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

КУРС ЛЕКЦИЙ

ПМ.01 Монтаж, ввод в действие и эксплуатация устройств транспортного радиоэлектронного оборудования

МДК.01.01. Теоретические основы монтажа, ввода в действие и эксплуатации устройств транспортного радиоэлектронного оборудования

Тема 1.2 Цифровая схемотехника

для студентов специальности

11.02.06 Техническая эксплуатация транспортного радиоэлектронного оборудования (по видам транспорта)

Саратов 2023

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

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

_____/Ю.А. Солдатенко/

УТВЕРЖДАЮ
Заместитель директора по учебной работе

_____/С.А. Крижановский/

Автор (составитель):

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

Рецензент:

Соболева А.Б **-** преподаватель высшей квалификационной категории филиала СамГУПС в г. Саратове

Содержание

1 Введение. Основные особенности систем счисления для представления информации в устройствах цифровой схемотехники.	4
2 Формы представления чисел с фиксированной и плавающей запятой	15
3 Особенности выполнения арифметических операций смногоразрядными двоичными кодированными числами	22
4 Элементарные (основные, базисные функции И, ИЛИ, НЕ)	33
5 Основы аналитического и графического способов минимизации функций.	42
6 Общие сведения о цифровых интегральных микросхемах (ЦИМС)	58
7 Триггеры	65
8 Счетчики	78
9 Регистры	86
10 Шифраторы и дешифраторы	91
11 Преобразователи кодов	99
12 Мультиплексоры и демультиплексоры	106
13 Сумматоры	118
14 Компараторы	123
15 Классификация и параметры запоминающих устройств(ЗУ)	127
16 Оперативные запоминающие устройства	134
17 Постоянные запоминающие устройства	146
18 Цифро-аналоговые преобразователи (ЦАП)	153
19 Аналого-цифровые преобразователи (АЦП)	161
20 Общие сведения о микропроцессорах и микропроцессорных системах	167
21 Однокристальные микропроцессоры	171
Список источников	175

Лекция 1

Введение(1 час)

В цифровой технике пользуются *кодовыми словами.* Особенность этих слов состоит в том, что все они имеют одинаковую длину (т. е. последовательность букв одинаковой длины) и для их построения используется простейший алфавит, состоящий лишь из двух букв. Эти буквы принято обозначать символами 0 и 1. Таким образом,кодовое слово в цифровой технике есть последовательность символов 0 и 1 определенной длины, например 10111011. Такими словами могут

представляться и числа, в этом случае 0 и 1 совпадают по смыслу с обычными арабскими цифрами. При представлении кодовым словом некоторой нечисловой информации, чтобы отличать буквы 0 и 1 от цифр,будем эти буквы называть соответственно *логическим нулем* и *логической единицей.*

Если длина кодовых слов составляет *n* разрядов, то можно построить 2^n различных комбинаций — кодовых слов. Например, при $n = 3$ можно построить $2^3 = 8$ слов: 000, 001, 010, 011, 100, 101, 110, 111.

Информация, которая передается между отдельными узлами (блоками) сложного цифрового устройства, представляется в виде кодовых слов. Таким образом, на входы каждого узла поступают кодовые слова,на выходе узла образуется новое кодовое слово, представляющее собой результат обработки входных слов. Выходное слово зависит от того, какие слова поступают на входы узла. Поэтому можно говорить, что выходное слово есть функция, для которой аргументами являются входные слова. Для того чтобы подчеркнуть отличительную особенность такихфункций, состоящую в том, что сама функция и ее аргументы могут принимать значения логических 0 и 1, будем эти функции называть *функциями алгебры логики {ФАЛ).*

Устройства, предназначенные для формирования функции алгебры логики, в дальнейшем будем называть *логические устройства* или *цифровые устройства.*

Цифровые устройства (либо их узлы) можно делить на типы по разным признакам.

По способу ввода и вывода кодовых слов различают логические устройства последовательного, параллельного и смешанного действия.

На входы *устройства последовательного действия* символы кодовых слов поступают не одновременно, а последовательно символ за символом (в так называемой *последовательной форме).* В такой же последовательной форме формируется выходное слово. Пример такого устройства показан на рис. *2.1а.*

На входы *устройства параллельного действия* все *n* символов каждого входного кодового слова подаются одновременно (в так называемой*параллельной форме).* В такой же форме образуется на выходе

выходноеслово. Очевидно, при параллельной форме приема и выдачи кодовых слов в устройстве необходимо иметь для каждого разряда входного (выходного) слова отдельный вход (выход).

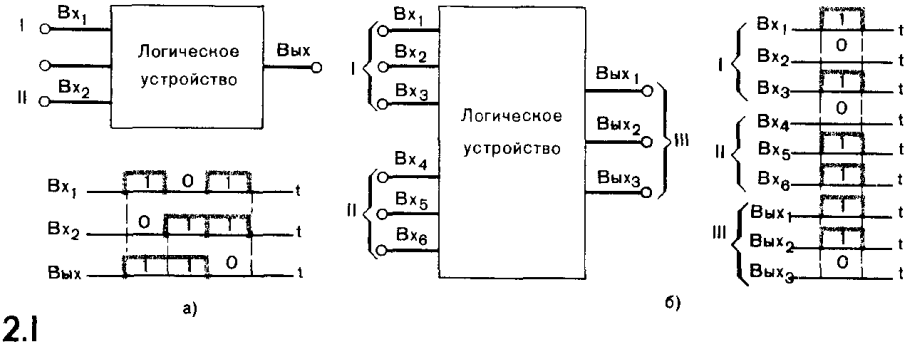
Пример такого устройства показан на рис. 2.1 б. Устройство выполняет над разрядами входных слов ту же логическую операцию (выявляя несовпадение символов соответствующих разрядов входных слов),что и устройство, показанное на рис. *2.1а,* но в параллельной форме. Входы устройства разделены на две группы *(I* и *II),* каждая из которых предназначена для приема трехразрядного входного кодового слова в параллельной форме. На выходах устройства также в параллельной форме получается трехразрядное выходное кодовое слово.

В *устройствах смешанного действия* входные и выходные кодовые слова представляются в разных формах. Например, входные слова — в последовательной форме, выходные — в параллельной. Устройства смешанного действия могут использоваться для преобразования кодовых слов из одной формы представления в другую (из последовательной формы в параллельную либо наоборот).

По типу схемного построения логические устройства (и их схемы) делят на два класса: *комбинационные устройства* (и соответственно*комбинационные схемы)* и *последовательностные устройства* (*последовательностные схемы).*

В комбинационном устройстве (называемом также *автоматом без памяти)* каждый символ на выходе (логический 0 или логическая 1) определяется лишь символами (логический 0 или логическая 1), действующими в данный момент времени на входах устройства, и не зависит от того, какие символы ранее действовали на этих входах. В этом смысле комбинационные устройства лишены памяти (они не хранят сведений о прошлом работы устройства).

В последовательностных устройствах (или *автоматах с памятью)*выходной сигнал определяется не только набором символов, действующих в данный момент времени на входах, но и внутренним состоянием устройства, а последнее зависит от того, какие наборы символов действовали во все предшествующие моменты времени. Из-за этого свойства устройство находится в одном из возможных состояний (в зависимости от символов, действовавших на входах в предшествующие моменты времени), и можно говорить, что последовательностные устройстваобладают памятью (они хранят сведения о прошлом работы устройства)

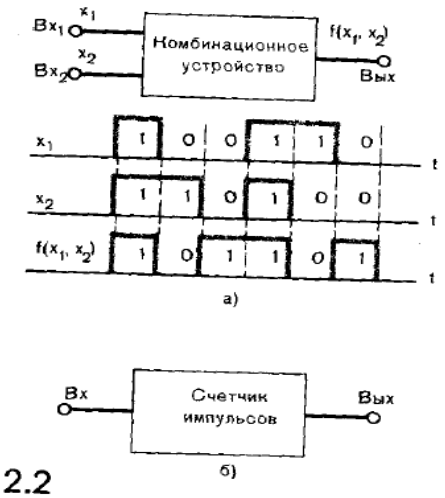


2.1

Рассмотрим примеры комбинационного и последовательного устройств

Пусть устройство (рис. 2.2а) предназначено для формирования навыходе сигнала, определяющего совпадение сигналов на входах; на выходе формируется логическая 1 в случаях, когда на обоих входах действует логическая 1 либо логический 0; если на одном из входов действует логическая 1, а на другом логический 0, то на выходе устройства образуется логический 0. Такое устройство представляет собой пример комбинационного устройства, в котором значение формируемой на выходе логической функции определяется лишь значениями ее аргументов в данный момент времени.

Рассмотрим другой пример Счетчик на рис.2.2б подсчитывает импульсы. В каждый момент времени его состояние соответствует числу поступивших на вход импульсов. Выходная информация определяется тем, каково было состояние счетчика до данного момента времени и поступает или нет на вход импульс в этот момент. Таким образом, данное устройство является последовательным устройством.



2.2

Основные особенности систем счисления для представления (записи) информации в устройствах цифровой схемотехники (1 час)

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 $\mathbf{30_{(10)}}$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*.

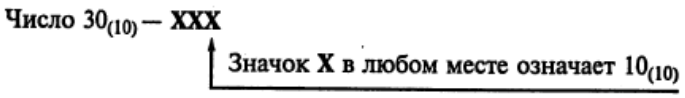
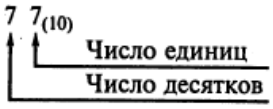


Рис. 1.1



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 $к$ — номер разряда, равный индексу при символах, изображающих цифры разрядов.

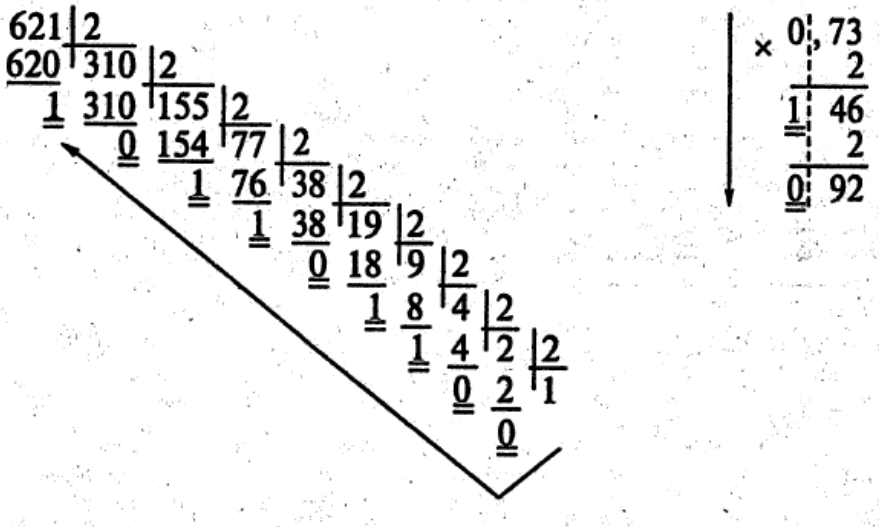
1.2. Правила перевода из одной системы счисления в другую

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):

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

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

3) (8) → (2):

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

4) (2) → (16):

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

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

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="3">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)
$25_{(10)} \mid 2$
$24 \quad 12 \mid 2$
$1 \quad 12 \quad 6 \mid 2$
$0 \quad 6 \quad 3 \mid 2$
$0 \quad 2 \mid 1$
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="3">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>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 $0,83_{(10)} \rightarrow 0,650_{(8)}$ 2) (10)→(2) $0,83_{(10)} \rightarrow 0,110_{(2)}$ 3) (10)→(16) $0,83_{(10)} \rightarrow 0,D_{(16)}$	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

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) $\underline{00}1\ \ 101\ \ 101\ ,\ 1\underline{00}_{(2)}\to(8)$ $1\ \ 5\ \ 5\ ,\ 4_{(8)}$	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) $\underline{00}10\ \underline{1100}\ \underline{1010}\ \underline{0110}\ ,\ 001\underline{0}_{(2)}$ $2\ \ C\ \ A\ \ 6\ ,\ 2_{(16)}$	Дополнение нулями до тетрады осуществляется влево и вправо от запятой

Окончание табл. 1.2

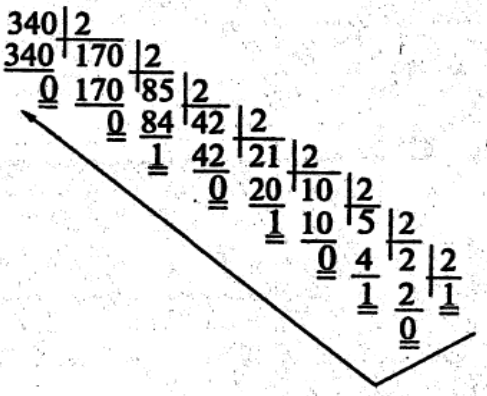
Значения чисел	Перевод	Правило перевода	Пример перевода	Примечание
Целые и дробные	(10)→(2–10)	Каждая цифра десятичного числа записывается в виде его четырехразрядного двоичного эквивалента	$2467{,}39_{(10)\rightarrow(2-10)}$ $\underbrace{0010}_{2}\ \underbrace{0100}_{4}\ \underbrace{0110}_{6}\ \underbrace{0111}_{7}\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

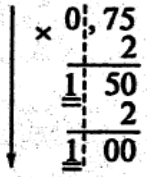
Пример 2. $524{,}6_{(8)} \to (10) \to (2) \to (16) \to (10) \to (8)$.

