». В данном режиме адресное обращение к ЗЭ не производится, ключи доступа должны быть разомкнуты, что достигается подачей лог. 0 на адресные шины *X* и *Y*.

Режим «Считывание». С учетом того, что разрядные шины с входами *D0* и *D1* используются как для записи, так и для считывания информации, управление потоком информации осуществляется через внешние ключи-усилители (записи—считывания) (рис. 18.12, *а*). При этом ключи записи размыкаются, а ключи считывания замыкаются. Источником информации в этом случае является триггер; происходит изменение расположения истока и стока в адресных ключах *VT3* и *VT4*.

Малая потребляемая мощность ЗЭ на КМОП-транзисторах объясняется тем, что в любом состоянии в инверторах триггера всегда открыт только один из транзисторов пары. Во втором транзисторе канал не образуется, а цепь для протекания тока оказывается ра-

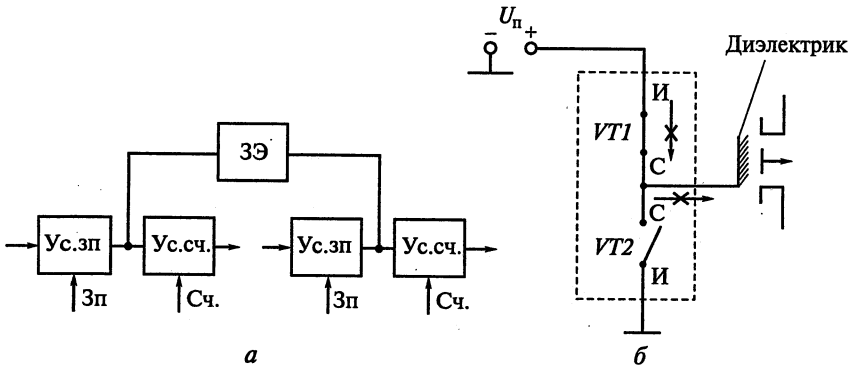


Рис. 18.12

зомкнутой. Кроме того, нагрузкой таких схем являются также МОП-транзисторы (рис. 18.12, *б*) с изолированным диэлектриком затвором, т. е. на нагрузке тока также не будет.

ЗЭ динамического типа. Информация в таких ЗЭ хранится в виде заряда на конденсаторе. Простота схемных решений позволяет им занимать меньшую площадь на кристалле микросхемы и тем самым увеличивать объем БИС ЗУ. Фрагмент такого БИС ЗУ представлен на рис. 18.13, *а*, а его УГО — на рис. 18.13, *б*.

Рассмотрим организацию ЗЭ на одном транзисторе. В качестве ЗЭ используется конденсатор, одно из состояний которого принято за 1, а другое — за 0. При адресном обращении ЗЭ (конденсатор) в точке *A* подсоединяется через адресные ключи к разрядной шине (РШ). При записи лог. 1 на РШ подается высокий потенциал, заряжающий конденсатор, а при записи лог. 0 — низкий по-

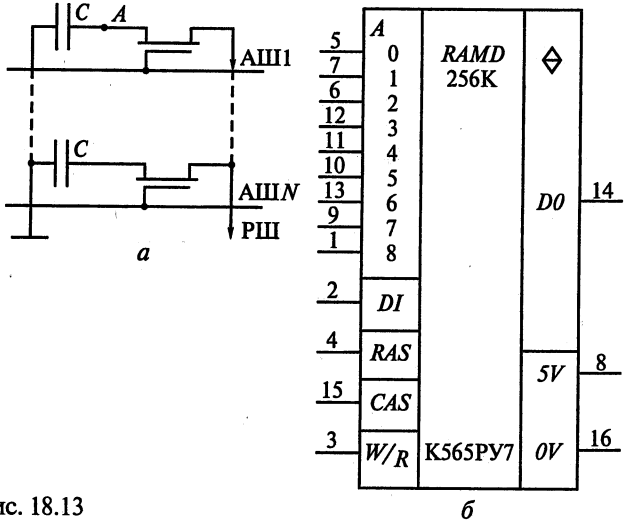


Рис. 18.13



Рис. 18.14

тенциал, разряжающий конденсатор. Наличие тока в разрядной шине при разряде конденсатора воспринимается как лог. 1.

При считывании происходит опрос состояния конденсатора. Состояние ЗЭ при считывании определяется по наличию или отсутствию тока считывания в разрядной шине. Наличие тока в разрядной шине при разряде конденсатора воспринимается как лог. 1. Считывание происходит с разрушением информации, поэтому необходимо время для ее восстановления. Такой процесс называется *регенерацией*. Кроме того, в любом случае в таких ЗУ периодически необходимо производить самостоятельный процесс опроса и подзарядки конденсаторов в зависимости от хранимой информации.

Часто в качестве ЗЭ (конденсатора) используется емкость обедненного слоя затвор — подложка МОП-транзистора (рис. 18.14).

Контрольные вопросы

1. Каким образом производится режим записи лог.1 в ЗЭ на биполярных транзисторах?
2. В чем заключается сущность режима хранения информации в запоминающих элементах?
3. Происходит ли разрушение информации в ЗЭ на биполярных транзисторах при считывании?
4. Меняется ли потребление мощности в ЗЭ на биполярных транзисторах в режимах хранения и считывании?
5. Как взаимосвязаны потребляемая мощность ЗЭ и плотность размещения хранимой информации на кристалле?
6. Почему ЗЭ на КМОП-транзисторах получили широкое распространение?
7. Может ли протекать ток от затвора МОП-транзистора к истоку (стоку)?
8. При каких условиях открывается МОП-транзистор? Какие процессы в нем происходят в этот момент?
9. Как организуется режим считывания в ЗЭ на КМОП-транзисторах?
10. С помощью какого элемента удается хранить информацию в ЗУ динамического типа?
11. Чем различаются режимы считывания и регенерации в ЗЭ динамического типа?
12. Как обеспечивается режим хранения в ЗЭ динамического типа?
13. Как на базе МОП-транзистора можно организовать накапливающий конденсатор?

18.6. Организация модулей ЗУ

На основе БИС ЗУ можно организовать ЗУ большого объема, построить модуль ЗУ. В зависимости от структуры БИС ЗУ возможны три варианта увеличения емкости: за счет увеличения разряд-

ности слов, количества слов, а также разрядности и количества слов. Рассмотрим каждый из этих вариантов.

Построение модуля памяти за счет увеличения разрядности слов. Например, микросхема ЗУ имеет объем 256 одноразрядных слов. Необходимо организовать ЗУ емкостью 256 восьмиразрядных слов (рис. 18.15) или накопитель должен иметь емкость $256 \times 8 = 2048$ бит информации.

Для организации такого ЗУ потребуется восемь БИС ЗУ 256×1, которые определят количество групп разрядов, соответствующих разрядности ЗУ. Для адресации к такому объему памяти потребуется восемь разрядов адреса:

$$2^A = 256;\ A = 8.$$

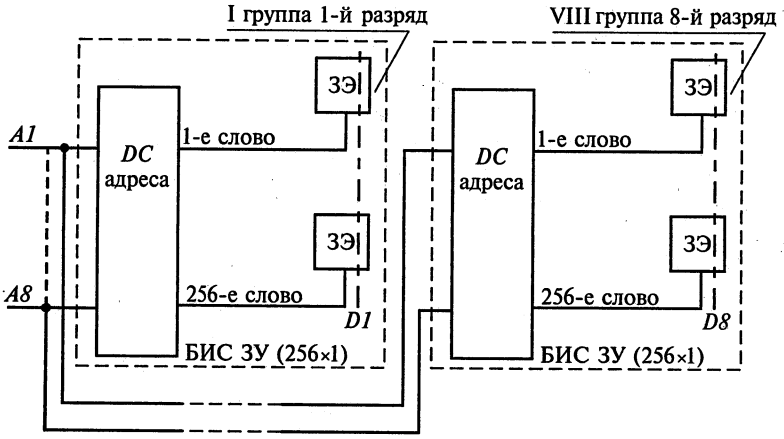


Рис. 18.15

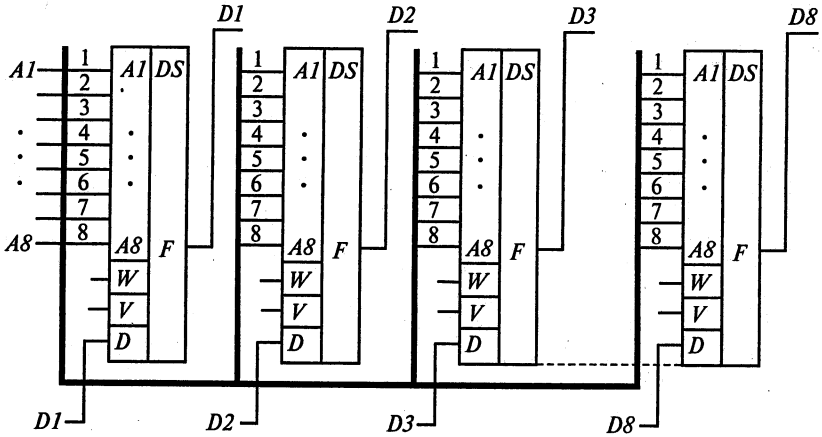


Рис. 18.16

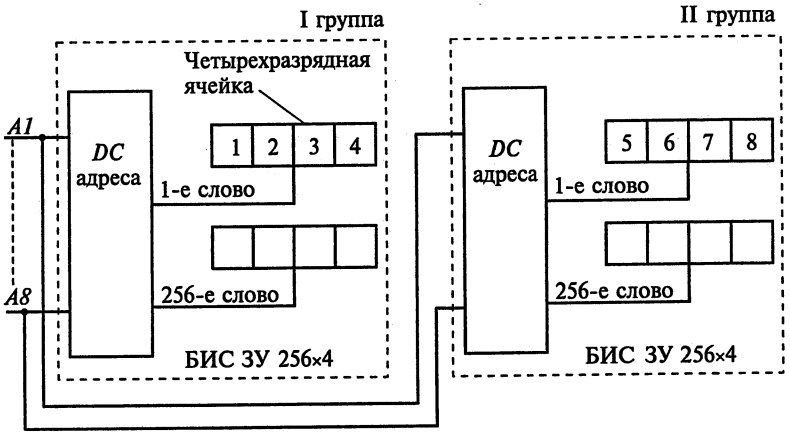


Рис. 18.17

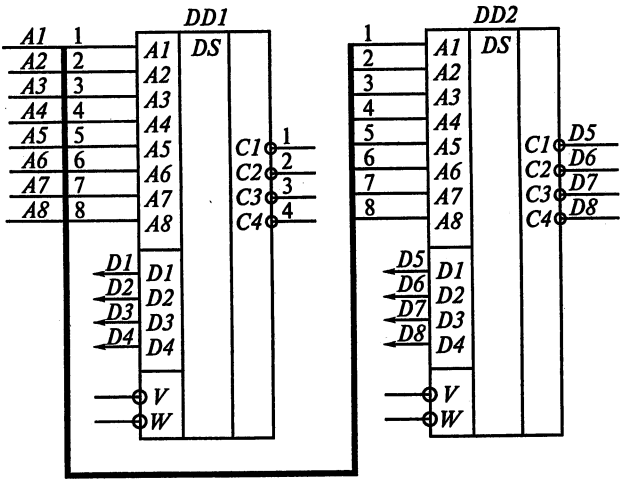


Рис. 18.18

Увеличение количества разрядов ЗУ осуществляется за счет объединения адресных входов БИС ЗУ. Информационные (разрядные) входы и выходы являются входами и выходами модуля ЗУ увеличенной разрядности (рис. 18.16).

Другой пример. Микросхема ЗУ имеет объем памяти 256 четырехразрядных слов. Необходимо организовать ЗУ емкостью 256 восьмиразрядных слов (рис. 18.17) или накопитель должен иметь емкость $256 \times 8 = 2048$ бит информации. Для организации такого ЗУ потребуется две БИС ЗУ 256×4, которые определят количество групп разрядов. Для обращения к такому объему памяти потребуется восемь разрядов адреса:

$$2^A = 256;\ A = 8.$$

Увеличение количества разрядов ЗУ как и в предыдущем примере осуществляется за счет объединения адресных входов ЗУ. Дешифрация адреса происходит внутри каждой микросхемы памяти (рис. 18.18).

Построение модуля памяти за счет увеличения количества слов. Например, микросхема ЗУ имеет объем 16 четырехразрядных слов. Необходимо организовать ЗУ емкостью 64 четырехразрядных слова. Для того чтобы иметь возможность обращения к одной из 16 запоминающих ячеек микросхемы памяти, необходим четырехразрядный адрес. Для увеличения емкости ЗУ до 64 слов необходимо иметь четыре таких микросхемы с последовательно нарастающим шестиразрядным адресом. Следовательно, первая БИС ЗУ должна быть доступна для адресов, содержащих в пятом и шестом разрядах 00; вторая — 01; третья — 10; четвертая — 11. Выбор одной из четырех микросхем происходит с использованием входов «Выбор микросхемы» (*V*), сигналы для которых формируются на дополнительном дешифраторе пятого и шестого разрядов адреса (рис. 18.19).

Четырехразрядные входные данные *D* и выходные данные *C* имеют у всех четырех микросхем общие цепи.

Построение модуля памяти за счет увеличения разрядности и количества слов. При организации таких ЗУ объединяются ранее рассмотренные способы построения модулей памяти: увеличение разрядности (увеличение БИС ЗУ по горизонтали) и увеличение количества слов (увеличение БИС ЗУ по вертикали).

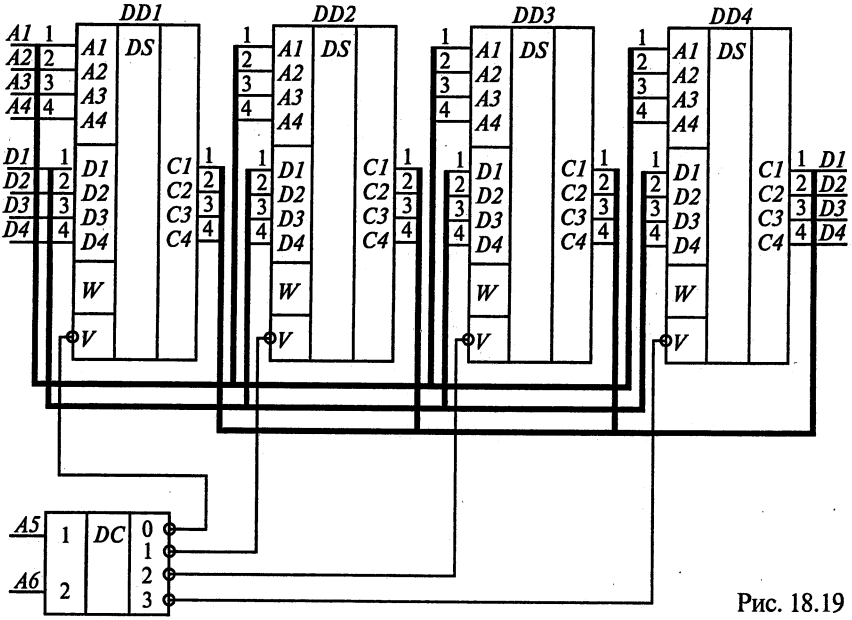


Рис. 18.19

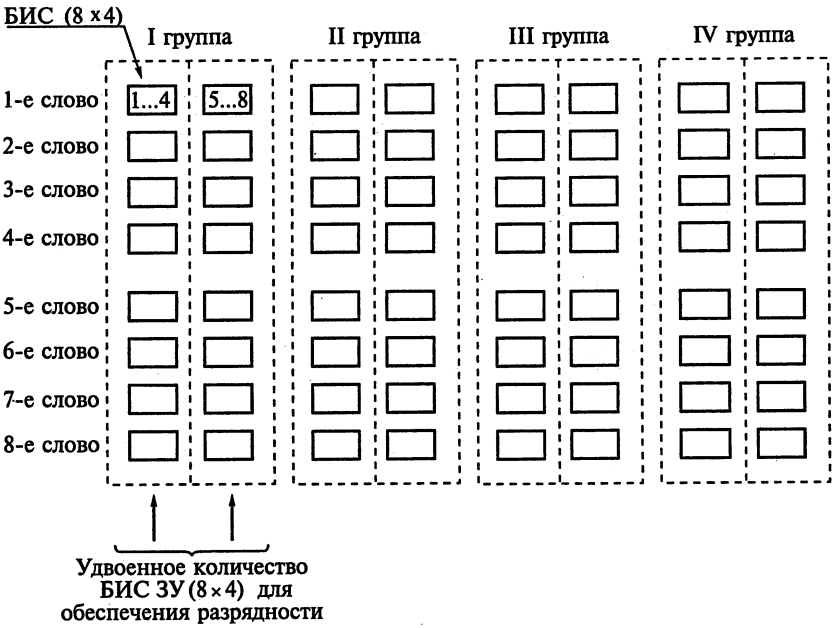


Рис. 18.20

Например, микросхема ЗУ имеет объем памяти 8 четырехразрядных слов. Необходимо организовать ЗУ емкостью 32 восьмиразрядных слова или накопитель должен иметь емкость 32 слова × 8 разрядов = 256 бит информации (рис. 18.20).

Организация восьмиразрядной ячейки потребует наращивания разрядности, для чего используются две четырехразрядных микросхемы, образующих одну группу ЗУ емкостью 8 слов × 8 разрядов. Для увеличения количества слов до 32 необходимы еще три группы ЗУ емкостью 8 слов × 8 разрядов каждая (II... IV). Соответственно в них будут находиться ячейки для хранения слов с 9 по 16, с 17 по 24 и с 25 по 32.

Чтобы иметь возможность обращения к одной из восьми запоминающих ячеек микросхемы памяти, необходим трехразрядный адрес. Для увеличения емкости ЗУ до 32 слов необходимо иметь четыре таких микросхемы с последовательно нарастающим пятиразрядным адресом. Следовательно, первая группа, состоящая из двух БИС ЗУ, образующая восьмиразрядные ячейки, должна быть доступна для адресов, содержащих в четвертом и пятом разрядах 00, вторая — 01, третья — 10, четвертая — 11. Выбор одной из групп происходит с использованием входов «Выбор микросхемы» ($\overline{V}$), сигналы для которых формируются на дополнительном дешифраторе четвертого и пятого разрядов адреса (рис. 18.21).

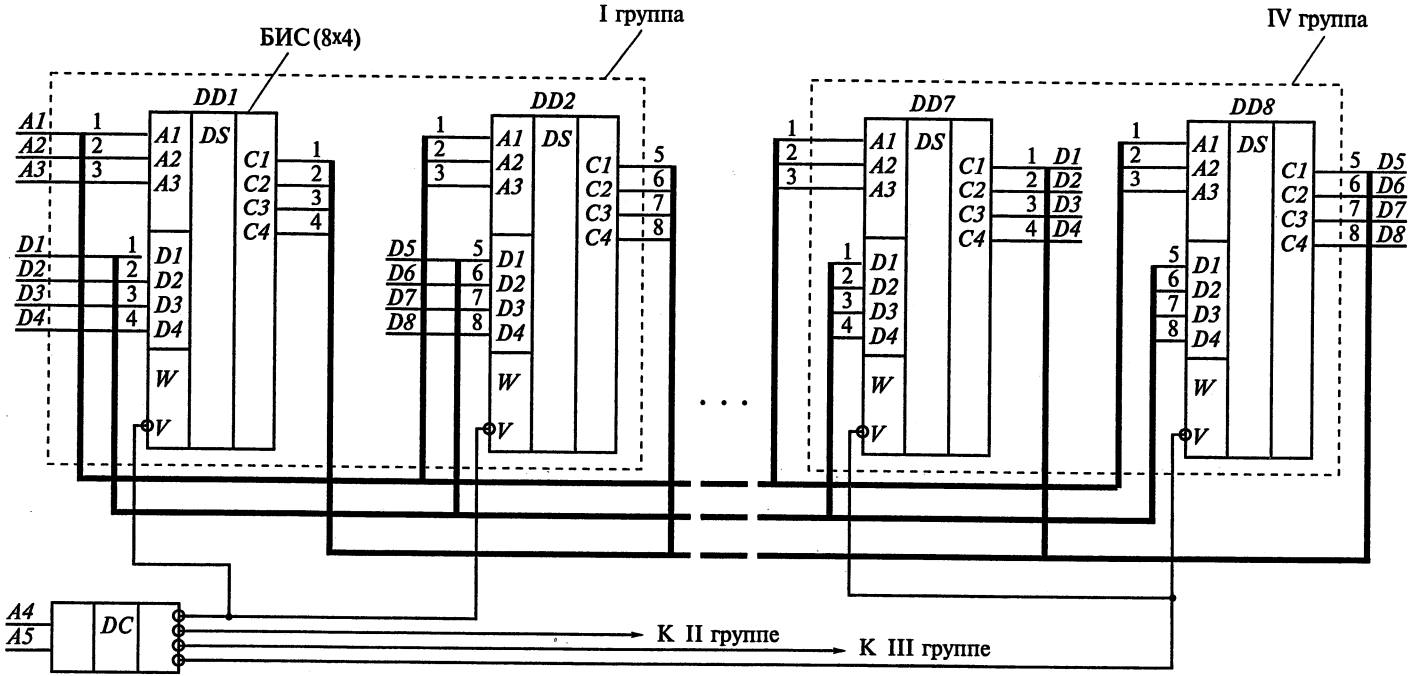

I группа
IV группа
БИС (8x4)
DD1
DD2
DD7
DD8
A1
A2
A3
D1
D2
D3
D4
DS
C1
C2
C3
C4
W
V
D5
D6
D7
D8
A4
A5
DC
К II группе
К III группе

Рис. 18.21

Контрольные задания

1. Построить ЗУ емкостью 128 четырехразрядных слов с использованием БИС ЗУ:

$$128 \times 1;\ 64 \times 4;\ 64 \times 2;\ 32 \times 4.$$

2. Построить ЗУ емкостью 16 восьмиразрядных слов с использованием БИС ЗУ:

$$16 \times 1;\ 4 \times 8;\ 8 \times 4;\ 16 \times 4.$$

Глава 19
ПОСТОЯННЫЕ ЗАПОМИНАЮЩИЕ УСТРОЙСТВА

19.1. Общие сведения

Постоянные запоминающие устройства предназначены для хранения постоянной или редко меняющейся информации: таблиц, функций, констант, управляющей информации. Информация в ПЗУ сохраняется при отключении источника питания.

ПЗУ создаются на основе полупроводниковых БИС. Организация БИС ПЗУ схожа с организацией БИС ОЗУ, в которой отсутствуют элементы схемы, связанные с записью информации при выполнении вычислительного процесса. Основным элементом, используемым для обеспечения хранения информации, является перемычка на определенном участке электрической цепи. Действительно, при наличии перемычки возникает цепь для протекания тока (состояние лог. 1); при отсутствии перемычки цепь для протекания тока отсутствует (состояние лог. 0). Процесс организации перемычек называется программированием ПЗУ.

В ПЗУ запись информации (программирование) происходит заранее, вне вычислительного устройства с применением дополнительных технологических операций, таких как напыление перемычек или их разрушение на специальной установке, называемой программатором. Поэтому процесс программирования ПЗУ сводится к нескольким вариантам.

1. При изготовлении масочных ПЗУ (ПЗУМ) на заводе-изготовителе производится нанесение перемычек в нужных участках схемы с помощью фотошаблонов — масок, которые делают по заказу пользователя микросхемы. Этот способ программирования является самым дешевым и предназначен для крупносерийного производства. ПЗУМ изготавливают по ТТЛ, ТТЛШ, *n*-, *p*-канальной МОП и КМОП-технологиям.

2. ПЗУ, программируемые пользователем (ППЗУ), отличаются тем, что микросхема поступает пользователю с полным набором возможных перемычек, и пользователь программирует (пережига-

ет) перемычки на специальных установках в соответствии со своими задачами. ППЗУ могут быть изготовлены по ТТЛШ, $И^2Л$, *n*-МОП, ЭСЛ и КМОП-технологиям.

Рассмотренные виды ПЗУ допускают только однократное программирование.

3. Перепрограммируемые ПЗУ, или репрограммируемые ПЗУ (РПЗУ) позволяют многократно изменять хранимую информацию.

19.2. Однократно программируемые ПЗУ

Как правило, ПЗУ организовано по словарному принципу, и перемычки представляют собой ряд, образующий ячейку памяти. Следовательно, все запоминающие элементы ряда, образующие ячейку памяти, должны иметь ключи доступа, в качестве которых выступают диоды или транзисторы, выполненные по различным технологиям. Например, в эмиттерной цепи биполярного транзистора находится перемычка, обеспечивающая протекание тока в транзисторе (рис. 19.1, *а*). В схеме диодного ПЗУ (рис. 19.1, *б*) ток протекает только при наличии перемычки в цепи адресная шина (АШ) — диод — перемычка — разрядная шина (РШ).

В приведенных примерах ПЗУ наличие перемычек или их отсутствие можно запрограммировать любым способом.

Рассмотрим процессы, происходящие при считывании.

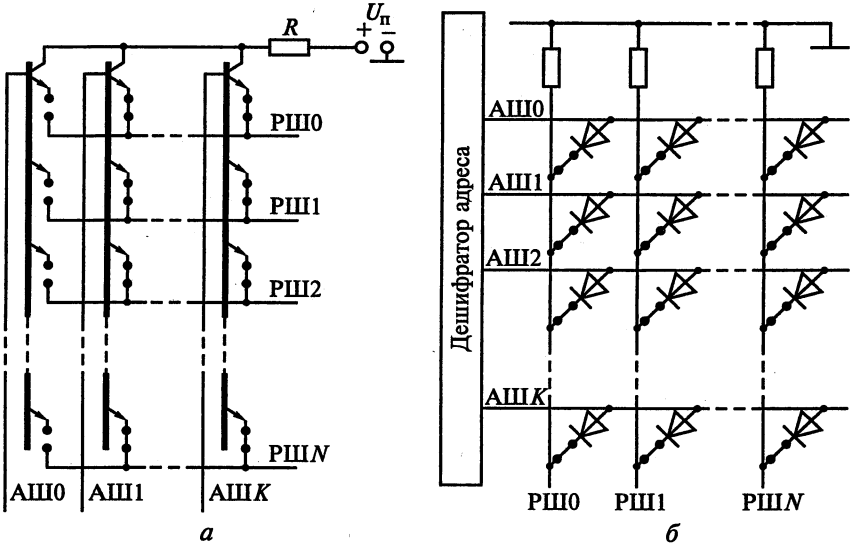


Рис. 19.1

В схеме на рис. 19.1, *а* показано ПЗУ на основе многоэмиттерного транзистора (МЭТ). При подаче необходимого адресного сигнала обращения к выбранной ячейке на базу выбранного транзисторного ключа поступает отпирающее напряжение, в результате чего в зависимости от наличия перемычек в эмиттерных цепях транзистора и соответственно в выходных разрядных шинах будет присутствовать или отсутствовать ток, что воспринимается как считанные лог. 1 или лог. 0. Структура ПЗУ на основе МЭТ в интегральном исполнении будет рассмотрена далее.

Схемы ПЗУ на основе биполярных диодных матриц работают аналогично схемам с использованием транзисторных ключей. При возбуждении адресной шины токи появляются только в тех разрядных шинах, в цепи диодов которых есть перемычки.

Схема ПЗУ на МОП-транзисторах работает аналогично и представлена на рис. 19.2. При адресном обращении лог. 0 будет считываться только с тех разрядных шин, которые через МОП-транзистор и соответствующую ему перемычку соединены с общей шиной. В остальных случаях с разрядных шин считывается лог. 1.

На рис. 19.3 приведен пример программирования ПЗУМ на основе МОП-транзистора с тонким и толстым слоями диэлектрика.

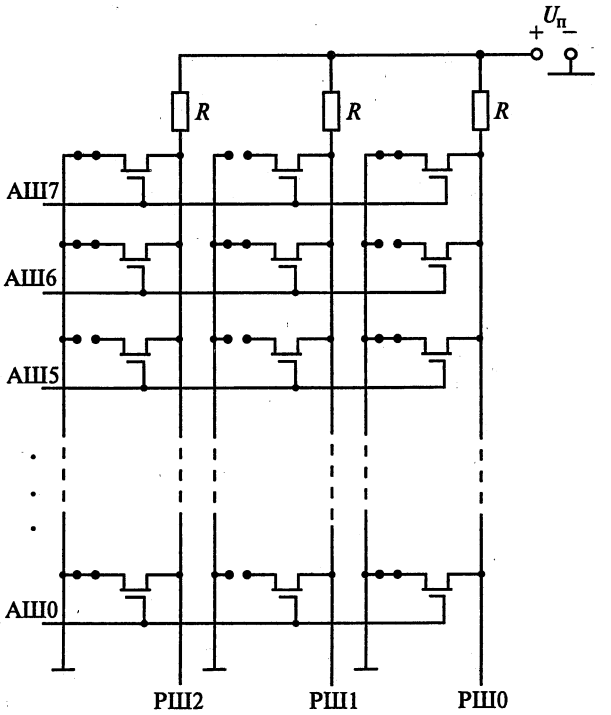


Рис. 19.2

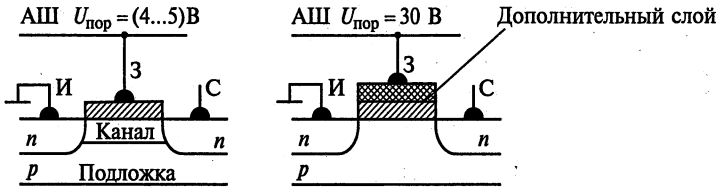


Рис. 19.3

ЗЭ накопителя выполняется на МОП-транзисторе с тонким или толстым слоем диэлектрика под затвором в зависимости от того, какая информация должна храниться в нем.

Запись информации в ПЗУМ на МОП-транзисторах производится с помощью сменного, заказного фотошаблона. В соответствии с заказом осуществляется наращивание слоя диэлектрика.

ПЗУ организовано аналогично схеме, показанной на рис. 19.2. При обращении по адресу на адресную шину соответствующего слова подается напряжение 4...5 В, при котором в МОП-транзисторе с тонким слоем диэлектрика образуется канал, соединяющий области истока и стока, а в транзисторе с толстым слоем диэлектрика такой канал не образуется. Истоки всех транзисторов подсоединены к корпусу, поэтому с соответствующих разрядных шин будут считываться лог. 0. В МОП-транзисторах с толстым слоем диэлектрика под затвором канал образуется при пороговом напряжении порядка 30 В, т. е. напряжения на затворе порядка 5 В недостаточно для образования канала, что соответствует отсутствию транзистора. Поэтому при обращении к данному слову в соответствующем разряде будет считан высокий уровень напряжения источника питания — лог. 1.

Контрольные вопросы

1. Чем отличается программирование ПЗУ на заводе-изготовителе и пользователем?
2. Какой способ организации (словарный или матричный) используется при построении ПЗУ?
3. Какой вид ЗЭ используется для хранения информации в ПЗУ?
4. Для хранения каких видов информации используются ПЗУ?
5. Через какой элемент схемы ПЗУ происходит адресная выборка ячейки памяти?
6. Какую роль играет дополнительное напыление диэлектрика в МОП-транзисторах ПЗУМ?

19.3. Однократно программируемые ПЗУ в интегральном исполнении серии КР556

Одним из наиболее распространенных способов введения информации в ППЗУ является принцип плавкого предохранителя,

когда для образования необходимой комбинации лог. 0 и лог. 1 производится выжигание дозированным током, например, нихромовых соединений между матричными элементами микросхемы. ППЗУ на биполярных транзисторах программируется следующим образом: все эмиттеры транзисторов соединены с разрядными шинами плавкими перемычками; в местах, где должен быть записан лог. 0, через эмиттеры пропускают импульс тока, достаточный для разрушения плавкой перемычки. В процессе выжигания перемычек на адресные входы последовательно подаются адреса, а на разрядные выходы подаются импульсы пережигания в соответствии с заданной таблицей истинности.

Рассмотрим БИС ПЗУ КР556РТ4 информационной емкостью 1024 бита (256 × 4), которая представлена на рис. 19.4, *а*. Условное графическое обозначение этой схемы показано на рис. 19.4, *б*. На рис. 19.5 дана ее упрощенная организация. Такие ПЗУ могут применяться в микропроцессорных системах с обработкой данных в размере байта. Транзисторы матрицы включены по схеме эмиттерного повторителя; 32 шины металлизации проходят поперек 32 МЭТ над эмиттерами. Вскрытие окна к эмиттеру МЭТ (подключение к шине металлизации) соответствует записи лог.1, отсутствие окна (неподключение к шине металлизации) — записи лог. 0.

Матрица содержит 32 МЭТ с 32 эмиттерами каждый. Эмиттеры и включенные в их цепь перемычки разделены на четыре группы, каждая из которых содержит восемь одноразрядных слов. Таким образом, в первой группе находятся восемь слов первого разряда, во второй — второго разряда, в третьей — третьего разряда, в четвертой — четвертого разряда. Для обращения к такому объему памяти и выбору одного из 32 МЭТ необходим дешифратор строк на пять входов и 32 выхода. При адресной выборке будет производиться обращение к одному из 32 МЭТ, т.е. к восьми четырехразрядным словам. Чтобы на выход поступило только одно четырехразрядное слово, необходимо осуществить выбор по одному ЗЭ из каждой разрядной группы.

Для этой цели используются селекторы *CL0... CL3*, каждый из которых представляет собой восемь ключей. Замыкание одного из ключей в каждой группе происходит под воздействием управляющего сигнала с выхода дополнительного дешифратора, на вход которого поступают три младших разряда адреса, после чего считанная информация проходит на усилители считывания. Усилители считывания предназначены для преобразования уровней сигналов, считываемых с матрицы ЗЭ, в уровни, требуемые для работы схемы на внешнюю нагрузку. Схема разрешения выборки запрещает обращение к разрядной группе, если хотя бы на один из ее входов $\overline{CS1}$ или $\overline{CS2}$ подано напряжение высокого уровня, соответствующее лог. 1. С помощью этой схемы происходит стробирование усилителей считывания. Если выборка запрещена, то на выходах усилителей считывания образуется напряжение высокого уровня.