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):

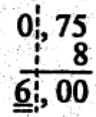
$\overset{000}{}1\ 0101\ 0100{,}11\overset{00}{}_{(2)}$

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

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 задания

Перевести исходные числа в другие системы счисления:

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

2) $278{,}3_{(10)} \to (8) \to (2) \to (10)$;

3) $51{,}4_{(10)} \to (8) \to (2) \to (10)$;

4) $69{,}18_{(10)} \to (8) \to (2) \to (16)$;

5) $5C_{(16)} \to (2) \to (10)$;

6) $1AB_{(16)} \to (2) \to (10)$;

7) $A5{,}7_{(16)} \to (2) \to (8) \to (10) \to (16)$;

8) $1011{,}01_{(2)} \to (10) \to (16) \to (2)$;

9) $C96{,}E_{(16)} \to (2) \to (10) \to (16)$;

10) $76{,}36_{(10)} \to (16) \to (2) \to (10) \to (2\text{—}10) \to (10)$;

11) $0011{,}01_{(2\text{—}10)} \to (10) \to (16) \to (2) \to (10) \to (2\text{—}10)$;

12) $31{,}7_{(8)} \to (10) \to (2\text{—}10) \to (10) \to (16) \to (2) \to (8)$.

Лекция 2

Формы представления чисел с фиксированной и плавающей запятой. Понятие о разрядной сетке.

Цифроваясхемотехника 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.***

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 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.

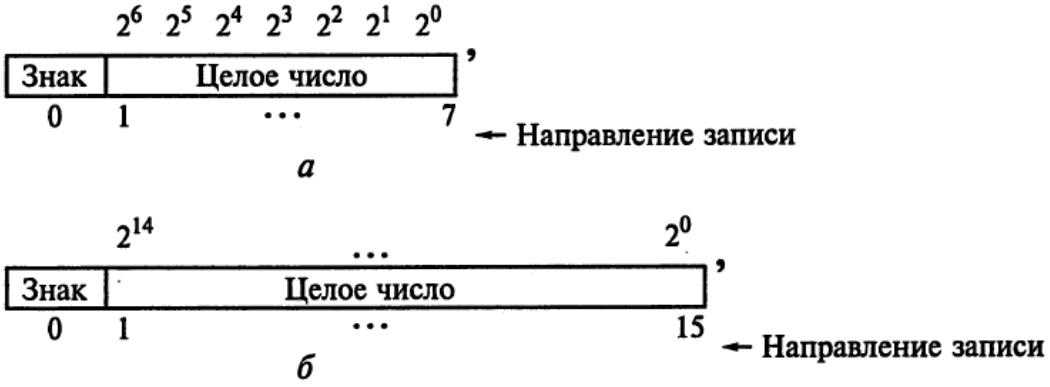


Рис. 2.2

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

$$A_1' = A_1 \cdot M_1 = +1101,\ 001 \cdot 1000 = +1101001,\ 00;$$
$$A_2' = A_2 \cdot M_1 = -10,\ 11101 \cdot 1000 = -0010111,\ 01.$$

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,

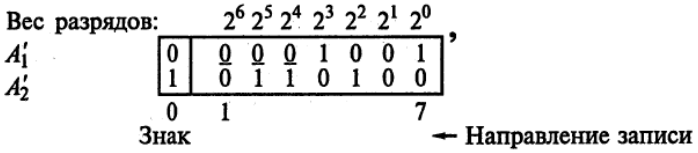


Рис. 2.3

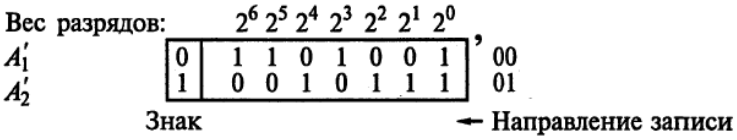


Рис. 2.4

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.

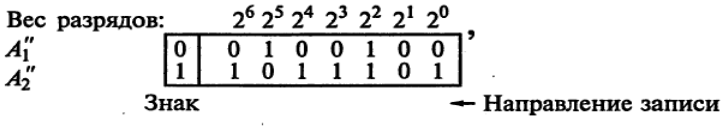


Рис. 2.5

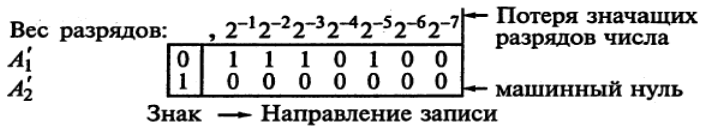


Рис. 2.6

Контрольные 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) $A_1 = -10{,}10101$; $A_2 = +0{,}0001$.

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 совпадают 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»

Кодирование отрицательных чисел в форме с фиксированной запятой

Коды отрицательных чисел (в отличие отположительных) 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

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 дополнительный 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
   \+ 1
   1. 0001010.

Таким образом прямые коды совпали.

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.

Лекция 3
Особенности выполнения арифметических операций с многоразрядными двоичными кодированными числами

Арифметика

Арифметические операции в позиционной системе с любым основанием производятся по одним и тем же правилам: сложение, вычитарние и умножение «в столбик», а деление — «уголком». Рассмотрим пример выполнения действий сложения и вычитания в двоичной, восьмеричной и шестнадцатеричной системах счисления.

Сложение

Двоичная система:

		•	•	•	•			(перенос)
1	0	0	1	1	0	1	1	
	1	0	0	1	1	1	0	
1	1	1	0	1	0	0	1	
7	6	5	4	3	2	1	0	(номера разрядов)

В нулевом разряде: 1 + 0 = 0

В первом разряде: 1 + 1 = 2. 2 переносится в старший (2-й) разряд, обращаясь в единицу переноса. В первом разряде остается 2 - 2 = 0.

Во втором разряде: 0 + 1 + 1 (перенос) = 2; Переносим в старший разряд,

В третьем разряде: 1 + 1 + 1 (перенос) = 3; В старший разряд переносим 2, здесь остается 3 - 2 = 1.

Продолжая вычисления, получим:

$$10011011_2 + 1001110_2 = 11101001_2$$

Восьмеричная система:

•		•			(перенос)
3	4	2	6	1	
	4	4	3	5	
4	0	7	1	6	
4	3	2	1	0	(номера разрядов)

Выполняем вычисления аналогично двоичной системе, но в старший разряд переносим 8. Получаем:

$$34261_8 + 4435_8 = 40716_8$$

Шестнадцатеричная система:

•					(перенос)
	A	3	9	1	
	8	5	3	4	
1	2	8	C	5	
4	3	2	1	0	(номера разрядов)

$$A391_{16} + 8534_{16} = 128C5_{16}$$

Вычитание

Двоичная система:

•			•	•				(перенос)
1	0	0	1	1	0	1	1	
	1	0	0	1	1	1	0	
	1	0	0	1	1	0	1	
7	6	5	4	3	2	1	0	(номера разрядов)

В нулевом разряде: 1 - 0 = 1

В первом разряде: 1 - 1 = 0.

Во втором разряде: 0 - 1; необходимо занять единицу старшего разряда. Поскольку веса разрядов двоичной системы отличаются в 2 раза: 2 + 0 - 1 = 1

Из третьего разряда занимали единицу, там остался 0, поэтому вновь нужно занимать из старшего разряда.

Продолжая вычисления, получим:

$$10011011_2 - 1001110_2 = 1001101_2$$

Восьмеричная система:

•	•		•		(перенос)
3	4	2	6	1	
	4	4	3	5	
2	7	6	2	4	
4	3	2	1	0	(номера разрядов)

Выполняем вычисления аналогично двоичной системе, но, занимая из старшего разряда, получаем 8. В результате:

$$34261_8 - 4435_8 = 27624_8$$

Шестнадцатеричная система:

	•		•	(перенос)
	A	3	9	1
	8	5	3	4
	1	E	3	D
4	3	2	1	0 (номера разрядов)

$$A391_{16} - 8534_{16} = 1E3D_{16}$$

Вариант 1 3. Выполните сложение чисел. а) $1110101010_2+10111001_2$; б) $10111010_2+10010100_2$; в) $111101110,1011_2+1111011110,1_2$; г) $1153,2_8+1147,32_8$; д) $40F,4_{16}+160,4_{16}$. 4. Выполните вычитание чисел. а) $1000000100_2-101010001_2$; б) $1010111101_2-111000010_2$; в) $1101000000,01_2-1001011010,011_2$; г) $2023,5_8-527,4_8$; д) $25E,6_{16}-1B1,5_{16}$.	**Вариант 2** 3. Выполните сложение чисел. а) $10111111_2+110010000_2$; б) $110010100_2+1011100001_2$; в) $1000000101,0101_2+1010000110,01_2$; г) $1512,4_8+1015,2_8$; д) $274,5_{16}+DD,4_{16}$. 4. Выполните вычитание чисел. а) $1000001001_2-111110100_2$; б) $1111000101_2-1100110101_2$; в) $1100110101,1_2-1011100011,01_2$; г) $1501,34_8-1374,5_8$; д) $12D,3_{16}-39,6_{16}$.
Вариант 3 3. Выполните сложение чисел. а) $1000100001_2+1011100110_2$; б) $1101110011_2+111000101_2$; в) $1011011,01_2+1000101110,1001_2$; г) $665,1_8+1217,2_8$; д) $30C,7_{16}+2A1,8_{16}$. 4. Выполните вычитание чисел. а) $11110010_2-10101001_2$; б) $1110100001_2-1011001001_2$; в) $1101001010,1_2-1011101001,11011_2$; г) $166,14_8-143,2_8$; д) $287,A_{16}-62,8_{16}$.	**Вариант 4** 3. Выполните сложение чисел. а) $11110100_2+110100001_2$; б) $1101110_2+101001000_2$; в) $1100110011,1_2+111000011,101_2$; г) $1455,04_8+203,3_8$; д) $14E,8_{16}+184,3_{16}$. 4. Выполните вычитание чисел. а) $1000010101_2-100101000_2$; б) $1001011011_2-101001110_2$; в) $111111011,101_2-100000010,01_2$; г) $341,2_8-275,2_8$; д) $249,5_{16}-EE,A_{16}$.
Вариант 1 ответы **Задание 3** а) 100011000112; б) 1010011102; в) 10111001101,001112; г) 2322,528; д) 56F,816. **Задание 4** а) 101100112; б) 111110112; в)11100101,1112;	**Вариант 2 ответы** **Задание 3** а) 10010011112; б) 100011101012; в)10010001011,100112; г) 2527,68; д) 351,916. **Задание 4** а) 101012; б) 100100002; в) 010010,012;

г) 1274,18; д) AD,116.	г) 104,648; д) F3,D16.
Вариант 3 ответы **Задание 3** а) 101000001112; б) 101001110002; в)1010001001,11012; г) 2104,38; д) 5AD,F16. **Задание 4** а) 10010012; б) 110110002; в) 1100000,101012 г) 22,748; д) 225,216.	**Вариант 4 ответы** **Задание 3** а) 10100101012; б) 1101101102; в) 1001111011,0012; г) 1660,348; д) 2D2,B16. **Задание 4** а) 111011012; б) 1000011012; в) 11111001,0112; г) 448; д) 15A,B16.

4.1. Арифметические операции над двоичными числами с фиксированной запятой

Сложение чисел. Сложение чисел с фиксированной запятой осуществляется в одном из машинных кодов -обратном или дополнительном. Причем, операции в этих кодах выполняются обычно над двоичными, числами, являющимися правильными дробями. Последовательность выполнения операции сложения следующая:

- исходные числа записываются в принятом для данной машины коде;

- производится поразрядное сложение кодов чисел, включая и знаковые разряды;

- единица переноса добавляется к младшему разряду суммы, если операция выполнялась над числами в обратном коде, или не учитывается (отбрасывается) в дополнительном коде;

- производится анализ на переполнение разрядной сетки. В случае переполнения вырабатывается сигнал на прерывание программы.

Переполнение разрядной сетки устраняется путем изменения масштабных коэффициентов.

Операция вычитания в ЭВМ заменяется операцией сложения в модифицированном обратном или дополнительном коде.

Рассмотрим сложение положительных и отрицательных чисел.

Пример 4.4. Сложить числа x=0,101001 и y=0,011011 в модифицированном коде.

Решение

$$\begin{array}{r} 00,101001 \\ +\,00,011011 \\ \hline 01,000100 \end{array}$$

Сочетание 01 в знаковых разрядах свидетельствует о переполнении. Полученная сумма есть число, по модулю превышающее единицу.

Пример 4.5. Сложить числа x=0,101111 и y=1,110010 в модифицированных обратном и дополнительном кодах. Результат сложения представить в прямом коде.

$$\begin{array}{r} [x]^{M}_{обр} = 00,101111 \\ +\,[y]^{M}_{обр} = 11,001101 \\ \hline [x+y]^{M}_{обр} = 11,111100 \end{array} \quad \text{и} \quad \begin{array}{r} [x]^{M}_{доп} = 00,101111 \\ +\,[y]^{M}_{доп} = 11,001110 \\ \hline [x+y]^{M}_{доп} = 11,111101 \end{array}$$

$$[x+y]^{M}_{обр} \rightarrow [x+y]^{M}_{доп} \rightarrow [x+y]_{пр} = 1,000011$$

Пример 4.5. Сложить числа x=1,1010 и y=1,1101

$$\begin{array}{r} [x]^{M}_{обр} = 11,0101 \\ +\,[y]^{M}_{обр} = 11,0010 \\ \hline [x+y]^{M}_{обр} = 110,0111 \\ \qquad\qquad \dots\dots\rightarrow 1 \\ \hline 10,1000 \end{array} \quad \text{и} \quad \begin{array}{r} [x]^{M}_{доп} = 11,0110 \\ +\,[y]^{M}_{доп} = 11,0011 \\ \hline [x+y]^{M}_{доп} = 110,0111 \end{array}$$

Различные значения цифр (10) в знаковых разрядах сумм. модифицированных обратного и дополнительного кодов означают переполнение разрядной сетки машины.

Пример 4.7. Выполнить вычитание чисел, используя дополнительный и обратный коды, уменьшаемое число x=0,101011, а вычитаемое число y=0,001111.

Решение. Изменяем знак *y* на обратный, получим $-y$=–0,001111. Образуем дополнительный и обратный коды:

$$[y]_{доп} = 11,110001 \text{ и } [y]_{обр} = 11,11000.$$

Затем выполняем операцию сложения $(x + [y]^{M}_{доп})$ и $(x + +[y]^{M}_{обр})$.

В дополнительном коде: $[x]^{м}_{доп} = 0{,}101011$, $[y]^{м}_{доп} = 1{,}110001$

$$\begin{array}{r} 00{,}101011 \\ + \quad 11{,}110001 \\ \hline 100{,}011100 \end{array}$$

Получаем результат в дополнительном коде 00,011100, что соответствует положительному числу +0,011100.

В обратном коде

$[x]^{м}_{доп} = 0{,}101011$, $[y]^{м}_{доп} = 1{,}110001$

$$\begin{array}{r} 00{,}101011 \\ + \quad 11{,}110000 \\ \hline 1 \\ \hline 00{,}011100 \end{array}$$ циклический перенос

Производя циклический перенос, получим результат в обратном коде 00,011100, что соответствует положительному числу +0,011100.

Умножение чисел. Умножение осуществляется в прямом коде. Процесс умножения состоит из операций сложения со сдвигами множимого или сумм частных произведений. Умножение может производиться начиная с младшего или старшего разряда множителя. В первом случае сдвиг частных произведений производится влево, а во втором – вправо. Частные произведения могут быть равны нулю, если в разряде множителя стоит нуль, или множимому, если в разряде множителя стоит единица. Знак произведения получается в результате сложения знаков сомножителей. При одинаковых знаках сомножителей знак произведения положительный, при разных знаках - отрицательный (0+0=0, 1+1=0, 1+0=1, 0+1=1).

Разрядность произведения определяется суммой разрядов сомножителей (n+n=2n). При округлении произведения младшие разряды отбрасываются, а старшие - округляются.

Пример 4.8. Найти произведение двух чисел x=0,1101 и y=1,1011.

а) определяем знак произведения 0+1=1;

б) производим умножение абсолютных значений сомножителей:

со сдвигом влево

$$\begin{array}{r} 0{,}1101 \\ \times \quad 0{,}1011 \\ \hline 1101 \\ + \quad 1101 \\ \quad 1101 \end{array}$$

со сдвигом вправо

$$\begin{array}{r} 0{,}1101 \\ \times \quad 0{,}1011 \\ \hline 1101 \\ + \quad 1101 \\ \quad 1101 \end{array}$$

Искомое произведение будет 1,10001111.

Деление чисел. Деление производится путем последовательного вычитания делителя из делимого или из образовавшихся остатков со сдвигом их на один разряд влево. Знак частного, как и при умножении, получается в результате сложения по модулю 2 знаков делимого и делителя.

В ЭВМ используются два метода деления: деление с восстановлением остатка и деление без восстановления остатка.

Деление с восстановлением остатка состоит в том, что если при вычитании делителя из делимого или очередного остатка получается положительный остаток (больше или равен делителю), то в разряд частного записывается 1, и полученный остаток сдвигается на один разряд влево. Если же остаток получится отрицательным (меньше делителя), то в разряд частного записывается нуль, а к остатку добавляется делитель для восстановления предыдущего остатка, и результат сдвигается на один разряд влево.

При делении без восстановления остатка полученный в результате вычитания остаток, независимо от его знака, сдвигается на один разряд влево. Из сдвинутого остатка, если он положительный, вычитается делитель и в разряд частного записывается 1, а если он отрицательный, то к нему прибавляется делитель и в разряд частного записывается нуль.

Для определения каждой цифры частного при делении без восстановления остатка необходимо выполнить 2 такта (сложение или вычитание и сдвиг).

Для выполнения деления без восстановления остатка чисел, заданных в дополнительном коде, при произвольном сочетании знаков делимого и делителя выполняются (n +l) тактов для получения 1 разряда знака частного и *n* разрядов мантиссы частного; первый такт отличается от последующих *N* тактов: он состоит только из одной операции-вычитания (или сложения) делителя и делимого (без предварительного сдвига делимого); все последующие *N* тактов одинаковы и состоят из двух операций: а) сдвига частичного остатка, полученного в предыдущем такте, на 1 .разряд влево, и б) вычитания (или сложения) делителя и полученного сдвинутого частичного остатка.

При этом:

1) в случае совпадения знаков делителя и очередного остатка (в первом такте - делимого) - сочетания знаков 00 'и 11 - код регистра делителя преобразуется в дополнительный и передается на сумматор;

в младший разряд частного записывается первая очередная цифра частного. В следующем такте частное и частичный остаток сдвигаются на один разряд влево;

2) в случае несовпадения знаков делителя и очередного остатка (в первом такте - делимого) - сочетания знаков 01 и 10-код с регистра делителя передается на сумматор без преобразования; в младший разряд частного записывается нуль. В следующем такте частное и частичный остаток сдвигаются на один разряд влево.

Пример 4.9. Выполнить в дополнительном коде деление без восстановления остатка двух чисел x=0,10011 и y=0,11001.

Решение для деления без восстановления остатка: делимое x=0,10011, делитель y=0,11001, $[-y]_{доп}$= 1,00111.

Сумматор (см)		Регистр частного	Примечание
00	10011	00000	начальное положение: на СМ – делимое x частное равно 0
01	00110		сдвиг содержимого СМ влево
+ 11	00111	00001	$[-y]_{доп}$ на СМ сочетание знаков СМ и y – 00
00	01101		частное равно 1
00	11010		сдвиг содержимого СМ влево
+ 11	00111	00011	$[-y]_{доп}$ на СМ сочетание знаков СМ и y – 00
00	00001		частное равно 1
00	00010		сдвиг содержимого СМ влево
+ 11	00111	00110	$[y]_{доп}$ на СМ сочетание знаков СМ и y – 10
00	01001		частное равно 0
10	10010		сдвиг содержимого СМ влево
+ 00	11001	01100	$[y]_{пр}$ на СМ сочетание знаков СМ и y – 10
11	01011		частное равно 0
10	10110		сдвиг содержимого СМ влево
+ 00	11001	11000	$[y]_{пр}$ на СМ сочетание знаков СМ и y – 10
11	01111		частное равно 0 и т. д.

Деление продолжается до тех пор пока не будет получена

требуемая точность частного.

Ответ: частное 0,11000, остаток 2^{-3} ⁄0,10001.

Примеры для упражнений

1. Выполнить сложение в модифицированном коде:

а) 0,101001+0,011011
б) 0,101011+0,010011
в) 0,110011+0,001111
г) 0,100001+0,011111,
д) 0,101010+0,000111,
е) 0,101110+0,010111,
ж) 0,100100+0,111011.

2. Выполнить сложение, используя в каждом случае дополнительный и обратный модифицированные коды:

а) 0,110001+(–0,001011);
б) (–0,010111)+(–0,000001);
в) (–0,001010)+(–0,011011);
г) (–0,100111)+0,001001;
д) (–0,101010)+(–0,011011);
е) (–0,011101)+(0,001010);
ж) (–0,100101)+ (0,010111).

3. Выполнить вычитание, используя в каждом случае дополнительный и обратный модифицированные коды:

а) 0,101011–0,001111,
б) 0,111011–0,101110,
в) 0,111011–0,110101,
г) (–0,101111) –0,101011,
д) 0,100110–0,011000.
е) 0,101110–0,010111,
ж) 0,100001–0,001110.

4. Выполнить умножение двух чисел x и y в прямом коде:

а) x=0,1101
y=0.0101;
б) x =0,1110
y=1,0011
в) x=0,1000
y=1.1010;
г) x=1,0111
y=1,1011;
д) x= 1,1011;
y=0,1101;
е) x=1,0010;
y=1,1111;
ж) x=0,0110;
y=1,1001.

Таблица 3

Варианты значений десятичных чисел для перевода в другие системы счисления и выполнения арифметических действий в двоичном коде

Вариант	Числа в десятичной системе счисления			
	для перевода		для арифметических действий	
1	65,28	1258	67	41
2	73.9	1931	75	39
3	82,625	2013	71	50
4	68,25	1240	68	42
5	92,3	1824	77	40
6	74,3125	2020	72	51
7	82,6	1351	69	43
8	69,125	1443	78	41
9	81,5625	1735	73	39
10	121,1	2015	70	44
11	99,27	1565	79	42
12	61,3	1191	74	40
13	109.25	1218	71	45
14	72.1	2018	80	43
15	98.9	1322	75	41
16	83,2	1979	72	39
17	77,8	2034	81	44
18	94,3	1463	76	42
19	86,7	1840	73	40
20	105,4	2025	68	45
21	65,6	1554	77	52
22	16,51	1741	74	41
23	74,3	1665	77	45
24	107,6	2035	78	43
25	83,5	1572	75	42
26	68,7	1726	69	47
27	92,4	1486	79	44
28	79,8	1813	76	58
29	101,125	2097	70	55
30	80,9	1907	80	49

Лекция 4

Физическое представление логических значений двоичных чисел электрическими сигналами. Элементарные (основные, базисные функции И, ИЛИ, НЕ)

Интегральная микросхема — это изделие, выполняющее определенную функцию преобразования и обработки сигналов, имеющее высокую плотность упаковки электрических элементов, рассматриваемых в итоге как единое целое.

Логический элемент интегральной микросхемы содержит группу электрически соединенных элементов, реализующих одну простейшую функцию алгебры логики.

Цифровая интегральная микросхема предназначена для преобразования и обработки сигналов, изменяющихся по закону дискретной функции.

Степень интеграции является показателем степени сложности интегральной микросхемы, характеризующимся числом элементов.

1-я степень сложности — 10 элементов и компонентов,
2-я — 10—100
3-я — 100—1000
4-я — 1000—10000
5-я — 10000—100000 и более

Различаются микросхемы разной степени интеграции:
Малой — 1—2-й степени сложности,
Средней — 2—3-й
Большой — 3—4-й
Сверхбольшой — 5-й

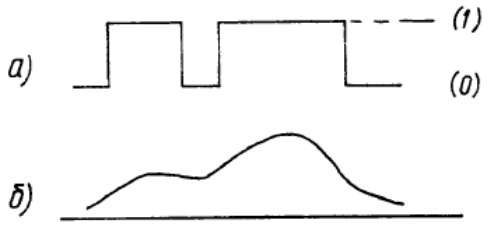


Рис. 1. Пример цифрового (а) и аналогового (б) сигналов

Что же такое цифровая техника? Эта отрасль техники(электроники), в которой сигналы, действующие в схемах, могут, как правило, иметь лишь два крайних дискретных уровня — высокий и низкий — в отличие от аналоговых сигналов, которые имеют произвольные уровни и изменяются непрерывно (рис. 1).

Элементысхем (лампы, транзисторы, диоды) работают как электрические ключи и находятся в одном из двух крайних состояний: пропускание (включение) или запирание (выключение).

Главные достоинства цифровой техники заключаются в высокой надежности и помехоустойчивости. Цифровые сигналы исключают ошибки при передаче и воспроизведении информации, содержащейся в них,

поскольку распознавание двух крайних уровней сигнала являетсянадежным даже при наличии больших искажений и помех.

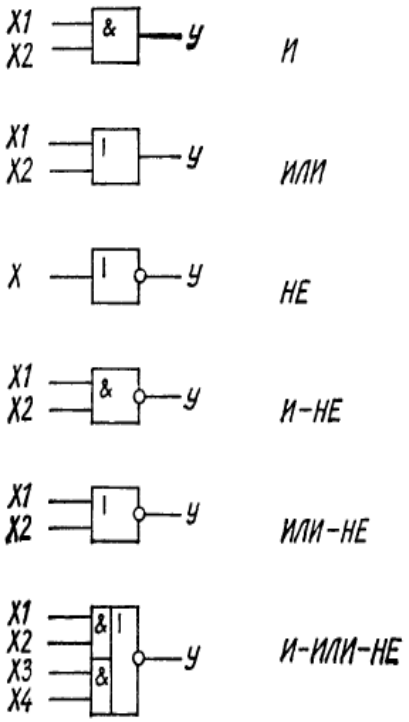


Рис. 2. Условные обозначения основных логических элементов

Рис. 3. Логический элемент И (а), его аналоги (б, в) и временные диаграммы (г)

В основе цифровой техники, как уже говорилось, лежитдвоичная система выражения цифр, именуемая еще бинарной, и связанный с ней математический аппарат — алгебра логики, или булевая алгебра, названная так в честь Д. Буля, английского математика XIX века, основоположника математической логики. В двоичной системе счисления удается любое число записать с помощью 1 или 0. Например, двоичное число 11 101011 соответствует 235. Каждая позиция числа, записанного в двоичной системе счисления, представляет одно из двух состояний (1 или 0).

Относительно электрических сигналов двоичная система счисления также соответствует двум состояниям или уровням: высокому (более положительному) и низкому (менее положительному нулевому или даже отрицательному)(рис. 1). Если высокий уровень рассматривать как 1, а низкий — 0, то в результате мы получим так называемую положительную логику. В случае отрицательной — высоким уровням присваивается 0, а низким— 1. Мы же будем пользоваться в дальнейшем только положительной логикой.

На практике невозможно выполнить условие, при которомвсе цифровые сигналы точно соответствовали бы одному из принятых уровней. Поэтому разрешаются некоторые допуски, так что правильнее было бы говорить о двух интервалах, в которых находятся сигналы. Логический ноль в этом случае для рассматриваемых в дальнейшем микросхем может иметь значение от —0,5 В до + 0,5 В, а логическая единица от + 2,4 В до +5,5 В.

Основу цифровой техники составляют логическиеэлементы. К ним относятся элементы, в которых существует определенная логическая связь между входными и выходными сигналами, принимающими значения логических ноля или единицы. Связь между сигналами определяется логической функцией. Для ее описания используется математическая логика. Для простоты анализа логических элементов или для уяснения ее функциональных возможностей служат таблицы состояний входных и выходных величин — сигналов. По этим таблицам можно построить и временные диаграммы работы логического элемента.