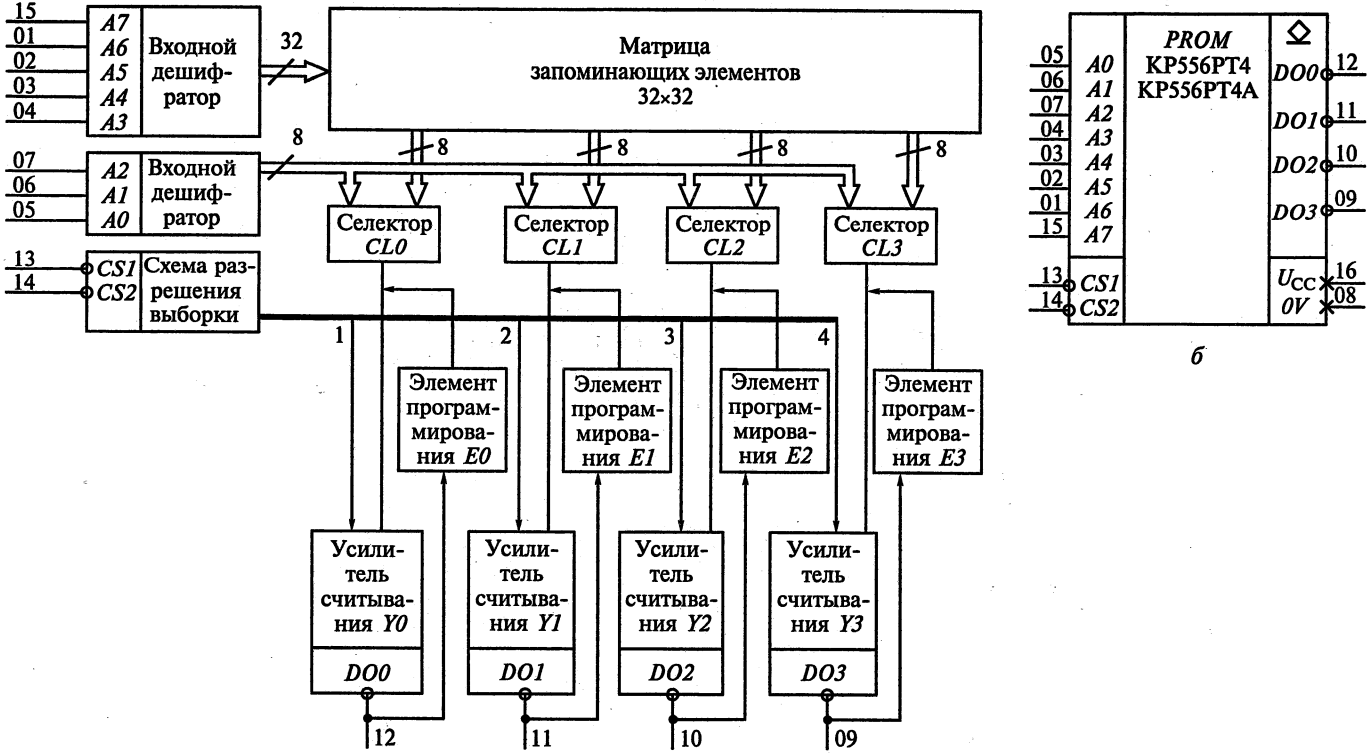

15
01
02
03
04
A7
A6
A5
A4
A3
Входной дешифратор
32
Матрица запоминающих элементов 32×32
8
07
06
05
A2
A1
A0
Входной дешифратор
8
8
8
8
13
14
CS1
CS2
Схема разрешения выборки
Селектор CL0
Селектор CL1
Селектор CL2
Селектор CL3
1
2
3
4
Элемент программирования E0
Элемент программирования E1
Элемент программирования E2
Элемент программирования E3
Усилитель считывания Y0
Усилитель считывания Y1
Усилитель считывания Y2
Усилитель считывания Y3
DO0
DO1
DO2
DO3
12
11
10
09
а
05
06
07
04
03
02
01
15
A0
A1
A2
A3
A4
A5
A6
A7
13
14
CS1
CS2
PROM
КР556РТ4
КР556РТ4А
DO0
DO1
DO2
DO3
12
11
10
09
UCC
0V
16
08
б

Рис. 19.4

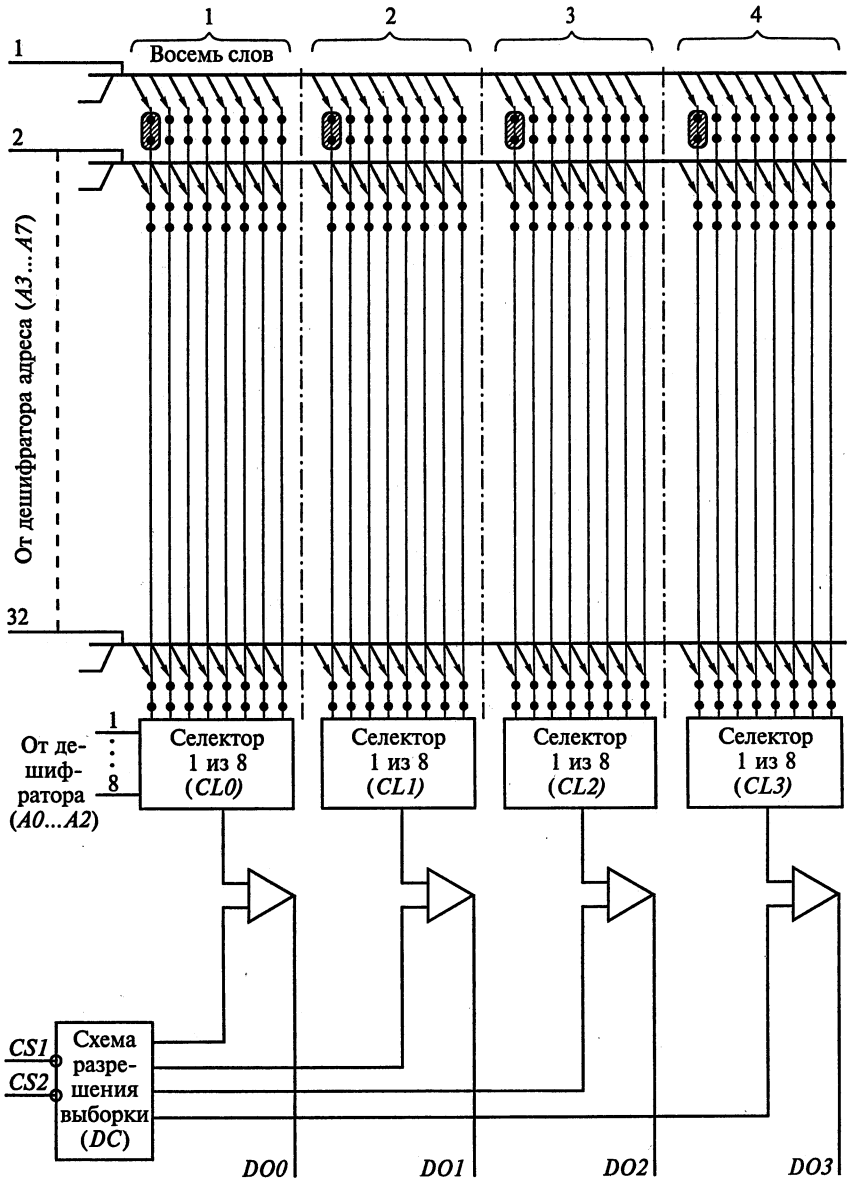


Рис. 19.5

При программировании микросхемы код адреса подается на адресные входы *A0...A7*, а данные — на выходы *DO0... DO3*. Запись лог. 1 в соответствующие разряды осуществляется через элементы программирования (*E0...E3*) путем пережигания перемычек.

Контрольные вопросы

1. Из чего состоит матрица ЗЭ ПЗУ КР556РТ4?
2. Каков объем матрицы памяти и из чего он складывается? Какой должна быть разрядность адреса?
3. Какие разряды адреса используются для непосредственного обращения к матрице памяти?
4. Для чего предназначен дополнительный дешифратор, на вход которого подаются младшие разряды адреса?
5. Сколько слов содержится в выбранной строке и чему равна разрядность каждого слова?
6. Каким образом происходит выбор одного из слов?
7. Для чего предназначены сигналы $\overline{CS1}$ и $\overline{CS2}$?

19.4. Репрограммируемые ПЗУ

Существенный недостаток БИС полупроводниковых ПЗУ заключается в том, что программирование в них возможно только один раз. При необходимости коррекции информации, хранимой в ПЗУ, например при отладке программ или уточнении задачи, решаемой ЭВМ, возникает необходимость в частичной замене БИС ПЗУ. Эти проблемы можно решить, применив репрограммируемые ПЗУ (РПЗУ), способные не только практически бесконечно долго хранить информацию при отключении питания (энергонезависимые РПЗУ), но и при необходимости допускающие запись новой информации электрическим способом. Считывание хранимой в РПЗУ информации осуществляется обычно за сотни наносекунд, а стирание старой информации и последующая запись новой требуют гораздо больше времени.

По способу стирания информации различают РПЗУ со стиранием ультрафиолетовыми лучами (РПЗУ-УФ) и РПЗУ с электрическим стиранием (РПЗУ-ЭС). В РПЗУ-УФ в качестве запоминающего элемента используется МОП-транзистор с одним так называемым плавающим затвором или чаще МОП-транзистор с двойным затвором — плавающим и управляющим. В РПЗУ-ЭС используется МОП-транзистор с двойным затвором или транзистор со структурой металл-нитрид кремния (Si_3N_4)—диоксид кремния (SiO_2)—полупроводник (МНОП-транзистор).

Структура МОП-транзисторов с плавающими затворами (рис. 19.6 и 19.7) характеризуется наличием проводящего слоя (обычно из поликристаллического кремния), расположенного над областью канала и изолированного от остальных областей транзистора диоксидом кремния SiO_2.

Принцип работы этих приборов основан на лавинном пробое p—n-перехода (сток—подложка) при подаче на сток достаточно большого обратного напряжения (15...30 В). При этом у прибора с двойным затвором на управляющий затвор подается достаточно высокое напряжение той же полярности (см. рис. 19.7).

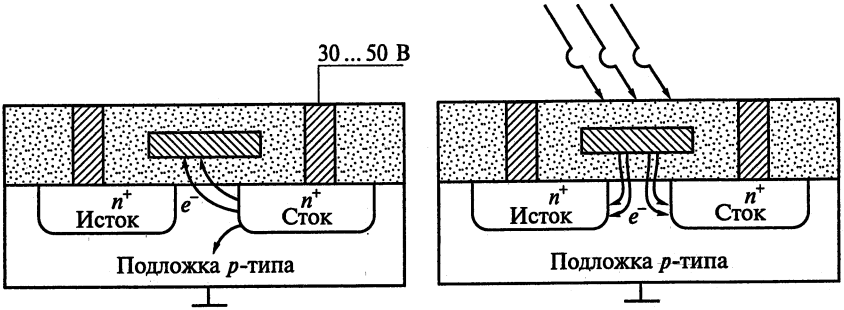


Рис. 19.6

Электроны, попадая на плавающий затвор, оказываются как бы в ловушке и после снятия напряжения со стока и управляющего затвора (у МОП-транзисторов с двойными затворами) могут сохраняться в нем в течение длительного времени (в высококачественных приборах многие годы).

На рис. 19.8 показаны кривые, характеризующие состояния n-МОП-транзистора: исходное (*1*) и после записи информации (*2*). Наличие электронов на плавающем затворе n-МОП-транзистора значительно увеличивает его пороговое напряжение по сравнению с n-МОП-транзистором, у которого нет или очень мало таких электронов.

Для того чтобы исключить накопление заряда на плавающем затворе и обеспечить малое значение порогового напряжения n-МОП-транзистора, при его программировании на сток прибора с одним плавающим затвором не подается напряжение обратного смещения перехода сток—подложка (программирующее напряжение). Для этой цели у прибора с двойным затвором высокое напряжение подается только на управляющий затвор. Стирание информации (заряды на плавающем затворе) у МОП-транзистора с одним плавающим затвором может производиться только ультра-

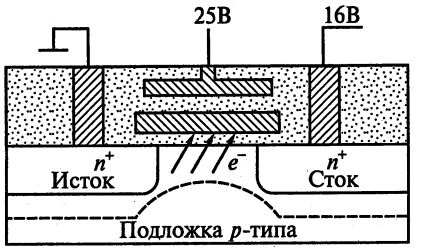


Рис. 19.7

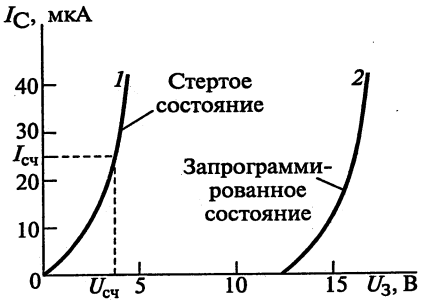


Рис. 19.8

фиолетовым облучением, а у МОП-транзистора с двойным затвором — и ультрафиолетовым облучением, и электрическими сигналами.



Рис. 19.9

Для стирания информации ультрафиолетовым облучением кристалла корпус микросхемы имеет специальное прозрачное окошко. Лучи вызывают в области плавающего затвора транзистора фототоки и тепловые токи, что делает эти области проводящими и приводит к стеканию заряда с плавающего затвора в подложку. Стирание информации происходит сразу на всем кристалле, но для выполнения этой операции требуется достаточно большое время (десятки минут). В РПЗУ-УФ число операций стирания при перепрограммировании ограничено (10...100 для отечественных микросхем), так как под действием ультрафиолетовых лучей свойства материалов в кристалле постепенно изменяются. РПЗУ-УФ с запоминающим элементом в виде прибора с двойным затвором используются гораздо чаще, так как в запоминающем элементе на МОП-транзисторе с одним плавающим затвором необходимы дополнительный МОП-транзистор для считывания информации (рис. 19.9) и большее значение программирующего напряжения.

Использование в РПЗУ приборов с двойными затворами позволяет осуществлять выборочное электрическое стирание информации не на всем кристалле, а только в подлежащих перепрограммированию запоминающих элементах. Электрическое стирание информации осуществляется подачей высокого напряжения программирования только на управляющий затвор. При электрическом способе процесс стирания происходит гораздо быстрее, а число циклов перепрограммирования увеличивается до $10^4...10^6$. Кроме того, для перепрограммирования не требуется извлекать микросхему из устройства, в котором она работает. Поэтому БИС РПЗУ-ЭС постепенно вытесняют БИС РПЗУ-УФ, хотя количество запоминающих элементов в последних пока еще несколько больше из-за более простой структуры управления.

МНОП-транзистор (рис. 19.10) отличается от обычного МОП-транзистора наличием под затвором двух слоев диэлектрика: слоя диоксида кремния (SiO_2) толщиной примерно 0,005 мкм, нанесенного на поверхность кристалла, и поверх него слоя нитрида кремния (Si_3N_4) толщиной 0,1 мкм. На границе диэлектрических слоев возникают центры захвата заряда, куда благодаря туннельному эффекту через тонкий слой SiO_2 при достаточной для его возникновения напряженности электрического поля между затвором и подложкой могут проходить носители заряда и скапливаться на границе этих слоев. Меняя направленность электрического поля, можно создать заряд любого знака на границе диэлектрических слоев.

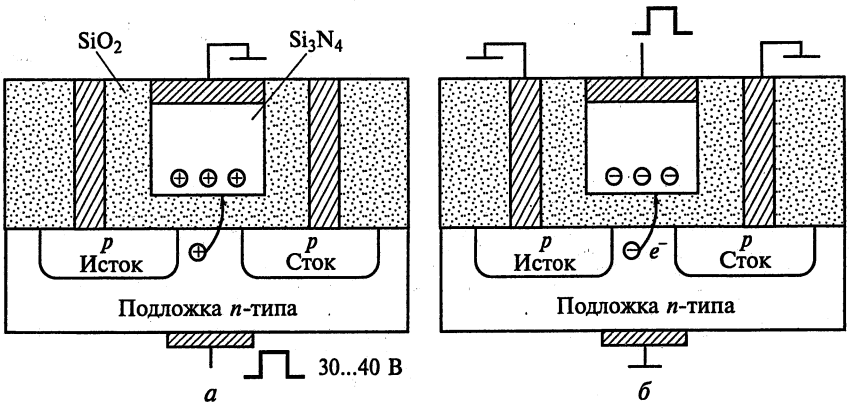


Рис. 19.10

В МНОП-транзисторе с *p*-каналом положительный заряд увеличивает его пороговое напряжение настолько, что рабочее напряжение на затворе прибора не в состоянии его открыть. Транзистор, в котором заряд отсутствует или имеет другой знак, легко открывается рабочим значением напряжения. Так осуществляется хранение 1 бита информации в МНОП-транзисторе: одно из состояний трактуется как отображение лог. 1, другое — как лог. 0.

В режиме стирания (см. рис. 19.10, *а*) — записи большого значения порогового напряжения — затвор заземляется, а на подложку подается положительный импульс амплитудой 20...30 В. Стирание может осуществляться также подачей отрицательного импульса напряжения амплитудой 20...30 В на управляющий затвор. В режиме программирования — записи малого значения порогового напряжения — на затвор *p*-МНОП-транзистора подается положительное напряжение (см. рис. 19.10, *б*) при заземленных выводах подложки, стока и истока. В режиме считывания на затвор прибора подается напряжение, находящееся между этими двумя значениями пороговых напряжений.

Контрольные вопросы

1. Каковы достоинства и недостатки РПЗУ по сравнению с ПЗУ?
2. Какие преимущества дает использование РПЗУ с электрическим стиранием информации по сравнению с РПЗУ с ультрафиолетовым стиранием?
3. Как влияет на пороговое напряжение появление электронов на плавающем затворе МОП-транзистора *n*-типа?
4. Какие преимущества обеспечивает использование РПЗУ на МОП-транзисторах с двойным затвором по сравнению с МОП-транзисторами с одним плавающим затвором?
5. Почему использование МОП-транзисторов в РПЗУ позволяет существенно уменьшить потребляемую ими мощность в режиме хранения?

19.5. Различия между ПЗУ и ПЛМ

В организации ПЗУ и ПЛМ много общего. Выявим общий подход в построении этих схем на примере.

Предположим, что необходимо построить устройство, которое обеспечивает выдачу сигнала на выходе *Y1* при поступлении на вход кодов 000, 001; на выходе *Y2* при кодах 010, 100, 110; на выходе *Y3* при кодах 011, 101, 110, 111. Подаваемые на вход устройства коды можно рассматривать как коды адреса одноразрядных ячеек ПЗУ, из которых считываемые единицы через элемент ИЛИ поступают на один из выходов Y_i. Рассмотрим взаимосвязь между адресами и данными — функциями (табл. 19.1).

На рис. 19.11, *а* представлена схема ПЗУ, состоящая из дешифратора адреса на логических элементах и запоминающих элементов в виде диодно-резистивных схем, в цепи которых включены перемычки. Переменные *X3*, *X2*, *X1* рассматриваются как коды адресов различных ячеек памяти. Из табл. 19.1 видно, что в дешифраторе при определенных адресах возбуждаются соответствующие выходные шины, которые должны быть объединены на одном из выходов схемы: *Y1*, *Y2*, *Y3*. Элементы ИЛИ, с помощью которых формируются сигналы Y_i, представляют собой неполный шифратор.

На рис. 19.11, *б* представлена тажа схема ПЗУ в виде двух матриц. Матрица *A1* представляет собой полный линейный дешифратор на восемь выходов. Каждая вертикальная линия в *A1* соответствует элементу И с тремя входами, на каждом из которых реализовано одно из сочетаний входных переменных *X3*, *X2*, *X1*. Матрица *A2* представляет собой неполный шифратор. Каждая горизонтальная линия в *A2* соответствует восьмивходовому элементу ИЛИ. О формировании необходимых сигналов на каждом из его входов говорит точка в месте пересечения вертикальной линии матрицы *A1* и горизонтальной линии матрицы *A2*.

Таблица 19.1

Адрес	Входы			Выходы		
	X3	*X2*	*X1*	*Y1*	*Y2*	*Y3*
A0	0	0	0	1	0	0
A1	0	0	1	1	0	0
A2	0	1	0	0	1	0
A3	0	1	1	0	0	1
A4	1	0	0	0	1	0
A5	1	0	1	0	0	1
A6	1	1	0	0	1	1
A7	1	1	1	0	0	1

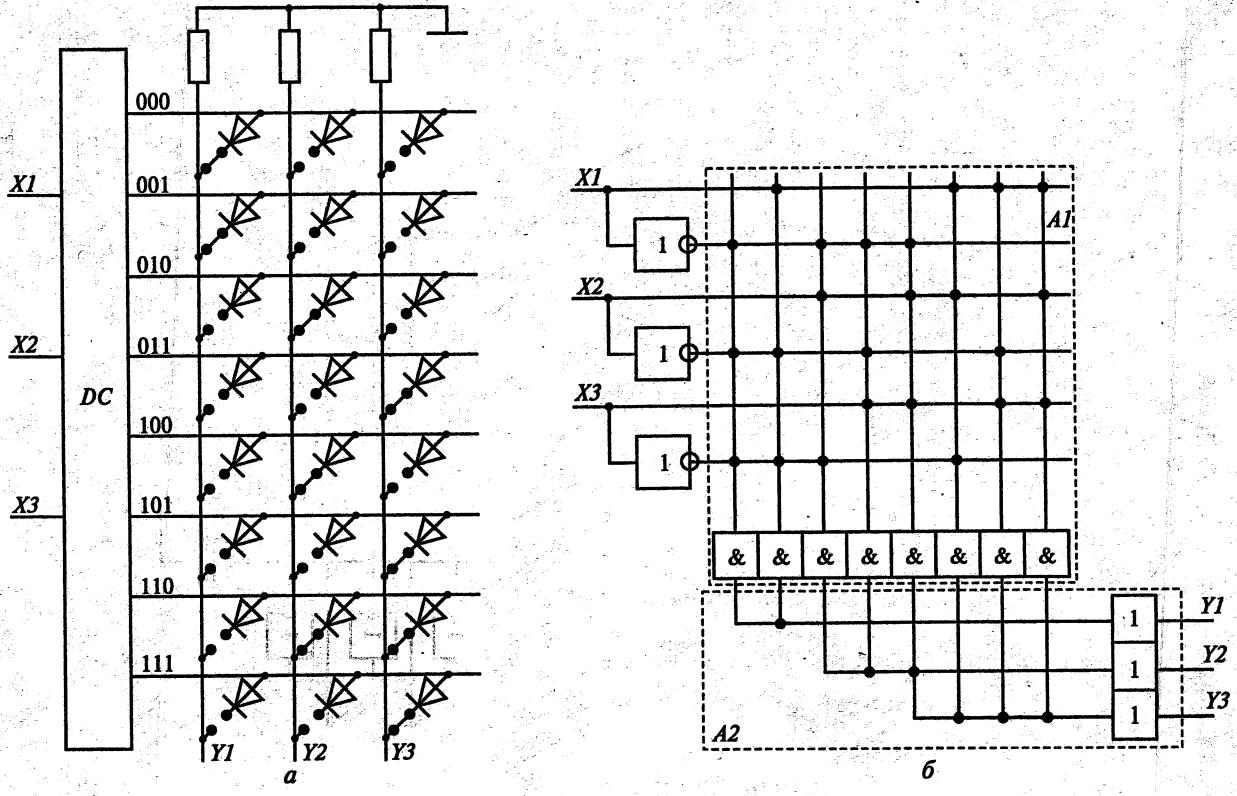

X1
X2
X3
DC
000
001
010
011
100
101
110
111
Y1
Y2
Y3
а
X1
X2
X3
1
1
1
A1
&
&
&
&
&
&
&
&
A2
1
1
1
Y1
Y2
Y3
б

Рис. 19.11

С другой стороны, данные табл. 19.1 представляют собой форму задания функций, т. е. их можно преобразовать в формулы и затем упростить:

$$Y1 = \bar{X}1\ \bar{X}2\ \bar{X}3 \vee X1\ \bar{X}2\ \bar{X}3 = \bar{X}2\ \bar{X}3\left(\overset{1}{\cancel{\bar{X}1 \vee X1}}\right) = \bar{X}2\ \bar{X}3;$$

$$Y2 = \bar{X}1\ X2\ \bar{X}3 \vee \bar{X}1\ \bar{X}2\ X3 \vee \bar{X}1\ X2\ X3 = \bar{X}1\ X2\left(\overset{1}{\cancel{\bar{X}3 \vee X3}}\right) \vee \bar{X}1\ \bar{X}2\ X3 =$$

$$= X1\left(X2 \vee \bar{X}2\ X3\right) = X1\left(X2 \vee X3\right)\left(\overset{1}{\cancel{\bar{X}2 \vee X2}}\right) = X1\ X2 \vee X1\ X3;$$

$$Y3 = X1\ X2\ \bar{X}3 \vee X1\ \bar{X}2\ X3 \vee \bar{X}1\ X2\ X3 \vee X1\ X2\ X3 = X1\ X2\left(\overset{1}{\cancel{\bar{X}3 \vee X3}}\right) \vee$$

$$\vee X1\ \bar{X}2\ X3 \vee \bar{X}1\ X2\ X3 = X1\ X2\left(\overset{1}{\cancel{\bar{X}3 \vee X3}}\right) \vee X1\ \bar{X}2\ X3 \vee \bar{X}1\ X2\ X3 =$$

$$= X1\left(X2 \vee \bar{X}2\ X3\right) \vee \bar{X}1\ X2\ X3 = X1\left(X2 \vee X3\right)\left(\overset{1}{\cancel{X2 \vee \bar{X}2}}\right) \vee \bar{X}1\ X2\ X3 =$$

$$= X1\ X2 \vee X1\ X3 \vee \bar{X}1\ X2\ X3 = X2\left(X1 \vee \bar{X}1\ X3\right) \vee X1\ X3 =$$

$$= X2\left(X1 \vee X3\right)\left(\overset{1}{\cancel{X1 \vee \bar{X}1}}\right) \vee X1\ X3 = X2\left(X1 \vee X3\right) \vee X1\ X3 =$$

$$= X1\ X2 \vee X2\ X3 \vee X1\ X3.$$

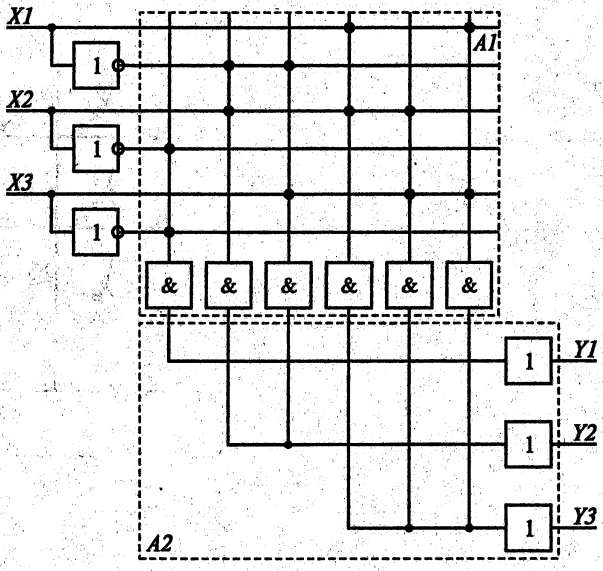


Рис. 19.12

Полученные формулы могут быть реализованы в виде комбинационной схемы на ПЛМ (рис. 19.12).

Сравнивая две схемы, выполняющие одни и те же функции (см. рис. 19.11, *б* и 19.12), видим, что схема, реализованная в виде ПЛМ, проще. Матрица *А1* в ПЗУ — это полный, жестко программируемый дешифратор, в матрице ПЛМ — это программируемые под функции минтермы. Затраты на оборудование принято определять площадью полупроводникового кристалла, занимаемого схемой. Таким образом, схемы, выполненные на ПЛМ, обеспечивают большую степень интеграции и тем самым расширяют функциональные возможности микросхемы.

Контрольное задание

Реализовать в виде ПЗУ и ПЛМ следующие функции:

$$Y1 = X1\,\bar{X}2\,X3\,X4 \vee X1\,X2\,X3\,X4 \vee X1\,\bar{X}2\,X3\,\bar{X}4 \vee \bar{X}1\,X2\,\bar{X}3\,X4 \vee \bar{X}1\,\bar{X}2\,\bar{X}3\,X4 \vee \vee \bar{X}1\,X2\,\bar{X}3\,\bar{X}4 \vee \bar{X}1\,\bar{X}2\,\bar{X}3\,\bar{X}4 \vee X1\,X2\,X3\,\bar{X}4;$$

$$Y2 = \bar{X}1\,\bar{X}2\,X3\,\bar{X}4 \vee \bar{X}1\,\bar{X}2\,X3\,X4 \vee \bar{X}1\,\bar{X}2\,\bar{X}3\,\bar{X}4 \vee \bar{X}1\,X2\,X3\,\bar{X}4 \vee \bar{X}1\,X2\,\bar{X}3\,\bar{X}4 \vee \vee \bar{X}1\,X2\,\bar{X}3\,X4 \vee \bar{X}1\,\bar{X}2\,\bar{X}3\,X4.$$

Раздел VII
ОСНОВЫ МИКРОПРОЦЕССОРНОЙ ТЕХНИКИ

Глава 20
ОРГАНИЗАЦИЯ РАБОТЫ УСТРОЙСТВ НА БАЗЕ МИКРОПРОЦЕССОРОВ

20.1. Общие сведения

Современные достижения в области микроэлектроники позволили создать на базе больших и сверхбольших интегральных схем микроЭВМ, обладающие достаточно большим объемом памяти, малыми стоимостью, энергопотреблением и размерами, но при этом имеющие большие вычислительные мощности. Эти факторы позволили использовать микроЭВМ в различных сферах экономики, непосредственно на рабочих местах или в исполнительных механизмах, например в станках, приборах, устройствах бытовой, измерительной техники и т.д. При этом функции микроЭВМ постоянно расширяются, так как появляется возможность управления происходящими процессами в исполнительных устройствах. В процессе выполнения работы происходит отслеживание результатов, их обработка, выработка корректирующих сигналов и дальнейшее продолжение работы до тех пор, пока не будет получен требуемый результат. Специализация микроЭВМ с учетом конкретных функций, выполняемых аппаратурой, достигается с помощью записи в ее память соответствующих программ. При этом различная по назначению микропроцессорная аппаратура может иметь схожую структуру построения. Использование встроенных микроконтроллеров (управляющих устройств) освобождает разработчиков от создания аппаратурной части устройств, что заменяется разработкой и введением специальных программ, с помощью которых обеспечиваются необходимые режимы работы исполнительных механизмов. В результате удается на 60...70 % сократить время на разработку новой техники, повысить эксплуатационные характеристики, уменьшить массу, габаритные размеры, повысить надежность, расширить функциональные возможности изделий.

20.2. Организация микроЭВМ

МикроЭВМ представляет собой законченную вычислительную систему. Рассмотрим организацию микроЭВМ на примере универ-

сальной цифровой вычислительной машины — персонального компьютера. В состав компьютера (рис. 20.1) входят:

центральный процессор (ЦП), или просто *микропроцессор*, состоящий из устройства управления, которое следит за порядком выполнения операций, и арифметико-логического устройства, используемого для выполнения арифметических и логических операций;

оперативное запоминающее устройство (или оперативная память — ОП), предназначенное для временного хранения программы и данных пользователя на момент исполнения, благодаря которому ЦП может последовательно выбирать команды программы и подвергать их обработке;

постоянное запоминающее устройство, в котором хранятся наиболее важные и часто используемые программы, необходимые для работы компьютера, например: программы, обеспечивающие начальную активизацию работы вычислительной системы и поддерживающие взаимодействие ЦП с устройствами ввода-вывода, прикладные программы, транслятор языка программирования и т.д. Обычно ПЗУ содержит в себе базовую систему ввода-вывода BIOS (*Base Input Output System*), которая отвечает прежде всего за процедуру самопроверки, выполняемую после включения питания, и выполнение программы самозагрузки. Кроме того, там же расположена таблица знакогенератора, которая используется для формирования изображения символов на экране монитора.

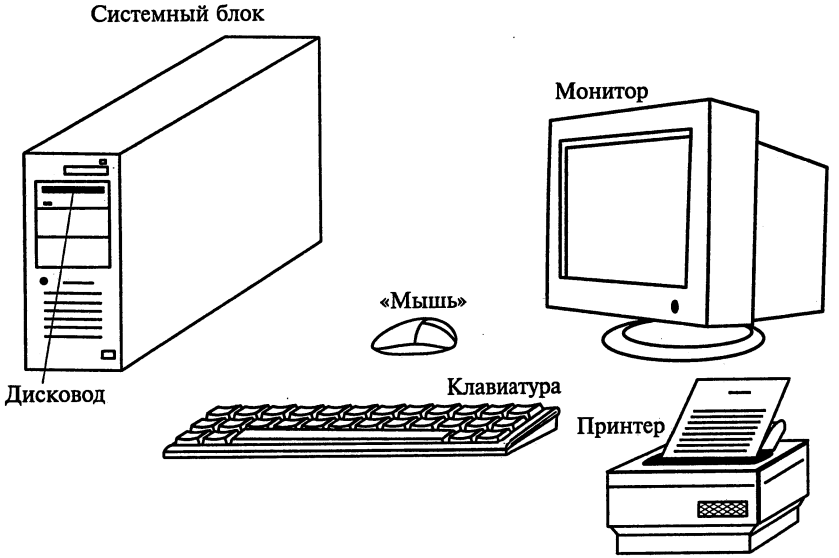


Рис. 20.1

кэш-память, представляющая собой небольшой блок (объемом порядка 256 бит) быстродействующей памяти, которая находится между процессором и более мощной ОП. Контроллер кэш-памяти старается предугадать запросы процессора и заполняет ее той информацией из ОП, которая с наибольшей вероятностью потребуется процессору в ближайшее время;

устройства внешней памяти на магнитных носителях, которые обеспечивают долговременное хранение программных и информационных файлов. Эти устройства совмещают в себе функции ввода и вывода информации. С их помощью компьютер может скопировать информацию с магнитного носителя в ОЗУ для обработки, а полученные результаты записать обратно на магнитный носитель. Для этой цели используются дисководы (накопители) на гибких магнитных дисках (дискетах). С помощью гибких дисков возможно переносить информацию с одного компьютера на другой и делать архивные копии информации, содержащейся на жестком диске. Накопитель на жестком магнитном диске (винчестер) предназначен для постоянного хранения информации, используемой при работе с компьютером: программ операционной системы, часто используемых пакетов программ, редакторов документов. Винчестер имеет большую емкость хранимой информации (около 1 Гбайта) и может надежно сохранять информацию при отключенном питании.

устройства ввода, в качестве которых используют клавиатуру (с ее помощью вычислительной системе задаются команды и данные) и «мышь» — устройство, которое облегчает ввод информации в компьютер;

устройства вывода, используемые для отражения результатов обработки информации. К ним можно отнести экран дисплея (монитор), который служит для отображения текстовой и графической информации. (С помощью клавиатуры и монитора значительно облегчается диалог между пользователем и компьютером.) Кроме того, находит широкое использование принтер — устройство для вывода на печать текстовой и графической информации;

порты — параллельные и последовательные, используемые для подключения к компьютеру внешних периферийных устройств (принтеров, модема, «мыши» и т.д.).

Процессор и все виды полупроводниковой памяти расположены на так называемой *системной*, или *материнской плате*. Внешняя память (накопители) и соответствующий контроллер, видеоконтроллер, порты последовательного ввода («мышь», модем), порты параллельного ввода (принтер) находятся вне материнской платы, но тем не менее образуют с материнской платой *системный блок*, который имеет конструктивно законченное оформление.

Центральное место в большинстве вычислительных систем занимает совокупность программ, называемых *операционной системой*, которые координируют работу компьютера и управляют раз-

мещением программ и данных в оперативной памяти. Она дает указания компьютеру, как интерпретировать команды и данные, как распределять аппаратные ресурсы для выполнения задания и как управлять устройствами ввода-вывода. Операционная система также обеспечивает возможность непосредственного взаимодействия человека и компьютера.

Кроме того, существует огромное количество различных *прикладных программ*, с помощью которых компьютер приобретает большую гибкость:

редакторы для подготовки текстов (документов) на компьютере;

программы, используемые в полиграфии;

программы, используемые при обработке массивов информации — система управления базами данных;

программы экономического назначения — бухгалтерские программы, программы финансового анализа;

программы юридического назначения;

программы для создания графических работ, анимационных и видеофильмов;

системы автоматизированного проектирования (САПР) разработок;

программы для обработки статистических данных;

обучающие программы, справочники и т.д.

Объем используемых программ определяют их версии, например: Windows 95, Windows 98, Windows XP 2002.

Программное обеспечение включает в себя также такие программы как трансляторы и утилиты. *Трансляторы* помогают программистам создавать прикладное программное обеспечение, с их помощью так называемые языки высокого уровня, отдаленно напоминающие язык человека, преобразуются в машинный язык, представляющий собой комбинацию нулей и единиц, которые компьютер обрабатывает как последовательность электрических импульсов. *Утилиты* выполняют рутинные, но часто необходимые функции, например сортировку и слияние файлов, чем помогают прикладным программам решать довольно сложные задачи.

Контрольные вопросы

1. Каковы предпосылки создания микроЭВМ?
2. В чем заключается процесс взаимодействия микроЭВМ и исполнительного механизма при их совместной работе?
3. Привести примеры использования микропроцессоров и микроЭВМ в бытовой технике.
4. Чем отличается микроконтроллер от микроЭВМ?
5. Привести пример замены механического управления исполнительным механизмом на электронное управление.
6. В чем состоят преимущества радиоприемников с использованием цифровых схем по сравнению с радиоприемниками, которые выпускались в середине прошлого века?

7. Какие виды информации могут храниться в ОЗУ, ПЗУ, ВЗУ?
8. Какие виды информации размещаются на гибких и жестких магнитных дисках?
9. С помощью какого устройства производится ввод программы пользователя?
10. Из чего состоит материнская плата?
11. Какую роль играют контроллеры, работающие на внешние устройства?
12. Чем отличается системный блок от материнской платы?
13. Каковы основные функции операционной системы?
14. Для чего служат прикладные программы?
15. Для чего предназначены трансляторы и утилиты?

20.3. Классификация микропроцессоров

Микропроцессоры (МП) классифицируются по многим признакам.

По назначению микропроцессоры подразделяются на универсальные и специализированные (рис. 20.2).

Универсальные МП предназначены для широкого применения, а специализированные — для решения определенных задач.

Возможности МП определяются в первую очередь *разрядностью обрабатываемой информации* и системой команд. Конструктивно МП подразделяются на однокристальные, многокристальные и многокристальные секционированные.

Однокристальный МП представляет собой большую интегральную схему. Такие МП оперируют с информацией фиксированной разрядности (в пределах от 4 до 16).

Многокристальные МП представляют собой отдельные, самостоятельные функциональные БИС, каждая из которых содержит операционную, управляющую и интерфейсную части. Возможности таких МП шире. Они также имеют фиксированную разрядность, которая может достигать 32 (рис. 20.3, *а*).



Рис. 20.2

Многокристальные секционированные МП (рис. 20.3, *б*) представляют собой БИС, в которые входят отдельные секции АЛУ, УУ и интерфейса. Каждая секция используется для выполнения определенных операций над данными фиксированной разрядности от 4 до 16. Параллельное объединение — наращивание позволяет увеличивать разрядность обрабатываемой информации до $n \times k$, где n — число разрядов, k — число секций.

Однокристальные микроЭВМ размещаются на одном кристалле, имеют все основные элементы ЭВМ и могут быть встроены в различную аппаратуру.

По технологии изготовления различают микропроцессоры на основе:

униполярной технологии с использованием полевых транзисторов (*р*-МОП, *n*-МОП и КМОП);

биполярной технологии с использованием биполярных транзисторов (ТТЛ, ТТЛШ, $И^2Л$ и ЭСЛ).

В зависимости от *быстродействия* различают МП следующих поколений:

I — медленно действующие четырехразрядные МП БИС;

II — четырех- и восьмиразрядные со средним быстродействием МП БИС;

III — быстродействующие секционированные МП БИС;

IV — сверхбыстродействующие секционированные МП БИС и однокристальные микроЭВМ.

Различают МП с доступом к памяти через канал прямого доступа и МП с доступом к памяти через арифметическую операцию в АЛУ, а также МП с фиксированной разрядностью и МП с наращиваемой разрядностью.

По способу реализации команд различают МП с жесткой и гибкой логикой.

В МП с жесткой логикой система команд и набор управляющих сигналов не могут быть изменены (однокристальные и многокристальные МП БИС с фиксированной разрядностью).

В МП с гибкой логикой система команд и набор управляющих сигналов могут быть изменены (секционированные МП БИС).

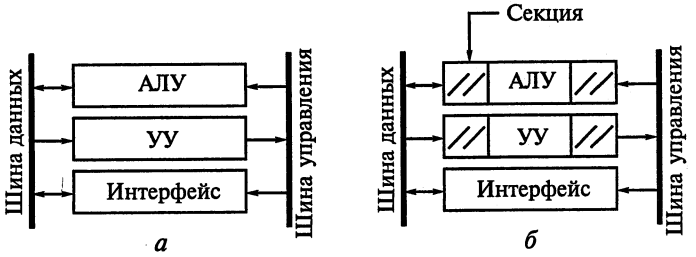


Рис. 20.3

Кроме того, МП классифицируются по числу регистров СОЗУ, числу шин, используемой разрядности, числу команд и их длине, видам адресации, числу прерываний и т.д.

Контрольные вопросы

1. К какому виду по возможности наращивания можно отнести однокристальный МП?
2. Чем отличается организация однокристальных МП от многокристальных?
3. Чем отличаются многокристальные МП от многокристальных секционированных?
4. В чем заключается различие в управлении с жесткой и гибкой логикой?

20.4. Работа устройств на базе микропроцессора с фиксированной разрядностью

Взаимодействие между МП и внешними по отношению к нему устройствами осуществляется по шинам. Количество шин, связывающих МП с внешними устройствами, может быть различным в зависимости от его внутренней организации и структуры вычислительного устройства.

Рассмотрим структуру микроЭВМ, взяв за основу простейшую из микропроцессорных серий, построенную на основе однокристального МП КР580ВМ80А (Intel 8080/8085), схема которой представлена на рис. 20.4. На предлагаемом примере изучим основные принципы взаимодействия устройств на базе МП.

МП имеет три типа шин: адресную шину, шину данных и управляющую шину. Каждая шина представляет собой совокупность проводников, количество которых определяется разрядностью передаваемой информации. В данном случае имеем шестнадцатиразрядную шину адреса и восьмиразрядную шину данных. Взаимодействие между устройствами по шинам осуществляется следующим образом. Чтобы обратиться к памяти и произвести запись или чтение информации, МП должен выставить на адресную шину адрес ячейки памяти, где хранится информация, и указать направление передаваемой информации по шине данных, выставив на шине управления один из двух сигналов «Запись» или «Чтение». Аналогично происходит общение МП с устройствами ввода-вывода. При этом необходимо иметь в виду, что все устройства микроЭВМ по отношению к МП определяются как внешние.

В любой момент времени выход на шину имеют только два устройства, между которыми происходит обмен информацией, причем одно из устройств всегда активно, а другое — пассивно. Активность состоит в проявлении инициативы в обмене информацией с любым пассивным источником. Как правило, активную роль вы-

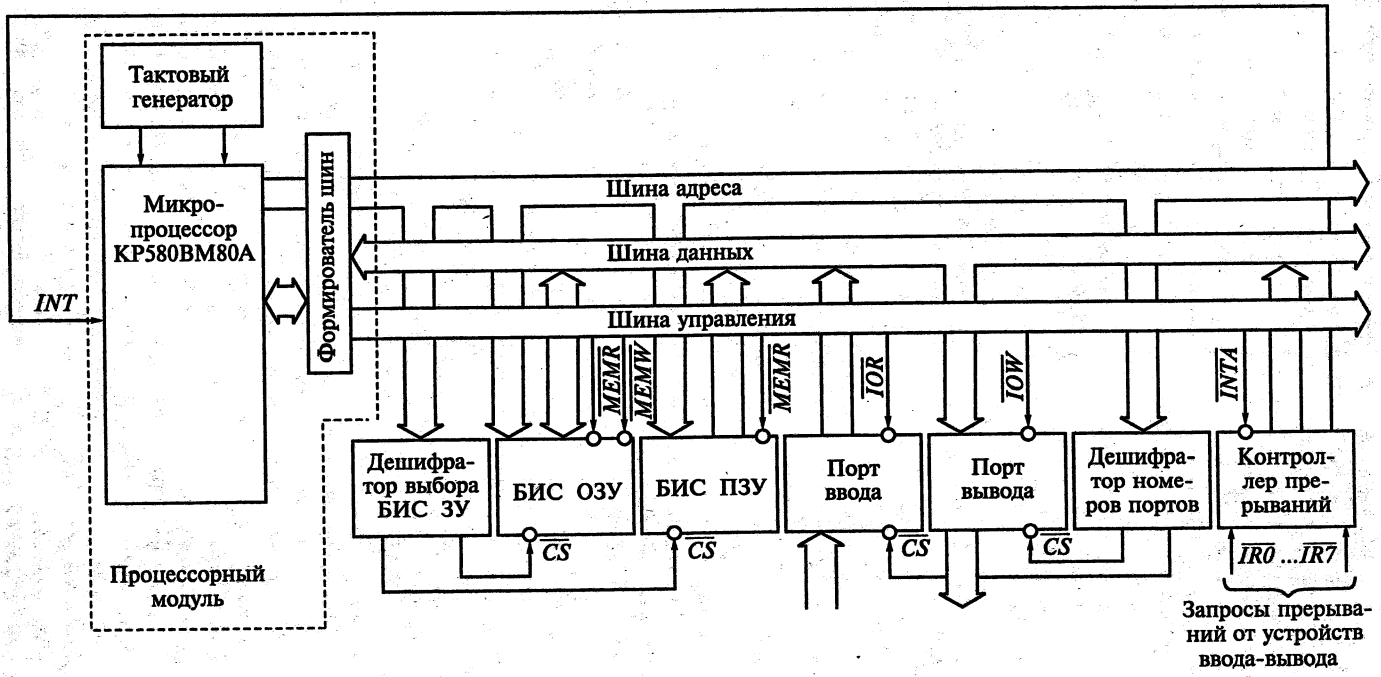

Тактовый генератор
Микро-процессор КР580ВМ80А
INT
Формирователь шин
Процессорный модуль
Шина адреса
Шина данных
Шина управления
MEMR
MEMW
MEMR
IOR
IOW
INTA
Дешифратор выбора БИС ЗУ
БИС ОЗУ
БИС ПЗУ
Порт ввода
Порт вывода
Дешифратор номеров портов
Контроллер прерываний
CS
CS
CS
CS
IR0 ...IR7
Запросы прерываний от устройств ввода-вывода

Рис. 20.4

полняет процессор. Арбитр процессора определяет приоритет каждого из пассивных устройств, стремящихся к обмену информацией. Каждое подключаемое к шине устройство имеет на выходе ключи — схемы с тремя состояниями, с помощью которых в любой момент времени оно может отсоединяться (третье состояние) или подсоединяться к шине, способствуя передаче сигналов лог. 0 (первое состояние) или лог. 1 (второе состояние). При передаче информации в том или другом направлении используется мультиплексирование, т.е. разделение по времени.

Адресная шина, или *шина адреса* (ША) однонаправленная. По ней МП передает адрес пассивного устройства, участвующего в обмене информацией в данный момент времени.

Шина данных (ШД) двунаправленная. По ней обеспечивается передача данных между МП и памятью или внешними устройствами. Ее разрядность определяется разрядностью МП. В любой момент времени информация может передаваться только в одном из направлений, которое определяется одним из двух управляющих сигналов «Запись» или «Чтение». Все устройства микроЭВМ по отношению к процессору рассматриваются как внешние, поэтому и значения этих сигналов рассматриваются относительно внешних устройств:

по сигналу «Запись» (*WR*) МП посылает (записывает) данные во внешнее устройство;

по сигналу «Чтение» (*RD*) МП получает (считывает) данные от внешнего устройства.

Рассмотрим *шину управления* (ШУ). Микропроцессор и некоторые устройства ввода-вывода генерируют управляющие сигналы, предназначенные для синхронизации операций, выполняемых устройствами. Управляющие сигналы задают начало и последовательность срабатывания различных устройств, а также направление передачи информации. МП настраивает подключаемое устройство на необходимый режим обмена информацией. При работе с памятью — это сигналы $\overline{MEMW}$, $\overline{MEMR}$; при работе с устройствами ввода-вывода — это сигналы $\overline{IOW}$, $\overline{IOR}$ и ряд других.

Каждому устройству ввода-вывода соответствует свой регистр (порт), имеющий выход на шину. В зависимости от источника информации к шине подключается только один из регистров. Его выбор производится по адресу, передаваемому из МП по ША, под воздействием разрешающего сигнала $\overline{CS}$, который вырабатывается дешифратором ввода-вывода. Порт выполняет роль буфера, согласующего скорость передачи информации между МП и устройствами ввода-вывода. При вводе в нем накапливается информация от медленно действующих устройств ввода с последующей быстрой передачей данных в процессор. Процессор как быстродействующее устройство подключается к шине на короткое время (порядка 1 мкс). При выводе информации процессор быстро выда-

ет информацию через шину данных в порт вывода, а затем принятая информация со скоростью, соизмеримой со скоростью работы устройства, медленно выводится на устройство вывода. Аналогично режимам ввода-вывода информации осуществляется обмен информацией между МП и памятью.

Контрольные вопросы

1. Почему ША является однонаправленной, а ШД — двунаправленной?
2. Какое устройство управляет направлением передаваемой информации? Под действием каких сигналов это происходит?
3. Какие условия обеспечивают считывание информации из памяти?
4. Для чего предназначен порт ввода-вывода?
5. Какие существуют этапы выполнения программы?
6. На основании какой информации происходит выполнение команды?
7. Что является источником информации при выполнении команды «Чтение» при совместной работе МП с памятью?
8. Какие устройства участвуют в обмене информацией и в каком направлении перемещается информация при наличии сигналов $\overline{IOR}$, $\overline{MEMW}$?

Глава 21
ОРГАНИЗАЦИЯ ФУНКЦИОНИРОВАНИЯ МИКРОПРОЦЕССОРНОЙ СИСТЕМЫ

21.1. Структурная схема микропроцессора

На рис. 21.1 представлена структурная схема микропроцессора КР580ВМ80.

Микропроцессорная БИС состоит из трех функциональных частей:

устройства управления и синхронизации;

восьмиразрядного арифметико-логического устройства;

блока регистров.

Взаимодействие между этими функциональными частями осуществляется по внутренней шине. Информация в МП поступает с внешней восьмиразрядной шины данных. Один выход МП БИС сориентирован на подключение к его контакту одного выхода микросхемы типа ТТЛ ($I^1_{\text{вых}} \approx 0{,}1$ мА; $I^0_{\text{вых}} \approx 1{,}6$ мА). Низкая нагрузочная способность выходов МП объясняется тем, что реальная нагрузка МП значительно превышает указанные значения. Поэтому на выходе МП со стороны ШД устанавливаются дополнительные усилители-буферы с тристабильными выходными каскадами.

Аналогичную роль играет и буфер адреса.

Устройство управления состоит из регистра команд, в который поступает код операции очередной команды (КОп), программи-

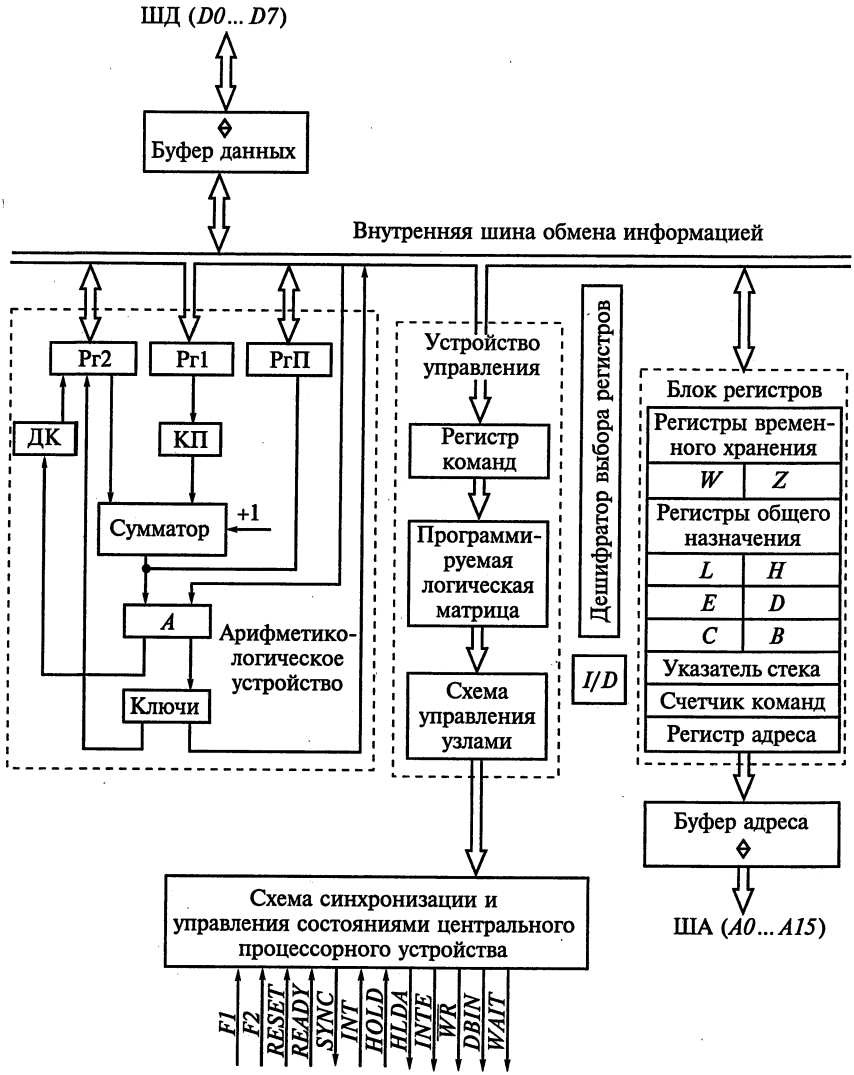


Рис. 21.1

руемой логической матрицы и схемы управления внутренними и внешними узлами. В зависимости от кода операции в схеме управления будет инициировано формирование определенной последовательности управляющих сигналов, синхронизированных тактовыми сигналами *F1* и *F2* для выполнения заданной команды.

Арифметико-логическое устройство предназначено для выполнения арифметических и логических операций.

В состав АЛУ входят:

восьмиразрядные буферные регистры Рг1 и Рг2 для приема и временного хранения операндов при обмене информацией между памятью и МП. Рг2 используется также для выдачи результата операции на внутреннюю шину данных. Данные регистры программно недоступны;

кодопреобразователь (КП) для передачи содержимого регистра Рг1 в сумматор в прямом или обратном коде;

комбинационный сумматор для выполнения арифметических и логических операций над восьмиразрядными операндами;

регистр-аккумулятор (Рг*А*), являющийся главным регистром МП. Любая операция, выполняемая в АЛУ, предполагает размещение одного из операндов в аккумуляторе. Так аккумулятор хранит первый считанный из памяти операнд, который на момент поступления в Рг1 второго операнда переписывается в Рг2, после чего происходит их сложение в сумматоре. Результат операции в АЛУ тоже обычно хранится в аккумуляторе. (Он может быть сохранен в аккумуляторе для дальнейших вычислений или может быть выведен за пределы МП.) В процессоре с аккумулятором все арифметические и логические операции, а также операции по передаче данных выполняются только через размещение данных в аккумуляторе;

десятичный корректор (ДК) для формирования избытка 6 при переводе результата в случае необходимости в двоично-десятичный код;

ключи, которые позволяют передавать содержимое аккумулятора в Рг2 и на внутреннюю шину данных;

регистр признаков (РгП) для выдачи признаков результата операции (флагов).

При выполнении команд вырабатывается пять условных флагов. Каждый флаг, представляющий собой одноразрядный регистр (рис. 21.2), отражает некоторые особенности результата операции и может быть использован в дальнейшей работе или контроле правильности результата.

Рассмотрим назначение каждого разряда.

Разряд *С* (перенос) устанавливается в состояние, соответствующее переносу из старшего разряда байта при выполнении арифметических операций и равное значению выдвигаемого из аккумулятора разряда при выполнении операции сдвига.

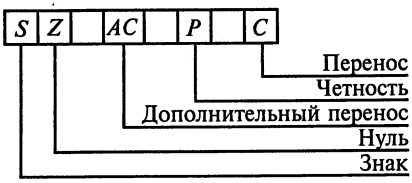


Рис. 21.2

Разряд *Р* (четность) устанавливается в единичное состояние, которое говорит о том, что число единиц в разрядах результата четно.

Разряд *АС* (дополнительный перенос) хранит значение переноса из младшей тетрады результата.

Разряд *Z* (нуль) устанавливается в единичное состояние, которое говорит о нулевом значении результата.

Разряд *S* (знак) устанавливается в состояние, соответствующее знаку результата.

Блок регистров выполняет несколько самостоятельных функций, например сверхоперативного запоминающего устройства для временного хранения операндов в процессе вычислений. Располагаясь в процессоре и являясь местной памятью, он освобождает ОЗУ от временного хранения промежуточной информации.

Каждый из семи программно доступных регистров *A* (расположенного в АЛУ), *B*, *C*, *D*, *E*, *H*, *L* может хранить обрабатываемую информацию и подключаться к внутренней шине через мультиплексор. Результат вычислений может храниться в одном из внутренних регистров или может быть выведен на внешнюю шину. При обработке шестнадцатиразрядных слов из регистров образуются пары *BC*, *DE* и *HL*, причем сдвоенный регистр называется первой буквой — *B*, *D* и *H*. Кроме того, необходимо помнить, что в регистровой паре старшим является первый регистр. Особым свойством обладают программно-доступные регистры *H* и *L*, используемые как для хранения данных, так и для хранения шестнадцатиразрядных адресов ячеек ОЗУ. Программно-недоступные регистры (регистры недоступные пользователю) *W* и *Z* используются как буферные для приема и временного хранения адресов, содержащихся в командах переходов. На рис. 21.3 представлены регистры, которые могут участвовать в формировании адреса.

Счетчик команд (*PC*) используется для приема и хранения текущего адреса команды. Содержимое счетчика команд автоматически увеличивается на единицу (инкрементируется) сразу же после выборки очередного байта команды.

Инкремент 1 (*INC* 1) — операция сложения 1 с операндом, декремент 1 (*DEC* 1) — операция вычитания 1 из операнда. В блоке регистров для увеличения или уменьшения значения адреса на единицу применяется специальная схема (*I/D*).

Регистр адреса (РА) служит для приема, хранения и образования исполнительного адреса данных или устройств ввода-вывода. Адрес может поступать в РА из различных регистров, но чаще из *PC* и *H*, *L*. Как правило, адрес сохраняется в РА в течение одного машинного цикла.

Указатель стека (*SP*) будет рассмотрен далее.

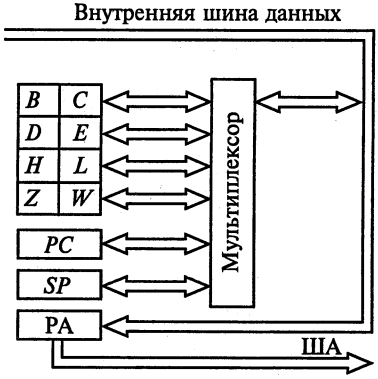


Рис. 21.3

Контрольные вопросы

1. Какие основные функции выполняет микропроцессор?
2. Какое устройство используется для управления вычислительными процессами?
3. Как называется главный регистр МП? Какую роль он играет при выполнении операций в АЛУ?
4. Какие управляющие сигналы должны быть сформированы устройством управления при считывании первого операнда из ОЗУ в аккумулятор АЛУ, при перезаписи содержимого аккумулятора в Рг2, при считывании второго операнда из ОЗУ в Рг1?
5. Будет ли участвовать в вычислительном процессе кодопреобразователь, если производится сложение положительных чисел?
6. Каким образом организуется двоично-десятичное сложение в АЛУ?
7. По каким цепям между устройствами МП происходит обмен данными?
8. Для чего предназначены буфер данных и буфер адреса?
9. Какая информация фиксируется в счетчике команд и как она там может изменяться в зависимости от формата подготавливаемой к выполнению команды?
10. Какая информация фиксируется в регистре адреса и как она там может изменяться?

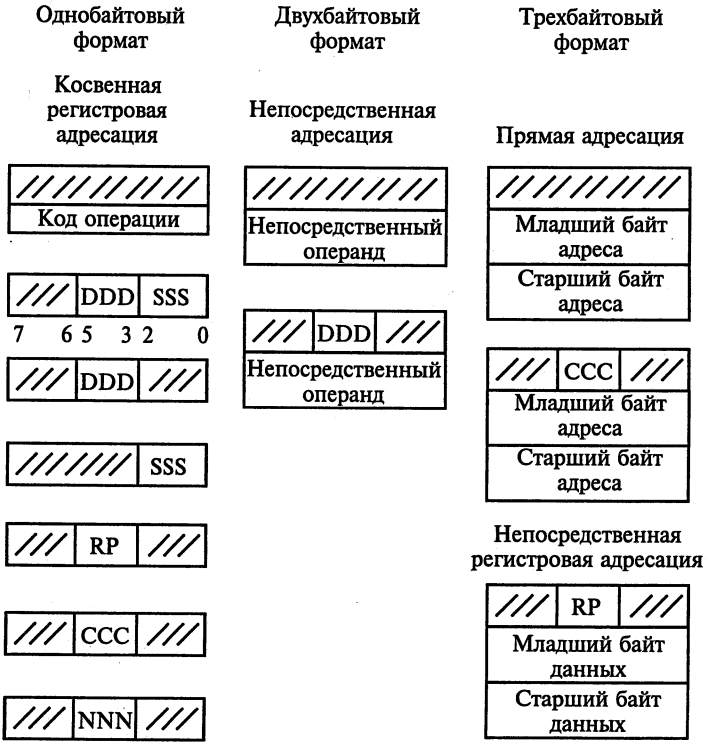
21.2. Система команд

Общие сведения. Для определенной серии МП комплекта используется индивидуальный язык составления программ. Для МП КР580 применяется язык ассемблера. В командах на языке ассемблера код операции имеет сокращенную форму записи английских слов — мнемоническое обозначение. *Мнемоника* (от греч. *mnemonic* — искусство запоминания) позволяет легче запомнить команды по их функциональному назначению. Перед исполнением исходная программа переводится с помощью *программы трансляции*, называемой ассемблер, на язык кодовых комбинаций — *машинный язык*, в таком виде размещается в памяти МП и далее используется при выполнении команды. Например, команда очистки аккумулятора на языке ассемблера имеет вид CLA A (от англ. *clear* — очищать); она же на машинном языке будет иметь вид 0100 1111. Сравнив две эти команды, видим очевидные преимущества написания программы на языке ассемблера, обеспечивающем в первую очередь простоту записи и возможность считывания команд.

Система команд микропроцессора — это полный перечень элементарных действий, которые способен производить МП. Управляемый этими командами МП выполняет простые действия, такие как элементарные арифметические и логические операции, пересылку данных, сравнение двух величин и др. Система команд для МП КР580ВМ80 представлена 244 кодами операций. Команда характеризуется форматом, т. е. числом отведенных для нее разрядов, которые разделены побайтно на определенные функциональ-

ные поля. Для команд используются одно-, двух- и трехбайтовые форматы (рис. 21.4). Многобайтовые команды должны храниться в соседних ячейках памяти.

Первый байт команды содержит код операции, записанный в мнемоническом виде. Он определяет формат команды и те действия, которые должны быть выполнены МП над данными в процессе ее выполнения, и способ адресации, а также может содержать информацию о нахождении данных. Кроме мнемонической записи кода операции остальная информация этого поля для упрощения записывается в шестнадцатеричном виде, который затем легко может быть заменен (транслирован) на двоичный код путем перезаписи каждого символа его четырехразрядным эквивалентом.



Условные обозначения:

DDD – номер регистра приемника;
SSS – номер регистра источника;
CCC – указатель условия передачи управления;
NNN – код, определяющий адрес вектора прерывания;
RP – указатель, адресующий пару регистров.

Рис. 21.4

	0	1	2	3	4	5	6	7
0	NOP	LXI B <B2> <B3>	STAX B	INX B	INR B	DCR B	MVI B <B2>	RLC
1	—	LXI D <B2> <B3>	STAX D	INX D	INR D	DCR D	MVI D <B2>	RAL
2	—	LXI H <B2> <B3>	SHLD <L ADR> <H ADR>	INX H	INR H	DCR H	MVI H <B2>	DAA
3	—	LXI SP <B2> <B3>	STA <L ADR> <H ADR>	INX SP	INR M	DCR M	MVI M <B2>	STC
4	MOV B, B	MOV B, C	MOV B, D	MOV B, E	MOV B, H	MOV B, L	MOV B, M	MOV B, A
5	MOV D, B	MOV D, C	MOV D, D	MOV D, E	MOV D, H	MOV D, L	MOV D, M	MOV D, A
6	MOV H, B	MOV H, C	MOV H, D	MOV H, E	MOV H, H	MOV H, L	MOV H, M	MOV H, A
7	MOV M, B	MOV M, C	MOV M, D	MOV M, E	MOV M, H	MOV M, L	HLT	MOV M, A
8	ADD B	ADD C	ADD D	ADD E	ADD H	ADD L	ADD M	ADD A
9	SUB B	SUB C	SUB D	SUB E	SUB H	SUB L	SUB M	SUB A
A	ANA B	ANA C	ANA D	ANA E	ANA H	ANA L	ANA M	ANA A
B	ORA B	ORA C	ORA D	ORA E	ORA H	ORA L	ORA M	ORA A
C	RNZ	POP B	JNZ <L ADR> <H ADR>	JMP <L ADR> <H ADR>	CNZ <L ADR> <H ADR>	PUSH B	ADI <B2>	RST 0
D	RNC	POP D	JNC <L ADR> <H ADR>	OUT <Port №>	CNC <L ADR> <H ADR>	PUSH D	SUI <B2>	RST 2
E	RPO	POP H	JPO <L ADR> <H ADR>	XTHL	CPO <L ADR> <H ADR>	PUSH H	ANI <B2>	RST 4
F	RP	POP PSW	JP <L ADR> <H ADR>	DI	CP <L ADR> <H ADR>	PUSH PSW	ORI <B2>	RST 6

Во втором и третьем байтах могут находиться данные, над которыми производятся операции, или адреса, указывающие местонахождение данных. Данные, над которыми производятся действия, называются *операндами*. Для упрощения записи содержимое второго и третьего байтов также записывается в шестнадцатеричном виде.

Программа на языке ассемблер — это последовательность строк, в каждой из которых записывается команда, представляющая собой предложение, состоящее из четырех полей: поля метки, поля кода операции, поля операнда и поля комментария.

Например,

LOOP1: _ MOV _ D, A _ ; D ← (A) (содержимое Рг*A* заносится в Рг*D*).

Таблица 21.1

8	9	A	B	C	D	E	F
—	DAD B	LDAX B	DCX B	INR C	DCR C	MVI C <B2>	RRC
—	DAD D	LDAX D	DCX D	INR E	DCR E	MVI E <B2>	RAR
—	DAD H	LHLD <L ADR> <H ADR>	DCX H	INR L	DCR L	MVI L <B2>	CMA
—	DAD SP	LDA <L ADR> <H ADR>	DCX SP	INR A	DCR A	MVI A <B2>	CMC

MOV C, B
MOV E, B
MOV L, B
MOV A, B
ADC B
SBB B
XRA B
CMP B
	MOV C, C
	MOV E, C
	MOV L, C
	MOV A, C
	ADC C
	SBB C
	XRA C
	CMP C
		MOV C, D
		MOV E, D
		MOV L, D
		MOV A, D
		ADC D
		SBB D
		XRA D
		CMP D
			MOV C, E
			MOV E, E
			MOV L, E
			MOV A, E
			ADC E
			SBB E
			XRA E
			CMP E
				MOV C, H
				MOV E, H
				MOV L, H
				MOV A, H
				ADC H
				SBB H
				XRA H
				CMP H
					MOV C, L
					MOV E, L
					MOV L, L
					MOV A, L
					ADC L
					SBB L
					XRA L
					CMP L
						MOV C, M
						MOV E, M
						MOV L, M
						MOV A, M
						ADC M
						SBB M
						XRA M
						CMP M
							MOV C, A
							MOV E, A
							MOV L, A
							MOV A, A
							ADC A
							SBB A
							XRA A
							CMP A

RZ	RET	JZ <L ADR> <H ADR>	—	CZ <L ADR> <H ADR>	CALL <L ADR> <H ADR>	ACI <B2>	RST 1
RC	—	JC <L ADR> <H ADR>	IN <Port №>	CC <L ADR> <H ADR>	—	SBI <B2>	RST 3
RPE	PCHL	JPE <L ADR> <H ADR>	HCHG	CPE <L ADR> <H ADR>	—	XRI <B2>	RST 5
RM	SPHL	JM <L ADR> <H ADR>	EI	CM <L ADR> <H ADR>	—	CPI <B2>	RST 7

Поле метки представляет собой условное имя или набор символов, начинающихся с определенной буквы. После имени метки обязательно ставится двоеточие. Метка присваивается команде в тех случаях, когда при выполнении программы необходимо к ней вернуться (возврат по метке). Имя метки возврата записывается в командах условного и безусловного переходов.

Поле кода операции используется для мнемонического обозначения кода операции команды.

В *поле операнда*, как правило, дается информация о данных, над которыми производятся операции. Данные, над которыми должны быть произведены операции, могут быть заданы непосредственно

в поле команды (непосредственные операнды). В этих полях могут быть заданы адреса источника данных (SSS) — регистры или память, где хранятся требуемые данные, или адреса регистров. Кроме того, в этом поле может быть размещена информация о приемнике результата выполнения операции (DDD). В командах условных и безусловных переходов в этом поле хранится имя метки возврата.

Поле комментария начинается символом «;». Это поле имеет вспомогательные функции и используется только программистом при составлении программы.

Команда состоит из двух частей: кода операции и адреса. Код операции сообщает микропроцессору, что надо делать; адрес указывает местоположение данных, участвующих в операции. В зависимости от формата команды адрес указывается различным количеством полей.

Классификация полей по формату команды и виду адресации показана на рис. 21.4. Заштрихованные на рисунке разряды первого байта команд относятся к коду операции.

Помимо системы команд, приведенной в Приложении, удобно пользоваться табл. 21.1, по которой, зная мнемонику команды, легко определить код ее операции. Например, чтобы по мнемонике определить код команды ADD B, необходимо сначала найти ее в таблице, а затем в крайнем левом столбце и верхней строке — соответствующие ей символы, т.е. мнемонике ADD B соответствует код 80.

Все слова, образующие программу, должны следовать одно за другим и располагаться в соседних ячейках памяти (рис. 21.5).

При регистровой адресации один байт информации может храниться в одном из регистров, коды адресов которых представлены в табл. 21.2. Двухбайтовая информация хранится в регистровых парах, коды адресов которых представлены в табл. 21.3, причем старший байт информации (с 8-го по 15-й разряд) размещается в первом регистре пары, а младший байт (с 0 по 7-й) — во втором регистре пары.

Адрес байта в памяти	Байт	
0800	Код операции	Однобайтовая команда
0801	Код операции	Двухбайтовая команда
0802	Операнд или адрес внешнего устройства	
0803	Код операции	Трехбайтовая команда
0804	Младший байт адреса внешнего устройства или ячейки	
0805	Старший байт адреса внешнего устройства или ячейки	

Рис. 21.5

Таблица 21.2

Регистр	Коды DDD, SSS	Регистр	Коды DDD, SSS
A	111	*E*	011
B	000	*H*	100
C	001	*L*	101
D	010	*M*	110

Таблица 21.3

Пара регистров	Код RP	Пара регистров	Код RP
BC	00	*HL*	10
DE	01	*SP*	11

Под регистром *M* понимается ячейка оперативной памяти, адрес которой находится в паре регистров *HL* или байте команды.

Классификация команд. Различают следующие группы команд: пересылки данных, арифметических и логических операций, перехода, управления и вызова подпрограмм. Рассмотрим различные форматы команд, полный объем которых представлен в Приложении.

Команды пересылки данных организуют перемещение данных между следующими устройствами:

регистр — регистр;

регистр — память;

данные, содержащиеся в команде, — регистр;

данные, содержащиеся в команде, — память.

Команды пересылки данных могут быть однобайтовыми, двухбайтовыми и трехбайтовыми.

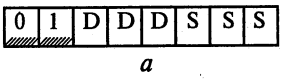
Рассмотрим некоторые из них.

С помощью однобайтовых команд производится обмен данными между внутренними регистрами микропроцессора, а также между внутренними регистрами и ячейками памяти.

Команды пересылки данных из регистра в регистр **MOV r1, r2.** Содержимое регистра **r2** (источник операнда — S) пересылается в регистр **r1** (приемник операнда — D). В качестве регистров S и D может быть использован любой из регистров *A*, *B*, *C*, *D*, *H*, *L*. Формат команды представлен на рис. 21.6, *а*; одна из команд в машинных кодах — на рис. 21.6, *б*.

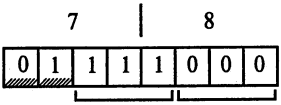
Содержимое регистра SSS при этом не меняется. В регистре DDD просто появляется копия содержимого регистра SSS.

Команды пересылки данных из регистра в память **MOV M, r.** Содержимое регистра **r** (источник операнда — S) пересылается в ячейку памяти (приемник операнда — M) по адресу, указанному



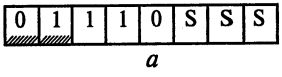
а

MOV A, B
Код команды 78



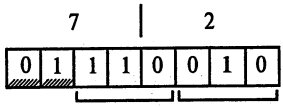
б

Рис. 21.6



а

MOV M, D
Код команды 72



б

Рис. 21.7

в регистровой паре *HL*. В этой команде происходит замена кода символа D на код символа M. После выполнения команды в ячейке памяти появится копия содержимого регистра-источника. Содержимое самого регистра не изменится. Формат команды представлен на рис. 21.7, *а*; одна из команд в машинных кодах — на рис. 21.7, *б*.

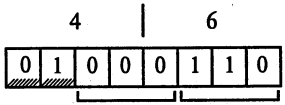
Команды пересылки данных из памяти в регистр **MOV r, M**. Содержимое ячейки памяти (источник операнда — M) пересылается в регистр **r** (приемник операнда — D). Пересылка данных осуществляется из ячейки памяти по адресу, указанному в регистровой паре *HL*, в регистр-приемник, в качестве которого может выступать любой из регистров *B*, *C*, *D*, *H*, *L* или аккумулятор *A*. Формат команды представлен на рис. 21.8, *а*; одна из команд в машинных кодах — на рис. 21.8, *б*.

При пересылке данных между аккумулятором и ячейкой памяти в качестве адреса ячейки памяти может быть использовано также содержимое регистровых пар *BC* или *DE*. Тогда для записи в память данных или чтения данных из памяти могут быть использованы команды типа **STAX** или **LDAX**.