Всего логических элементов, применяемых в настоящее время в цифровой технике, около 30, включая и различные типы триггеров. Мы же изучим сначала наиболее распространенные из них: И, ИЛИ, НЕ, И-НЕ, ИЛИ-НЕ, И-ИЛИ-НЕ и триггер. На рис. 2 показаны эти логические элементы, из которых, как из кирпичиков, мы будем строить различные конструкции.Рассматривая логические элементы, мы прежде всегодолжны себе задавать вопрос: а что же происходитс тем или иным логическим элементом, когда на входе(или входах) будет 1 или 0?

Элемент И. Его изображение, электрический аналог,временная диаграмма работы показаны на рис. 3. Состояние входных и выходных сигналов изображены в табл. 1. Как уже догадался пытливый читатель, ***X1***и ***X2***— входные сигналы. В условном обозначении линии электрической связи, символизирующие входы, изображены слева, выход — справа. Отличительным признаком символа И является условный знак, заменяющий союз ≪И≫ в английском языке и помещенный в левом верхнемуглу прямоугольника. Этот элемент может иметь два иболее входов и один выход (рис. 3, а). Сигнал, соответствующий логической 1, появляется на выходе элемента только в том случае, если такие же сигналы поданы на все его входы. И, действительно, обратившись к электрическому аналогу, убеждаемся в том, что лампочка ***HL*** загорится тогда, и только тогда, когда будут замкнуты одновременно и контакты SA1, и SA2 (рис. 3, б). Взгляните, пожалуйста, на последнюю строчку табл. 1. Выходной сигнал, равный логической 1, будет тогда, когдаи X1, и ***X2***равны 1. Теперь ясно, что любой логический 0 на входе ячейки (логического элемента) даст на выходе тот же 0. Это еще раз можно увидеть на другом аналоге логического элемента (рис. 3, в). Замкнутое состояние контакта SA1 или ***SA2*** эквивалентно логическому 0.Если только какой-либо контакт замкнут, то напряжения на выходе с резистора этого аналога нет.

А как же будет выглядеть во времени действие сигналов?Это можно проследить по диаграмме (рис. 3, г).

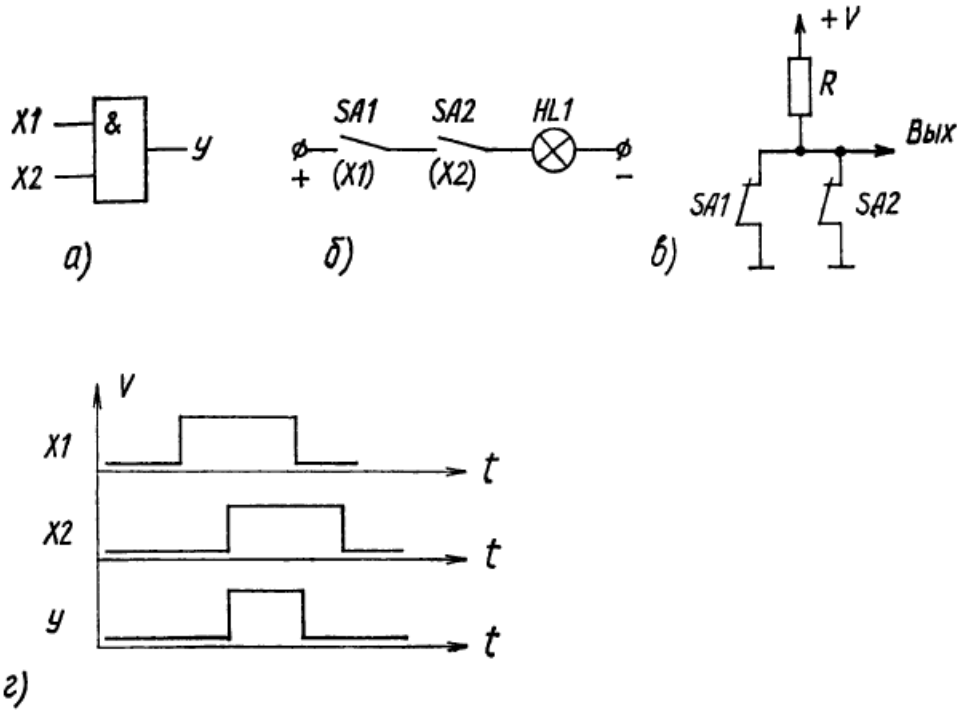


Рисунок 3 – Элемент «И»

Таблица 1

X1	X2	Y
0	0	0
1	0	0
0	1	0
1	1	1

Таблица 2

X1	X2	Y
0	0	0
1	0	1
0	1	1
1	1	1

Таблица 3

X	Y
0	1
1	0

Входной сигнал *XI* появляется первым и первым исчезает. Как только появится сигнал *X2*, тут же возникает выходной. Он будет существовать до тех пор, пока будут присутствовать оба сигнала на входе.

Надо заметить, что в электрических схемах от моментавоздействия какой-то электрической величины до моментареакции на нее проходит какое-то время. Для различныхэлементов оно различно. Поэтому выходная величинав логическом элементе микросхемы появится снекоторой задержкой, которая характеризует его частотныеили динамические характеристики. Однако пустьнас сейчас это не волнует, будем считать, что все происходитмгновенно.

Элемент ИЛИ. Здесь могут быть два или более входов.Сигнал на выходе появляется при поступлении управляющего сигнала хотя бы на один из его входов.В условном обозначении этого элемента (рис. 4, а) вместо упомянутого знака (союза≪И≫) используют 1. Взглянув на электрический аналог этого элемента'(рис. 4, б), становится ясным, что лампочка *HL*

будетгореть при замыкании любого из контактов *SA1* или *SA2.* Совершенно очевидным является и то, что в другом аналоге логического элемента (рис. 4, в) напряжение на резисторе будет иметь место, когда есть напряжение на входе *X1* , либо на входе *X2*, либо одновременно.

Это же самое подтверждается и таблицей истинности (табл. 2), а также временной диаграммой (рис. 4, г).

Как вы уже заметили, управляющими в обоих элементах являются логические единицы, хотя в первом случае их должно быть обязательно две (по числу входов),а здесь достаточно одной.

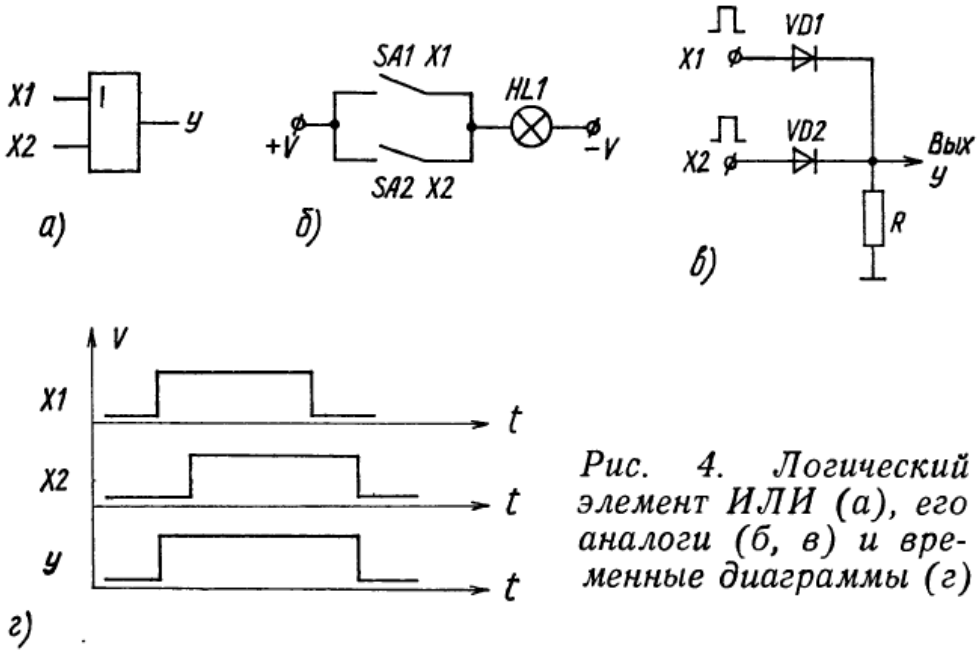


Рис. 4. Логический элемент ИЛИ (а), его аналоги (б, в) и временные диаграммы (г)

Элемент НЕ. Его чаще называют инвертором. В отличие от элементов Ии ИЛИ он имеет один вход. Как увидит дальше читатель, инверторомможет быть и другой логический элемент, у которого несколько входов.

Параллельное соединение их превращает, например,элемент И-НЕ в инвертор. Но это чуть впереди. А пока сигнал на выходе этого устройства существует только при отсутствии сигнала на входе и исчезает с его появлением. Стоп, стоп, читатель! Не опережай событие! Чтозначит отсутствует и какой сигнал? Дело в том, что сигнал, равный логическому нолю, тоже сигнал. Вот почему в практике отсутствие сигнала не значит, что вход никуда не подключен, ≪висит в воздухе≫, хотя вывод логического элемента припаян к печатной плате. Спешу заметить, что неприсоединение вывода логического элемента ни к какому другому для ТТЛ микросхем равносильно подаче на этот вывод логической 1.

Итак, инвертор дает на выходе 1 в том случае, когда на его входе низкий логический уровень — элемент ≪все наоборот≫. Он выполняет функцию логического отрицанияНЕ. Его обозначают небольшим кружочком в местеприсоединения линии, символизирующей выход устройства(рис. 5, а). Нет смысла долго останавливаться нааналогах этого логического элемента (рис. 5, б, в): сработаетреле — лампочка погаснет; откроется транзистор—

светодиод потухнет, так как транзистор своимоткрытым состоянием его зашунтирует. Временная диаграмма(рис. 5, г) и таблица истинности (табл. 3) настолькопросты, что не требуют пояснений.

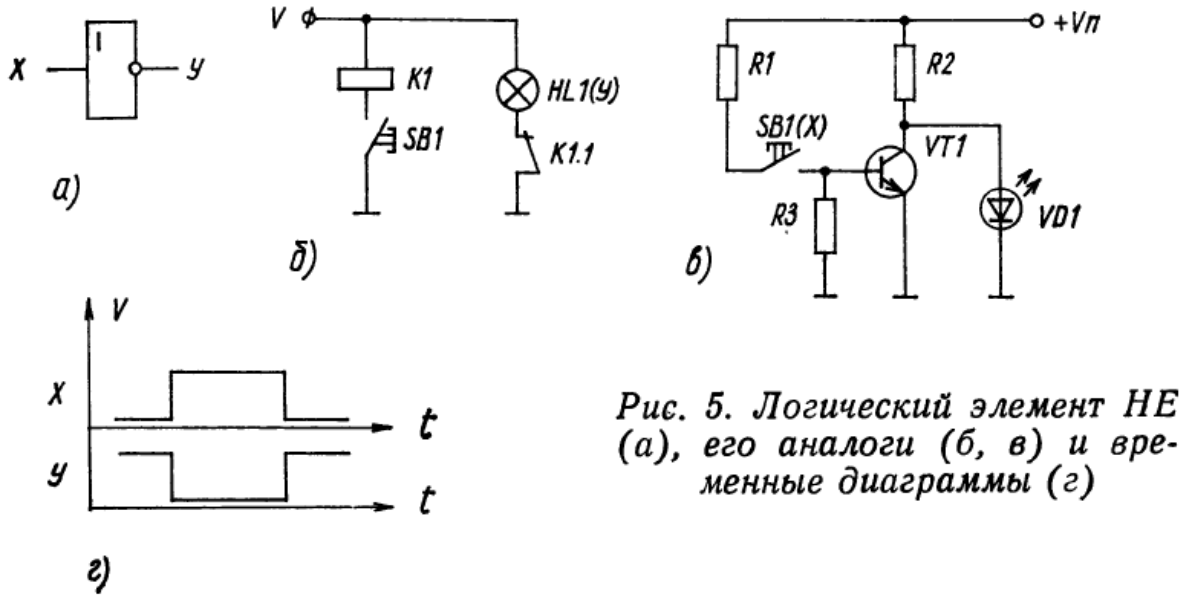


Рис. 5. Логический элемент НЕ (а), его аналоги (б, в) и временные диаграммы (г)

Элемент И-НЕ. Несколько более сложной зависимостьюсвязаны входные и выходные сигналы в логическом элементе И-НЕ, являющемся комбинацией элементовИ, НЕ. Здесь при наличии нолевых сигналов на входах или хотя бы одного нолевого сигнала на выходе -будет высокий логический уровень. Если на всех входах будет по 1, выходной сигнал становится равным 0. Это свойство логического элемента И-НЕ как элемента совпадения логических единиц с инверсией на выходе широко используется. Условное обозначение и его временнаядиаграмма изображены на рис. 6, а, г. В исходном состоянии лампочка на схеме электрического аналога (рис. 6, б) горит. Замкнув и SA1, и SA2, мы гасимее, т. е. выходная величина становится равной 0. Точно так же происходит в другом электрическом аналоге(рис. 6, в), если на вход закрытого транзистораподать положительный уровень и на вход ***X1,*** и на вход ***X2,*** то транзистор откроется, светодиод погаснет. Самое важное в этом элементе, а это хорошо виднои таблицы истинности (табл. 4) и на временной диаграмме,то, что при совпадении единичных сигналов выходной становится равным. 0. Условное обозначение элемента И-НЕ очень похоже на логический элемент И.

Единственное отличие — кружок, символизирующий логическое отрицание на выходе. Как уже было замечено выше, логический элемент И-НЕ может стать инвертором, если все входы этой ячейки объединить вместе.