а

MOV B, M
Код команды 46



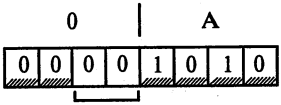
б

Рис. 21.8



а

LDAX B
Код команды 0A



б

Рис. 21.9

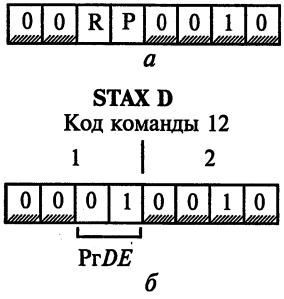


Рис. 21.10

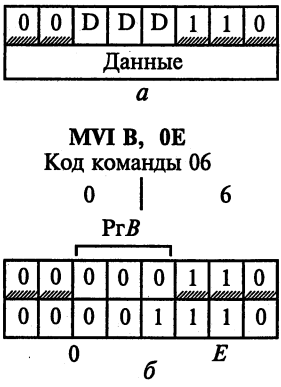


Рис. 21.11

Команды косвенной загрузки аккумулятора **LDAX гр**. Содержимое ячейки памяти, адрес которой указан в паре регистров **гр**, пересылается в регистр *A*. Формат команды представлен на рис. 21.9, *а*; одна из команд в машинных кодах — на рис. 21.9, *б*.

Команды косвенной записи содержимого аккумулятора в память **STAX гр**. Содержимое аккумулятора пересылается в ячейку памяти, адрес которой определен в паре регистров **гр** (рис. 21.10, *а* и *б*).

К двухбайтовым командам относятся два формата команд, с помощью которых можно переслать содержимое второго байта в регистр или ячейку памяти.

Команды загрузки регистра вторым байтом команды **MVI, г байт** (*непосредственная пересылка второго байта команды в регистр*). Содержимое второго байта команды пересылается в регистр **г**. Формат команды представлен на рис. 21.11, *а*; одна из команд в машинных кодах — на рис. 21.11, *б*.

Команда загрузки ячейки памяти вторым байтом команды **MVI M, байт** (*непосредственная пересылка второго байта в память*). Содержимое второго байта команды пересылается в ячейку памяти, адрес которой указан в регистрах *H* и *L*. Формат команды представлен на рис. 21.12, *а*; команда, представленная в машинных кодах, — на рис. 21.12, *б*.

Трехбайтовые команды служат для непосредственной записи в соответствующие регистровые пары данных из второго и третьего байтов.

Команда непосредственной загрузки пары регистров **LXI гр, 2**. Третий байт команды посылается в старший регистр (**rh**) пары регистров, второй — в младший регистр (**rl**) пары регистров. Формат команды представлен на рис. 21.13, *а*; одна из команд в машинных кодах — на рис. 21.13, *б*.

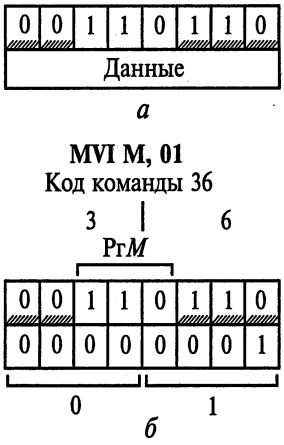


Рис. 21.12



Рис. 21.13

Помимо ранее рассмотренных команд, используемых для пересылки данных между аккумулятором и ячейкой памяти для записи в память данных или чтения данных из памяти, могут быть использованы команды типа **STA** или **LDA**.

Команда прямой записи содержимого аккумулятора в память **STA [адр.]**. Содержимое аккумулятора пересылается в ячейку памяти, адрес которой указан во втором и третьем байтах команды. Формат команды представлен на рис. 21.14, *а*; одна из команд в машинных кодах — на рис. 21.14, *б*.

Команды арифметических операций предназначены для выполнения арифметических операций над данными, хранимыми в регистрах и ячейках памяти. Перед началом выполнения любой команды один из операндов должен быть размещен в аккумуляторе, а другой (если команда однобайтовая) — в одном из восьмиразрядных внутренних регистров МП или ячейке памяти с адресом, содержащимся в регистровой паре *HL*. В двухбайтовой команде значение второго операнда задается непосредственно во втором байте команды. Полученный результат операции фиксируется в аккуму-

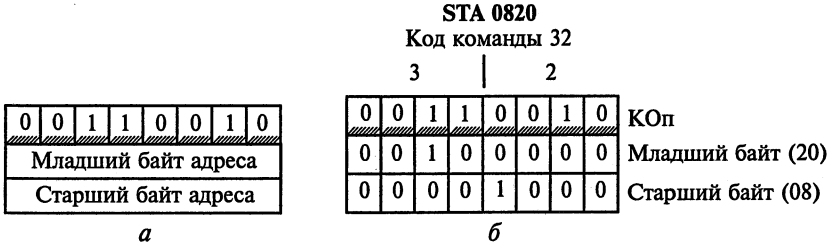


Рис. 21.14

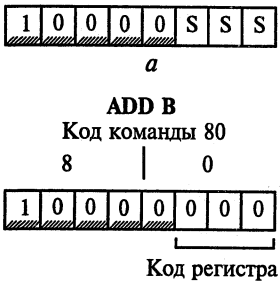


Рис. 21.15

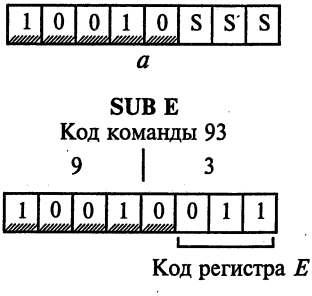


Рис. 21.16

ляторе, анализируется, и результат анализа заносится в регистр признаков (флаговый регистр), что может повлиять на порядок выполнения вычислительного процесса. Рассмотрим некоторые команды.

Команды сложения содержимого аккумулятора с содержимым регистра **ADD r**. Команды выполняют сложение содержимого регистра-источника S с содержимым аккумулятора *A*. Результат сложения помещается в аккумулятор. Чтобы получить конкретную команду, необходимо в формате команды вместо символов S проставить код регистра-источника. Формат команды представлен на рис. 21.15, *а*; одна из команд в машинных кодах — на рис. 21.15, *б*.

Команды вычитания содержимого регистра **SUB r**. Команды выполняют вычитание содержимого регистра-источника S из содержимого аккумулятора. Результат помещается в аккумулятор. Формат команды представлен на рис. 21.16, *а*; одна из команд в машинных кодах — на рис. 21.16, *б*.

Команда непосредственного вычитания **SUB байт**. Содержимое второго байта команды вычитается из содержимого аккумулятора. Результат помещается в аккумулятор. Формат команды представлен на рис. 21.17, *а*; команда в машинных кодах — на рис. 21.17, *б*.

Команды выполнения логических операций предназначены для выполнения логических операций над данными, содержащимися в регистрах, ячейках памяти, а также над флагами условий. К этим операциям относятся: логическое сложение (ИЛИ), логическое умножение (И), суммирование по модулю два, сравнение, сдвиг, дополнение до 1 и до 2. Как и команды предыдущей группы, все логические команды оказывают влияние на флаги.

Операция И над содержимым регистра и аккумулятора **ANA r**. Команда выполняет параллельно поразрядное логическое И над содержимым регистра-источника и аккумулятора. Результат операции заносится в аккумулятор. Формат команды представлен на рис. 21.18, *а*; одна из команд в машинных кодах — на рис. 21.18, *б*.

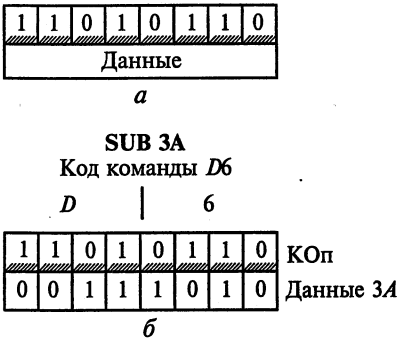


Рис. 21.17



Рис. 21.18

Операция И над содержимым второго байта команды **ANI байт**. Команда является двухбайтовой и также выполняет поразрядную операцию логического И, но над содержимым второго байта команды и аккумулятора. Результат помещается в аккумулятор. Формат команды представлен на рис. 21.19, *а*; команда в машинных кодах — на рис. 21.19, *б*.

Операция ИЛИ над содержимым аккумулятора и одного из регистров **ORA r**. Команда аналогична команде **ANA r**, но в отличие от нее выполняет логическое сложение содержимого регистра и аккумулятора. Результат помещается в аккумулятор. Формат команды представлен на рис. 21.20, *а*; одна из команд в машинных кодах — на рис. 21.20, *б*.

Команды перехода позволяют изменить последовательность выполнения команд программы за счет выполнения безусловных и условных переходов. При выполнении команд *безусловного перехода* текущая последовательность команд нарушается и происходит переход к выполнению другой последовательности команд, указанной в команде. Для организации этого перехода необходимо



Рис. 21.19

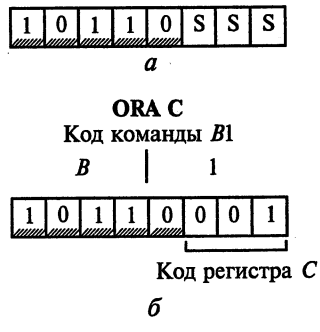


Рис. 21.20

Таблица 21.4

Условие	Код
NZ — не нуль (Z = 0)	000
Z — нуль (Z = 1)	001
NC — нет переноса (CY = 0)	010
C — есть перенос (CY = 1)	011
PO — нечетный результат (P = 0)	100
PE — четный результат (P = 1)	101
P — плюс (S = 0)	110
M — минус (S = 1)	111

изменить адрес обращения за очередной командой в счетчике команд. При этом флаги влияния на организацию перехода не оказывают. При выполнении команд *условного перехода* дальнейший выбор одной из последовательностей команд зависит от значения одного из четырех флагов: нуля, знака, четности, переноса, т.е. сначала проверяется состояние одного из четырех флагов МП на выполнение указанного в команде условия, а затем происходит выполнение команды. Например, если результат арифметической операции равен нулю, то необходимо осуществить переход к выполнению другой последовательности команд, начальный адрес которой указан в команде, если не равен — продолжить выполнение текущей программы. Условия перехода приведены в табл. 21.4. На рис. 21.21 *а* и *б* представлен вид команды безусловного перехода **JMP [адр.]**, на рис. 21.21, *в* — вид команды условного перехода **J (условие)[адр.]**.

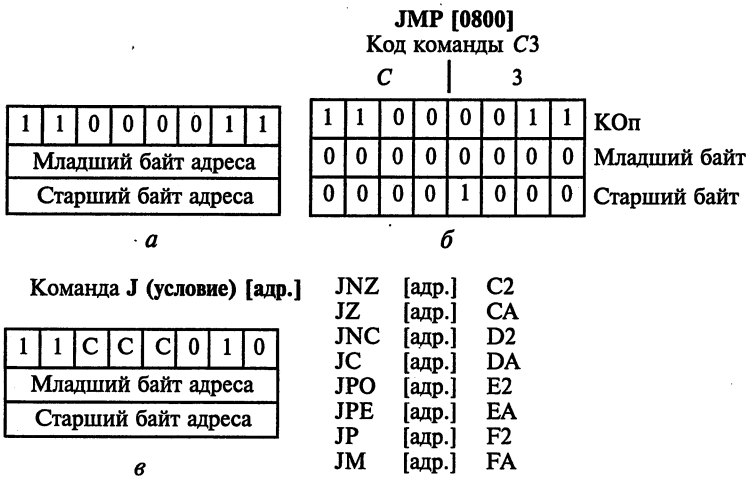
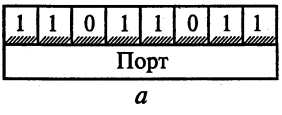


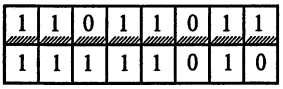
Рис. 21.21



а

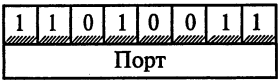
IN FA
Код команды *DB*

D | *B*



б

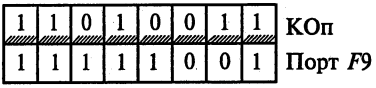
Рис. 21.22



а

OUT F9
Код команды *D*3

D | 3



б

Рис. 21.23

Команды управления предназначены для управления работой МП, устройствами ввода-вывода, стеком. Команды этой группы не оказывают влияния на флаги. Рассмотрим некоторые из них.

Команда ввода данных **IN порт** предназначена для ввода данных от входного порта по ШД в аккумулятор. Формат команды представлен на рис. 21.22, *а*; команда в машинных кодах — на рис. 21.22, *б*.

Команда читается следующим образом: «Считывание данных из порта ввода (*FA*) в аккумулятор».

Команда вывода данных **OUT порт** предназначена для вывода данных из аккумулятора в выходной порт. Формат команды представлен на рис. 21.23, *а*; команда в машинных кодах — на рис. 21.23, *б*.

Команды **PUSH** используются для временного сохранения содержимого внутренних регистров МП в стеке при переходе к выполнению подпрограммы. По команде **POP** происходит извлечение из стека временно хранящихся данных после выполнения подпрограммы.

С помощью команд **XTHL** и **SPHL** происходит обмен данными между указателем стека *SP* и парой регистров *H* и *L* в том или ином направлении, соответственно.

По команде **HLT** (останов) останавливается выполнение текущей программы до тех пор, пока не появится запрос на прерывание от устройств ввода-вывода.

По команде **DI** (запрещение прерываний) МП игнорирует запросы на прерывание до тех пор, пока не получит команду **EI** (разрешение прерываний).

По команде **NOP** (нет операции) не производится никаких вычислений, изменяется лишь содержимое счетчика команд *PC*. Команда используется для организации временных задержек и часто применяется для организации обращения к медленно действующим устройствам.

Команды вызова подпрограмм будут рассмотрены в подразд. 21.3.

Примеры составления простейших программ. На основании рассмотренной системы команд составим несколько простейших программ.

Программа 1.

1. Очистить аккумулятор.
2. Записать содержимое аккумулятора в ячейку 0820.
3. Выполнить временную задержку.
4. Прибавить к содержимому аккумулятора единицу.
5. Записать содержимое аккумулятора в ячейку 0820 и т.д.

Возврат осуществляется по метке **LOOP**. Команда **NOP** не выполняет никаких действий, а используется для организации временной задержки. Структурная схема программы представлена на рис. 21.24.

На основе структурной схемы составляется программа:

```
      MVI A, 0
LOOP: STA 0820
      NOP
      INR A
      JMP LOOP
```

Программа 2.

Загрузить две соседние ячейки памяти с адресами 0800 и 0801 числами 5A и 17, для чего выполнить следующие действия.

1. При обращении к памяти задать адрес обращения, который фиксируется в паре регистров *HL*, причем в старший регистр *H* занести старший байт адреса — 08, а в младший *L* — 00:

```
MVI H, 08
MVI L, 00
```

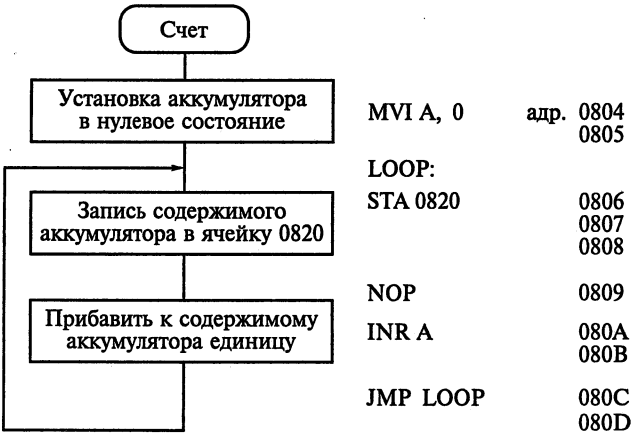


Рис. 21.24

2. По адресу, хранящемуся в паре регистров *HL*, записать число 5A:

```
MVI M, 5A
```

3. Сформировать адрес соседней ячейки путем приращения единицы к содержимому пары регистров *HL*:

```
INX H
```

4. По адресу 0801 записать число 17:

```
MVI M, 17
```

В целом, программа будет иметь следующий вид:

```
MVI H, 08
MVI L, 00
MVI M, 5A
INX H
MVI M,17
RST 1
```

Программа 3.

1. Загрузить аккумулятор числом FF.
2. Содержимое аккумулятора переслать в ячейку памяти по адресу 0820.
3. Содержимое ячейки памяти по адресу 0820 загрузить в аккумулятор.
4. Инвертировать содержимое аккумулятора.
5. Переписать содержимое аккумулятора по адресу 0821:

```
MVI A, FF
STA 0820
LDA 0820
CMA
STA 0821
RST 1
```

В табл. 21.5 представлено расположение программы в памяти.

Программа 4. Рассмотрим другой вариант построения программы 3. Разница заключается в том, что адрес обращения к памяти находится не в самой команде, а в паре регистров *HL*:

```
MVI A, FF
STA 0820
LXI H, 0820
MOV A, M
CMA
INX H
MOV M, A
RST 1
```

В табл. 21.6 представлено расположение программы в памяти.

Таблица 21.5

Адрес	Команда	Содержимое байта
0800	MVI A, FF	3E
0801		FF
0802	STA 0820	32
0803		20
0804		08
0805	LDA 0820	3A
0806		20
0807		08
0808	CMA	2F
0809	STA 0821	32
080A		21
080B		08
080C	RST 1	CF

Таблица 21.6

Адрес	Команда	Содержимое байта
0800	MVI A, FF	3E
0801		FF
0802	STA 0820	32
0803		20
0804		08
0805	LXI H, 0820	21
0806		20
0807		08
0808	MOV A, M	7E
0809	CMA	2F
080A	INX H	23
080B	MOV M, A	77
080C	RST 1	CF

Контрольные вопросы

1. Для чего предназначены поля в однобайтовой команде, представленной на рис. 21.4, *а*; в двухбайтовой команде — на рис. 21.4, *б*; в трехбайтовой команде — на рис. 21.4, *в*?
2. Почему команды пересылки данных могут быть названы командами копирования данных?
3. При помощи какой команды регистр *B* загружается данными, содержащимися во втором байте команды?
4. При помощи какой команды содержимое аккумулятора записывается в регистр *D*?
5. Какая мнемоника команд используется при перезаписи данных из одного регистра в другой?
6. Чем отличаются команды LDA и STA?

7. С помощью каких команд можно записать в ячейку памяти с адресом 0821 данные, находящиеся в регистре *D*?

8. Где должен находиться первоначальный источник адреса перед обращением в память при необходимости выполнения безусловного перехода?

9. Куда и для чего поступают считанные из памяти данные?

10. Какой регистр используется для перезаписи информации в ячейку памяти?

11. Какие операции выполняются при реализации команд программ, представленных в табл. 21.5 и 21.6?

Контрольные задания

Составить четыре программы в виде таблиц в соответствии с заданием:

Программа 1.

1.Загрузить регистр *C* числом 07.

2. Загрузить регистр *B* числом 05.

3. Произвести сложение содержимого регистров.

4. Переслать результат сложения в ячейку памяти 0820.

5. Инвертировать содержимое регистра *A* и разместить его по адресу 0821.

Программа 2.

1. Загрузить аккумулятор числом 0*A*.

2. Загрузить ячейку памяти с адресом 0820 числом 10.

3. Выполнить считывание данных из ячейки 0820 и направить их в регистр *C*.

4. Произвести сложение содержимого регистров *A* и *C*.

5. Уменьшить результат на 1 и отправить его по адресу памяти 0821.

Программа 3.

1. Последовательно загрузить регистры числами: *B* — 01; *C* — 02; *D* — FF; *E* — 00.

2. Переслать содержимое регистров *D* и *F* в ячейки памяти с адресами 0820 и 0821 соответственно.

3. Переслать содержимое регистра *B* в аккумулятор.

4. Сложить содержимое регистров *A* и *C*.

5. Отправить результат в ячейку 0822.

Программа 4.

1. Загрузить ячейку памяти 0820 числом 01.

2. Загрузить регистр аккумулятора числом FF.

3. Считать содержимое ячейки 0820 в регистр *C*.

4. Вычесть содержимое регистра *C* из содержимого аккумулятора.

5. Отправить результат в ячейку памяти 0821.

Таблица 21.7

Регистры					ОЗУ		
A	B	C	D	E	0820	0821	0822

Для каждой программы оформить табл. 21.7, и занести в нее предполагаемое содержимое используемых регистров и ячеек памяти после выполнения программы.

21.3. Способы адресации

При составлении программ важно знать особенности процедур, позволяющих преобразовать информацию об адресах команд и данных в физические адреса. Для МП КР580ВМ80 существуют следующие способы адресации: непосредственная, прямая, косвенная. Вид адресации для каждой команды указан в системе команд (см. Приложение).

Непосредственная адресация используется в двух- и трехбайтовых командах (рис. 21.25, *а*). Данные содержатся во втором или втором и третьем байтах команды, причем младший байт данных находится во втором байте команды, а старший — в третьем.

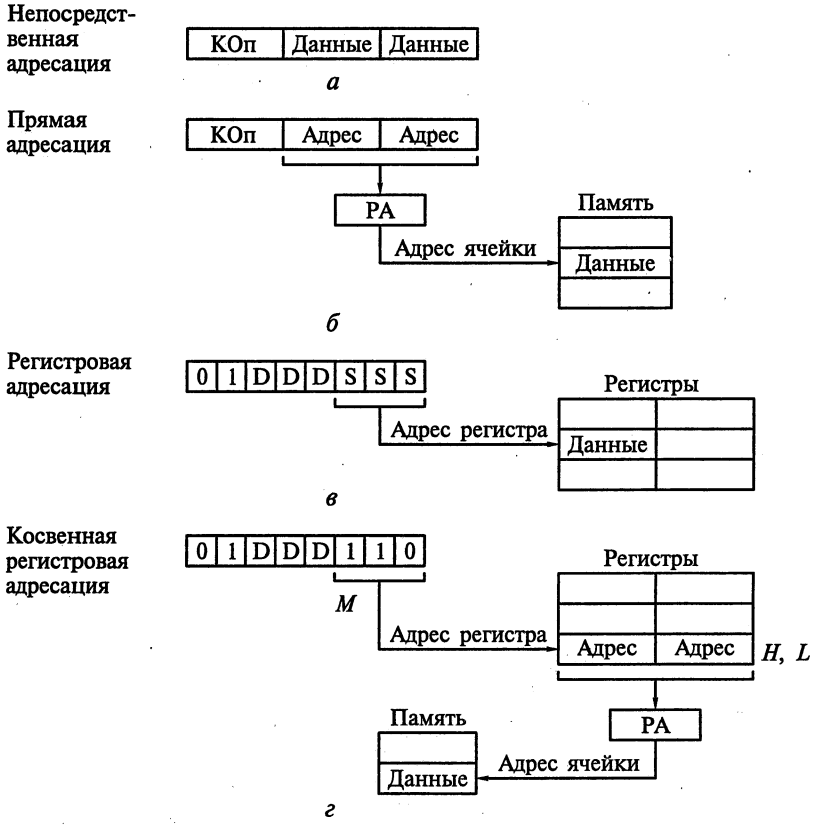


Рис. 21.25

Прямая адресация используется в трехбайтовых командах. Во втором и третьем байтах записан шестнадцатиразрядный адрес обращения к ячейке памяти, где хранятся данные (рис. 21.25, *б*). Содержимое этих байтов доносится в регистр адреса, а затем выводится на ША.

Регистровая адресация используется в однобайтовых командах, в определенных полях которых находится информация об одном или паре регистров, где хранятся необходимые данные (рис. 21.25, *в*).

Косвенная (*неявная*) *регистровая адресация* является разновидностью регистровой адресации. В этом случае в разрядах команды, отведенных под источник (SSS), находится информация о паре регистров, хранящих адрес ячейки памяти с необходимыми данными. Как правило, адреса хранятся в регистрах *H* и *L*, но иногда используются и пары регистров *BD* (рис. 21.25, *г*). Использование косвенной адресации дает наибольший эффект при записи и чтении следующих одна за другой областей памяти.

Кроме описанных способов адресации возможна адресация к ячейкам памяти по содержимому шестнадцатиразрядного регистра *SP* —указателя стека.

Стек — это память с определенной (упрощенной) формой адресации.

Под стеком в микроЭВМ на базе МП КР580ВМ80 понимается любая область ОЗУ, служащая для хранения содержимого внутренних регистров МП со специальным упрощенным режимом доступа. Основная задача стека — обслуживание прерываний. В случае, когда нужно прекратить выполняемую последовательность команд, адрес подготовленной, но не выполненной команды, посылается на хранение в стек. При необходимости возврата к прерванной программе адрес команды, с которой возобновляется прерванная программа, извлекается из стека. Так можно вы-

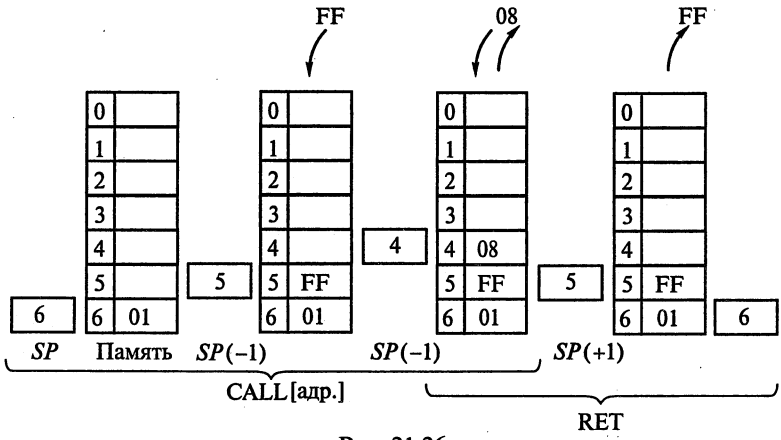


Рис. 21.26

полнить последовательный ряд подпрограмм, в которых также предполагаются прерывания (вложенные прерывания). При этом стек заполняется адресами возврата к прерванным подпрограммам. Затем по мере извлечения адресов возврата к прерванным подпрограммам будет происходить их последовательное выполнение.

На рис. 21.26 представлена упрощенная модель стека. Ячейки стека имеют последовательно убывающую адресацию. Для организации обращения в верхнюю свободную ячейку памяти (дно свободного пространства стека) используется двухбайтовый указатель стека *SP*, расположенный в блоке регистров МП. По мере заполнения стека содержимое *SP* уменьшается на единицу, когда же данные извлекаются из стека, содержимое *SP* увеличивается на единицу, причем *SP* всегда указывает на адрес последней заполненной ячейки стека. Увеличение или уменьшение содержимого стека производится с помощью схемы инкремента/декремента (*I*/*D*).

CALL (адр.)
Код команды *CD*

C				*D*			
1	1	0	0	1	1	0	1
Младший байт адреса							
Старший байт адреса							

а

RET
Код команды *C9*

C				*9*			
1	1	0	0	1	0	0	1

б

Рис. 21.27

Рассмотрим работу стека на примере выполнения команды передачи управления подпрограмме **CALL [адр.]** и команды передачи управления главной программе **RET**.

При выполнении прерывания необходимо сохранить адрес команды подготовленной к выполнению, но не выполненной прерванной программы (по команде CALL), с тем чтобы в дальнейшем иметь возможность возврата к ней (по команде RET). Чтобы разместить в свободной ячейке старший байт адреса команды, надо подготовить адресное обращение к стеку, для чего содержимое указателя стека *SP* уменьшается на единицу *SP* (–1). После размещения старшего байта адреса содержимое указателя стека *SP* еще раз уменьшается на единицу *SP* (–2). По подготовленному адресу размещается младший байт адреса команды.

Формат команды CALL представлен на рис. 21.27, *а*.

По команде RET происходит извлечение адреса команды, с которой необходимо продолжить выполнение прерванной программы. Старший байт адреса возврата извлекается по адресу, хранящемуся в *SP*; затем показание *SP* увеличивается на единицу *SP* (+1). Младший байт адреса возврата извлекается по только что полученному адресу в *SP*, после чего показание *SP* опять увеличивается на единицу *SP* (+2). Формат команды RET представлен на рис. 21.27, *б*.

Контрольные вопросы

1. Какой способ адресации команды применяется, если данные находятся в ее полях?

2. Какой способ адресации команды применяется, когда адрес обращения к памяти находится в ее полях?

3. Какой способ адресации команды применяется, когда адрес обращения к регистрам находится в формате команды?

4. За какими символами закреплено условное обозначение регистров приемников и регистров источников данных? Какие разряды занимают эти символы в однобайтовой команде?

5. Назвать коды регистров МП *A*, *D*, *H*. В каком месте в однобайтовой команде эти коды могут находиться в качестве источников и приемников информации?

6. Какие регистры образуют пары? Какими символами обозначаются эти регистры? В каких разрядах команды расположены коды таких пар?

7. Для чего предназначены регистры *H* и *L*?

8. Привести пример команды, предполагающей использование регистров *H* и *L*. Как называется способ адресации для этой команды?

9. В чем заключается сущность механизма обращения к памяти при косвенной адресации?

10. Какие существуют виды адресации, предполагающие обращение к памяти?

11. Для чего предназначен стек?

12. Какой вид информации может находиться в указателе стека *SP*?

13. В каком месте на схеме МП расположен указатель стека? С помощью какого устройства происходит увеличение и уменьшение содержимого *SP*?

14. В какой момент времени происходит уменьшение содержимого стека на единицу?

15. В какой момент времени происходит увеличение содержимого стека на единицу?

21.4. Процесс выполнения команды в микропроцессорном устройстве

Формирование сигналов для ШУ. Работа МП по реализации команд программы основана на принципе микропрограммного управления. Считываемая из памяти микропроцессором команда реализуется как некоторая последовательность микрокоманд или микроопераций. Из памяти команда поступает в регистр команд. Код операции команды дешифрируется, в результате чего формируются управляющие сигналы для выполнения последовательности микрокоманд — микропрограммы, соответствующей команде. Выполнение отдельных микрокоманд синхронизируется двумя последовательностями тактовых импульсов *F1* и *F2*, которые формируются генератором тактовых импульсов и называются *машинными тактами*. За такт принят период следования импульсов с тактовой частотой *F1*.

Выполнение команды происходит в два этапа:

на первом этапе осуществляется обмен байтами информации по ШД между МП и памятью, МП и устройствами ввода-вывода;

на втором — непосредственное выполнение команды.

Время, необходимое для обмена информацией по шине данных, называется *машинным циклом* (М).

В МП существуют десять типов машинных циклов:

1. Извлечение из памяти кода операции команды (М1).
2. Считывание данных из памяти.
3. Запись данных в память.
4. Считывание данных из стека.
5. Запись данных в стек.
6. Ввод данных с устройства ввода.
7. Вывод данных на устройство вывода.
8. Цикл обслуживания прерывания.
9. Останов.
10. Обслуживание прерывания в режиме «Останов».

Команда может выполняться за один или несколько (до пяти) циклов в зависимости от ее вида. В свою очередь машинный цикл делится на 3...5 тактов.

Каждому типу машинного цикла соответствует восьмиразрядная кодовая комбинация ССП — слово состояния процессора (или управляющее слово), которое хранится в специальном регистре (РгСС). ССП выставляется в начале каждого машинного цикла и остается в регистре на всем его протяжении, определяя имя одного из десяти выполняемых циклов ССП и обеспечивая получение необходимой информации в виде кодовой комбинации (табл. 21.8).

На основе ССП формируются необходимые управляющие сигналы воздействия на внешние по отношению к МП устройства (рис. 21.28). Необходимо иметь в виду, что ССП первого цикла любой команды не зависит от тех операций, которые должны быть выполнены при ее реализации. ССП первого цикла (М1) всегда

Таблица 21.8

Кодовая комбинация ССП								Тип цикла
D7	*D6*	*D5*	*D4*	*D3*	*D2*	*D1*	*D0*	
1	0	1	0	0	0	1	0	Выборка первого байта команды (М1)
1	0	0	0	0	0	1	0	Считывание из памяти
0	0	0	0	0	0	0	0	Запись в память
1	0	0	0	0	1	1	0	Считывание из стека
0	0	0	0	0	1	0	0	Запись в стек
0	1	0	0	0	0	1	0	Ввод из УВВ
0	0	0	1	0	0	0	0	Вывод в УВВ
0	0	0	0	0	0	1	1	Подтверждение прерывания
1	0	0	0	1	0	1	0	Подтверждение останова
0	0	1	0	1	0	1	1	Подтверждение прерывания на останове

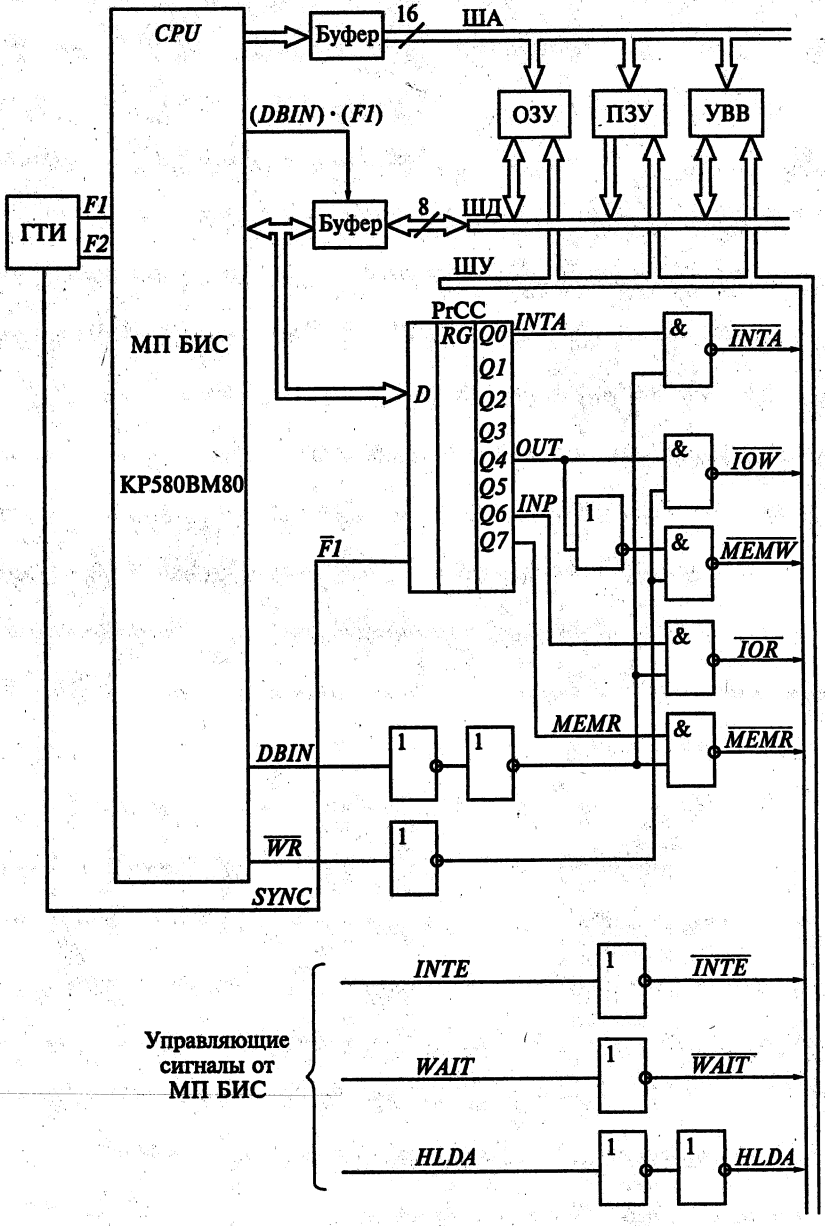

CPU
Буфер
16
ША
ОЗУ
ПЗУ
УВВ
(DBIN) · (F1)
ГТИ
F1
F2
Буфер
8
ШД
ШУ
РгСС
RG
Q0
Q1
Q2
Q3
Q4
Q5
Q6
Q7
D
МП БИС
КР580ВМ80
INTA
OUT
INP
MEMR
F1
DBIN
WR
SYNC
IOW
MEMW
IOR
1
&
Управляющие сигналы от МП БИС
INTE
WAIT
HLDA

Рис. 21.28

соответствует извлечению из памяти кода операции команды выполняемой программы. ССП, выставляемые в последующих циклах выполняемой команды, будут зависеть от извлеченного кода операции.

Для того чтобы показать, что по ШД идет процесс передачи именно ССП, параллельно выставляется сигнал *SYNC*, по которому происходит запись ССП в регистр. В табл. 21.8 представлены кодовые комбинации ССП, соответствующие десяти машинным циклам. Наличие единицы в каждом разряде несет определенную смысловую нагрузку (табл. 21.9).

На рис. 21.29 даны два варианта условного графического обозначения МП КР580ВМ80, а в табл. 21.10 — функциональное назначение сигналов МП. В дальнейшем необходимо иметь в виду, что все устройства микроЭВМ по отношению к МП определяются как внешние.

На основе определенных разрядов ССП (*D0*, *D4*, *D6*, *D7*) и сигналов управления *SYNC*, *WR*, *DBIN* происходит формирование управляющих сигналов для внешних устройств. Для лучшего понимания принципа формирования управляющих сигналов на рис. 21.28 показана схема с использованием дополнительной логики, а на рис. 21.30 — с использованием интегральной микросхемы КР580ВК28.

Таблица 21.9

Разряды	Назначение сигнала
D0 (*INTA*)	*Обслуживание прерывания.* МП выдает сигнал подтверждения прерывания
D1 (*WR*)	*Запись.* При *D1* = 0 происходит запись данных во внешнее по отношению к МП устройство, при *D1* = 1 — считывание
D2 (*STACK*)	*Работа со стеком.* На ША находится адрес обращения к стеку, поступивший из *SP*
D3 (*HLTA*)	*Обслуживание останова.* При *D3* = 1 процессор находится в режиме «Останов» по команде HLT
D4 (*OUT*)	*Вывод.* Данные с ШД выводятся на внешнее устройство вывода, адрес которого находится на ША
D5 (M1)	Из памяти в процессоре на первом цикле M1 передается код операции команды
D6 (*INP*)	*Ввод.* Данные с устройства ввода через ШД вводятся в аккумулятор МП, адрес которого находится на ША
D7 (*MEMR*)	*Память.* ШД используется для считывания данных из памяти

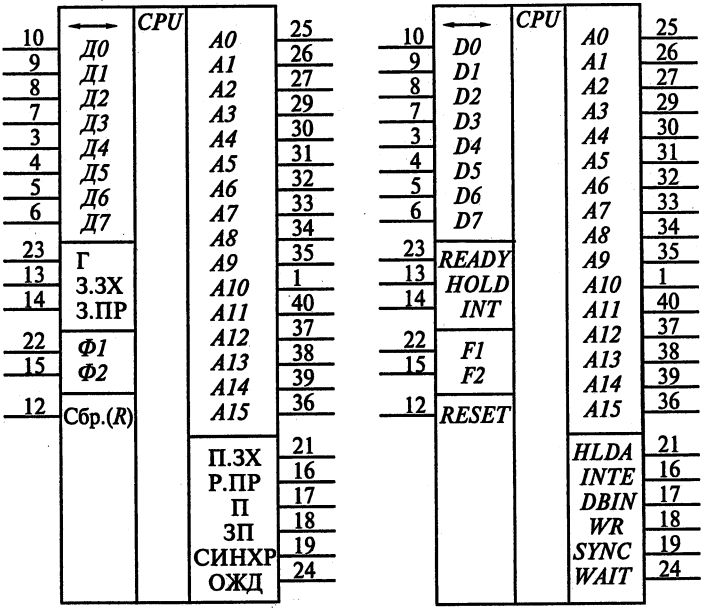


Рис. 21.29

Таблица 21.10

Вывод		Функциональное назначение вывода
Англ.	Русск.	
A0...A15	*A0...A15*	Однонаправленная ША. Выходные сигналы с МП используются для адресации памяти и портов ввода-вывода. Адрес выставляется на ША в такте *T1* фронта *F2* и длится до фронта *F2* в такте, следующем за тактом *T3*
D0... D7	*Д0...Д7*	Двунаправленная ШД. Используется при обмене данными между МП и памятью, между МП и устройствами ввода-вывода. Байт данных сопровождается управляющими сигналами *DBIN* или $\overline{WR}$. При считывании время нахождения байта данных на ШД соответствует длительности управляющего сигнала *DBIN* (от такта *T2* фронта *F2* до такта *T3* фронта *F2*). Байт данных должен быть установлен за 30 нс до среза *F1* такта *T2* и снят не менее чем через 130 нс после такта *T3* фронта *F2*. При записи время нахождения байта данных на ШД (от такта *T2* фронта *F2*) длится, пока существует сигнал $\overline{WR}$ (до такта *T3* или $T_{ож}$ фронта *F1*), и снимается в такте, следующем за тактом *T3*.

Продолжение табл. 21.10

Вывод		Функциональное назначение вывода
Англ.	Русск.	
		Кроме того, ШД используется для вывода ССП в начале каждого машинного цикла, который сопровождается сигналом управления *SYNC* и соответствует его длительности. Сигнал снимается через 220...280 нс после такта *T2* фронта *F2*
SYNC	СИНХР (С)	Выходной сигнал «Синхронизация». Сопровождает выдачу МП ССП на вход РгСС в начале каждого машинного цикла и длится от такта *T1* фронта *F2* до такта *T2* фронта *F2*. Сигнал снимается через 130...200 нс после такта *T2* фронта *F2*
DBIN	П(рием)	Выходной сигнал «Прием информации». Равен лог. 1 и длится от такта *T2* фронта *F2* до такта *T3* фронта *F2*. Сигнал устанавливается с задержкой 130...200 нс относительно фронта *F2*; в такте *T2* снимается с максимальной задержкой 200 нс относительно такта *T3* фронта *F2*. При совместном использовании сигнала *DBIN* с *D7* или *D6* ССП формируются сигналы управления для ШУ: считывание данных из памяти — $\overline{MEMR}$ или считывание данных с УВВ — $\overline{IOR}$ соответственно. В конце действия сигнала *DBIN* в момент прихода тактового импульса *F1* через двунаправленный шинный формирователь (буфер данных) происходит ввод байта данных с ШД в один из внутренних регистров МП
$\overline{WR}$	$\overline{\text{ЗП}}$	Выходной сигнал «Запись информации». Длится от такта *T3* или $T_{ож}$ фронта *F1* и снимается в такте, следующем за тактом *T3*. При совместном использовании сигнала с $\overline{D4}$ или *D4* ССП формируются сигналы управления ШУ: запись данных в память — $\overline{MEMW}$ или запись данных в УВВ — $\overline{IOW}$. При наличии $\overline{WR} = 0$ момент записи во внешнее устройство определяется тактовым импульсом *F2*
READY	Г(отов)	Входной сигнал «Готов» от модулей памяти или портов ввода-вывода. Указывает на их готовность к обмену данными с МП. Анализ готовности внешних устройств производится МП в машинных циклах в такте *T2* по фронту *F2*. Если сигнал равен лог. 0, то после такта *T2* МП выполняет вспомогательные такты ожидания готовности внешних устройств к обмену данными — $T_{ож}$, позволяя тем самым общение МП с медленно действующими УВВ. Если сигнал равен лог. 1, то МП переходит к вы-

Продолжение табл. 21.10

Вывод		Функциональное назначение вывода
Англ.	Русск.	
		полнению такта *T3*. Сигнал устанавливается минимум за 180 нс до такта *T2* фронта *F2*
WAIT	ОЖД	Выходной сигнал «Ожидание». МП ожидает готов-ности внешних устройств к обмену данными. Если в такте *T2* в момент действия *F2* сигнал «Готов» будет равен лог. 0, то МП в следующем за тактом *T3* такте $T_{ож}$ в момент действия *F1* переходит в состояние «Ожидание», выставляя лог. 1 на выводе *WAIT*. Находясь в этом режиме, МП по срезу тактовых импульсов *F2* будет постоянно проверять значение сигнала «Готов», пока он не станет равным лог. 1. После чего по фронту очередного тактового импульса *F1* МП выходит из состояния «Ожидание», выставив на вывод *WAIT* лог. 0
HOLD	З.ЗХ	Входной сигнал «Захват». Инициирует перевод шинных формирователей адреса и данных МП в высокоимпедансное состояние Z. Выходные ключи шинных формирователей размыкаются, в результате чего происходит отключение МП от ША и ШД. Это позволяет производить обмен данными между ячейками памяти и портами ввода-вывода, минуя МП (режим прямого доступа в память). Сигнал воспринимается только в том случае, если МП находится в состоянии «Останов» или «Ожидание». Контроль сигнала происходит в тактах *T2* в момент действия фронта импульса *F2* аналогично контролю сигнала *READY*. Сигнал должен быть выставлен за 180 нс до окончания импульса *F2*.
HLDA	П.ЗХ	Выходной сигнал «Подтверждение захвата». Является ответом МП на сигнал «Захват», который подтверждает перевод ША и ШД МП в высокоимпедансное состояние Z. Формирование сигнала происходит с задержкой в 70...120 нс относительно такта *T3* фронта *F1* или в следующем за ним такте (такте *T4* или *T1* следующего цикла) в зависимости от выполнения цикла считывания или записи соответственно, после чего с задержкой в 200 нс линии ШД и ША переводятся в высокоимпедансное состояние Z
INT	З.ПР	Входной сигнал «Запрос прерывания». Используется внешним устройством для запроса обслу-

Окончание табл. 21.10

Вывод		Функциональное назначение вывода
Англ.	Русск.	
		живания у МП. Наличие сигнала анализируется только в конце текущей команды или в состоянии «Останов»
*INTE**	Р.ПР	Выходной сигнал «Разрешение прерывания» (обслуживание прерывания). Указывает, что МП приступил к обслуживанию прерывания
RESET	Сбр. (*R*)	Входной сигнал «Сброс». Устанавливает триггерные схемы в нулевое исходное состояние, при этом ША и ШД находятся в состоянии Z

* Встречается также обозначение сигнала *INTA — INTERUPT ACKNOWLEGES.*

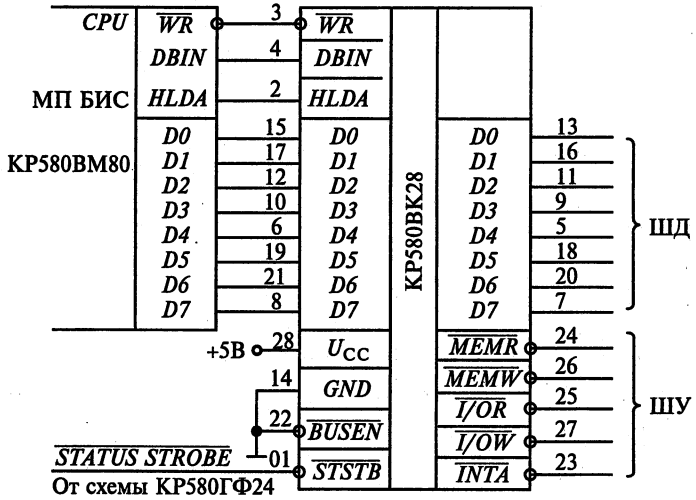


Рис. 21.30

Система прерываний. Система прерываний предназначена для переключения внимания МП на особые условия, возникающие в нем самом или во внешних устройствах, и начала выполнения других задач.

Источники прерывания делятся на три класса.

1. Внутренние прерывания, например, аппаратный сбой, ошибка от схем контроля и т.д. При внутренних прерываниях не происходит формирования сигнала *INTE*. Внутренние прерывания имеют более высокий приоритет по сравнению с внешними прерываниями.

2. Внутрисистемные прерывания, возникающие вследствие событий во внешних устройствах, например, прерывания от тайме-

ра, устройства ввода-вывода в момент его готовности к обмену информацией и т.д.

3. Прерывания, предусмотренные в выполняемой текущей программе с помощью специальных команд.

Существуют два типа обслуживания прерываний: считываемый и векторный.

При *считываемом* типе обслуживания МП использует подпрограмму периодического опроса возможных источников прерывания и выявляет устройства, нуждающиеся в обслуживании. Выявленное устройство выставляет начальный адрес подпрограммы его обслуживания по прерыванию, после чего МП приступает к обработке прерывания.

При *векторном* типе обслуживания прерываний внешние устройства, требующие к себе внимания со стороны МП, выставляют сигнал *INT* (Запрос прерывания). Восприятие запросов устанавливается определенным разрядом ССП при выполнении команды EI (Разрешение прерывания). МП не воспринимает запросы на прерывание при начальной установке по входу *RESET*, когда все триггеры, в том числе и триггеры ССП, устанавливаются в нулевое состояние. Поэтому, если МП приступил к выполнению программы, то он и в дальнейшем не будет реагировать на запросы прерывания до тех пор, пока в программе не появится команда EI (Разрешение прерывания). При дальнейшем продолжении выполнения программы, в случае, если необходимо проигнорировать прерывание, вводят команду DI (Запрещение прерываний). Удовлетворение запроса на прерывание зависит от значения сигнала *D0* ССП. Если *D0* = 1, то при наличии сигнала *DBIN* (прием информации) формируется сигнал *INTE* (Разрешение прерывания) (см. рис. 21.29). Таким образом, изменяя значение разряда *D0*, программист может запретить или разрешить МП реагировать на прерывание (маскирование прерываний). Если МП может воспринимать этот сигнал в данный момент времени, то начинается режим обмена информацией по прерыванию в следующем порядке.

1. МП завершает выполнение текущей программы, сохраняя в стеке адрес следующей подготовленной к выполнению команды, к которой МП вернется после обмена информацией по прерыванию. Обычно в стеке сохраняется и последнее содержимое аккумулятора.

2. МП посылает устройствам ввода-вывода ответный сигнал *INTE* (Разрешение прерывания), давая понять, что он готов к обмену данными.

3. Устройство ввода-вывода сообщает начальный адрес подпрограммы обслуживания прерывания.

4. Получив адрес, МП начинает подпрограмму обмена данными.

5. В конце подпрограммы всегда должна стоять команда выхода из подпрограммы и возврата управления прерванной основной программе по команде RETURN (RET).

6. Получив команду возврата RETURN, МП извлекает из стека адрес возврата к прерванной программе и продолжает ее выполнение.

Если одновременно возникают запросы на прерывание от нескольких устройств, необходимо выработать приоритет по их обслуживанию. Для этой цели вводят специальный блок приоритетных прерываний (БПП), который берет на себя посреднические услуги между устройством ввода-вывода и МП. Для подключения БПП используется вход *INT* МП.

БПП выполняет следующие функции:

получает запросы на прерывание от нескольких источников, если они пришли одновременно и формирует один сигнал, который поступает на вход *INT* МП;

выставляет запрет (маскирование) на восприятие сигналов от определенных источников прерываний;

устанавливает и меняет приоритеты в обслуживании поступивших запросов на прерывание;

сообщает МП начальный адрес подпрограммы обслуживания прерывания внешнего устройства, причем адрес подпрограммы обслуживания прерывания сообщается МП по ШД.

Организация рабочего цикла микропроцессора. Для любого машинного цикла характерны следующие особенности.

1. Выполнение машинного цикла начинается с подачи сигнала *RESET* (Сброс), в результате чего все триггерные схемы устанавливаются в состояние лог. 0.

2. Выработка синхронных импульсов управления в определенные моменты времени относительно тактовых сигналов *F1* и *F2* происходит при их совмещении (выполнении логического умножения) с функциональными сигналами микропроцессора.

3. На ШД выставляется ССП, характеризующее данный цикл, ограниченный временным интервалом от такта *T1* фронта *F2* до такта *T2* фронта *F2* (рис. 21.31, *а*). ССП используется для формирования необходимых управляющих сигналов для внешних устройств и сопровождается сигналом *SYNC*, на базе которого в такте *T2* с использованием тактовых импульсов *F1* формируется сигнал «Импульс записи в РгСС» (рис. 21.31, *б*).

4. На ША в момент такта *T1* фронта *F2* выставляется адрес устройства источника или приемника информации. Адрес находится на ША не менее трех тактов. Если команда выполняется более трех тактов, адрес снимается с ША в следующем машинном цикле в момент такта *T1* фронта *F2* (рис. 21.31, *в*).

5. К моменту фиксации ССП в РгСС по сигналу «Импульс записи в РгСС» ШД освобождается и можно готовиться к обмену информацией по ШД.

Рассмотрим более подробно момент действия *T2* фронта *F2* любого машинного цикла, в который МП анализирует возможность

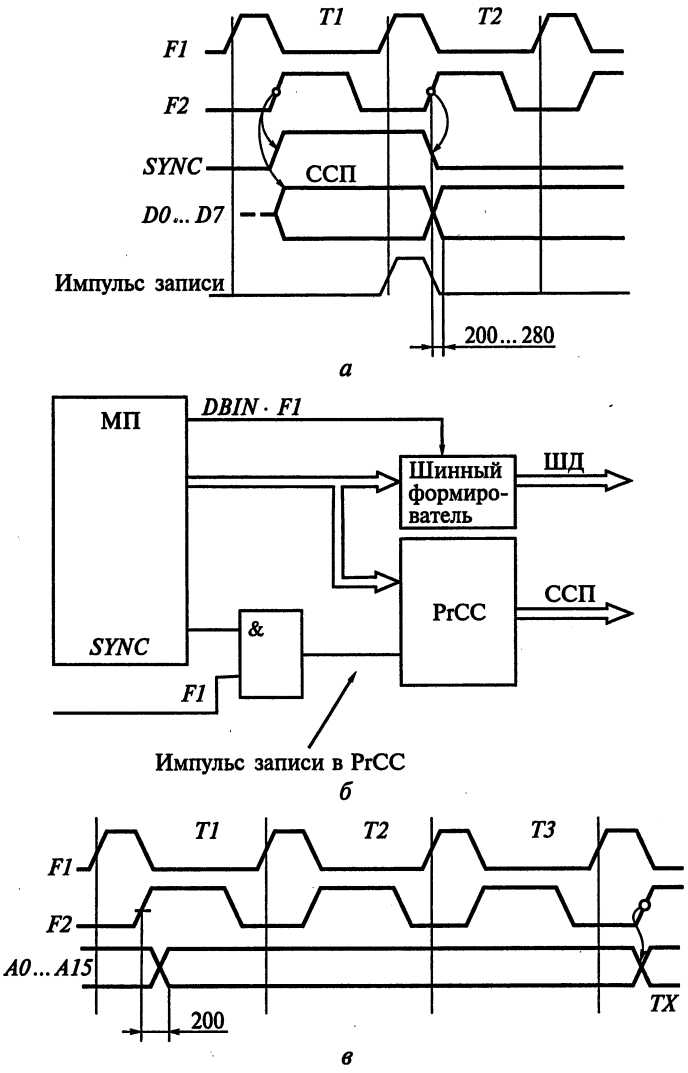


Рис. 21.31

перехода в одно из четырех состояний: «Останов», «Готов», «Ожидание», «Захват».

Состояние «Останов». При выполнении команды HLT (Останов) МП переходит в состояние «Останов» (ССП — «Подтверждение останова»). Находясь в этом режиме, МП приостанавливает свою работу, сохраняя все уровни сигналов на своих выводах. Такое состояние сходно с состоянием «Ожидание», которое будет рассмотрено позже, и отличается от него только тем, что устанавливается

программно. Как правило, введение команды HLT предполагает общение с медленно действующими внешними устройствами. Режим «Останов» длится до тех пор, пока внешние устройства не выставят запрос на прерывание — *INT*. Если такой сигнал не приходит или его восприятие запрещено командой DI, то МП выводится из этого состояния начальной установкой по входу *RESET* или переходит в состояние *HOLD* (Захват) при поступлении на соответствующий вывод лог. 1. Из состояния «Захват» можно вернуться в состояние «Останов» при подаче на вывод *HOLD* лог. 0.

Состояния «Готов» и «Ожидание». Если внешние устройства будут готовы к обмену данными, то в момент такта *T2* фронта *F2* они устанавливают на входе МП *READY* (Готов) сигнал лог. 1.

При передаче информации с внешних устройств в МП в период от такта *T2* фронта *F2* до такта *T3* фронта *F2* на входе *DBIN* (Прием информации) появляется сигнал лог. 1, который используется для формирования управляющих сигналов $\overline{MEMR}$ и $\overline{IOR}$ для внешних устройств (см. рис. 21.28). При наличии этих управляющих сигналов от одного из источников информации, адрес которого установлен на ША, будет произведено чтение данных с выбранного внешнего устройства и размещение этих данных в одном из внутренних регистров МП (рис. 21.32, *а*).

Если внешние устройства не будут готовы к обмену данными, то к моменту появления среза *F2* в такте *T2* на входе МП *READY* (Готов) должен появиться сигнал лог. 0. В этом случае после такта *T2* следуют такты ожидания — $T_{ож}$, т.е. МП прекращает работу, сохраняя на всех выходах имеющиеся уровни сигналов. Такой режим называется режимом «Ожидание». Для подтверждения своего состояния МП в такте *T2* в момент действия фронта *F1* выдает сигнал *WAIT* (Ожидание), равный лог. 1. Находясь в этом режиме, МП по срезу тактовых импульсов *F2* будет постоянно проверять значение сигнала *READY* (Готов) до тех пор, пока внешнее устройство не будет готово к обмену информацией. Далее МП продолжает свою работу, предварительно по фронту очередного тактового импульса *F1* изменив значение сигнала *WAIT* (Ожидание) на лог. 0. На рис. 21.32, *б* показан цикл считывания данных с внешнего устройства с переходом в режим «Ожидание».

При готовности к записи информации внешнее устройство по фронту очередного тактового сигнала *F2* выставляет на входе МП *READY* (Готов) сигнал лог. 1. Цикл «Запись в память УВВ» (рис. 21.33, *а*) обычно выполняется в четвертом или пятом машинном цикле. Тогда перед моментом времени *T3* фронта *F1* МП помещает на ШД данные, а в период от такта *T3* фронта *F1* до такта *T4* фронта *F1* на выводе *WR* (Выдача информации) появляется сигнал, равный лог. 0, с помощью которого внешнему устройству сообщается о готовности к обмену информацией. Сигнал *WR* используется для формирования управляющих сигналов $\overline{MEMW}$ и

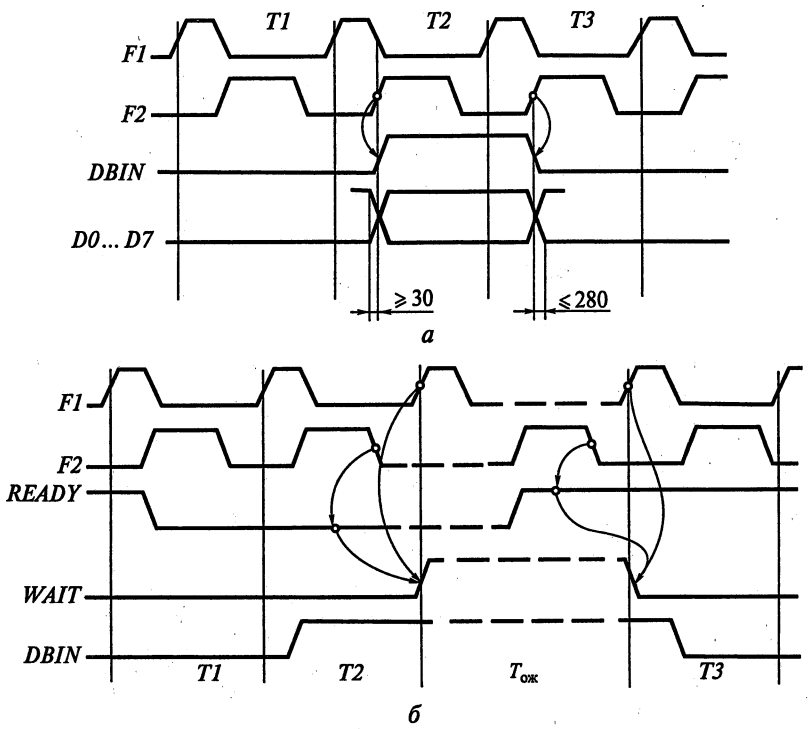


Рис. 21.32

$\overline{IOW}$ для внешних устройств (см. рис. 21.28). При наличии этих сигналов в один из приемников информации, адрес которого установлен на ША, будут переданы данные от МП (см. рис. 21.33, *а*). На рис. 21.33, *б* показан цикл «Запись информации во внешнее устройство» с переходом в режим «Ожидание».

Состояние «Захват» (режим «Прямой доступ в память»). Если внешнее устройство готово к обмену информацией и на входе *READY* (Готов) лог. 1, то необходимо определить, с какими устройствами УВВ собирается взаимодействовать. Режим обмена задается сигналом *HOLD* (Захват), который проверяется в момент действия такта *T2* фронта *F1* (рис. 21.34).

Если на выводе *HOLD* (Захват) лог. 0, то происходит обмен информацией с МП.

Если на выводе *HOLD* (Захват) лог. 1, то происходит обмен информацией с устройствами памяти. Такое состояние возникает, когда устройство ввода-вывода инициирует обмен информацией с памятью без участия МП. Режим работы при этом называется режимом прямого доступа в память (режим ПДП). Так как МП исключен из работы, то все управляющие функции лежат на дополнительном устройстве, которое называется устройством прямого

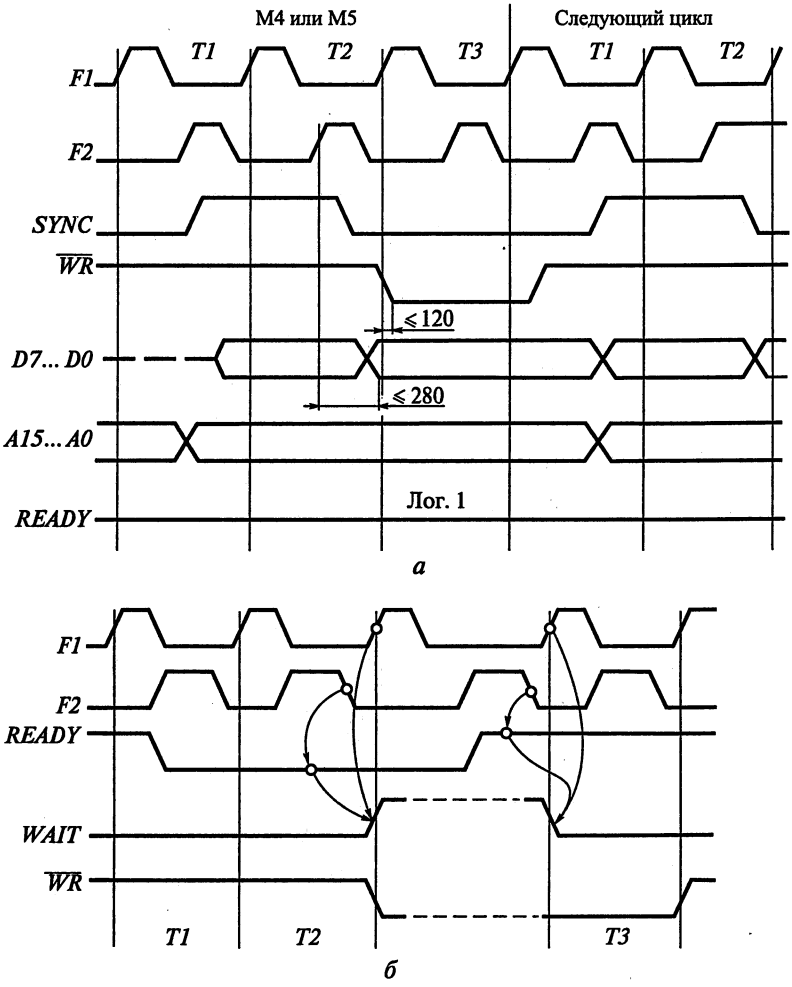


Рис. 21.33

доступа к памяти, а обмен данными — обменом по каналу прямого доступа к памяти микроЭВМ.

Для организации режима ПДП выходные ключи шинных формирователей адреса и данных МП должны быть разомкнуты — высокоимпедансное состояние Z, что и происходит под воздействием сигнала *HOLD* (Захват). После перехода в состояние Z МП выдает сигнал *HLDA* (Подтверждение захвата), равный лог. 1, в момент действия такта *T3* фронта *F1* или в следующем такте *T4* фронта *F1* в зависимости от циклов «Считывание» или «Запись» соответственно. Так как перед выполнением циклов ПДП МП должен закончить работу в текущем машинном цикле, то сигнал *READY* (Готов) име-

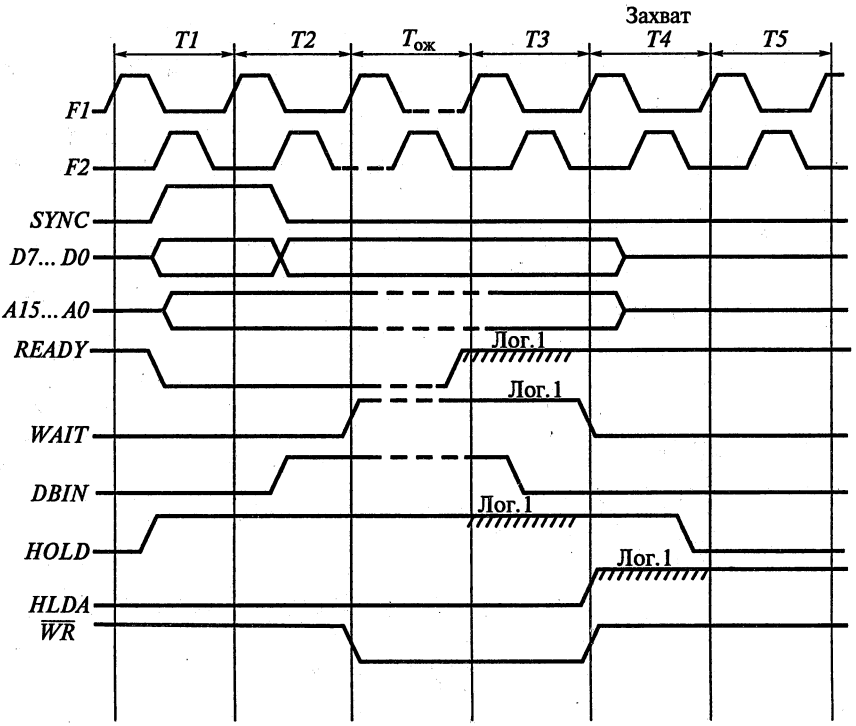


Рис. 21.34

ет более высокий приоритет, чем *HOLD* (Захват). После обмена данными в режиме прямого доступа к памяти значение сигнала «Захват» меняется на лог. 0 (в момент действия среза *F1* любого цикла).

Таким образом, при выполнении любого машинного цикла происходят следующие процессы:

в такте *T1* МП на ША выставляет адрес устройства, с которым необходимо осуществить обмен информацией, а на ШД выводит ССП;

в такте *T2* МП контролирует значение сигналов на выводах *READY* и *HOLD* и наличие ССП «Подтверждение останова», а также формирует адрес для обмена байтом данных, после чего происходит чтение данных с внешних устройств с переходом в такт *T3*;

в такте *T3* происходит обмен байтом информации с памятью или портами ввода-вывода с переходом при записи в такт *T4*;

в такте *T4* производится запись данных;

в такте *T5* выполняются команды внутри МП, т.е. происходят межрегистровая передача и обработка данных в соответствии с командой.

В конце машинного цикла вновь анализируется сигнал *HOLD*.

Для примера рассмотрим процесс выполнения двухбайтовой команды «Ввод данных с устройства ввода». В результате выполнения этой команды вводимые через устройство ввода данные поступают на ШД и далее пересылаются в аккумулятор:

IN [адр. порта ввода] — IN FA,
1-й байт 2-й байт 1-й байт 2-й байт

где FA — адрес порта устройства ввода.

Команда выполняется за три машинных цикла (рис. 21.35). В двух первых циклах происходит считывание двухбайтовой команды и размещение ее в МП, в третьем — выполнение.

Первый машинный цикл (М1 — считывание первого байта команды — кода операции команды).

Такт T1. 1. МП выводит на ШД ССП (см. табл. 21.8), соответствующее выполняемому циклу М1 (от такта *T1* фронта *F2* до такта *T2* фронта *F2*).

2. Для записи ССП в РгСС МП выдает сигнал сопровождения *SYNC* (от такта *T1* фронта *F2* до такта *T2* фронта *F2*), на основе которого в такте *T2* с использованием импульсов тактовой частоты *F1* формируется сигнал «Импульс записи в РгСС».

3. Счетчик команд *PC* через регистр адреса выдает на ША адрес ячейки памяти, в которой находится код операции команды (от такта *T1* фронта *F2* до момента *F2* в такте, следующим за тактом *T3*), после чего схема приращения адреса МП автоматически увеличивает содержимое *PC* на единицу, подготавливая тем самым адрес второго байта команды для последующего считывания его из памяти.

Такт T2. 1. МП проверяет значения сигналов *READ* (Готов), *HOLD* (Захват) и ССП «Подтверждение останова». Если в такте *T2* в момент действия *F2* сигнал *READY* (Готов) равен лог. 0, то в такте *T3* в момент действия *F1* МП переходит в режим «Ожидание», если этот сигнал равен лог. 1, то происходит нормальный цикл чтения информации.

2. ССП записывается в РгСС под воздействием сформированного в такте *T1* сигнала «Импульс записи в РгСС». ССП будет сохраняться в регистре до следующего машинного цикла, а ШД освобождается для новой информации.

3. Появляется сигнал *DBIN* (Прием), действующий в период от такта *T2* фронта *F2* до такта *T3* фронта *F2*. На основе этого импульса и лог. 1 в седьмом разряде ССП происходит формирование управляющего сигнала $\overline{MEMR}$. С помощью этого сигнала из памяти на ШД будет считан код операции команды (см. рис. 21.22), но его запись в регистр команд МП произойдет только в такте *T3*.

Такт T3. В момент среза сигнала *F1* производится запись кода операции команды в регистр команд МП.

Такт T4. В дешифраторе команд расшифровывается код операции, на основании чего определяется количество байтов в коман-

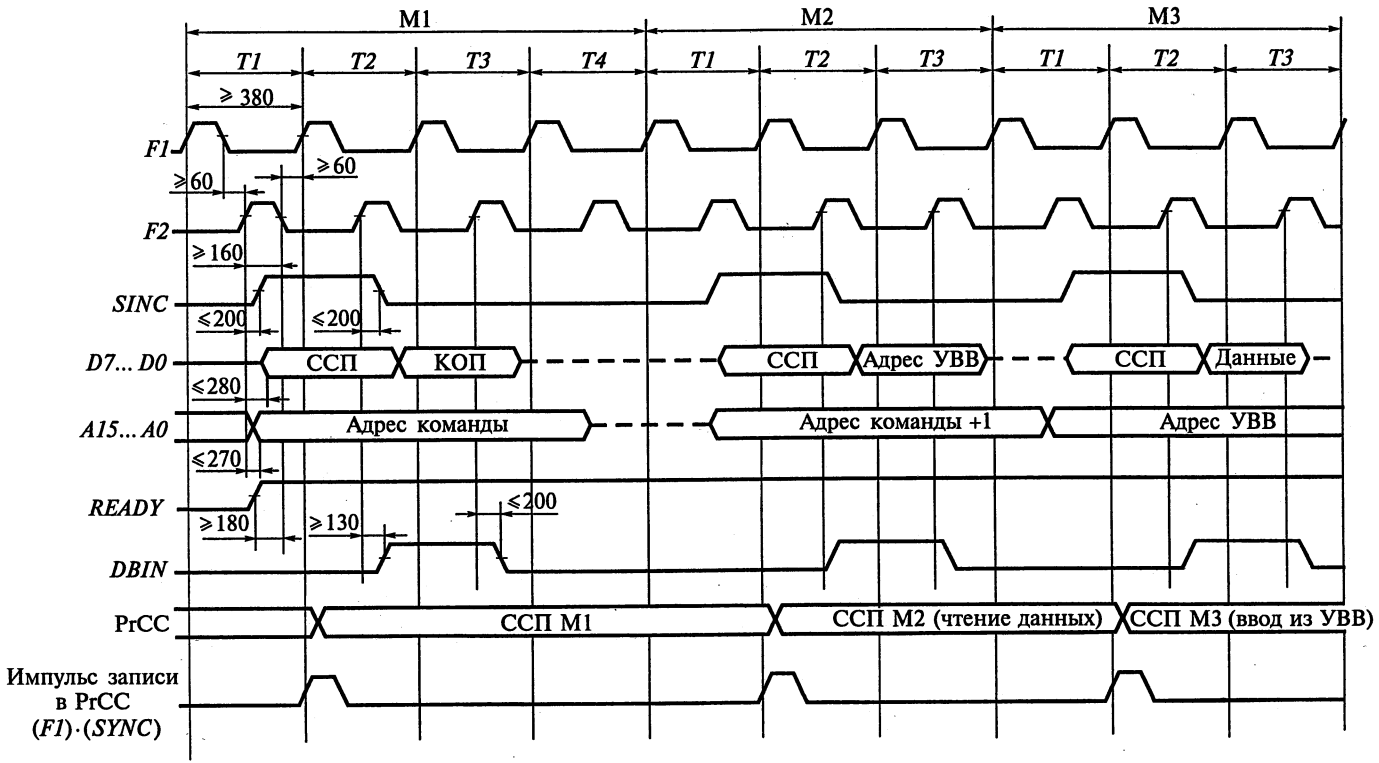

M1
M2
M3
T1
T2
T3
T4
T1
T2
T3
T1
T2
T3
≥ 380
F1
≥60
≥60
F2
≥160
SINC
≤200
≤200
D7... D0
ССП
КОП
ССП
Адрес УВВ
ССП
Данные
≤280
A15... A0
Адрес команды
Адрес команды +1
Адрес УВВ
≤270
READY
≥180
≥130
≤200
DBIN
PrCC
ССП М1
ССП М2 (чтение данных)
ССП М3 (ввод из УВВ)
Импульс записи
в PrCC
(F1)·(SYNC)

Рис. 21.35

де и формируется необходимая последовательность управляющих сигналов.