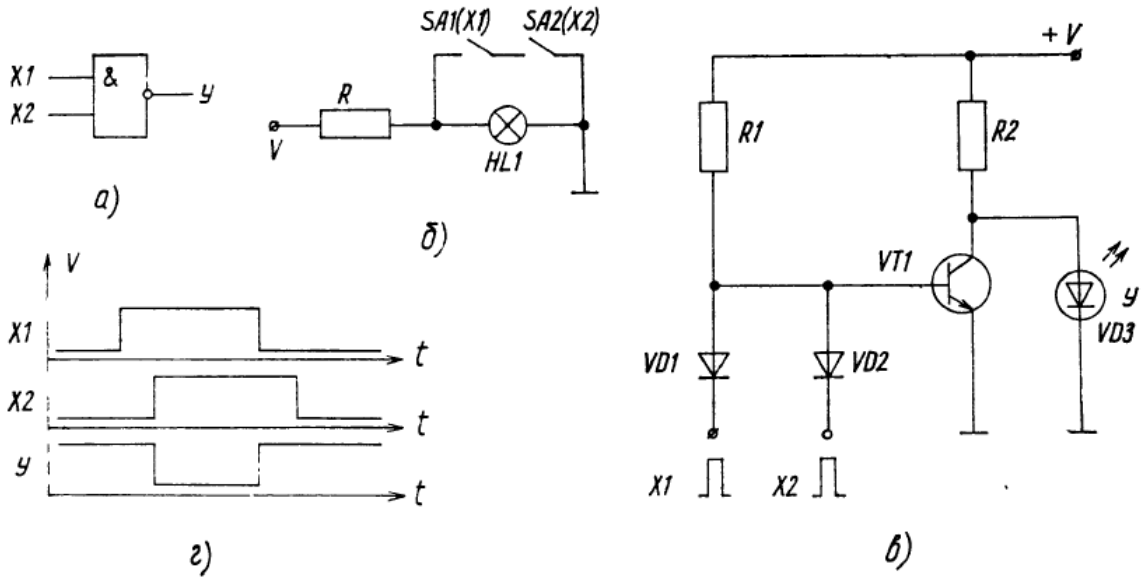


Рис. 6. Логический элемент И-НЕ (а), его аналоги (б, в) и временные диаграммы (г)

Таблица 4

X1	X2	Y
0	0	1
1	0	1
0	1	1
1	1	0

Таблица 5

X1	X2	Y
0	0	1
1	0	0
0	1	0
1	1	0

Элемент ИЛИ-НЕ. Условное обозначение логическогоэлемента ИЛИ-НЕ получают путем добавления кружочкак линии выхода (рис. 7, а). Единичный сигнал на любом входе всегда дает на выходе нолевой логический уровень. Это видно на временной диаграмме (рис. 7, г)- и из таблицы истинности (табл. 5). Для того чтобы выполнить логическую функцию ИЛИ-НЕ в электрическоманалоге этого элемента (рис. 7, б), контактыSA1 и SA2 должны быть соединены параллельно друг другу.А инверсия сигнала символизируется тем, что любоевключение контакта, эквивалентное логической 1, гаситлампочку. В другом электрическом аналоге (рис. 7, в) любой положительный уровень на входе заставляеттранзистор открыться и шунтировать светодиод, который при этом гаснет.

Как видно из таблицы истинности, логический элементИЛИ-НЕ может служить элементом совпадениялогических нолей.

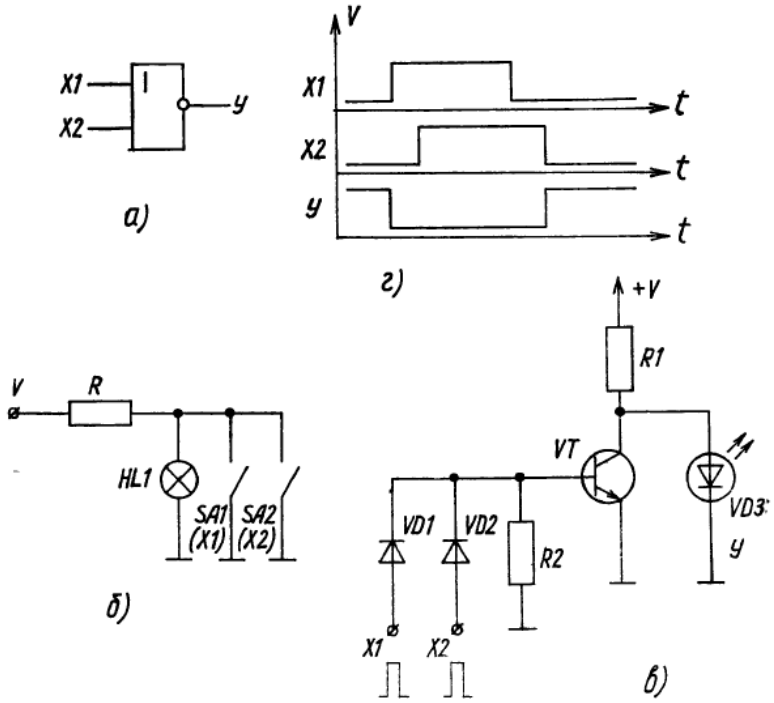


Рис. 7. Логический элемент ИЛИ-НЕ (а), его аналоги (б, в) и временные диаграммы (г)

Еще более сложно связаны входные и выходные сигналыьв комбинированных логических элементах типа И-ИЛИ, ИЛИ-И, И-ИЛИ-НЕ и др. Условные графическиеобозначения построены на основе прямоугольника, разделенного на два поля (рис. 8, а) : основное (правое) и вспомогательное (левое). Во вспомогательном поле напротив первого (сверху) входа помещают условный знак-метку, символизирующую первую логическую операцию, а в основном — знак-метку второй операции. Зная это, нетрудно расшифровать принцип действия, например, комбинационного элемента И-ИЛИ-НЕ.

Элемент И-ИЛИ-НЕ. Сигнал на выходе этого элемента(рис. 8, а) будет выработан при условии, чтовходные сигналы поданы одновременно на один из входовкаждой группы. И, действительно, просмотрев аналоги(рис. 8, б, в), убеждаемся, что достаточно соединитьцепь через любой из контактов пары XI или ***X2,***Х3 или ***X4,*** как лампочка погаснет. Ввиду того что контактов четыре, то количество комбинаций состояний изних будет 16. Это наглядно отображено в таблице состояний(табл. 6).

Работа этого логического элемента и двух предыдущих(И-НЕ, ИЛИ-НЕ) убеждает нас в том, что логическийноль — тоже сигнал, и не менее важный в работеэтих элементов. Мало того, в логическом элементе И-НЕон (О) играет роль блокирующего сигнала. Иначе говоря, ничто не изменится с выходным сигналом, пока наодном из входов будет логический ноль.
Временная диаграмма элемента И-ИЛИ-НЕ выглядитболее сложно (рис. 8, г), однако, сопоставляя таблицусостояний и выходные сигналы, можно найти простыезакономерности, доступные для понимания работыэтого элемента.

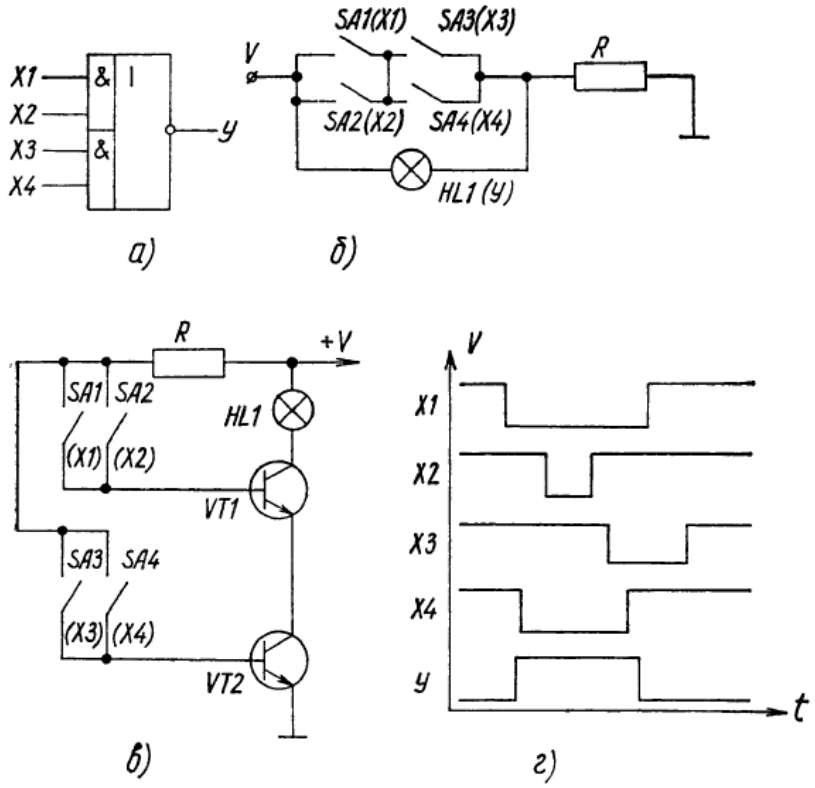


Рис. 8. Логический элемент И-ИЛИ-НЕ (а), его аналоги (б, в) и временные диаграммы (г)

А теперь осталось повторить названия наиболее распространенныхлогических элементов и дополнить ихбуквенными символами маркировки. И-ЛИ, ИЛИ-ЛЛ, НЕ-ЛН, И-НЕ-ЛА, ИЛИ-НЕ-ЛЕ, И-ИЛИ-НЕ-ЛР.

Назовем обозначения других цифровых устройств, которые в дальнейшем будут встречаться на пути освоения микросхем. Триггеры — ТВ, ТМ; регистры — ИР;,сумматоры — ИМ; счетчики — ИЕ, шифраторы — ИВ;,дешифраторы — ИД.

Лекция 5
Основы аналитического и графического способов минимизации функций.

Алгебра логики (алгебра высказываний) — раздел математической логики, в котором изучаются логические операции над высказываниями. Алгебра логики – это раздел математики, возникший в XIX веке благодаря усилиям английского математика Дж. Буля.
Законы и аппарат алгебры логики используются при проектировании различных частей компьютеров (память, процессор).

Логическое высказывание - любое повествовательное предложение, в отношении которого однозначно можно сказать, истинно оно или ложно.
Пример 1: предложение «6 - четное число» является высказыванием, т.к. оно истинное.

Существуют предложения, в которых для выяснения истинности или ложности требуются дополнительные сведения. Такие предложения являются высказывательными формами.

Высказывательная форма - повествовательное предложение, которое прямо или косвенно содержит хотя бы одну переменную и становится высказыванием, когда все переменные замещаются своими значениями.
Пример 2: предложение «площадь поверхности Индийского океана равна 75 млн.км2» - и истинно (значение приближенное, приемлемо на практике) и ложно (указанное значение неточное)
Из логических высказываний составляются логические формулы.
Например, $X \to Y = \overline{X} \vee Y$.

Синтез логического устройства распадается на несколько этапов. На первом этапе требуется функцию, заданную в словесной, табличной и других формах, представить в виде логического выражения с использованием некоторого базиса. Дальнейшие этапы сводятся к получению минимальных форм функций, обеспечивающих при синтезе наименьшее количество электронного оборудования и рациональное построение функциональной схемы устройства.

Для начального представления функции обычно используется базис И, ИЛИ, НЕ независимо от того, какой базис будет использован для построения логического устройства.

Исходными, из соображений удобства последующих преобразований, приняты следующие две канонические формы представления функций: совершенная дизъюнктивная нормальная форма (СДНФ) и совершенная конъюнктивная нормальная форма (СКНФ).

Функция в СДНФ представляет собой дизъюнкцию конституант единицы и имеет вид

$$f_{\text{СДНФ}} = \sum_{i=1}^{n} \text{КЕ}_i,$$

где КЕ — конституанта единицы (минтерма) — функция, равная «1» и представляющая собой конъюнкцию входных аргументов, причем, если аргумент равен «0», то его записывают с инверсией;

n — количество КЕ.

Функция в СКНФ представляет собой конъюнкцию дизъюнктуант нуля и имеет вид

$$f_{\text{СКНФ}} = \prod_{i=1}^{n} \text{ДН}_i,$$

где ДН — дизъюнктуант нуля (макстерма) — функция, равная «0» и представляющая собой дизъюнкцию входных аргументов, причем, если аргумент равен «1», то его записывают с инверсией.

Пример. По заданной ТИ (табл. 2.12) записать функции в СДНФ, СКНФ.

1. Найдем $\text{КЕ}_1 = \bar{A} \cdot \bar{B}$, $\text{КЕ}_2 = A \cdot B$, запишем $f_{\text{СДНФ}} = (A \vee \bar{B}) \cdot (\bar{A} \vee B)$.

2. Найдем $\text{ДН}_1 = A \vee \bar{B}$, $\text{ДН}_2 = \bar{A} \vee B$, запишем $f_{\text{СКНФ}} = (A \vee \bar{B}) \cdot (\bar{A} \vee B)$.

Таблица 2.12

Исходные данные

A	B	f
0	0	1
0	1	0
1	0	0
1	1	1

7.3. Тождества и законы алгебры логики

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

$$\begin{gathered} A \vee 1 = 1; \quad A \cdot 1 = A; \quad A \vee \bar{A} = 1; \\ A \vee 0 = A; \quad A \cdot 0 = 0; \quad A \cdot \bar{A} = 0; \\ A \vee A \vee A \vee \ldots \vee A = A; \\ A \cdot A \cdot A \cdot \ldots \cdot A = A. \end{gathered} \tag{7.1}$$

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):

$$\begin{aligned} &1)\ (A \vee B) \cdot C = AC \vee BC; \\ &2)\ A \cdot B \vee C = (A \vee C) \cdot (B \vee C), \end{aligned} \tag{7.2}$$

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:

$$\bar{\bar{A}} = A.$$

Закон инверсии или правило де Моргана (двойственности):

$$1)\ \overline{A \vee B} = \bar{A} \cdot \bar{B}; \tag{7.3}$$

$$2)\ \overline{A \cdot B} = \bar{A} \vee \bar{B}. \tag{7.4}$$

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

$$A \vee B = \overline{\bar{A} \cdot \bar{B}}; \tag{7.5}$$

$$A \cdot B = \overline{\bar{A} \vee \bar{B}}. \tag{7.6}$$

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

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

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

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

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

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

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

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

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

$= AC \vee B\left(C \vee A\right)\overset{1}{\cancel{(C \vee \bar{C})}} = AC \vee BC \vee AB.$

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

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

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

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:

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

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

Контрольные вопросы

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

Контрольные задания

1. Упростить следующие выражения:

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Минимизацией называют процедуру такого упрощения логической функции, чтобы ее реализация имела наименьшее число логических элементов. Это экономит элементную базу и время монтажа, снижает стоимость построения логического устройства при обеспечении заданного уровня надежности. Наиболее используемые методы получения МДНФ или МКНФ — метод карт Вейча, метод карт Карно, метод Квайна.

Метод Квайна содержит два этапа преобразования выражения функции: на первом этапе осуществляется переход от СДНФ или СКНФ к сокращенной форме, на втором этапе — переход от сокращенной формы логического выражения к минимальной форме.

Порядок нахождения функции в МДНФ методом Квайна.

1. Записать по заданным значениям $f(A,B,C)$ в ТИ $f_{\text{СДНФ}}$;

2. По найденной $f_{СДНФ}$ найти члены сокращенной формы функции (простые импликанты) по следующему принципу: посредством операции склеивания выделить простые импликанты, получаемые путем сравнения каждого члена функции f с каждым из последующих, т.е $КЕ_1$ сравнить с $КЕ_2$, с $КЕ_3$, с $КЕ_i$; $КЕ_2$ сравнить с $КЕ_3$, с $КЕ_i$; и т.д. Таким образом выявить пары членов, различающиеся лишь тем, что один из аргументов в одном из членов представлен без инверсии, а в другом — с инверсией.
3. Составить импликантную матрицу (табл. 2.13); отметить крестиками столбцы членов СДНФ, поглощаемых отдельными простыми импликантами.

Таблица 2.13

Заголовок импликантной матрицы

Простые импликанты	Члены СДНФ		
	$КЕ_1$	...	$КЕ_i$
1	2	...	n

4. Записать $f_{МДНФ}$ сумму простых импликант, составляющих ядро функции (импликанты, для каждой из которых имеете^ хотя бы один столбец, перекрываемый только ею) и добавить импликанты, не входящие в ядро, но обеспечивающие перекрытие всех столбцов импликантной матрицы.

Пример. Минимизировать ФАЛ, заданную ТИ (табл. 2.14) методом Квайна

Таблица 2.14

Переключательная таблица

Аргументы	Номера наборов аргументов							
	0	1	2	3	4	5	6	7
C	0	1	0	1	0	1	0	1
B	0	0	1	1	0	0	1	1
A	0	0	0	0	1	1	1	1
$f(A, B, C)$	1	0	1	0	0	1	1	1

1. Запишем $f_{СДНФ}$:

$$f_{СДНФ} = КЕ_1 \vee КЕ_2 \vee КЕ_3 \vee КЕ_4 \vee КЕ_5 =$$
$$= \overline{A} \cdot \overline{B} \cdot \overline{C} + \overline{A} \cdot B \cdot \overline{C} + A \cdot \overline{B} \cdot C + A \cdot B \cdot \overline{C} + A \cdot B \cdot C.$$

2. Найдем члены сокращенной формы функции. Сравниваем KE_i:

KE_1 с KE_2: $\overline{A}\cdot\overline{B}\cdot\overline{C}+\overline{A}\cdot B\cdot\overline{C}=\overline{A}\cdot\overline{C}$.

KE_2 с KE_4: $\overline{A}\cdot B\cdot\overline{C}+A\cdot B\cdot\overline{C}=B\cdot\overline{C}$.

KE_3 с KE_5: $A\cdot\overline{B}\cdot C+A\cdot B\cdot C=A\cdot C$.

KE_4 с KE_5: $A\cdot B\cdot\overline{C}+A\cdot B\cdot C=A\cdot B$.

Сокращенная форма функции будет иметь вид $f=\overline{A}\cdot\overline{C}+B\cdot\overline{C}+A\cdot C+A\cdot B$.

3. Составим импликантную матрицу (табл. 2.15).

Таблица 2.15

Импликантная матрица для СДНФ

Простые импликанты (п. 2)	Члены СДНФ (п. 1)				
	$\overline{A}\cdot\overline{B}\cdot\overline{C}$	$\overline{A}\cdot B\cdot\overline{C}$	$A\cdot\overline{B}\cdot C$	$A\cdot B\cdot\overline{C}$	$A\cdot B\cdot C$
$\overline{A}\cdot\overline{C}$	+	+			
$B\cdot\overline{C}$		+		+	
$A\cdot C$			+		+
$A\cdot B$				+	+

4. Ядро составляют импликанты $\overline{A}\cdot\overline{C}$ и $A\cdot C$ (только ими перекрываются первый и третий столбцы матрицы). Дополнительно требуется перекрытие четвертого столбца матрицы, что можно сделать любой из импликант $B\cdot\overline{C}$ или $A\cdot B$. Функция примет вид:

$$f_{\text{МДНФ}}=\overline{A}\cdot\overline{C}+B\cdot\overline{C}+A\cdot C.$$

Метод Вейча обеспечивает простоту получения результатов. Используется при минимизации относительно несложных функций (с числом аргументов до пяти). Карта Вейча (рис. 2.13) представляет собой определенную форму ТИ. Каждая клетка карты соответствует некоторому набору значений аргументов. Число клеток карты равно числу всех возможных наборов значений аргументов 2^n *(где n-*число аргументов функций). Цифры внутри ячеек означают номера наборов аргументов в ТИ.

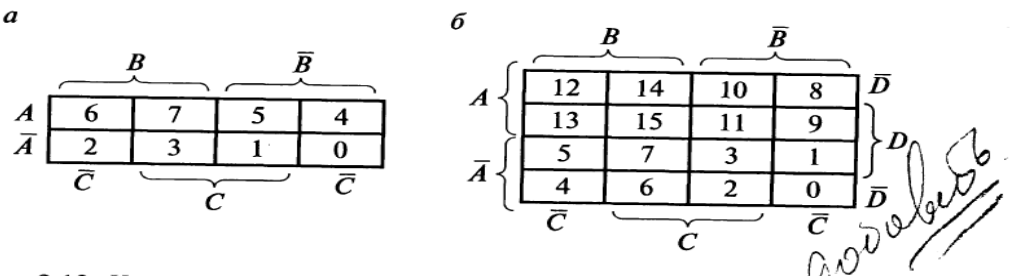


Рис. 2.13. Карты Вейча: *а* — трех аргументов; *б* — четырех аргументов

Порядок нахождения функции в МДНФ методом Вейча:

1. Все клетки ДВ, содержащие «1», объединить в замкнутые области Допустимое число клеток в области может быть равно 1, 2, 4, 8, ... Области могут пересекаться и одни и те же клетки могут входить в разные области.
2. Для каждой замкнутой *i*-й области записать функцию f_i, составленуюкак конъюнкция лишь тех аргументов (рис. 2.13, б), которые не меняют своего значения в пределах соответствующей области.
3. Записать $f_{МДНФ}$ как дизъюнкцию (сумму) найденных f_i.

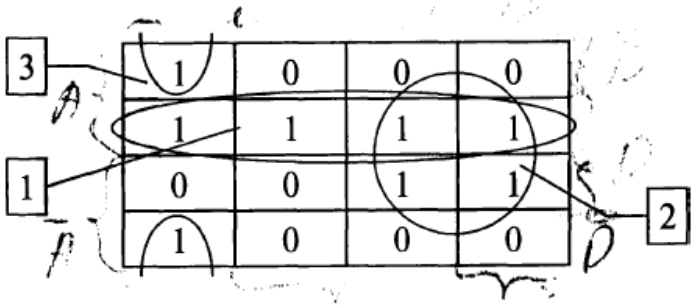
Пример. По заданной диаграмме Вейча (табл. 2.16) найти $f_{МДНФ}$.

Таблица 2.16

Карта Вейча

1	0	0	0
1	1	1	1
0	0	1	1
1	0	0	0

1. Объединим в ДВ «1» в замкнутые области, получим три объединения:



2. Запишем функции f_i для трех замкнутых областей:

$$f_1 = A \cdot D;\ f_2 = \overline{B} \cdot D;\ f_3 = B \cdot \overline{C} \cdot \overline{D}.$$

3. Запишем МДНФ функции:

$$f_{МДНФ} = f_1 \vee f_2 \vee f_3 = A \cdot D + \overline{B} \cdot D + B \cdot \overline{C} \cdot \overline{D}.$$

Порядок нахождения функции в МКНФ методом Вейча:

1. Все клетки ДВ, содержащие «0», объединить в замкнутые области.
2. Для каждой замкнутой *i*-й области записать функцию f_i составленую как дизьюнкциялишь тех инверсий аргументов (рис. 2.13, б),которые не меняют своего значения в пределах соответствующей области.
3. Записать $f_{МКНФ}$ как дизъюнкцию (сумму) найденных f_i.

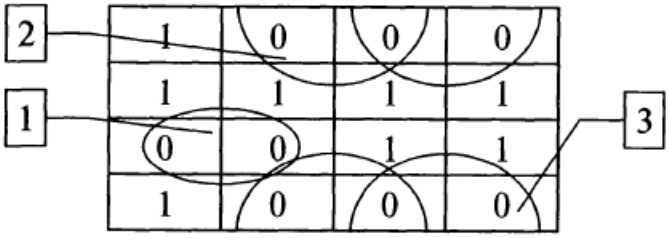
Пример. По заданной карте Вейча (табл. 2.17) найти МКНФ функции *f*.

Таблица 2.17

Карта Вейча

1	0	0	0
1	1	1	1
0	0	1	1
1	0	0	0

1. Объединим в ДВ «0» в замкнутые области, получим три объединения:



2. Запишем функции f_i для трех замкнутых областей:

$$f_1 = A + \overline{B} + \overline{D};\ f_2 = \overline{C} + D;\ f_3 = B + D.$$

3. Запишем МКНФ функции:

$$f_{\text{МКНФ}} = f_1 \wedge f_2 \wedge f_3 = \left(A + \overline{B} + \overline{D}\right) \cdot \left(\overline{C} + D\right) \cdot \left(B + D\right).$$

Метод Карно (табл. 2.18) отличается от метода Вейча способом обозначения строк и столбцов ТИ. Аргументы функции делятся на две группы, комбинации значений аргументов одной группы приписываются столбцам таблицы, комбинации значений аргументов другой группы — строкам.

Столбцы и строки обозначаются комбинациями, соответствующими последовательности чисел в коде Грея (это сделано для того, чтобы склеивающиеся клетки находились рядом). Обозначения столбца и строки, на пересечении которых находится клетка таблицы, образуют набор, значение функции на этом наборе записывается в клетку. Цифры внутри ячеек табл. 2.18 означают номера

Таблица 2.18

Карты Карно

а

AB / *C*	00 $(\bar{A}\bar{B})$	01 $(\bar{A}B)$	11 (AB)	10 $(A\bar{B})$
0 $(\bar{C})$	0	2	6	4
1 (C)	1	3	7	5

б

AB / *CD*	00 $(\bar{A}\bar{B})$	01 $(\bar{A}B)$	11 (AB)	10 $(A\bar{B})$
00 $(\bar{C}\bar{D})$	0	4	12	8
01 $(\bar{C}D)$	1	5	13	9
11 (CD)	3	7	15	11
10 $(C\bar{D})$	2	6	14	10

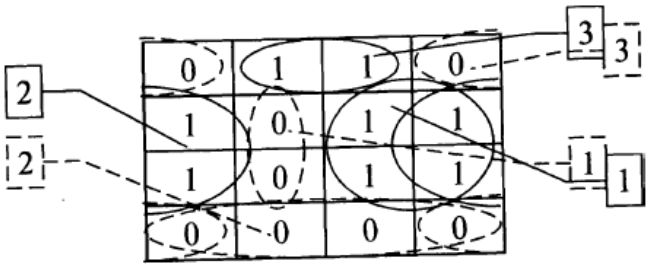
Пример. По заданной ТИ (табл. 2.19) составить карту Карно (КК) и найти $f_{МДНФ}$, $f_{МКНФ}$.

Таблица 2.19

Переключательная таблица

Аргументы	Цифры (номера наборов аргументов)															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
D	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
C	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
B	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
A	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
f(*A*, *B*, *C*, *D*)	0	1	0	1	1	0	0	0	0	1	0	1	1	1	0	1

1. Карта Карно с объединениями «1» и «0 » имеет вид:



2. По составленной КК запишем МДНФ функции *f*:
– объединим «1» в замкнутые области, получим три объединения;
– запишем функции f_i для трех замкнутых областей:

$$f_1 = A \cdot D;\ f_2 = \overline{B} \cdot D;\ f_3 = B \cdot \overline{C} \cdot \overline{D};$$

– запишем МДНФ функции:

$$f_{\text{МДНФ}} = f_1 \vee f_2 \vee f_3 = A \cdot D + \overline{B} \cdot D + B \cdot \overline{C} \cdot \overline{D}.$$

3. По составленной КК запишем МКНФ функции *f*:
– объединим «0» в замкнутые области, получим три объединения;
– запишем функции f_i для трех замкнутых областей:

$$f_1 = A + \overline{B} + \overline{D};\ f_2 = \overline{C} + D;\ f_3 = B + D;$$

– запишем МКНФ функции:

$$f_{\text{МКНФ}} = f_1 \wedge f_2 \wedge f_3 = \left(A + \overline{B} + \overline{D}\right) \cdot \left(\overline{C} + D\right) \cdot \left(B + D\right).$$

Синтез логических устройств с несколькими выходами

Логические схемы с несколькими выходами могут быть заданы в виде системы ФАЛ или ТИ. Схема, реализующая систему функций, составляется в простых и минимальных базисах. Она состоит из заданного числа аргументов, логических элементов и заданного количества выходных функций в системе.