Второй машинный цикл (считывание второго байта команды).

Такт T1. 1. МП выводит на ШД ССП, соответствующее выполняемому циклу «Считывание данных из памяти» длительностью от такта *T1* фронта *F2* до такта *T2* фронта *F2.*

2. Для записи ССП в РгСС МП выдает сигнал *SYNC* в период от такта *T1* фронта *F2* до такта *T2* фронта *F2.*

3. Счетчик команд *PC* через регистр адреса выдает на ША адрес ячейки памяти, в которой находится второй байт команды длительностью от такта *T1* фронта *F2* до момента *F2* в такте, следующим за тактом *T3*, т.е. до такта *T1* фронта *F2* в третьем машинном такте.

Такт T2. 1. МП проверяет значения сигналов *READ* (Готов), *HOLD* (Захват) и ССП «Подтверждение останова».

2. ССП записывается в РгСС под воздействием сформированного в такте *T1* сигнала «Импульс записи в РгСС». ССП будет сохраняться в регистре до следующего машинного цикла, а ШД освобождается для новой информации.

3. Появляется сигнал *DBIN* (Прием), действующий в период от такта *T2* фронта *F2* до такта *T3* фронта *F2.* На основе этого импульса и лог. 1 в седьмом разряде ССП происходит формирование управляющего сигнала $\overline{MEMR}$. С помощью этого сигнала из памяти на ШД будет считан второй байт команды, содержащий адрес (FA) порта устройства ввода-вывода, но его фиксация в одном из внутренних регистров МП произойдет только в такте *T3.*

Такт T3. В момент среза сигнала *F1* производится запись второго байта команды — адреса порта УВВ в один из регистров временного хранения.

Третий машинный цикл (считывание третьего байта команды).

Такт T1. 1. МП выводит на ШД ССП, соответствующее выполняемому циклу «Ввод из УВВ» в период от такта *T1* фронта *F2* до такта *T3* фронта *F2.*

2. Для записи ССП в РгСС МП выдает сигнал *SYNC* в период от такта *T1* фронта *F2* до такта *T2* фронта *F2.*

3. Из регистра временного хранения производится перезапись адреса УВВ в регистр адреса, который затем поступает на ША в период от такта *T1* фронта *F2* до момента *F2* в такте, следующим за тактом *T3.*

4. Содержимое счетчика адреса *PC* увеличивается на единицу (подготовка адреса следующей команды).

Такт T2. 1. МП проверяет значение сигналов *READ* (Готов), *HOLD* (Захват) и ССП «Подтверждение останова».

2. ССП записывается в РгСС под воздействием сформированного в такте *T1* сигнала «Импульс записи в РгСС». ССП будет сохраняться в регистре до следующего машинного цикла, а ШД освобождается для новой информации.

3. Появляется сигнал *DBIN* (Прием), действующий в период от такта *T2* фронта *F2* до такта *T3* фронта *F2*. На основе этого импульса и лог. 1 в шестом разряде ССП происходит формирование управляющего сигнала $\overline{IOR}$. С помощью этого сигнала из устройства ввода, связанного с портом, адрес которого FA, на ШД будет считан байт данных, но его запись в регистр *A* (аккумулятор) произойдет только в такте *T3*.

Такт T3. Прием данных с ШД в аккумулятор произойдет при наличии сигнала *DBIN* (Прием) в момент действия среза сигнала *F1*.

Контрольные вопросы

1. Как расшифровывается аббревиатура ССП?
2. Какая взаимосвязь между машинным циклом и ССП?
3. Что является источником ССП? Почему первое ССП в первом машинном цикле не зависит от вида выполняемой команды?
4. Какая существует взаимосвязь между ССП и сигналом *SYNC*?
5. Для чего предназначен регистр РгСС и как он связан с ШУ?
6. Какой разряд ССП и сигнал сопровождения необходимы для записи данных в память? Как называется образованный от их сочетания сигнал управления?
7. Какие разряд ССП и сигнал сопровождения необходимы для ввода данных через устройство ввода и как называется образованный от их сочетания сигнал управления?
8. Какова длительность сигнала ССП на ШД?
9. Какова длительность хранения ССП в РгСС?
10. Какой сигнал выставляется внешними устройствами в случае особых условий?
11. Какие условия необходимы для восприятия запроса на прерывание работы МП?
12. Какова последовательность действий при возникновении прерывания?
13. Что такое маскирование прерываний?
14. Какова длительность адреса на ША? Какие процессы происходят в период, ограниченный этим промежутком времени?
15. Как называется сигнал, по которому МП судит о том, что внешнее устройство готово или не готово к обмену данными? Каково его значение и какому моменту времени это соответствует?
16. Какие причины вызывают состояния «Останов» и «Ожидание»? Что между ними общего?
17. В какие моменты времени может происходить ввод данных с внешних устройств в течение машинного цикла? Какие сигналы сопровождения на контактах МП для этого используются?
18. В какие моменты времени может происходить вывод данных во внешнее устройство в течение машинного цикла? Какие сигналы сопровождения для этого используются?
19. Почему одно из состояний называется «Захват»?
20. Что является источником адреса обращения в режиме прямого доступа в память?

21.5. Интерфейсные интегральные схемы

Схемы интерфейса обеспечивают функциональную электрическую совместимость при обмене информацией между МП и устройствами, участвующими в передаче или приеме информации. К ним относятся: шинные формирователи, буферные регистры, параллельные и последовательные адаптеры (интерфейсы), блоки обслуживания прерываний, блоки прямого доступа к памяти, таймеры и др.

Шинные формирователи (ШФ) позволяют осуществить связь между микропроцессором и периферийными устройствами. Они усиливают сигналы по мощности, обеспечивают высокую нагрузочную способность, отключение устройств от шин (состояние Z), формируют требуемые уровни сигналов. На рис. 21.36 представлена схема двунаправленного шинного формирователя КР580ВА86.

ШФ может обеспечивать передачу информации как при вводе, так и при выводе. Режимы его работы (табл. 21.11) зависят от сочетания управляющих сигналов T и $\overline{OE}$. Выходы A и B имеют разную нагрузочную способность. Шина A используется для связи с МП и обеспечивает ток 16 и 1 мА; шина B, связанная с внешними устройствами, обладает высокой нагрузочной способностью и обеспечивает большой ток, равный 32 и 5 мА (лог. 1 и лог. 0 соответственно).

Для ввода и вывода информации используются порты ввода-вывода, хранящие информацию до тех пор, пока она не будет востребована МП или периферийным устройством. Для повышения надежности хранения применяется устройство, называемое защелкой, в качестве которого можно использовать *многорежимный буферный регистр* (МБР). Он представляет собой восьмиразрядный фиксатор сигналов со схемами буферов на входе и может не только выполнять роль буфера, но и хранить передаваемую информацию. В составе МПК серии КР580 имеются МБР КР580ИР82 и КР580ИР83. Схемы имеют три состояния на выходе, из которых первое соответствует прямым значениям сигналов, второе — инверсным, а третье — состоянию Z.

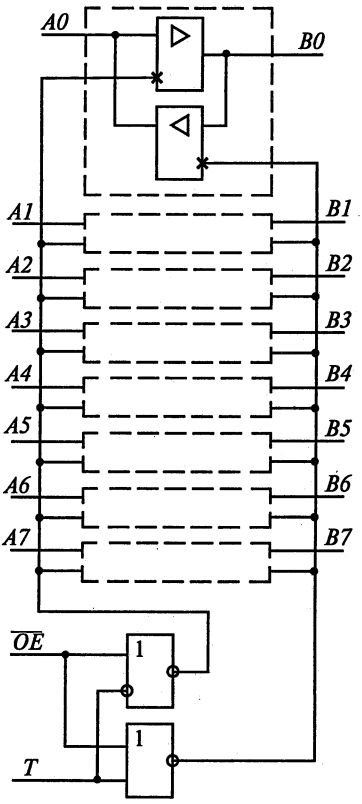


Рис. 21.36

Таблица 21.11

Управляющие входы		Режим работы ШФ
$\overline{OE}$	T	
1	0	Нет передачи
1	1	Нет передачи
0	0	Передача от A к B
0	1	Передача от B к A

На рис. 21.37, *а* представлена схема регистра КР580ИР82, на рис. 21.37, *б* — его условное графическое обозначение, а в табл. 21.12 — назначение выводов. Схема содержит восемь *D*-триггеров, на выходе которых установлены мощные ключи. В зависимости от значения сигнала на входе *STB* схема работает в режиме шинного формирователя или хранения.

Работа схемы в режиме шинного формирователя происходит при сочетании сигналов *STB* на лог. 0 равного лог. 1, и $\overline{OE}$, равного лог. 0. При изменении состояния сигнала *STB* на лог. 0 по его срезу происходит фиксация данных в триггеры регистра, и информация сохраняется там, пока он равен лог. 0. В течение этого времени изменение данных на входах *D* не влияет на состояние триггеров.

Если на входе $\overline{OE}$ сигнал соответствует лог. 1, то все выходы триггеров переходят в третье состояние Z независимо от значений сигналов *STB* и *D*.

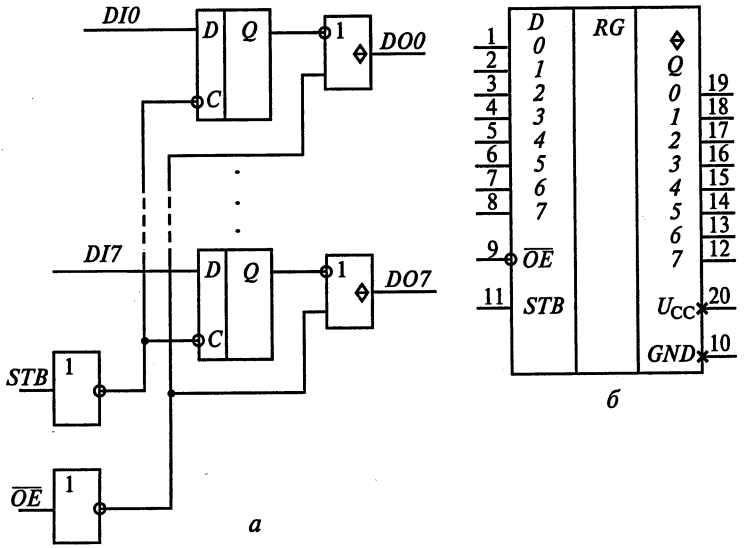


Рис. 21.37

Таблица 21.12

Выводы	Обозначения	Тип	Функциональное назначение
1...8	*D0... D7*	Вход	Информационная шина
9	$\overline{OE}$	Вход	Разрешение передачи (управление третьим состоянием)
10	*GND*	—	Общий
11	*STB*	Вход	Стробирующий сигнал
12...19	*Q7... Q0*	Выход	Информационная шина
20	U_{CC}	—	Напряжение питания +5 В ± 5 %

Подключение периферийных устройств к портам ввода-вывода производится через интерфейс, осуществляя согласование форматов сигналов. Такие микросхемы позволяют программировать их как порты ввода-вывода и обеспечивать более эффективный обмен информацией.

Программируемое устройство ввода-вывода параллельной информации КР580ВВ55А.

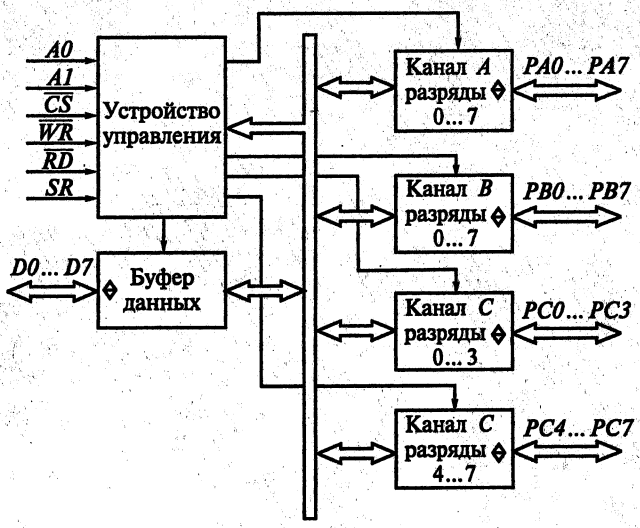
Программируемый параллельный интерфейс (ППИ) является более сложным устройством, которое берет на себя управляющие функции при вводе и выводе информации, освобождая от этой процедуры МП. На рис. 21.38, *а* приведена схема ППИ, на рис. 21.38, *б* — его условное графическое обозначение, а в табл. 21.13 — функциональное назначение выводов микросхемы.

В составе ППИ имеются три канала, снабженных буферными устройствами с тремя состояниями:

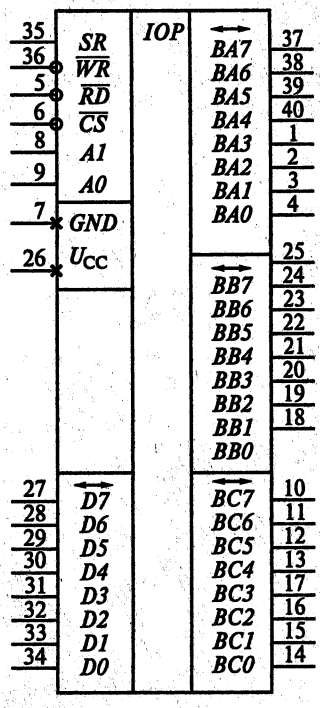
восьмиразрядный регистр (порт) ввода-вывода *A*;

восьмиразрядный регистр (порт) ввода-вывода *B*;

восьмиразрядный регистр *C*, который может функционировать и как единый восьмиразрядный, и как два самостоятельных четырехразрядных регистра *C*(3...0) и *C*(7...4). Причем четырехразрядные регистры могут использоваться одновременно для передачи информации в разных направлениях. Кроме того, триггеры канала *C* могут использоваться индивидуально для хранения специальных управляющих сигналов. Устройство управления ППИ содержит регистр управляющего слова (РУС), который используется для приема управляющей информации МП. Обмен данными между шиной данных и ППИ осуществляется через восьмиразрядный двунаправленный тристабильный канал — буфер данных (*D*). Выбор каналов (включая регистр управляющего слова — РУС) производится с помощью адресных сигналов *A0* и *A1* в соответствии с кодами, представленными в табл. 21.14.


A0
A1
CS
WR
RD
SR
Устройство управления
D0... D7
Буфер данных
Канал A разряды 0...7
PA0... PA7
Канал B разряды 0...7
PB0... PB7
Канал C разряды 0...3
PC0... PC3
Канал C разряды 4...7
PC4... PC7

а


IOP
35 SR
36 WR
5 RD
6 CS
8 A1
9 A0
7 GND
26 UCC
BA7 37
BA6 38
BA5 39
BA4 40
BA3 1
BA2 2
BA1 3
BA0 4
25 BB7
24 BB6
23 BB5
22 BB4
21 BB3
20 BB2
19 BB1
18 BB0
27 D7
28 D6
29 D5
30 D4
31 D3
32 D2
33 D1
34 D0
BC7 10
BC6 11
BC5 12
BC4 13
BC3 17
BC2 16
BC1 15
BC0 14

б

Рис. 21.38

Таблица 21.13

Выводы	Обозначения	Тип	Функциональное назначение
1...4, 37...40	*BA3*...*BA0*, *BA7*...*BA4*	Входы/выходы	Информационный канал *A*
5	$\overline{RD}$	Вход	Чтение информации
6	$\overline{CS}$	Вход	Выбор микросхемы
7	*GND*	—	Общий
8...9	*A1*, *A0*	Вход	Младшие разряды адреса
10...17	*BC7*...*BC4*, *BC0*...*BC3*	Входы/выходы	Информационный канал *C*
18...25	*BB0*...*BB7*	Входы/выходы	Информационный канал *B*
26	U_{CC}	—	Напряжение питания +5 В ± 5 %
27...34	*D7*...*D0*	Входы/выходы	Канал данных
35	*SR*	Вход	Установка в исходное состояние
36		Вход	Запись информации

Предварительно в РУС заносится информация, определяющая режим работы каналов, на основании чего устройство управления формирует необходимые сигналы для организации обмена информацией между регистрами каналов и периферийными устройствами ввода-вывода.

Каналы микросхемы могут работать в трех режимах, которые задаются определенными разрядами управляющего слова канала (УСК) (рис. 21.39).

Произведя запись УСК, можно перевести ППИ в один из трех режимов.

Режим 0 (простой ввод-вывод без стробирования). Используется для ввода и вывода медленно изменяющихся данных. Программно управляемая передача данных происходит через два независимых восьмиразрядных канала *A*, *B* и два независимых четырехразрядных канала *C*. В этом режиме выводимая информация запоминается в выходном регистре выбранного канала и хранится там до перезагрузки. Сигналов управления обменом информацией с периферийным устройством здесь не требуется.

Таблица 21.14

A1	*A0*	Канал
0	0	*A*
0	1	*B*
1	0	*C*
1	1	РУС

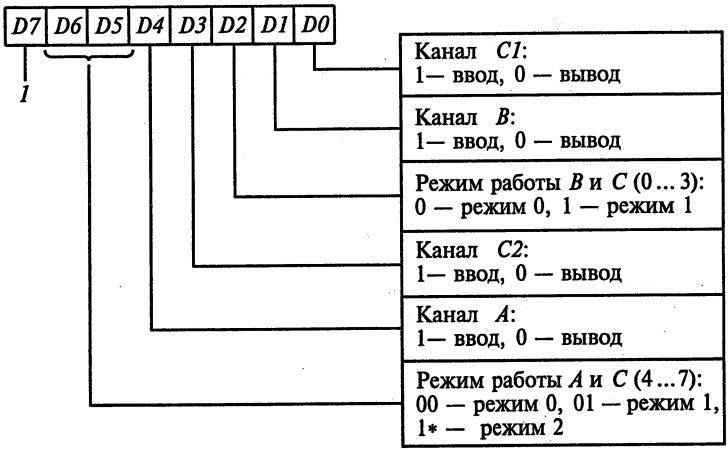


Рис. 21.39

Для записи УСК в ППИ используется режим 0 — вывод.

Режим 1. Организация ввода-вывода через каналы *A* и *B* по сигналам квитирования). Ввод или вывод информации осуществляется через два независимых восьмиразрядных канала *A* и *B* с использованием канала *C* для приема и выдачи необходимых сигналов управления (или квитирования — рукопожатия) при налаживании диалога между устройством ввода-вывода и периферийными устройствами.

Режим 2. Организация ввода-вывода информации через двунаправленный восьмиразрядный канал *A* с использованием канала *C* для формирования управляющих сигналов квитирования.

Режимы работы каналов можно менять как в начале, так и в процессе выполнения программы, что позволяет обслуживать различные периферийные устройства в определенном порядке с помощью одной микросхемы ППИ.

Направление передачи информации по каналам зависит от значения сигналов «Чтение» $\left(\overline{RD}\right)$ и «Запись» $\left(\overline{WR}\right)$ и задается определенными сигналами в соответствии с табл. 21.15.

При подаче сигнала $\overline{CS}$ РУС устанавливается в состояние, при котором все каналы настраиваются на работу в режиме 0.

Таблица 21.15

$\overline{CS}$	$\overline{RD}$	$\overline{WR}$	Операция ввода-вывода
0	0	1	Ввод (считывание)
0	1	0	Вывод (запись)
1	*	*	Канал данных — состояние Z

Обращение к регистрам выбранного канала производится по командам IN (ввод в МП) и OUT (вывод из МП).

Приведем пример.

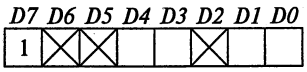


Рис. 21.40

Программирование ППИ

MVI A, 81 — занесение УСК (код УСК-81) в аккумулятор МП;

OUT, FB — вывод содержимого аккумулятора в РУС ППИ. (FB — адрес РУС)

Ввод-вывод данных

IN FA — считывание данных из порта *A* (адрес FA);

OUT, F9 — запись данных в порт *B* (адрес F9).

В режиме 0 происходит синхронная программно управляемая передача данных через два независимых канала *A* и *B* и два че-

Таблица 21.16

УСК				Направление передачи информации			
				Канал			
D4	*D3*	*D1*	*D0*	*A*	*B*	*C*	
						(7...4)	(3...0)
0	0	0	0	Вывод	Вывод	Вывод	Вывод
0	0	0	1	Вывод	Вывод	Вывод	Ввод
0	0	1	0	Вывод	Ввод	Вывод	Вывод
0	0	1	1	Вывод	Ввод	Вывод	Ввод
0	1	0	0	Вывод	Вывод	Ввод	Вывод
0	1	0	1	Вывод	Вывод	Ввод	Ввод
0	1	1	0	Вывод	Ввод	Ввод	Вывод
0	1	1	1	Вывод	Ввод	Ввод	Ввод
1	0	0	0	Ввод	Вывод	Вывод	Вывод
1	0	0	1	Ввод	Вывод	Вывод	Ввод
1	0	1	0	Ввод	Ввод	Вывод	Вывод
1	0	1	1	Ввод	Ввод	Вывод	Ввод
1	1	0	0	Ввод	Вывод	Ввод	Вывод
1	1	0	1	Ввод	Вывод	Ввод	Ввод
1	1	1	0	Ввод	Ввод	Ввод	Вывод
1	1	1	1	Ввод	Ввод	Ввод	Ввод

тырехразрядных канала *C*. На рис. 21. 40 представлен формат УСК для данного режима, а в табл. 21.16 показаны УСК для 16 различных комбинаций ввода-вывода работы каналов *A*, *B*, *C*.

На рис. 21.41 показаны временные диаграммы ввода-вывода информации.

Режим 1 сложнее режима 0. В этом режиме возможна пересылка информации через канал *A* или канал *B* по стробирующим сигналам. Линии канала *C* используются для приема и выдачи сигналов управления, служащих в качестве запросов прерывания работы МП.

При ***вводе*** информации устройство управления занимает по три линии канала *C* для управления каналами *A* и *B* соответственно. По этим линиям передаются сигналы квитирования (управления передачей информации между ПУ и ППИ), а также сигнал

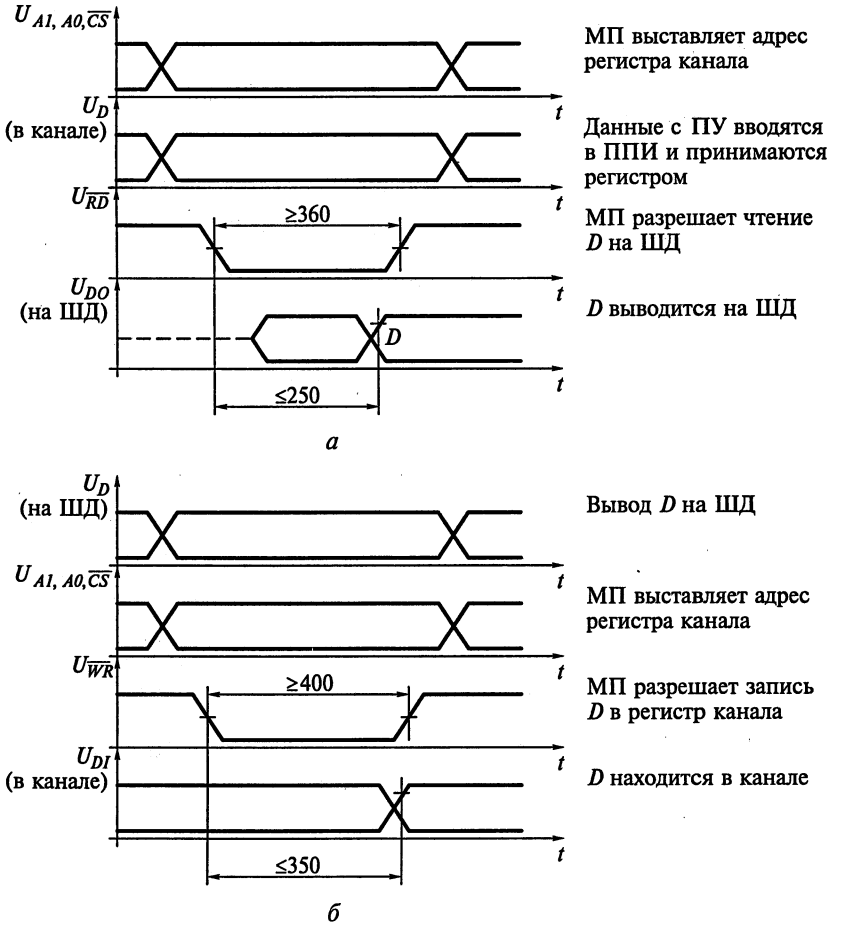


Рис. 21.41

Таблица 21.17

Разряд канала *C*	Обозначение сигнала		Функциональное назначение сигнала
	Русск.	Англ.	
C4 *C2*	Строб приема (СТР. П)	$\overline{STB}\ \overline{RD_A}$ $\overline{STB}\ \overline{RD_B}$	Нулевой сигнал стробирования сопровождает передачу данных с периферийного устройства в регистр канала *A* или *B*
C5 *C1*	Подтверждение приема (ПТВ. П)	$\overline{ASK}\ \overline{RD_A}$ $\overline{ASK}\ \overline{RD_B}$	Наличие единицы на этих контактах говорит о том, что запись данных во входной регистр канала произошла и поступление новых данных с ПУ заблокировано
C3 *C0*	Запрос прерывания (З. ПР)	IRQ_A (INT) IRQ_B	Наличие единицы используется для прерывания работы МП, тем самым ППИ сообщает, что данные подготовлены к передаче в МП. В ответ МП выдает сигнал $\overline{RD}$ (считывание) данных с ППИ
C6 *C7*	—	—	Могут использоваться как для ввода, так и для вывода данных

«Запрос прерывания» в МП. В соответствии с табл. 21.17 на одной из разрядных линий канала *C* выставляется сигнал, который несет определенную смысловую нагрузку. Если ПУ выставляет на линии выбранного канала *A* или *B* данные, то информирует об этом ППИ сигналом квитирования $\overline{STB}\ \overline{RD}$ (Строб приема), равным лог. 0, который поступает на *C4* или *C2*. С поступлением этого сигнала данные записываются в входной регистр соответствующего канала. В свою очередь ППИ посылает ПУ сигнал квитирования $\overline{ASK}\ \overline{RD}$ (Подтверждение приема), равный лог. 1 и говорящий о том, что данные каналом ППИ приняты. На время действия этого сигнала, т.е. до освобождения канала, ПУ запрещает ввод новых данных. Далее устройство управления ППИ передает сигнал «Запрос прерывания» *IRQ* (INT), равный лог. 1, в МП, указывая на то, что один из каналов ППИ готов к передаче в данных МП. Этот сигнал может быть равен лог. 1, только если сигналы $\overline{STB}$ и $\overline{ASK}$ равны лог. 1, т.е. когда данные находятся в канале. В ответ МП выдает сигнал «Считывание» ($\overline{DBIN}$) или $\overline{RD}$ ($\overline{IOR}$), равный лог. 0, по которому данные из канала поступают на ШД. Началу этого сигнала соответствует сигнал *IRQ*, а окончанию — $\overline{ASK}$.

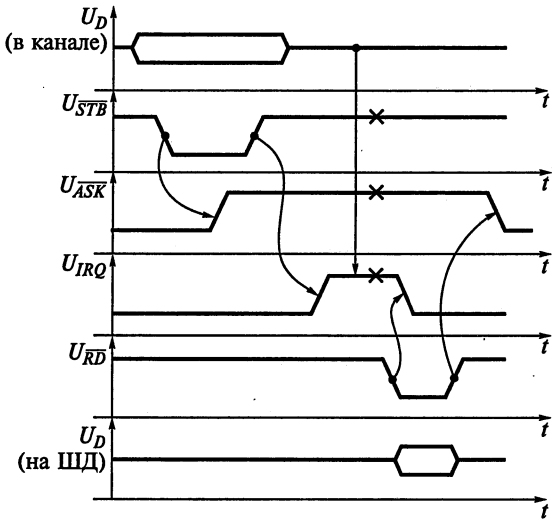


Рис. 21.42

На рис. 21.42 представлена временна́я диаграмма ввода данных в режиме 1.

При ***выводе*** информации разрядные линии канала *C* используются для передачи сигналов квитирования, смысловая нагрузка

Таблица 21.18

Разряд канала *C*	Обозначение сигнала		Функциональное назначение сигнала
	Русск.	Англ.	
C7 *C1*	Строб записи (СТР. З)	$\overline{STB}\ \overline{WR_A}$ $\overline{STB}\ \overline{WR_B}$	Единичный сигнал стробирования сопровождает ввод данных по сигналу $\overline{WR}$ (Запись) в канал. Нулевой сигнал стробирования говорит о том, что произведена запись данных в канал
C6 *C2*	Подтверждение приема (ПТВ. З)	$\overline{ASK}\ \overline{WR_A}$ $\overline{ASK}\ \overline{WR_B}$	Высокий уровень сигнала говорит о том, что ПУ принимает данные. Низкий уровень соответствует времени, когда ПУ уже приняло данные
C3 *C0*	Запрос прерывания (З. ПР)	IRQ_A IRQ_B	Наличие единицы говорит о связи МП с ППИ
C5 *C4*	—	—	Могут использоваться как для ввода, так и для вывода данных

которых показана в табл. 21.18. МП выставляет на ША адрес одного из каналов ППИ, а на ШД — выводимые данные, после чего

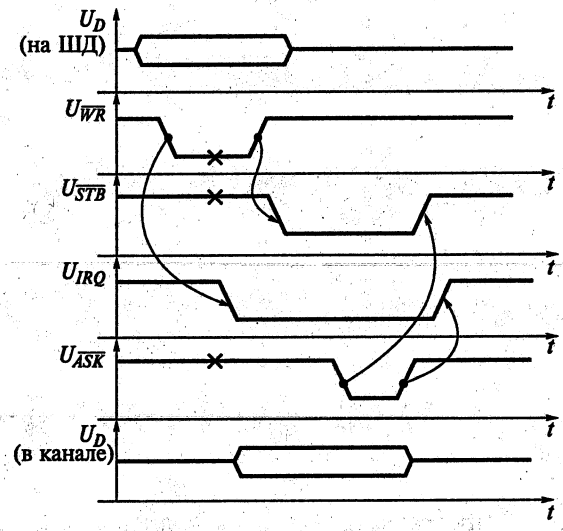


Рис. 21.43

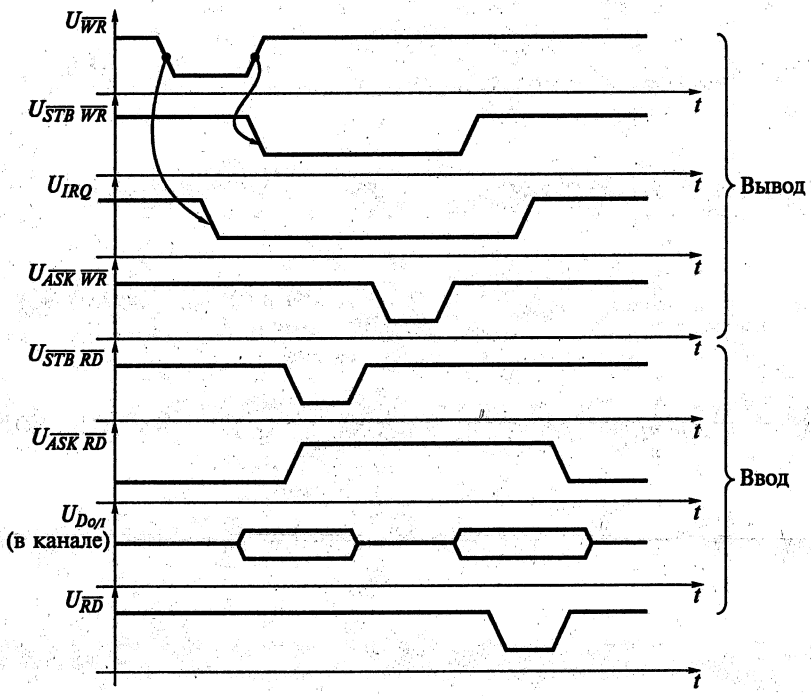


Рис. 21.44

вырабатывает сигнал «Запись» $\overline{WR}$ ($\overline{IOW}$). По этому сигналу при наличии сигналов $\overline{STB}$ и $\overline{ASK}$, равных лог. 1, происходит запись данных из МП в выходной регистр канала. По окончании записи данных (когда сигнал $\overline{WR}$ становится равен лог. 1) сигнал $\overline{STB}$ становится равным лог. 0, что свидетельствует о том, что МП произвел запись данных в выходной регистр канала, а появление сигнала $\overline{ASK}$, равного лог. 0, о том, что ПУ приняло данные, записанные в выходной регистр канала.

Значение сигнала «Запрос прерывания» *IRQ* (*INT*) зависит от сигналов $\overline{STB}$, $\overline{ASK}$ и $\overline{WR}$. В момент действия $\overline{WR}$ и при $\overline{STB}$, $\overline{ASK}$, равных лог. 0, сигнал *IRQ* также равен лог. 0.

На рис. 21.43 представлена временна́я диаграмма вывода данных в режиме 1.

В режиме 2 организуется передача информации в двух направлениях через канал *A*, т.е. информация может и вводиться, и вы-

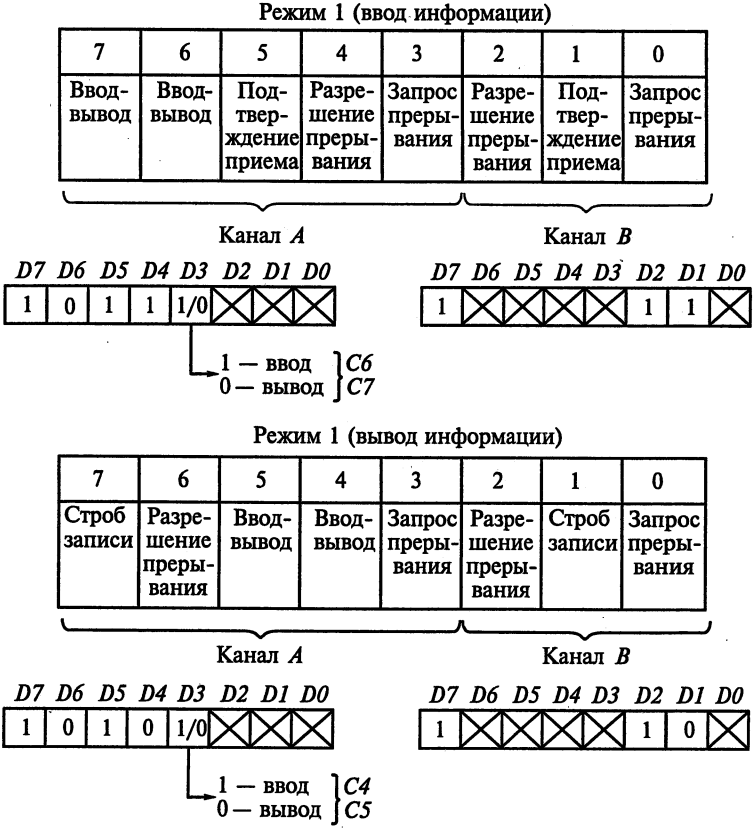


Рис. 21.45

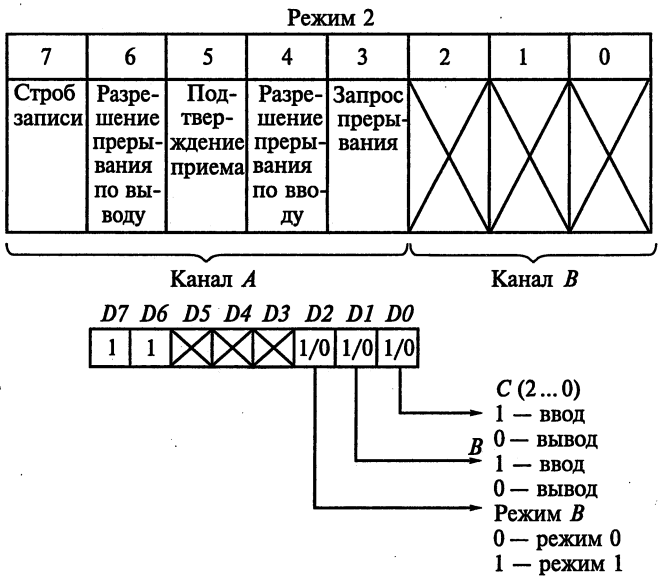


Рис. 21.46

водиться, в то время как в режиме 1 для изменения направления передачи информации необходима запись нового УСК. Режим 2 существует лишь для канала *A*, так как он имеет два регистра для записи данных, что необходимо при двунаправленной связи между ПУ и МП.