Пример. Минимизировать систему функций, используя метод Квайна.

$$f_1 = \overline{A} \cdot \overline{B} \cdot \overline{C} + \overline{A} \cdot B \cdot \overline{C} + A \cdot \overline{B} \cdot \overline{C} + A \cdot B \cdot \overline{C} + A \cdot B \cdot C;$$

$$f_2 = \overline{A} \cdot \overline{B} \cdot \overline{C} + \overline{A} \cdot \overline{B} \cdot C + A \cdot \overline{B} \cdot \overline{C} + A \cdot B \cdot \overline{C};$$

$$f_3 = \overline{A} \cdot B \cdot \overline{C} + \overline{A} \cdot B \cdot C + A \cdot \overline{B} \cdot \overline{C} + A \cdot \overline{B} \cdot \overline{C} + A \cdot \overline{B} \cdot C.$$

1. Запишем члены СДНФ и импликанты, укажем функции, содержащие названные элементы (табл. 2.20, табл. 2.21).

Таблица 2.20

Таблица конституант единицы

Конституанты	Функции
$\overline{A}\cdot\overline{B}\cdot\overline{C}$	f_1, f_2
$\overline{A}\cdot\overline{B}\cdot C$	f_2
$\overline{A}\cdot B\cdot\overline{C}$	f_1, f_3
$\overline{A}\cdot B\cdot C$	f_3
$A\cdot\overline{B}\cdot\overline{C}$	f_1, f_2, f_3
$A\cdot\overline{B}\cdot C$	f_3
$A\cdot B\cdot\overline{C}$	f_1, f_2
$A\cdot B\cdot C$	f_1

Таблица 2.21

Таблица импликант

Импликанты	Функции
$\overline{A}\cdot\overline{B}$	f_2
$\overline{B}\cdot\overline{C}$	f_1, f_2
$\overline{A}\cdot\overline{C}$	f_1
$\overline{A}\cdot B$	f_3
$B\cdot\overline{C}$	f_1
$A\cdot\overline{B}$	f_3
$A\cdot\overline{C}$	f_1, f_2
$A\cdot B$	f_1

2. Составим импликантную таблицу (табл. 2.22) и проведем операции поглощения.

Таблица 2.22

Импликантная таблица для системы функций

Импликанты (функции)	Конституанты							
	$\bar{A}\cdot\bar{B}\cdot\bar{C}$	$\bar{A}\cdot\bar{B}\cdot C$	$\bar{A}\cdot B\cdot\bar{C}$	$\bar{A}\cdot B\cdot C$	$A\cdot\bar{B}\cdot\bar{C}$	$A\cdot\bar{B}\cdot C$	$A\cdot B\cdot\bar{C}$	$A\cdot B\cdot C$
	f_1, f_2	f_2	f_1, f_3	f_3	f_1, f_2, f_3	f_3	f_1, f_2	f_1
$\bar{A}\cdot\bar{B}(f_2)$	+	+						
$\bar{B}\cdot\bar{C}(f_1, f_2)$	+ +				++			
$\bar{A}\cdot\bar{C}(f_1)$	+		+					
$\bar{A}\cdot B(f_3)$			+	+				
$B\cdot\bar{C}(f_1)$			+				+	
$A\cdot\bar{B}(f_3)$					+	+		
$A\cdot\bar{C}(f_1, f_2)$					++		+ +	
$A\cdot B(f_1)$							+	+
$\bar{C}(f_1)$	+		+		+		+	

3. В состав ФАЛ вводим импликанты, составляющие ядро функции, т.е. перекрывающие столбцы таблицы в единственном числе. Затем запишем импликанты, перекрывающие наибольшее количество столбцов, и в последнюю очередь в состав функций включим оставшиеся не поглощенными конституанты.

$$f_1 = A\cdot B + \bar{C};\ f_2 = \bar{A}\cdot\bar{B} + A\cdot\bar{C};\ f_3 = \bar{A}\cdot B + A\cdot\bar{B}.$$

Схема состоит из трех элементов «НЕ», пяти — «И», трех — «ИЛИ».

Синтез неполностью заданных логических функций

Используют методы карт Карно и Вейча. Для функции составляется диаграмма, где на запрещенных наборах аргументов функции задют такие значения, при которых клетки со значениями «1» и «0» охватываются минимальным числом областей с максимальным числом клеток в каждой области.

Пример. По заданной ТИ (табл. 2.23) минимизировать функцию.

Таблица 2.23

Переключательная таблица

I	0	0	0	0	1	1	1	1
II	0	0	1	1	0	0	1	1
C	0	1	0	1	0	1	0	1
f	1	1	0		1		1	

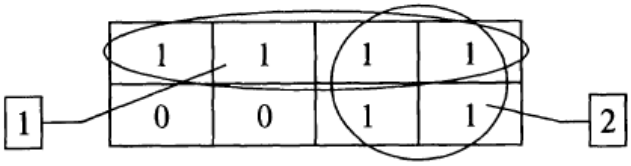
1. Запишем функцию $f_{\text{СДНФ}}$:

$$f_{\text{СДНФ}} = \overline{A}\cdot\overline{B}\cdot\overline{C} + \overline{A}\cdot\overline{B}\cdot C + A\cdot\overline{B}\cdot\overline{C} + A\cdot B\cdot\overline{C}.$$

2. Составим карту Вейча:

1	*	*	1
0	*	1	1

3. Заменим * удобными для склеивания значениями и объединим «1»:



4. Запишем $f_{\text{МДНФ}} = A + \overline{B}$.

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

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

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

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

Лекция 6

Общие сведения о цифровых интегральных микросхемах (ЦИМС) и область их применения.

Классификация интегральных микросхем

По функциональному назначению все ИМС делятся на два класса: цифровые и аналоговые.

Цифровые интегральные микросхемы (ЦИМС) предназначены для обработкиинформации, представленной в виде цифровых кодов. Характерной особенностью ЦИМС является то, что в виде цифровых кодов представлены и входные, и выходные сигналы. По этому признаку аналого-цифровые и цифроаналоговые преобра-зователи относятся к классу аналоговых ИМС.

Внутри каждого класса ИМС принята более детальная классификация микросхем по функциональному назначению и по целому ряду других признаков.

По функциональному назначению ЦИМС разделяют на подгруппы (логические элементы, триггеры и др.) и виды внутри подгрупп (триггеры: счетные, универсальные, Шмитта и т. д.).

По способу представления двоичной информации цифровые интегральные микросхемы подразделяют на импульсные, динамические, потенциальные. В потенциальных цифровых схемах значения «0» и «1» представляются двумя различными уровнями электрического потенциала: высоким и низким. Для потенциальных элементов используют понятия положительной и отрицательной логики, которые отражают принятый способ кодирования двоичных цифр. При положительной логике высокий уровень электрического потенциала соответствует логической единице, а низкий — логическому нулю. При отрицательной логике высокий уровеньэлектрического потенциала соответствует логическому нулю, а низкий — логической единице.

В импульсных цифровых схемах одно из значений логического сигнала («0»или «1») определяется наличием импульсов определенной длительности и амплитуды, а другое значение — отсутствием импульсов, то есть сохранением какого-либо постоянного потенциала. При положительной логике отсутствие импульсов соответствует логическому «0», а наличие — «1».

В динамических цифровых схемах логическая «1» представляется пачкой импульсов или возобновляемым через необходимый интервал времени потенциалом,а логический «0» — отсутствием импульсов (или наоборот).

В основе классификации цифровых микросхем *по типу логики* лежит принцип схемотехнического построения базового логического элемента серии микросхем. Потенциальные цифровые микросхемы, которые являются наиболее распространенными, по типу логики подразделяют на следующие

основные классы: *транзисторно-транзисторной логики* (ТТЛ) и *транзисторно-транзисторнойлогики с диодами Шоттки*(ТТЛШ), *логики на комплементарных МДП-транзисторах* (КМДП, КМОП), *на МДП-транзисторах с каналом n-типа* (*n*-МДП, *n*-МОП) и *на полевых транзисторах с затворомШоттки на основе арсенида галлия*(ПТШ-GaAs).

Основные показатели ЦИМС

Степень интеграции ЦИМС характеризуют коэффициентом компонентной интеграции $k_к$ и коэффициентом функциональной интеграции $k_ф$.

Коэффициент компонентной интеграции определяется выражением:

$$k_к = \lg N_к,$$

где $N_к$ — общее число элементов и компонентов, расположенных на кристалле,и характеризует, главным образом, уровень технологической сложности микросхемы. По величине коэффициента компонентной интеграции различают: ИМС
первой степени интеграции, если $k_к \leqslant 1$; ИМС второй степени интеграции, если $k_к \leqslant 2$; ИМС третьей степени интеграции, если $k_к \leqslant 3$; ИМС четвертой степени интеграции, если $k_к \leqslant 4$; ИМС пятой степени интеграции, если $k_к > 4$.
Для определения функциональной сложности ЦИМС используется ***коэффициент функциональной интеграции:***

$$k_ф = \lg N_э,$$

где $N_э$ — количество логических элементов И-НЕ либо ИЛИ-НЕ, расположенных на кристалле микросхемы. Если в качестве элементной базы используются другие логические элементы, то величина $N_э$ определяется числом элементов И-НЕ либо ИЛИ-НЕ, требуемых для реализации эквивалентной логической функции микросхемы. По величине коэффициента функциональной интеграции различают: малыеинтегральные схемы (МИС), содержащие один или несколько логических элементов, когда $k_ф \leqslant 1$ (триггер); средние интегральные схемы (СИС), содержащие один или несколько функциональных узлов, когда $k_ф \leqslant 2$ (счетчик, регистр, сумматор);
большие интегральные схемы (БИС), содержащие одно или несколько функциональных устройств, когда $2 \leqslant k_ф \leqslant 4$ (АЛУ, ЗУ); сверхбольшие интегральные схемы (СБИС), имеющие $k_ф > 4$ и выполняющие функции целых цифровых систем(микро-ЭВМ).

Для оценки сложности ЦИМС используется параметр, называемый ***«плотностью упаковки»*** $\gamma = N_к/V$, где V — объем кристалла без выводов.

Параметры ЦИМС

Характеристики ЦИМС делятся на статические и динамические.

Статические характеристики представляют собой зависимости между входными и выходными токамии напряжениями в установившемся режиме работы.

Динамические характеристики определяют поведение микросхем в переходных режимах, то есть при переключении из одного состояния в другое.

Схемотехнические и конструктивные параметры

Коэффициент kоб объединения по входу логического элемента *— число входов логического элемента, по которым реализуется логическая функция, в том числе с учетом входов логических расширителей.*

Для элементов многоступенчатой логики различают коэффициент объединенияпо логической функции ИЛИ $k_{об.\ или}$ и коэффициент объединения по логической функции И$k_{об.\ и.}$

Коэффициент kраз разветвления по выходу логического элемента (нагрузочная способность) *— число единичных нагрузок, которые можно одновременно подключить к выходу логического элемента.*

Единичной нагрузкой является один вход базового логического элемента данной серии ЦИМС. С увеличением числа нагрузок параметры ЦИМС ухудшаются. Допустимое количество входов элементов другой серии специально оговаривается.

Особенности выходных каскадов цифровых микросхем

Часто возникает необходимость подключения выходов нескольких цифровых микросхем к одной нагрузке. Одним из способов объединения выходов является использование в выходных каскадах микросхем транзисторов, один из выводов которых (коллектор, эмиттер, сток, исток) никуда не подключен. Такой вывод называют открытым.

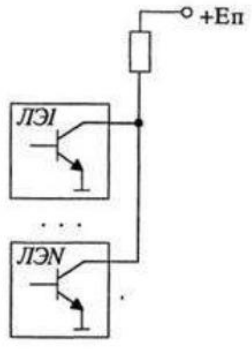


Рис. 3.25 Рис 7

Покажем схематически (рис.7), как объединяются выходы микросхем с открытым коллектором. Такую схему называют «монтажным (проводным)

ИЛИ».
Если открытым является коллектор транзистора *n-p-n*-типа, эмиттер транзистора*p-n-p*-типа, сток транзистора с каналом n-типа, исток транзистора с каналом p-типа, то вывод обозначают символом . Если открытым является коллектор транзистора p-n-p-типа, эмиттер транзистора n-p-n-типа, сток транзистора с каналом *p*-типа, исток транзистора с каналом n-типа, вывод обозначают символом .
Выходные каскады некоторых микросхем могут работать в таком режиме, когда микросхема оказывается фактически отключенной от нагрузки. Это так называемое третье (высокоимпедансное) состояние микросхемы. Использование третьего состояния является еще одним способом объединения выходов микросхем, который широко используется в вычислительной технике, при подключении к общей шине многих устройств. Приведем фрагмент схемы, поясняющей возникновение третьего состояния (рис. 3.26). Если оба транзистора закрыты, то микросхема и нагрузка фактически являются разъединенными. Наличие третьего состояния обозначают символом_

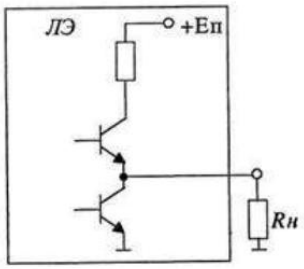


Рис. 3.26 рис.8

При использовании в едином цифровом устройстве микросхем различных серий, и в особенности различных логик, может возникнуть проблема согласования уровней входных и выходных напряжений. Для указанных целей производятся специальные микросхемы, которые называют преобразователями уровня сигналов.

Особенности логических элементов различных логик

Для конкретной серии микросхем характерно использование типового электронного узла — базового логического элемента. Этот элемент является основой построения самых разнообразных цифровых электронных устройств. Ниже рассмотрим особенности базовых логических элементов различных логик.