В режиме 2 устройство управления использует пять линий канала *C* для сигналов управления передачей информации так же, как и в режиме 1, оставшиеся линии канала *C* могут быть использованы для ввода и вывода информации в зависимости от УСК. Следует отметить, что в то время как канал *A* работает в режиме 2 или других режимах, канал *B* может работать как в режиме 0, так и в режиме 1. На рис. 21.44 представлена диаграмма обмена сигналами канала *A* при выводе и вводе информации.

На рис. 21.45 и 21.46 представлены форматы УСК соответственно для режимов 1 и 2.

Программируемое четырехканальное устройство прямого доступа к памяти (ПДП) КР580ВТ57. Эта микросхема предназначена для организации по требованию периферийного устройства высокоскоростного обмена данными между памятью системы и периферийными устройствами, минуя центральный процессор. Структурная схема устройства представлена на рис. 21.47; в табл. 21.19 приведено назначение выводов микросхемы.

Микросхема КР580ВТ57 может обслуживать до четырех периферийных устройств, запрашивающих режим ПДП, подачей на схему приема запросов асинхронных сигналов *DRQ0... DRQ3.* В пер-

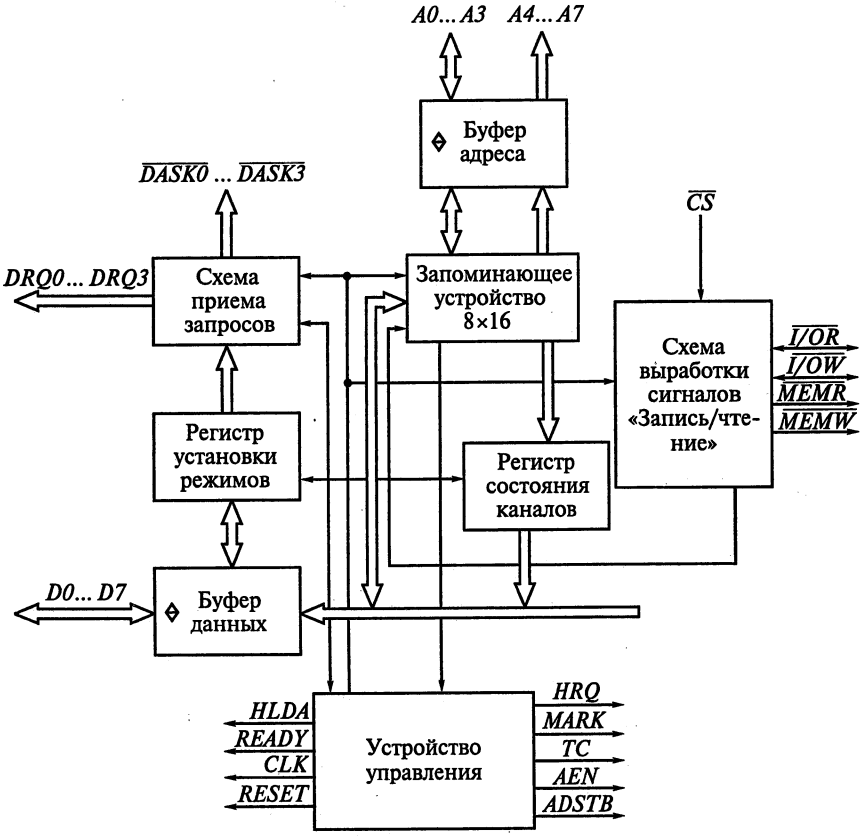


Рис. 21.47

вую очередь обслуживается устройство с наивысшим приоритетом, который устанавливается программно. При поступлении запроса ПДП формирует сигнал «Запрос захвата» *HRQ* (*HOLD*), запрашивающий процессор о возможности организации ПДП. При поступлении от МП сигнала разрешения *HLDA* схема приема запросов формирует один из сигналов $\overline{DASK0}...\overline{DASK3}$, подтверждающий запрашивающему периферийному устройству возможность ПДП.

Для хранения начального адреса и числа циклов ПДП используются два шестнадцатиразрядных регистра на каждый канал, которые образуют ЗУ емкостью 8 × 16 бит. Оба регистра должны быть загружены перед началом работы. В регистр адреса записывается адрес первой ячейки памяти, к которой должно быть обращение. Значения, записываемые в младшие 14 разрядов регистра числа циклов, указывают число ПДП минус 1 до появления сигнала *TC* (Конец счета) на выходе микросхемы. В двух старших разрядах ре-

Таблица 21.19

Выводы	Обозначения	Функциональное назначение
32...35, 37...40	*A0...A7*	Шина адреса
21...23, 25...30	*D0...D7*	Шина данных
1	*I/OR*	Считывание ввода-вывода
2	*I/OW*	Запись ввода-вывода
3	*MEMR*	Считывание памяти
4	*MEMW*	Запись в память
5	*MARK*	Маркер
6	*READY*	Готовность
7	*HLDA*	Подтверждение захвата
8	*ADSTB*	Строб адреса
9	*AEN*	Разрешение адреса
10	*HRQ*	Запрос захвата
11	*CS*	Выбор микросхемы
12	*CLK*	Тактовый сигнал
13	*RESET*	Установка
25, 24, 14, 15	*DASK0... DASK3*	Подтверждение ПДП
16...19	*DRQ0... DRQ3*	Запрос ПДП
20	*GND*	Общий
31	U_{CC}	+5 В
36	TC	Конец счета

гистра числа циклов указывается режим обмена данными: проверка, запись, чтение.

Устройство управления формирует необходимые управляющие сигналы для работы микросхемы в различных режимах. Схема выработки сигналов «Запись/чтение» обеспечивает прием, формирование и выдачу сигналов, осуществляющих обмен информацией между МП и схемой ПДП, между памятью и периферийными устройствами.

Записывая в восьмиразрядный регистр установки режимов определенную информацию, можно задать следующие режимы работы схемы ПДП: автозагрузка (повторение ранее выбранного массива адресов), конец счета, удлиненная (обычная) запись, форсированный (циклический) приоритет. Запись в регистр установки режима осуществляется в режиме программирования после загрузки регистра адреса и регистра числа циклов при установке схем в исходное состояние, в момент подачи сигнала *RESET* (Сброс).

Регистр состояния каналов указывает, какой канал достиг условия конца счета. Буфер адреса предназначен для приема и выдачи кода адреса. Двунаправленные адресные шины *A0...A3* в режиме программирования являются входами, выбирающими один из регистров, информация с которого должна быть считана или, на-

оборот, записана. В режиме обслуживания они являются младшими разрядами шестнадцатиразрядного адреса памяти. Выходы *A4...A7*, имеющие состояние Z, предназначены для выдачи 4...7 разрядов шестнадцатиразрядного адреса памяти.

Буфер данных состоит из восьми двунаправленных схем с тремя состояниями и предназначен для сопряжения устройства ПДП с шиной данных МП. При программировании загрузка регистра адреса, регистра числа циклов или регистра установки режимов осуществляется из МП через шину данных за два командных цикла.

При чтении содержимого регистра адреса, регистра числа циклов или регистра состояния каналов данные передаются в МП также через шину данных. В течение циклов ПДП микросхема КР580ВТ57 управляет системной шиной и выдает через буфер данных старшие восемь разрядов адреса памяти одного из регистров адреса ЗУ. Эти разряды адреса передаются в начале каждого цикла ПДП, а затем шина данных освобождается для обмена данными между памятью и периферийными устройствами в течение оставшейся части цикла ПДП.

Программируемый контроллер прерываний КР580ВН59. Эта микросхема предназначена для обслуживания до восьми запросов на прерывание МП, поступающих от внешних устройств. Предусмотрена возможность расширения числа обслуживаемых запросов до 64 путем каскадного соединения микросхем КР580ВН59.

Контроллер прерываний путем предварительной программной установки в определенный режим работы обеспечивает два способа обслуживания прерывания программ: по результатам опроса и по запросу.

В режиме обслуживания по результатам опроса производится опрос каждого периферийного устройства до тех пор, пока не будет обнаружено устройство, запрашивающее прерывание. Далее происходит переход в соответствующую программу обслуживания прерывания, которая выполняет обмен данными. В этом режиме приоритет устройства определяется его местом в последовательности опроса. Режим целесообразен в тех случаях, когда для ряда периферийных устройств имеется общая подпрограмма обслуживания или число запросов составляет более 64.

В режиме обслуживания по запросу обмен данными между ЦП и периферийными устройствами инициируется периферийными устройствами выработкой сигнала *IR* «Запрос прерывания», при поступлении которого управление передается соответствующей программе. Данный способ обмена особенно удобен при работе с периферийными устройствами, имеющими низкое быстродействие, а также в ситуациях, когда момент передачи непредсказуем, что в большинстве случаев и бывает. Алгоритм задания приоритета устанавливается программным путем. Приоритеты, закрепленные за

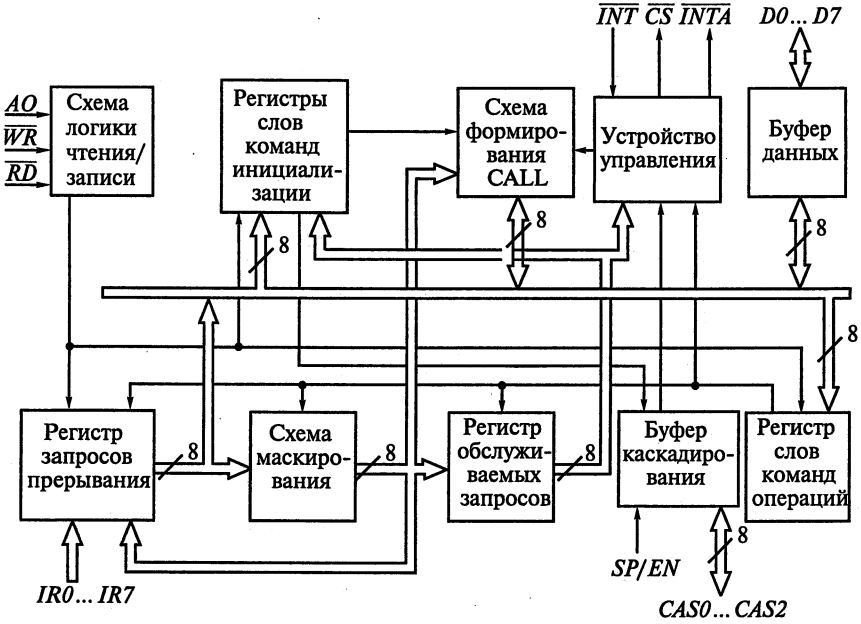


Рис. 21.48

внешними устройствами, могут быть изменены в процессе выполнения программ.

Структурная схема контроллера КР580ВН59 представлена на рис. 21.48; в табл. 21.20 приведено функциональное назначение выводов микросхемы.

Таблица 21.20

Выводы	Обозначения	Функциональное назначение
27	*A0*	Адрес
4...11	*D0... D7*	Шина данных
1	*CS*	Выбор микросхемы
2	*WR*	Считывание
3	*RD*	Запись
12, 13, 15	*CAS0... CAS2*	Каскадирование
14	*GND*	Общий
16	*SP/EN*	Признак подчинения
17	*INT*	Прерывание
18...25	*IR0... IR7*	Запрос прерывания
26	*INTA*	Подтверждение прерывания
28	U_{CC}	+5 В

Восьмиразрядный регистр запросов прерывания предназначен для записи и хранения сигналов прерывания *IR0... IR7*, поступающих от внешних устройств асинхронно, и которые поэтому должны удерживаться на входе до поступления с МП первого сигнала $\overline{INTA}$ «Подтверждение прерывания». Сброс разряда регистра запросов прерывания, соответствующего обслуживаемому запросу, осуществляется вторым сигналом $\overline{INTA}$, поступающим с МП.

Схема маскирования состоит из схемы анализа приоритета и регистра маски, в который предварительно записывается код маски прерываний, разрешающий или запрещающий прохождение определенных запросов на схему анализа приоритета. Сигналы, прошедшие через регистр маски, анализируются по приоритету. Запрос с наивысшим приоритетом записывается в регистр обслуживаемых запросов, запрещая прохождение через схему маскирования новых запросов, имеющих равный обслуживаемому запросу приоритет или более низкие. Схема логики чтения/записи позволяет записывать команды в различные регистры микросхемы, а также считывать содержимое регистров на шину данных.

Установка схемы в исходное состояние и установка фиксированного статуса уровней приоритета осуществляются командами инициализации, которые предварительно записываются и хранятся в регистрах слов команды инициализации. В зависимости от числа микросхем, примененных в системе, используется последовательность из двух или трех команд инициализации. Схема формирования команды CALL выдает на шину данных трехбайтовую команду CALL при поступлении с МП трех сигналов $\overline{INTA}$: при поступлении сигнала $\overline{INTA1}$ выдается первый байт (команда CALL), по сигналу $\overline{INTA2}$ — второй (младший байт адреса), по сигналу $\overline{INTA3}$ — третий (старший байт адреса).

Устройство управления формирует сигнал *INT* (Прерывание) при поступлении одного или нескольких запросов и вырабатывает управляющие сигналы, необходимые для выполнения команды CALL. Буфер каскадирования позволяет увеличить число обслуживаемых запросов до 64, при этом одна из микросхем КР580ВН59 включается как ведущая, а остальные — как ведомые. Каждой ведомой микросхеме присваивается номер, предварительно устанавливаемый программным путем.

Регистры слов команд операций предназначены для задания программным путем различных операций как в режиме обслуживания по запросу, так и в режиме обслуживания по результатам опроса: маскирование запросов прерывания (специальное маскирование); фиксированный приоритет (циклический сдвиг приоритета); обычный конец прерывания (специальный конец прерывания); установка чтения регистра запросов; установка чтения регистра обслуживаемых запросов.

Двунаправленный восьмиразрядный буфер данных имеет на выходе состояние Z. В режиме программирования через буфер данных в КР580ВН59 записываются содержимое регистра запросов, регистра обслуживаемых запросов, регистра маски и двоично-десятичный код запроса, по которому вырабатывается сигнал *INT*. В режиме прерывания по запросу после подтверждения прерывания через буфер данных в системную шину выдается трехбайтовая команда CALL. В остальное время выход буфера данных находится в состоянии Z.

В табл. 21.21 представлен состав микропроцессорного комплекта КР580.

Таблица 21.21

Тип микросхемы	Наименование микросхемы	Выполняемая функция
КР580ВМ80А	Восьмиразрядный параллельный центральный процессор	Центральный процессор с фиксированной системой команд для обработки параллельной восьмиразрядной информации
КР580ВВ51А	Программируемый последовательный интерфейс	Универсальное синхронно-асинхронное приемно-передающее устройство последовательной связи
КР580ВИ53	Программируемый таймер	Формирует программно-управляемые временны́е задержки для синхронизации управляемых объектов в реальном масштабе времени
КР580ВВ55А	Программируемый параллельный интерфейс	Программируемый ввод-вывод параллельной информации различного формата
КР580ВТ57	Программируемый контроллер прямого доступа к памяти	Высокоскоростной обмен информацией между памятью МП и периферийными устройствами
КР580ВН59	Программируемый контроллер прерываний	Обслуживает до восьми запросов на прерывания от внешних устройств
КР580ГФ24	Генератор тактовых импульсов	Формирует две последовательности тактовых импульсов, необходимые для работы центрального процессора

Окончание табл. 21.21

Тип микросхемы	Наименование микросхемы	Выполняемая функция
КР580ВК28 КР580ВК38	Системный контроллер	Формирует сигналы, предназначенные для управления различными устройствами, входящими в МП
КР580ВА86 КР580ВА87	Шинный формирователь	Двунаправленный восьмиразрядный шинный формирователь с высокой нагрузочной способностью и тремя состояниями
КР580ИР82 КР580ИР83	Буферный регистр	Восьмиразрядный буферный регистр с тремя состояниями
КР580ВГ75	Программируемый интерфейс электронно-лучевой трубки	Контроллер вывода информации из памяти МП на экран электронно-лучевой трубки
КР580ВВ79	Программируемый интерфейс клавиатуры и дисплея	Контроллер ввода-вывода для клавиатуры и дисплея
КР580ВК91А	Интерфейс микропроцессор — канал общего пользования	Устройство сопряжения микропроцессора с информационно-измерительной системой
КР580ВА93	Приемопередатчик микропроцессор — канал общего пользования	Программируемый приемопередатчик

Микропроцессорный комплект КР580 ориентирован на использование в микропроцессорных системах с относительно невысоким быстродействием (до 2,5 МГц). Комплект имеет фиксированную разрядность (восемь разрядов) и систему команд. Его микросхемы КР580ГФ24, КР580ВК38, КР580ИР82 (83), КР580ВА86 (87) выполнены по биполярной технологии ТТЛШ, остальные — по *n*-МОП-технологии.

Контрольные вопросы

1. Где на структурной схеме микроЭВМ (см. рис. 21.4) находятся шинные формирователи, МБР и ППИ?
2. С помощью каких сигналов устанавливается направление передачи информации в шинных формирователях?
3. Может ли МБР использоваться в качестве шинного формирователя?

4. В чем заключается особенность управления передачей информации в ППИ?

5. Каким образом задается направление передачи информации в ППИ?

6. Какие сигналы сопровождают данные при вводе и выводе информации с ПУ?

7. Какие сигналы подтверждают процесс ввода и вывода данных?

8. Каково назначение сигнала *IRQ*?

9. Какова взаимосвязь между разрядами УСК и разрядами канала *C*?

СИСТЕМА КОМАНД ДЛЯ КР580

Команды пересылки данных

Команды этой группы осуществляют пересылку данных в/из регистр(а) и ячейки памяти и не изменяют состояния флагов.

MOV r1, r2 (пересылка данных из регистра в регистр).

(r1) ← (r2)

Содержимое регистра r2 пересылается в регистр r1.

0	1	D	D	D	S	S	S

MOV A, A	7F	MOV B, A	47
MOV A, B	78	MOV B, B	40
MOV A, C	79	MOV B, C	41
MOV A, D	7A	MOV B, D	42
MOV A, E	7B	MOV B, E	43
MOV A, H	7C	MOV B, H	44
MOV A, L	7D	MOV B, L	45
MOV C, A	4F	MOV D, A	57
MOV C, B	48	MOV D, B	50
MOV C, C	49	MOV D, C	51
MOV C, D	4A	MOV D, D	52
MOV C, E	4B	MOV D, E	53
MOV C, H	4C	MOV D, H	54
MOV C, L	4D	MOV D, L	55
MOV E, A	5F	MOV H, A	67
MOV E, B	58	MOV H, B	60
MOV E, C	59	MOV H, C	61
MOV E, D	5A	MOV H, D	62
MOV E, E	5B	MOV H, E	63
MOV E, H	5C	MOV H, H	64
MOV E, L	5D	MOV H, L	65
MOV L, A	6F		
MOV L, B	68		
MOV L, C	69		
MOV L, D	6A		
MOV L, E	6B		
MOV L, H	6C		
MOV L, L	6D		

Циклы: 1.
Состояния: 4.
Адресация: регистровая.
Флаги: не используются.

MOV r, M (пересылка данных из памяти).

(r) ← (*H*, *L*)

Содержимое ячейки памяти, адрес которой хранится в регистрах *H* и *L*, пересылается в регистр r.

0	1	D	D	D	1	1	0

MOV A, M 7E
MOV B, M 46
MOV C, M 4E
MOV D, M 56
MOV E, M 5E
MOV H, M 66
MOV L, M 6E

Циклы: 2.
Состояния: 7.
Адресация: косвенная регистровая.
Флаги: нет.

MOV M, r (пересылка данных в память).

(*H*, *L*) ← (r)

Содержимое регистра r пересылается в ячейку памяти, адрес которой содержится в регистрах *H* и *L*.

0	1	1	1	0	S	S	S

MOV M, A 77
MOV M, B 70
MOV M, C 71
MOV M, D 72
MOV M, E 73
MOV M, H 74
MOV M, L 75

Циклы: 2.
Состояния: 7.
Адресация: косвенная регистровая.
Флаги: нет.

MVI, r байт (непосредственная пересылка).

(r) ← (байт 2)

Содержимое второго байта команды пересылается в регистр *r*.

0	0	D	D	D	1	1	0
Данные							

MVI A, байт 3E
MVI B, байт 06
MVI C, байт 0E
MVI D, байт 16
MVI E, байт 1E
MVI H, байт 26
MVI L, байт 2E

Циклы: 2.
Состояния: 7.
Адресация: непосредственная.
Флаги: нет.

MVI M, байт (непосредственная пересылка в память).

(*H*, *L*) ← (байт 2)

Содержимое второго байта команды пересылается в ячейку памяти, адрес которой указан в регистрах *H* и *L*.

0	0	1	1	0	1	1	0
Данные							

MVI M, байт 36.

Циклы: 3.
Состояния: 10.
Адресация: непосредственная/косвенная регистровая.
Флаги: нет.

LXI гр, 2 байта (непосредственная загрузка пары регистров).

(rh) ← (байт 3)
(rl) ← (байт 2)

Третий байт команды пересылается в старший регистр (rh) пары регистров, второй — в младший регистр (rl) пары регистров.

0	0	R	P	0	0	0	1
Младший байт данных							
Старший байт данных							

LXI B, 2 байта 01 (непосредственная загрузка пары регистров *B* и *C*).
LXI D, 2 байта 11 (непосредственная загрузка данных пары регистров *D* и *E*).

LXI H, 2 байта 21 (непосредственная загрузка пары регистров *H* и *L*).
LXI SP, 2 байта 31 (непосредственная загрузка указателя стека).

Циклы: 3.
Состояния: 10.
Адресация: непосредственная.
Флаги: нет.

LDA [адр.] (прямая загрузка аккумулятора).

(*A*) ← (байт 3, байт 2)

Содержимое ячейки памяти, адрес которой указан во втором и третьем байтах команды, загружается в аккумулятор.

0	0	1	1	1	0	1	0
Младший байт адреса							
Старший байт адреса							

LDA [адр.] 3A.

Циклы: 4.
Состояния: 13.
Адресация: прямая.
Флаги: нет.

STA [адр.] (прямая запись содержимого аккумулятора в память).

(байт 3, байт 2) ← (*A*)

Содержимое аккумулятора пересылается в ячейку памяти, адрес которой указан во втором и третьем байтах команды.

0	0	1	1	0	0	1	0
Младший байт адреса							
Старший байт адреса							

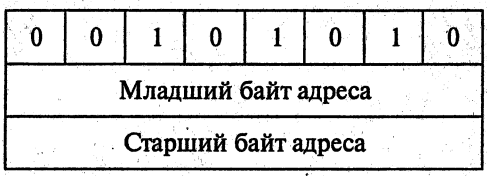
STA [адр.] 32.

Циклы: 4.
Состояния: 13.
Адресация: прямая.
Флаги: нет.

LHLD [адр.] (прямая загрузка *H* и *L*).

(*L*) ← (байт 3, байт 2)
(*H*) ← (байт 3, байт 2 + 1)

Содержимое ячейки памяти, адрес которой указан во втором и третьем байтах команды, пересылается в регистр *L*. Содержимое ячейки памяти со следующим адресом (адр. + 1) пересылается в регистр *H*.



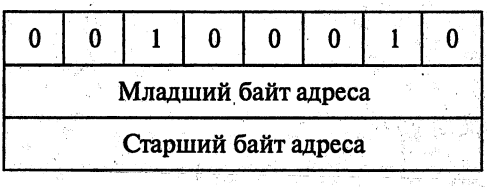
LHLD [адр.] 2A.

Циклы: 5.
Состояния: 16.
Адресация: прямая.
Флаги: нет.

SHLD [адр.] (прямая запись содержимого регистров *H*, *L* в память).

(байт 3, байт 2) ← (*L*)
(байт 3, байт 2 + 1) ← (*H*)

Содержимое регистра *L* пересылается в ячейку памяти, адрес которой определен во втором и третьем байтах команды. Содержимое регистра *H* пересылается в следующую ячейку памяти.



SHLD [адр.] 22.

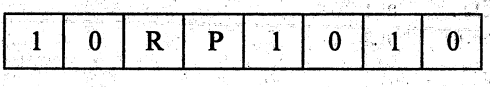
Циклы: 5.
Состояния: 16.
Адресация: прямая.
Флаги: нет.

LDAX гр (косвенная загрузка аккумулятора).

(*A*) ← (гр)

Содержимое ячейки памяти, адрес которой указан в паре регистров гр, пересылается в регистр *A*.

Примечание. Могут быть указаны только пары регистров гр = *B* (регистры *B* и *C*) и гр = *D* (регистры *D* и *E*).



LDAX B 0A.
LDAX D 1A.

Циклы: 2.
Состояния: 7.
Адресация: косвенная регистровая.
Флаги: нет.

STAX гр (косвенная запись содержимого аккумулятора в память).

(гр) ← (*A*)

Содержимое аккумулятора пересылается в ячейку памяти, адрес которой определен в паре регистров гр.

Примечание. Могут быть указаны только пары регистров гр = *B* (регистры *B* и *C*) или гр = *D* (регистры *D* и *E*).

0	0	R	P	0	0	1	0

STAX B 02.
STAX D 12.

Циклы: 2.
Состояния: 7.
Адресация: косвенная регистровая.
Флаги: нет.

XCHG (обмен данными между регистрами *H*, *L* и *D*, *E*).

(*H*) ↔ (*D*)
(*L*) ↔ (*E*)

Содержимое регистров *H* и *L* обменивается с содержимым регистров *D* и *E*.

1	1	1	0	1	0	1	1

XCHG EB.

Циклы: 1.
Состояния: 4.
Адресация: регистровая.
Флаги: нет.

Команды арифметических операций

Команды этой группы выполняют арифметические операции над данными в регистрах и ячейках памяти.

Если нет особых указаний, то команды изменяют состояния флагов нуля, четности, переноса и вспомогательного переноса в соответствии с стандартными правилами.

Все операции вычитания выполняются через арифметическое дополнение до 2, причем флаг переноса устанавливается в лог. 1, если имеется заем, и сбрасывается, если заема не было.

ADD r (сложение содержимого аккумулятора с содержимым регистра).

$(A) \leftarrow (A) + (r)$

Содержимое регистра r складывается с содержимым аккумулятора. Результат помещается в аккумулятор.

1	0	0	0	0	S	S	S

ADD A 87
ADD B 80
ADD C 81
ADD D 82
ADD E 83
ADD H 84
ADD L 85

Циклы: 1.
Состояния: 4.
Адресация: регистровая.
Флаги: Z, S, P, CY, AC.

ADD M (сложение содержимого аккумулятора с содержимым ячейки памяти).

$(A) \leftarrow (A) + (H, L)$

Содержимое ячейки памяти, адрес которой содержится в регистрах *H* и *L*, складывается с содержимым аккумулятора. Результат помещается в аккумулятор.

1	0	0	0	0	1	1	0

ADD M 86.

Циклы: 2.
Состояния: 7.
Адресация: косвенная регистровая.
Флаги: Z, S, P, CY, AC.

ADI байт (непосредственное сложение)

$(A) \leftarrow (A) + (\text{байт 2})$

Содержимое второго байта команды складывается с содержимым аккумулятора. Результат помещается в аккумулятор.

1	1	0	0	0	1	1	0
Данные							

ADI байт C6.

Циклы: 2.
Состояния: 7.
Адресация: непосредственная.
Флаги: Z, S, P, CY, AC.

ADC r (сложение содержимого аккумулятора с содержимым регистра и битом флага переноса).

(*A*) ← (*A*) + (r) + (CY)

Содержимое регистра r и бит флага переноса складываются с содержимым аккумулятора. Результат помещается в аккумулятор.

1	0	0	0	1	S	S	S

ADC A 8F
ADC B 88
ADC C 89
ADC D 8A
ADC E 8B
ADC H 8C
ADC L 8D

Циклы: 1.
Состояния: 4.
Адресация: регистровая.
Флаги: Z, S, P, CY, AC.

ADC M (сложение содержимого аккумулятора с содержимым ячейки памяти и битом переноса).

(*A*) ← (*A*) + (*H*, *L*) + (CY)

Содержимое ячейки памяти, адрес которой указан в регистрах *H* и *L*, и содержимое бита флага переноса складываются с содержимым аккумулятора. Результат помещается в аккумулятор.

1	0	0	0	1	1	1	0

ADC M 8E

Циклы: 2.
Состояния: 7.
Адресация: косвенная регистровая.
Флаги: Z, S, P, CY, AC.

ACI байт (непосредственное сложение с битом переноса).

(*A*) ← (*A*) + (байт 2) + (CY)

Содержимое второго байта команды и бита флага переноса складывается с содержимым аккумулятора. Результат помещается в аккумулятор.

1	1	0	0	1	1	1	0
Данные							

ACI байт CE.

Циклы: 2.
Состояния: 7.
Адресация: непосредственная.
Флаги: Z, S, P, CY, AC.

SUB r (вычитание содержимого регистра).

$(A) \leftarrow (A) - (r)$

Содержимое регистра r вычитается из содержимого аккумулятора. Результат помещается в аккумулятор.

1	0	0	1	0	S	S	S

SUB A 97
SUB B 90
SUB C 91
SUB D 92
SUB E 93
SUB H 94
SUB L 95

Циклы: 1.
Состояния: 4.
Адресация: регистровая.
Флаги: Z, S, P, CY, AC.

SUB M (вычитание содержимого ячейки памяти).

$(A) \leftarrow (A) - (H, L)$

Содержимое ячейки памяти, адрес которой содержится в регистрах *H* и *L*, вычитается из содержимого аккумулятора. Результат помещается в аккумулятор.

1	0	0	1	0	1	1	0

SUB M 96

Циклы: 2.
Состояния: 7.
Адресация: косвенная регистровая.
Флаги: Z, S, P, CY, AC.

SUI байт (непосредственное вычитание).

$(A) \leftarrow (A) -$ (байт 2)

Содержимое второго байта команды вычитается из содержимого аккумулятора. Результат помещается в аккумулятор.



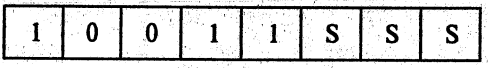
SUI байт D6.

Циклы: 2.
Состояния: 7.
Адресация: непосредственная.
Флаги: Z, S, P, CY, AC.

SBB r (вычитание содержимого регистра и бита переноса).

$(A) \leftarrow (A) - (r) - (CY)$

Содержимое регистра r и бит флага переноса CY вычитается из содержимого аккумулятора. Результат помещается в аккумулятор.



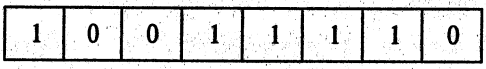
SBB A 9F
SBB B 98
SBB C 99
SBB D 9A
SBB E 9B
SBB H 9C
SBB L 9D

Циклы: 1.
Состояния: 4.
Адресация: регистровая.
Флаги: Z, S, P, CY, AC.

SBB M (вычитание содержимого ячейки памяти и бита переноса).

$(A) \leftarrow (A) - (H, L) - (CY)$

Содержимое ячейки памяти, адрес которой содержится в регистрах *H* и *L*, и содержимое бита флага переноса CY вычитаются из содержимого аккумулятора. Результат помещается в аккумулятор.



SBB M 9E.

Циклы: 2.
Состояния: 7.
Адресация: косвенная регистровая.
Флаги: Z, S, P, CY, AC.

SBI байт (непосредственное вычитание с заемом).

$(A) \leftarrow (A) - (\text{байт } 2) - (CY)$

Содержимое второго байта команды и бита флага переноса CY вычитается из аккумулятора. Результат помещается в аккумулятор.

1	1	0	1	1	1	1	0
Данные							

SBI байт DE.

Циклы: 2.
Состояния: 7.
Адресация: непосредственная.
Флаги: Z, S, P, CY, AC.

INR r (увеличение содержимого регистра).

$(r) \leftarrow (r) + 1$

Содержимое регистра r увеличивается на единицу.

Примечание. Команда влияет на состояние всех флагов, кроме флага переноса CY.

0	0	D	D	D	1	0	0

INR A 3C
INR B 04
INR C 0C
INR D 14
INR E 1C
INR H 24
INR L 26

Циклы: 1.
Состояния: 4.
Адресация: регистровая.
Флаги: Z, S, P, AC.

INR M (увеличение содержимого ячейки памяти).

$(H, L) \leftarrow (H, L) + 1$

Содержимое ячейки памяти, адрес которой содержится в регистрах *H* и *L*, увеличивается на единицу.

Примечание. Команда воздействует на все флаги, кроме флага переноса CY.

0	0	1	1	0	1	0	0

INR M 34.

Циклы: 3.
Состояния: 10.
Адресация: косвенная регистровая.
Флаги: Z, S, P, AC.

DCR r (уменьшение содержимого регистра).

(r) ← (r) – 1

Содержимое регистра r уменьшается на единицу.

Примечание. Команда воздействует на все флаги, кроме флага переноса CY.

0	0	D	D	D	1	0	1

DCR A 3D
DCR B 05
DCR C 0D
DCR D 15
DCR E 1D
DCR H 25
DCR L 2D

Циклы: 1.
Состояния: 4.
Адресация: регистровая.
Флаги: Z, S, P, AC.

DCR M (уменьшение содержимого ячейки памяти).

(*H*, *L*) ← (*H*, *L*) – 1

Содержимое ячейки памяти, адрес которой содержится в регистрах *H* и *L*, уменьшается на единицу.

Примечание. Команда воздействует на все флаги, кроме флага переноса CY.

0	0	1	1	0	1	0	1

DCR M 35.

Циклы: 3.
Состояния: 10.
Адресация: косвенная регистровая.
Флаги: Z, S, P, AC.

INX rp (увеличение содержимого пары регистров).

(rh)(rl) ← (rh)(rl) + 1

Содержимое пары регистров rp увеличивается на единицу.

Примечание. Команда не воздействует на флаги.

0	0	R	P	0	0	1	1

INX B 03 (пара регистров *B* и *C*)
INX D 13 (пара регистров *D* и *E*)
INX H 23 (пара регистров *H* и *L*)
INX SP 33 (указатель стека).

Циклы: 1.
Состояния: 6.
Адресация: регистровая.
Флаги: нет.

DCX rp (уменьшение содержимого пары регистров).

(rh)(rl) ← (rh)(rl) – 1

Содержимое пары регистров rp уменьшается на единицу.

Примечание. Команда не воздействует на флаги.

0	0	R	P	1	0	1	1

DCX B 0B (пара регистров *B* и *C*)
DCX D 1B (пара регистров *D* и *E*)
DCX H 2B (пара регистров *H* и *L*)
DCX SP 3B (указатель стека)

Циклы: 1.
Состояния: 6.
Адресация: регистровая.
Флаги: нет.

DAD rp (сложение содержимого пары регистров с содержимым регистров *H* и *L*).

(*H*)(*L*) ← (*H*)(*L*) + (rh)(rl)

Содержимое пары регистров rp складывается с содержимым регистров *H* и *L*. Результат помещается в пару регистров *H*, *L*.

Примечание. Используется только флаг переноса CY. Он устанавливается, если имеется перенос при сложении с двойной точностью, в противном случае он сбрасывается.

0	0	R	P	1	0	0	1

DAD B 09 (пара регистров *B* и *C*)
DAD D 19 (пара регистров *D* и *E*)
DAD H 29 (пара регистров *H* и *L*)
DAD SP 39 (указатель стека).

Циклы: 3.
Состояния: 10.
Адресация: регистровая.
Флаги: CY.

DAA (десятичное дополнение аккумулятора).

Восьмибитное число в аккумуляторе дополняется до представления в виде двух четырехбитных чисел в двоично-десятичном коде с помощью следующих операций.

1. Если число, представленное четырьмя младшими битами аккумулятора, больше 8 или установлен флаг вспомогательного переноса AC, то к содержимому аккумулятора добавляется 6.

2. Если теперь число, представленное четырьмя старшими битами аккумулятора, больше 9 или установлен флаг переноса CY, то число 6 добавляется к числу, образованному четырьмя старшими битами аккумулятора.

Примечание. Используются все флаги.

0	0	1	0	0	1	1	1

DAA 27.

Циклы: 1.
Состояния: 4.
Флаги: Z, S, P, CY, AC.

Команды логических операций

Команды этой группы выполняют логические (булевы) операции над данными, хранящимися в регистрах и ячейках памяти и флагами.

Если нет особых указаний, то все команды воздействуют на флаги: нуля, четности, знака, переноса и вспомогательного переноса в соответствии со стандартными правилами.

ANA r (операция «Логическое И» над содержимым регистра и аккумулятора).

$(A) \leftarrow (A) \wedge (r)$

Над содержимым регистра r и аккумулятора выполняется операция «Логическое И». Результат помещается в аккумулятор. Сбрасывается флаг переноса CY, устанавливается флаг вспомогательного переноса AC.

1	0	1	0	0	S	S	S

ANA A A7
ANA B A0
ANA C A1
ANA D A2
ANA E A3
ANA H A4
ANA L A5

Циклы: 1.
Состояния: 4.
Адресация: регистровая.

Флаги: Z, S, CY, AC, P.

ANA M (операция «Логическое И» над содержимым ячейки памяти и аккумулятора).

$(A) \leftarrow (A) \wedge (H, L)$

Над содержимым ячейки памяти, адрес которой содержится в регистрах *H* и *L*, и содержимым аккумулятора выполняется операция «Логическое И» . Результат помещается в аккумулятор. Сбрасывается флаг переноса CY и вспомогательного переноса AC.

1	0	1	0	0	1	1	0

ANA M A6.

Циклы: 2.
Состояния: 7.
Адресация: косвенная регистровая.
Флаги: Z, S, P, AC, CY.

ANI байт (непосредственная операция «И»)

$(A) \leftarrow (A) \wedge$ (байт 2)

Над содержимым второго байта команды и аккумулятора выполняется операция «Логическое И». Результат помещается в аккумулятор. Сбрасываются флаги переноса CY и вспомогательного переноса AC.

1	1	1	0	0	1	1	0
Данные							

ANI байт E6.

Циклы: 2.
Состояния: 7.
Адресация: непосредственная.
Флаги: Z, S, P, CY, AC.

XRA r (операция «Исключающее ИЛИ» над содержимым аккумулятора и регистра).

$(A) \leftarrow (A) \forall$ (r)

Над содержимым регистра r и аккумулятора выполняется операция «Исключающее ИЛИ». Результат помещается в аккумулятор. Сбрасываются флаги переноса CY и вспомогательного переноса AC.

1	0	1	0	1	S	S	S

XRA A AF
XRA B A8

XRA C A9
XRA D AA
XRA E AB
XRA H AC
XRA L AD

Циклы: 1.
Состояния: 4.
Адресация: регистровая.
Флаги: Z, S, P, CY, AC.

XRA M (операция «Исключающее ИЛИ» над содержимым ячейки памяти и аккумулятора).

$(A) \leftarrow (A) \vee (H, L)$

Над содержимым ячейки памяти, адрес которой содержится в регистрах *H* и *L*, и содержимым аккумулятора выполняется операция «Исключающее ИЛИ». Результат помещается в аккумулятор. Сбрасываются флаги переноса CY и вспомогательного переноса AC.

1	0	1	0	1	1	1	0

XRA M AE.

Циклы: 2.
Состояния: 7.
Адресация: косвенная регистровая.
Флаги: Z, S, P, CY, AC.

XRI байт (непосредственная операция «Исключающее ИЛИ»).

$(A) \leftarrow (A) \vee$ (байт 2)

Над содержимым аккумулятора и второго байта команды выполняется операция «Исключающее ИЛИ». Результат помещается в аккумулятор. Сбрасываются флаги переноса CY и вспомогательного переноса AC.

1	1	1	0	1	1	1	0
Данные							

XRI байт EE.

Циклы: 2.
Состояния: 7.
Адресация: непосредственная.
Флаги: Z, S, P, CY, AC.

ORA r (операция «Логическое ИЛИ» содержимого аккумулятора с содержимым регистра).

$(A) \leftarrow (A) \vee (r)$

Содержимое регистра r логически складывается с содержимым аккумулятора. Результат помещается в аккумулятор. Сбрасываются флаги переноса CY и вспомогательного переноса AC.

1	0	1	1	0	S	S	S

ORA A B7
ORA B B0
ORA C B1
ORA D B2
ORA E B3
ORA H B4
ORA L B5

Циклы: 1.
Состояния: 4.
Адресация: регистровая.
Флаги: Z, S, P, CY, AC.

ORA M (операция «Логическое ИЛИ» содержимого ячейки памяти с содержимым аккумулятора).

$(A) \leftarrow (A) \vee (H, L)$

Содержимое ячейки памяти, адрес которой определен в регистрах *H* и *L*, логически складывается с содержимым аккумулятора. Результат помещается в аккумулятор. Сбрасываются флаги переноса CY и вспомогательного переноса AC.

1	0	1	1	0	1	1	0

ORA M B6.

Циклы: 2.
Состояния: 7.
Адресация: косвенная регистровая.
Флаги: Z, S, P, CY, AC.

ORI байт (непосредственная операция «Логическое ИЛИ»).

$(A) \leftarrow (A) \vee$ (байт 2)

Содержимое второго байта команды складывается непосредственно с содержимым аккумулятора. Результат помещается в аккумулятор. Сбрасываются флаги переноса CY и вспомогательного переноса AC.

1	1	1	1	0	1	1	0
Данные							

ORI байт F6.

Циклы: 2.
Состояния: 7.
Адресация: непосредственная.
Флаги: Z, S, P, CY, AC.

CMP r (сравнение содержимого регистра с содержимым аккумулятора).

(*A*) – (r)

Содержимое регистра r вычитается из содержимого аккумулятора. При этом содержимое аккумулятора не изменяется. По результату вычитания флаги устанавливаются следующим образом: флаг нуля Z — в 1, если (*A*) = (r); флаг переноса CY — в 1, если (*A*) < (r).

1	0	1	1	1	S	S	S

CMP A BE
CMP B B8
CMP C B9
CMP D BA
CMP E BB
CMP H BC
CMP L BD

Циклы: 1.
Состояния: 4.
Адресация: регистровая.
Флаги: Z, S, P, CY, AC.

CMP M (сравнение содержимого ячейки памяти с содержимым аккумулятора).

(*A*) – (*H*, *L*)

Содержимое ячейки памяти, адрес которой определен в регистрах *H* и *L*, вычитается их содержимого аккумулятора. Аккумулятор не изменяется. По результату вычитания флаги устанавливаются следующим образом: флаг нуля Z — в 1, если (*A*) = (*H*, *L*); флаг переноса CY — в 1, если *A* < (*H*, *L*).

1	0	1	1	1	1	1	0

CMP M BE

Циклы: 2.
Состояния: 7.
Адресация: косвенная регистровая.
Флаги: Z, S, P, CY, AC.

CPI (байт) (непосредственное сравнение).

(A) – (байт 2)

Содержимое второго байта команды вычитается из содержимого аккумулятора. По результату вычитания флаги устанавливаются следующим образом: флаг нуля Z — в 1, если (*A*) = (байт 2); флаг переноса CY — в 1, если (*A*) < (байт 2).

1	1	1	1	1	1	1	0
Данные							

CPI байт FE.

Циклы: 2.
Состояния: 7.
Адресация: непосредственная.
Флаги: Z, S, P, CY, AC.

RLC (циклический сдвиг влево).

$A_{n+1} \leftarrow (A_n)$, $(A_0) \leftarrow (A_7)$, $(CY) \leftarrow (A_7)$

Содержимое аккумулятора сдвигается влево на одну позицию. Содержимое самого старшего бита заносится в младший бит и бит флага переноса. Команда воздействует только на флаг переноса CY.

0	0	0	0	0	1	1	1

RLC 07

Циклы: 1.
Состояния: 4.
Флаги: CY.

RRC (циклический сдвиг вправо).

$A_n \leftarrow (A_{n+1})$, $(A_7) \leftarrow (A_0)$, $(CY) \leftarrow (A_0)$

Содержимое аккумулятора сдвигается на одну позицию вправо. Содержимое младшего бита заносится в самый старший бит и бит флага переноса. Команда воздействует только на флаг переноса CY.

0	0	0	0	1	1	1	1

RRC OF

Циклы: 1.
Состояния: 4.
Флаги: CY.

RAL (сдвиг влево через перенос).

$(A_{n+1}) \leftarrow (A_n)$, $(CY) \leftarrow (A_7)$, $(A_0) \leftarrow (CY)$

Содержимое аккумулятора сдвигается на одну позицию влево через бит флага переноса CY. Младший бит устанавливается равным флагу переноса, а бит флага переноса CY — равным старшему биту аккумулятора. Команда воздействует только на флаг переноса CY.

0	0	0	1	0	1	1	1

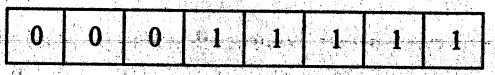
RAL 17.

Циклы: 1.
Состояния: 4.
Флаги: CY.

RAR (сдвиг вправо через перенос).

$(A_n) \leftarrow (A_{n+1})$, (CY) $\leftarrow (A_0)$, $(A_7) \leftarrow$ (CY)

Содержимое аккумулятора сдвигается на одну позицию вправо через бит флага переноса. Старший бит аккумулятора устанавливается равным флагу переноса, а бит флага переноса — равным младшему биту аккумулятора. Используется только флаг переноса CY.



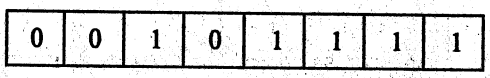
RAR 1F

Циклы: 1.
Состояния: 4.
Флаги: CY.

CMA (дополнение аккумулятора).

$(A) \leftarrow \overline{(A)}$

Содержимое аккумулятора инвертируется (бит, равный 1, становится равным 0; бит, равный 0, становится равным 1). Флаги не используются.



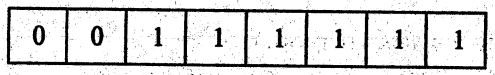
CMA 2F.

Циклы: 1.
Состояния: 4.
Флаги: нет.

CMC (дополнение флага переноса).

CY $\leftarrow \overline{(CY)}$

Инвертируется бит флага переноса CY. Другие флаги не используются.



CMC 3F.

Циклы: 1.
Состояния: 4.
Флаги: CY.

STC (установка флага переноса).

CY $\leftarrow$ 1

Бит флага переноса устанавливается в 1. Другие флаги не используются.

0	0	1	1	0	1	1	1

STC 37.

Циклы: 1.
Состояния: 4.
Флаги: CY.

Команды перехода

Команды этой группы изменяют нормальное последовательное исполнение программы. Исполнение команд не влияет на состояние флагов.

Имеются два типа команд перехода: безусловные и условные. Безусловные команды перехода осуществляют передачу управления, изменяя значение программного счетчика PC. Команды условного перехода сначала проверяют состояние одного из четырех флагов процессора, чтобы определить выполнено ли указанное в команде условие. Приведем имеющиеся условия перехода.

Условия:	Коды:
NZ — не нуль (Z = 0)	000
Z — нуль (Z = 1)	001
NC — нет переноса (CY = 0)	010
C — есть перенос (CY = 1)	011
PO — нечетный результат (P = 0)	100
PE — четный результат (P = 1)	101
P — плюс (S = 0)	110
M — минус (S = 1)	111

JMP [адр.] (переход).

(*PC*) ← (байт 3)(байт 2)

Управление передается команде, адрес которой указан во втором и третьем байтах команды перехода.

1	1	0	0	0	0	1	1
Младший байт адреса							
Старший байт адреса							

JMP [адр.] C3.

Циклы: 3.

Состояния: 10.
Адресация: непосредственная.
Флаги: нет.

J (условие) **[адр.]** (условный переход).

Если (CCC), то (*PC*) ← (байт 3)(байт 2).

Если указанное условие истинно, то управление передается команде, адрес которой указан во втором и третьем байтах команды перехода. Если условие ложно, то последовательный код программы не изменяется.

1	1	C	C	C	0	1	0
Младший байт адреса							
Старший байт адреса							

JNZ [адр.] C2
JZ [адр.] CA
JNC [адр.] D2
JC [адр.] DA
JPO [адр.] E2
JPE [адр.] EA
JP [адр.] F2
JM [адр.] FA

Циклы: 3.
Состояния: 10.
Адресация: непосредственная.
Флаги: нет.

CALL [адр.] (вызов).

(*SP*) – 1 ← (*PCH*)
(*SP*) – 2 ← (*PCL*)
(*SP*) ← (*SP*) – 2
(*PC*) ← (байт 3)(байт 2)

Старшие восемь разрядов адреса следующей команды пересылаются в ячейку памяти, адрес которой на 1 меньше содержимого указателя стека *SP*. Младшие восемь разрядов адреса следующей команды пересылаются в ячейку памяти, адрес которой на 2 меньше содержимого указателя стека *SP*. Содержимое указателя стека уменьшается на 2. Управление передается команде, адрес которой указан во втором и третьем байтах команды вызова.

1	1	0	0	1	1	0	1
Младший байт адреса							
Старший байт адреса							

CALL [адр.] CD

Циклы: 5.
Состояния: 18.
Адресация: непосредственная/косвенная регистровая.
Флаги: нет.

C (условие) **[адр.]** (условный вызов).

Если (CCC), то $(SP) - 1 \leftarrow (PCH)$,
$(SP) - 2 \leftarrow (PCL)$, $(SP) \leftarrow (SP) - 2$
$(PC) \leftarrow$ (байт 3)(байт 2)

Если указанное условие истинно, то выполняются действия, описанные в команде CALL, в противном случае последовательность исполнения команд не изменяется.

1	1	C	C	C	1	0	0
Младший байт адреса							
Старший байт адреса							

CNZ [адр.] C4
CZ [адр.] CC
CNC [адр.] D4
CC [адр.] DC
CPO [адр.] E4
CPE [адр.] EC
CP [адр.] F4
CM [адр.] FC

Циклы: 5.
Состояния: 18.
Адресация: непосредственная/косвенная регистровая.
Флаги: нет.

RET (возврат).

$(PCL) \leftarrow (SP)$
$(PCH) \leftarrow (SP) + 1$
$(SP) \leftarrow (SP) + 2$

Содержимое ячейки памяти, адрес которой содержится в указателе стека *SP*, пересылается в восемь младших разрядов программного счетчика *PC*. Содержимое ячейки памяти, адрес которой на 1 больше содержимого указателя стека, пересылается в восемь старших разрядов программного счетчика. Содержимое указателя стека увеличивается на 2.

1	1	0	0	1	0	0	1

RET C9.

Циклы: 3.

Состояния: 10.
Адресация: косвенная регистровая.
Флаги: нет.

R условие (условный возврат).

Если (CCC), то $(PCL) \leftarrow (SP)$,
$(PCH) \leftarrow (SP) + 1$, $(SP) \leftarrow (SP) + 2$

Если указанное условие истинно, то выполняются действия, описанные в команде RET, в противном случае последовательность выполнения команд не нарушается.

1	1	C	C	C	0	0	0

RNZ C0
RZ C8
RNC D0
RC D8
RPO E0
RPE E8
RP F0
RM F8

Циклы: 3.
Состояния: 12.
Адресация: косвенная регистровая.
Флаги: нет.

RST n (рестарт).

$(SP) - 1 \leftarrow (PCH)$
$(SP) - 2 \leftarrow (PCL)$
$(SP) \leftarrow (SP) - 2$
$(PC) \leftarrow 8(NNN)$

Старшие восемь разрядов адреса следующей команды пересылаются в ячейку памяти, адрес которой на 1 меньше содержимого указателя стека. Младшие восемь разрядов адреса следующей команды пересылаются в ячейку памяти, адрес которой на 2 меньше содержимого указателя стека. Содержимое указателя стека уменьшается на 2. Управление передается команде, адрес которой равен коду NNN, умноженному на 8.

1	1	N	N	N	1	1	1

RST 0 C7
RST 1 CF
RST 2 D7
RST 3 DF
RST 4 E7
RST 5 EF

RST 6 F7
RST 7 FF

Циклы: 3.
Состояния: 12.
Адресация: косвенная регистровая.
Флаги: нет.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	N	N	N	0	0	0

Программный счетчик после рестарта.

PCHL (загрузка содержимого регистров *H* и *L* в программный счетчик).

(*PCH*) ← (*H*)
(*PCL*) ← (*L*)

Содержимое регистра *H* пересылается в восемь старших разрядов программного счетчика *PC*. Содержимое регистра *L* пересылается в восемь младших разрядов программного счетчика.

1	1	1	0	1	0	0	1

PCHL E9.

Циклы: 1.
Состояния: 6.
Адресация: регистровая.
Флаги: нет.

Команды ввода-вывода, управления и работы со стеком

Команды этой группы выполняют операции ввода-вывода, изменяют содержимое стека и состояние флагов.

Регистр флагов

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
S	Z	*	AC	*	P	X	CY

* Безразлично.

PUSH rp (засылка в стек содержимого пары регистров).

(*SP*) – 1 ← (rh)
(*SP*) – 2 ← (rl)
(*SP*) ← (*SP*) – 2

Содержимое старшего регистра пары регистров rp пересылается в ячейку памяти, адрес которой на 1 меньше содержимого указателя стека *SP*. Содержимое младшего регистра пары регистров rp пересылается в ячейку памяти, адрес которой на 2 меньше содержимого указателя стека. Содержимое указателя стека уменьшается на 2.

Примечание. Пара регистров rp = *SP* не может быть указана в команде.

1	1	R	P	0	1	0	1

PUSH B C5 (заслать пару регистров *B* и *C*).
PUSH D D5 (заслать пару регистров *D* и *E*).
PUSH H E5 (заслать пару регистров *H* и *L*).

Циклы: 3.
Состояния: 12.
Адресация: косвенная регистровая.
Флаги: нет.

PUSH PSW (заслать в стек слово состояния программы).

$(SP - 1) \leftarrow (A)$
$(SP - 2)_0 \leftarrow (CY)$, $(SP) - 2)_1 \leftarrow *$
$(SP - 2)_2 \leftarrow (P)$, $(SP) - 2)_3 \leftarrow *$
$(SP - 2)_4 \leftarrow (AC)$, $(SP) - 2)_5 \leftarrow *$
$(SP - 2)_6 \leftarrow (Z)$, $(SP) - 2)_7 \leftarrow (S)$
$(SP) \leftarrow (SP - 2)$

* Безразлично.

Содержимое аккумулятора пересылается в ячейку памяти, адрес которой на 1 меньше содержимого указателя стека *SP*. Содержимое флагов объединяется в слово состояния, и это слово пересылается в ячейку памяти, адрес которой на 2 меньше содержимого указателя стека. Содержимое указателя стека уменьшается на 2.

1	1	1	1	0	1	0	1

PUSH PSW F5.

Циклы: 3.
Состояния: 12.
Адресация: косвенная регистровая.
Флаги: нет.

POP rp (считать из стека содержимое пары регистров).

(rl) $\leftarrow$ (*SP*)
(rh) $\leftarrow$ (*SP*) + 1
(*SP*) $\leftarrow$ (*SP*) + 2

Содержимое ячейки памяти, адрес которой определяется содержимым указателя стека *SP*, пересылается в младший регистр пары регистров rp. Содержимое ячейки памяти, адрес которой на 1 больше содержимого указателя стека, пересылается в старший регистр пары регистров rp. Содержимое указателя стека увеличивается на 2.

Примечание. В команде не может быть указана пара регистров rp = *SP*.

1	1	R	P	0	0	1

POP B C1 (считать из стека пару регистров *B* и *C*).
POP D D1 (считать из стека пару регистров *D* и *E*).
POP H E1 (считать из стека пару регистров *H* и *L*).

Циклы: 3.
Состояния: 10.
Адресация: косвенная регистровая.
Флаги: нет.

POP PSW (считать из стека слово состояния программы).

$(CY) \leftarrow (SP)_0$
$(P) \leftarrow (SP)_2$
$(AC) \leftarrow (SP)_4$
$(Z) \leftarrow (SP)_6$
$(S) \leftarrow (SP)_7$
$(A) \leftarrow (SP) + 1$
$(SP) \leftarrow (SP) + 2$

Содержимое ячейки памяти, адрес которой определяется содержимым указателя стека *SP*, используется для восстановления состояния флагов. Содержимое ячейки памяти, адрес которой на 1 больше содержимого указателя стека, переключается в аккумулятор. Содержимое указателя стека увеличивается на 2.

1	1	1	1	0	0	0	1

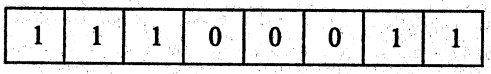
POP PSW F1.

Циклы: 3.
Состояния: 10.
Адресация: косвенная регистровая.
Флаги: Z, S, P, CY, AC.

XTHL (обмен содержимого верхушки стека и содержимого регистров *H* и *L*).

$(L) \leftrightarrow (SP)$
$(H) \leftrightarrow (SP) + 1$

Содержимое регистра *L* обменивается на содержимое ячейки памяти, адрес которой содержится в указателе стека *SP*. Содержимое регистра *H* обменивается на содержимое ячейки памяти, адрес которой на 1 больше содержимого указателя стека.



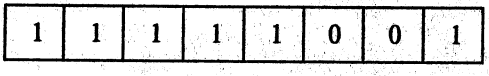
XTHL E3.

Циклы: 5.
Состояния: 16.
Адресация: косвенная регистровая.
Флаги: нет.

SPHL (пересылка содержимого регистров *H* и *L* в указатель стека).

(*SP*) ↔ (*H*)(*L*)

Содержимое регистров *H* и *L* (16 бит) пересылается в указатель стека.



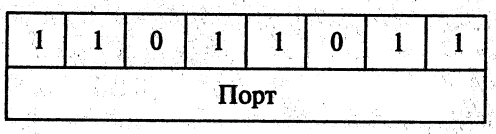
SPHL F9.

Циклы: 1.
Состояния: 6.
Адресация: регистровая.
Флаги: нет.

IN порт (ввод).

(*A*) ← (Данные)

Данные, имеющиеся на 8 битах двунаправленной шины данных указанного порта, пересылаются в аккумулятор.



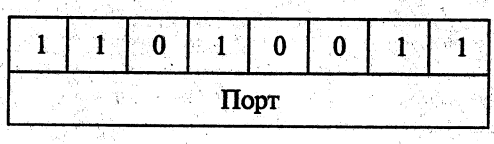
IN порт DB.

Циклы: 3.
Состояния: 10.
Адресация: прямая.
Флаги: нет.

OUT порт (вывод).

(Данные) ← (*A*)

Содержимое аккумулятора помещается на двунаправленную шину данных для передачи в указанный порт.



OUT порт D3.

Циклы: 3.
Состояния: 10.
Адресация: прямая.
Флаги: нет.

EI (разрешение прерываний).
Система прерываний разрешается при исполнении следующей команды.

1	1	1	1	1	0	1	1

EI FB.

Циклы: 1.
Состояния: 4.
Флаги: нет.

Примечание. Прерывания не распознаются во время исполнения команды EI.

DI (запрещение прерывания).
Система прерываний запрещается непосредственно при выполнении следующей за DI командой.

1	1	1	1	0	0	1	1

DI FB

Циклы: 1.
Состояния: 4.
Флаги: нет.

Примечание. Прерывания не распознаются во время исполнения команды.

HLT (останов).

В PC заносится адрес следующей команды. Процессор затем бездействует до прихода прерывания. Регистры и флаги не изменяют состояния.

0	1	1	1	0	1	1	0

HLT 76.

Циклы: 1.
Состояния: 5.
Флаги: нет.

NOP (нет операции).

В PC заносится адрес следующей команды, и МП переходит к ее обработке. Состояние флагов и регистров не изменяется.

0	0	0	0	0	0	0	0

NOP 00.

Циклы: 1.
Состояния: 4.
Флаги: нет.

СПИСОК ЛИТЕРАТУРЫ

1. *Алексенко А.Г., Шагурин И.И.* Микросхемотехника: Учеб. пособие для вузов. —2-е изд., перераб. и доп. — М.: Радио и связь, 1990.

2. Аналоговые и цифровые интегральные микросхемы: Справочник / С. В. Якубовский, Л. И. Ниссельсон, В. И. Кулешова и др.; Под ред. С. В. Якубовского. — М.: Радио и связь, 1989.

3. Большие интегральные схемы запоминающих устройств: Справочник / А. Ю. Гордонов, Н. В. Бекин, В. В. Цыкин и др.; Под ред. А. Ю. Гордонова, Ю. Н. Дьякова. — М.: Радио и связь, 1990.

4. *Брammер Ю.А., Пащук И.Н.* Импульсные и цифровые устройства: Учеб. для студентов электрорадиоприборостроительных сред. спец. учеб. заведений. — 6-е изд., перераб. и доп. — М.: Высш. шк., 2000.

5. *Горбунов В.Л., Панфилов Д.И., Преснухин Д.Л.* Справочное пособие по микропроцессорам и микроЭВМ / Под ред. Л. Н. Преснухина. — М.: Высш. шк., 1988.

6. *Гук М.* Аппаратные средства IBM PC: Энциклопедия. — СПб.: Питер, 2001.

7. *Зельдин Е.А.* Цифровые интегральные микросхемы в информационно-измерительной аппаратуре. — Л.: Энергоатомиздат, 1986.

8. *Калабеков Б.А.* Цифровые устройства и микропроцессорные системы: Учебник для техникумов связи. — М.: Горячая линия — Телеком, 2000.

9. *Лебедев О.Н.* Применение микросхем памяти в электронных устройствах: Справочное пособие. — М.: Радио и связь, 1994.

10. *Лебедев О.Н., Мирошниченко А.И., Телец В.А.* Изделия электронной техники. Цифровые микросхемы. Микросхемы памяти. Микросхемы ЦАП и АЦП: Справочник. — М.: Радио и связь, 1994.

11. Микропроцессоры: В 3 кн.: Учеб. для втузов / П. В. Нестеров, В. Г. Шаньгин, В. Л. Горбунов и др.; Под ред. Л. Н. Преснухина. — М.: Высш. шк., 1986.

12. Микропроцессоры: Справочное пособие для разработчиков судовой РЭА / Г. Г. Гришин, А. А. Мошков, О. В. Ольшанский, Ю. А. Овечкин. — Л.: Судостроение, 1987.

13. Микропроцессоры и микропроцессорные комплекты интегральных микросхем: Справочник: В 2 т. / В.-Б. Б. Абрайтис, Н. Н. Аверьянов, А. И. Белоус и др.; Под ред. В. А. Шахнова. — М.: Радио и связь, 1988.

14. Микросхемы памяти, ЦАП и АЦП: Справочник / О. Н. Лебедев, А.-Й. К. Марцинкявичус, Э.-А. К. Багданскис и др. — 2-е изд. — М.: КУбКа, 1996.

15. *Нефедов А. В.* Интегральные микросхемы и их зарубежные аналоги: Справочник: В 11 т. — М.: ИП РадиоСофт, 2001.

16. *Опадчий Ю. Ф., Глудкин О. П., Гуров А. И.* Аналоговая и цифровая электроника (полный курс): Учебник для вузов / Под ред. О. П. Глудкина. — М.: Горячая Линия — Телеком, 2000.

17. Полупроводниковые БИС запоминающих устройств: Справочник / В. В. Баранов, Н. В. Бекин, А. Ю. Гордонов и др.; Под ред. А. Ю. Гордонова и Ю. Н. Дьякова. — М.: Радио и связь, 1987.

18. *Потемкин И. С.* Функциональные узлы цифровой автоматики. — М.: Энергоатомиздат, 1988.

19. Применение интегральных микросхем в электронной вычислительной технике: Справочник / Р. В. Данилов, С. А. Ельцова, Ю. П. Иванов и др.; Под ред. Б. Н. Файзулаева, Б. В. Тарабрина. — М.: Радио и связь, 1986.

20. Применение интегральных микросхем памяти: Справочник / А. А. Дерюгин, В. В. Цыркин, В. Е. Красовский и др.; Под ред. А. Ю. Гордонова, А. А. Дерюгина. — М.: Радио и связь, 1994.

21. *Угрюмов Е. П.* Цифровая схемотехника. — СПб.: БХВ-Петербург, 2001.

22. Цифровые интегральные микросхемы: Справочник / П. П. Мальцев, Н. С. Долидзе, М. И. Критенко и др. — М.: Радио и связь, 1994.

23. *Фридман А., Менон П.* Теория и проектирование переключательных схем. — М.: МИР, 1978.

24. *Хвощ С. Т., Варлинский Н. И., Попов Е. А.* Микропроцессоры и микроЭВМ в системах автоматического управления. — Л.: Машиностроение, 1987.

ОГЛАВЛЕНИЕ

Предисловие 3

Раздел I
АРИФМЕТИЧЕСКИЕ ОСНОВЫ ТЕОРИИ ЦИФРОВЫХ УСТРОЙСТВ

Глава 1. Системы счисления 5

1.1. Основные понятия 5
1.2. Правила перевода из одной системы счисления в другую 9

Глава 2. Формы представления чисел 16

2.1. Форматы данных 16
2.2 Представление чисел в форме с плавающей запятой 16
2.3. Представление чисел в форме с фиксированной запятой 18

Глава 3. Машинные коды 21

3.1. Кодирование положительных чисел в форме с фиксированной запятой 21
3.2. Кодирование отрицательных чисел в форме с фиксированной запятой 22
3.3. Правило перевода отрицательных чисел из дополнительного и обратного кодов в прямой код 22

Глава 4. Арифметические действия над числами с фиксированной запятой 23

4.1. Сложение положительных чисел 23
4.2. Сложение отрицательных чисел или чисел с разными знаками 25
4.3. Умножение чисел 26
4.4. Деление чисел 31

Глава 5. Арифметические действия над числами с плавающей запятой 33

5.1. Сложение чисел 33
5.2. Умножение чисел 34
5.3. Деление чисел 36

Глава 6. Арифметические действия над числами в двоично-десятичной системе счисления 38

6.1. Особенности сложения в двоично-десятичной системе счисления 38
6.2. Сложение чисел с одинаковыми знаками 39
6.3. Сложение чисел с разными знаками 40

Раздел II
ЛОГИЧЕСКИЕ ОСНОВЫ ЦИФРОВОЙ СХЕМОТЕХНИКИ

Глава 7. Основные понятия алгебры логики 43

7.1. Общие сведения 43
7.2. Переключательные функции одной и двух переменных 44
7.3. Тождества и законы алгебры логики 50
7.4. Дизъюнктивные и конъюнктивные нормальные формы представления функций алгебры логики 53
7.5. Минимизация логических функций с использованием законов и тождеств 55
7.6. Минимизация логических функций с использованием карт Карно (диаграмм Вейча) 58
7.7. Метод испытаний 61
7.8. Анализ и синтез комбинационных схем 62
7.9. Правила оформления схем цифровых устройств 63
7.10. Функционально полные системы элементов (базис) 65
7.11. Особенности работы комбинационных схем. Гонки 70
7.12. Использование логического элемента в качестве ключа 71

Глава 8. Цифровые интегральные микросхемы 74

8.1. Общие сведения о цифровых микросхемах 74
8.2. Логические схемы на биполярных транзисторах 76
8.3. ТТЛ с транзисторами Шотки 81
8.4. Основные электрические параметры и характеристики логических элементов ТТЛ 84
8.5. Особенности применения микросхем ТТЛ при разработке цифровых устройств 87
8.6. Логические элементы на КМОП-транзисторах 92
8.7. Принципы построения логических элементов на КМОП-транзисторах 96
8.8. Физическая структура микросхемы на КМОП-транзисторах ... 100
8.9. Основные характеристики и параметры микросхем на КМОП-транзисторах 101
8.10. Особенности применения микросхем на КМОП-транзисторах при построении цифровых устройств 104

Раздел III
КОМБИНАЦИОННЫЕ УСТРОЙСТВА

Глава 9. Дешифраторы и шифраторы 108

9.1. Общие сведения о дешифраторах 108
9.2. Синтез линейного дешифратора 109
9.3. Схема прямоугольного дешифратора 111
9.4. Схема пирамидального дешифратора 114
9.5. Сравнительная оценка различных видов дешифраторов. Построение дешифраторов на большое количество входов 115

9.6. Области применения дешифраторов ... 116
9.7. Шифраторы ... 117

Глава 10. Мультиплексоры и демультиплексоры ... 121

10.1. Различные способы построения мультиплексоров ... 121
10.2. Универсальность использования мультиплексоров ... 123
10.3. Мультиплексное дерево ... 124
10.4. Демультиплексоры ... 126

Глава 11. Сумматоры ... 128

11.1. Общие сведения ... 128
11.2. Одноразрядный сумматор на два входа (полусумматор) ... 129
11.3. Одноразрядный сумматор на три входа ... 130
11.4. Организация сумматора на три входа на основе полусумматоров ... 132
11.5. Реализация сумматора на интегральных схемах ... 133
11.6. Организация многоразрядного параллельного сумматора с последовательным переносом на базе интегральных схем ИМ1, ИМ2, ИМ3 ... 134
11.7. Использование сумматоров в интегральном исполнении при выполнении различных арифметических операций ... 136
11.8. Комбинационный сумматор с ускоренным распространением переноса ... 139
11.9. Сумматор с ускоренным переносом на базе интегральной микросхемы ИМ6 ... 141
11.10. Сумматор последовательного типа ... 145
11.11. Накапливающий сумматор ... 146
11.12. Двоично-десятичный сумматор ... 147

Глава 12. Программируемые логические структуры ... 149

12.1. Общие сведения ... 149
12.2. Организация программируемой логической матрицы. ... 150
12.3. Программируемые матрицы логики ... 155

Раздел IV
ПОСЛЕДОВАТЕЛЬНОСТНЫЕ УСТРОЙСТВА

Глава 13. Триггеры ... 160

13.1. Общие сведения ... 160
13.2. Асинхронные *RS*-триггеры ... 164
13.3. Синхронные триггеры со статическим управлением ... 168
13.4. Двухступенчатые триггеры *MS*-типа с статическим управлением ... 172
13.5. Синхронные триггеры с динамическим управлением ... 179

Глава 14. Регистры ... 183

14.1. Общие сведения ... 183
14.2. Схемы простейших регистров ... 184
14.3. Реверсивные регистры ... 186

Глава 15. Счетчики ... 191
15.1. Общие сведения ... 191
15.2. Асинхронные счетчики ... 192
15.3. Синхронные счетчики ... 198
15.4. Счетчики с произвольным коэффициентом пересчета ... 208

Раздел V
ПРИНЦИПЫ ПОСТРОЕНИЯ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

Глава 16. Структуры вычислительных систем ... 224
16.1. Классическая структура ... 224
16.2. Магистральная структура ... 225

Глава 17. Организация устройств управления ... 227
17.1. Организация микропрограммных устройств управления ... 227
17.2. Организация устройств управления с жесткой логикой ... 233
17.3. Особенности построения устройств управления. Структура построения вычислительной системы на основе БИС секционированного микропроцессора ... 245
17.4. Внутренняя и внешняя память вычислительных систем ... 247

Раздел VI
ЗАПОМИНАЮЩИЕ УСТРОЙСТВА

Глава 18. Полупроводниковые запоминающие устройства ... 250
18.1. Классификация микросхем памяти ... 250
18.2. Режимы работы и характеристики статических БИС ЗУ ... 251
18.3. Организация статических ЗУ ... 253
18.4. Структурная организация БИС ЗУ ... 256
18.5. Построение разных видов запоминающих элементов ОЗУ ... 263
18.6. Организация модулей ЗУ ... 268

Глава 19. Постоянные запоминающие устройства ... 274
19.1. Общие сведения ... 274
19.2. Однократно программируемые ПЗУ ... 275
19.3. Однократно программируемые ПЗУ в интегральном исполнении серии КР556 ... 277
19.4. Репрограммируемые ПЗУ ... 281
19.5. Различия между ПЗУ и ПЛМ ... 285

Раздел VII
ОСНОВЫ МИКРОПРОЦЕССОРНОЙ ТЕХНИКИ

Глава 20. Организация работы устройств на базе микропроцессоров ... 289
20.1. Общие сведения ... 289
20.2. Организация микроЭВМ ... 289
20.3. Классификация микропроцессоров ... 293

20.4. Работа устройств на базе микропроцессора с фиксированной разрядностью 295

Глава 21. Организация функционирования микропроцессорной системы 298

21.1. Структурная схема микропроцессора 298
21.2. Система команд 302
21.3. Способы адресации 319
21.4. Процесс выполнения команды в микропроцессорном устройстве 322
21.5. Интерфейсные интегральные схемы 341

Приложение 362
Список литературы 392