Элементы транзисторно-транзисторной логики. **Характерной** особенностью ТТЛ является использование многоэмиттерных транзисторов. Эти транзисторы сконструированы таким образом, что отдельные эмиттеры не оказывают влияния друг на друга. Каждому эмиттеру соответствует свой p-n-переход. В первом приближении многоэмиттерный транзистор может моделироваться схемой на диодах (см. пунктир на рис. 9).

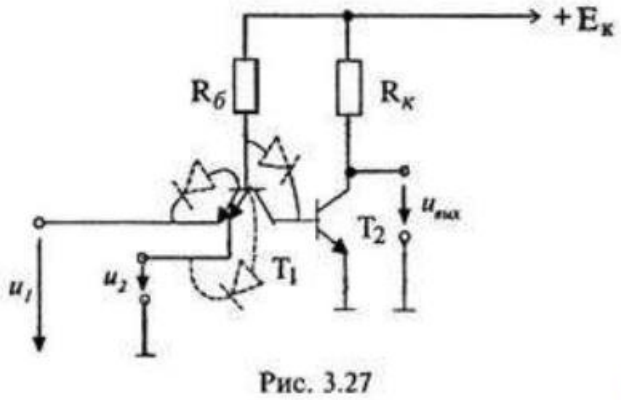


Рис. 3.27

Рис.9

Упрощенная схема ТТЛ-элемента приведена на рис. 9. При мысленной замене многоэмиттерного транзистора диодами получаем элемент диодно-транзисторной логики «И-НЕ». Из анализа схемы можно сделать вывод, что если на один из входов или на оба входа подать низкий уровень напряжения, то ток базы транзистора Т2 будет равен нулю, и на коллекторе транзистора Т2 будет высокий уровень напряжения. Если на оба входа подать высокий уровень напряжения, то через базу Т2 транзистора будет протекать большой базовый ток и на коллекторе транзистора Т2 будет низкий уровень напряжения, т. е. данный элемент реализует функцию И-НЕ: $u_{вых} = \overline{u_1 \cdot u_2}$

Базовый элемент ТТЛ содержит многоэмиттерный транзистор, выполняющий логическую операцию И, и сложный инвертор (рис.10). Если на один или оба входа одновременно подан низкий уровень напряжения, то многоэмиттерный транзистор находится в состоянии насыщения и транзистор Т2 закрыт, а следова-тельно, закрыт и транзистор Т4, т. е. на выходе будет высокий уровень напряжения. Если на обоих входах одновременно действует высокий уровень напряжения, то транзистор Т2 открывается и входит в режим насыщения, что приводит к открытию и насыщению транзистора Т4 и запиранию транзистора Т3, т. е. реализуется функция И-НЕ.

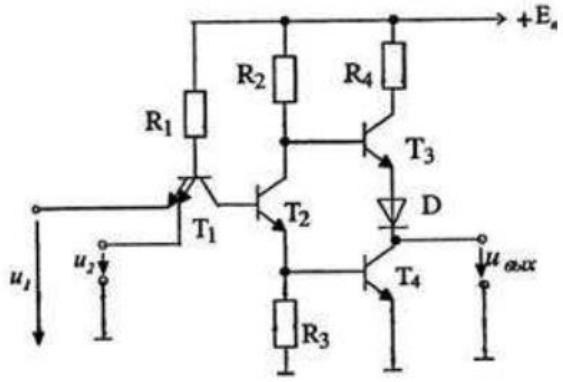


Рис. 3.28

Рис.10

***Базовый логический элемент ТТЛШ (на примере серии** К555).* В качестве базового элемента серии микросхем К555 использован элемент И-НЕ. На

рис. 3.29,а изображена схема этого элемента, а условное графическое обозначение транзистора Шоттки приведено на рис. 11. Такой транзистор эквивалентен рассмотренной выше паре из обычного транзистора и диода Шоттки. Транзистор VT4 — обычный биполярный транзистор.

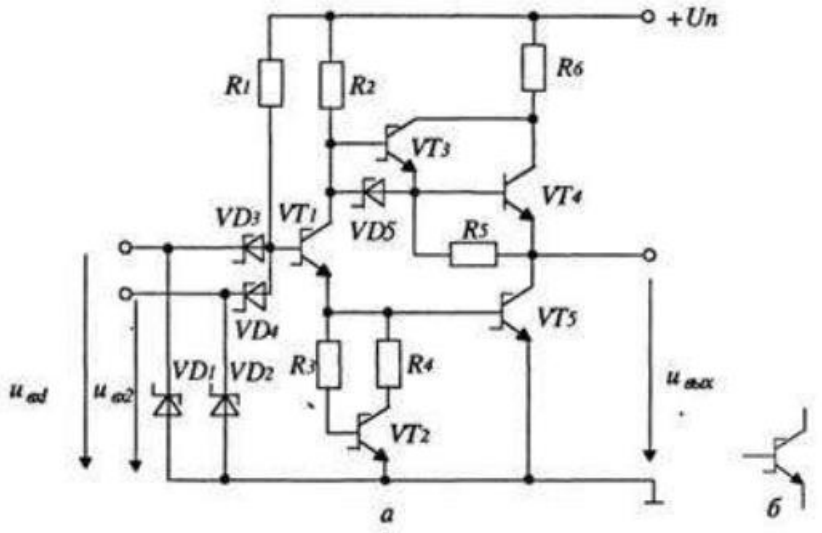


Рис. 3.29

Рис.11

Если оба входных напряжения ивх1 и *ивх2* имеют высокий уровень, то диоды VD3 и VD4 закрыты, транзисторы VT1, VT5 открыты и на выходе имеет место напряжение низкого уровня. Если хотя бы на одном входе имеется напряжение низкого уровня, то транзисторы VT1 и VT5 закрыты, а транзисторы VT3 и VT4 открыты, и на входе имеет место напряжение низкого уровня.

Особенности других логик. Основой базового логического элемента ЭСЛ является токовый ключ. Схема токового ключа (рис. 12) подобна схеме дифференциального усилителя. Необходимо обратить внимание на то, что микросхемы ЭСЛ питаются отрицательным напряжением (к примеру, -4,5 В для серии К1500). На базу транзистора VT2 подано отрицательное постоянное опорное напряжение *Uоп.* Изменение входного напряжения *ивх1*приводит к перераспределению постоянного тока *Iэ0*, заданного сопротивлением Rэ, между транзисторами, что имеет следствием изменение напряжений на их коллекторах. Транзисторы не входят в режим насыщения, и это является одной из причин высокого быстродействия элементов ЭСЛ.

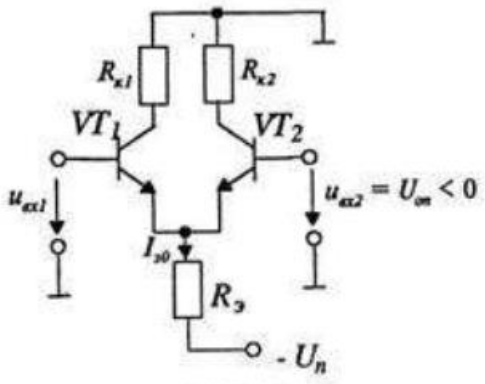


Рис. 3.30

Рис.12

В микросхемах *n*-МОП и *p-МОП* используются ключи соответственно на МОП-транзисторах с n-каналом и динамической нагрузкой (рассмотрены выше) и на МОП-транзисторах с p-каналом.

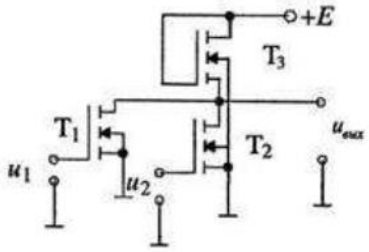


Рис. 3.31

Рис.13

В качестве примера рассмотрим элемент логики *n*-МОП, реализующий функцию ИЛИ-НЕ (рис. 13). Он состоит из нагрузочного транзистора Т3 и двух управляющих транзисторов Т1и Т2. Если оба транзистора Т1 и Т2 закрыты, то на выходе устанавливается высокий уровень напряжения. Если одно или оба напряжения *u1*и *u2* имеют высокий уровень, то открывается один или оба транзистора Т1и Т2 и на входе устанавливается низкий уровень напряжения, т. е. реализуется функция $u_{вых} = \overline{u_1 + u_2}$ (

Логика КМОП является очень перспективной. Рассмотренный ранее комплементарный ключ фактически является элементом НЕ (инвертором).

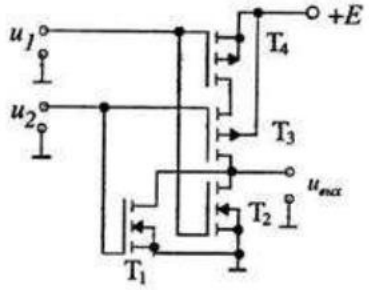


Рис. 3.32

Рис.14

Рассмотрим КМОП — логический элемент, реализующий функцию ИЛИ-НЕ (рис. 14), Если входные напряжения имеют низкие уровни *(u1*и *u2* меньше

порогового напряжения «-МОП-транзистора *Uзи*порогn), то транзисторы Т1и Т2 закрыты, транзисторы Т3 и Т4 открыты и выходное напряжение имеет высокий уровень. Если одно или оба входных напряжения *u1*и *u2* имеют высокий уровень, превышающий *Uзи*порогn, то открывается один или оба транзистора *T1*и Т2, а между истоком и затвором одного или обоих транзисторов Т3 и Т4 устанавливается низкое напряжение, что приводит к запиранию одного или обоих транзисторов Т3 и Т4, а следовательно, на выходе устанавливается низкое напряжение. Таким образом, этот элемент реализует функцию (7) и потребляет мощность от источника питания лишь в короткие промежутки времени, когда происходит его переключение.

Лекция 7Триггеры

Общие сведения

Такие логические уровни, которые, действуя на одном из входов элемента, однозначно задают логический уровень на его выходе независимо от уровней на других входах, будем называть ***активными логическими уровнями.*** Таким образом, активный логический уровень для элементов И—НЕ — логический 0, для элементов ИЛИ—НЕ — логическая 1.

Уровни, обратные активным, будем называть ***пассивными логическими уровнями.*** Пассивными уровнями для элементов И—НЕ служит логическая 1, для ИЛИ—НЕ — логический 0. При действии на одном из входов пассивного логического уровня уровень на выходе элемента определяется логическими уровнями на других его входах.

Триггер–это устройства с двумя состояниями. Они предназначены для запоминания двоичной информации. Использование триггеров позволяет реализовывать устройства оперативной памяти (то есть памяти, информация в которой хранится только на время вычислений)

Триггеры имеют два устойчивых состояния, обозначаемые как состояния 0 и 1. Эти состояния могут быть определены по логическим уровням на выходах. Триггер снабжается двумя выходами (рис. 3.1): прямым выходом *Q*и инверсным *Q*.Логический уровень на выходе *Q*совпадает с состоянием триггера: если триггер в состоянии 0, то и *Q*= 0, если триг гер в состоянии 1, то и = 1. Логический уровень на инверсном выходе *Q*представляет собой инверсию состояния триггера (в состоянии 0 $\bar{Q}$ = 1 и наоборот).

Триггеры снабжаются различными типами входов. Приводим обозначение и назначение входов, которые могут быть у триггеров:

R(от англ.RESET) — раздельный вход установки в состояние 0;

S(от англ.SET) — раздельный вход установки в состояние 1;

К — вход установки универсального триггера в состояние 0;

J— вход установки универсального триггера в состояние 1;

Т — счетный вход;

D(от англ.DELAY) — информационный вход установки триггера в состояние, соответствующее логическому уровню на этом входе;

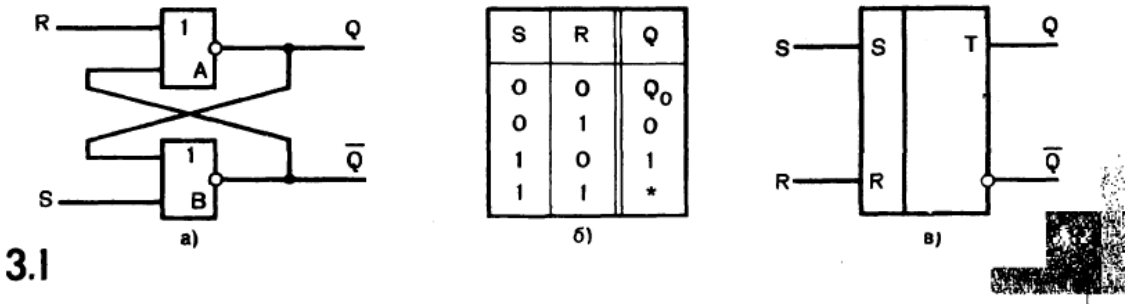
С — исполнительный управляющий (синхронизирующий) вход;

V — разрешающий управляющий вход.

Наименование триггера составляется из типов входов, которыми | он обладает. Например, RS-триггер — триггер, имеющий входы Rи S.

По характеру реакции на входные сигналы триггеры делятся на два типа :***асинхронные и синхронные.*** Асинхронные триггеры характеризуются тем, что входные сигналы на состояние триггера действуют с момента подачи их на входы, в синхронных триггерах только при подаче синхронизирующего сигнала на управляющий вход С.

АТ.RS-триггер с прямыми входами



3.1

Пусть на входах Rи Sдействуют пассивные для элементов ИЛИ—НЕ уровни логического 0, которые не влияют на состояние триггера. В состоянии 0 триггера на выходе элемента АQ= 0; этот уровень подается на вход элемента В; при этом на обоих входах элемента В действуетуровень логического 0 и на выходе элемента $\bar{Q} = 1$; с выхода элемента В уровень

логической 1 поступает на вх.А, что и обеспечивает на его вых. 0. Это одно из устойчивых состояний триггера. В состояний 1 триггера на вых элемента А $Q = 1$, что обуславливает на выходе элемента В $\overline{Q} = 0$, при этом на обоих входах элемента А действуют уровни 0, что и обеспечивает на выходе этого элемента уровень 1.

Таким образом, в каждом из состояний триггера элементы А и В оказываются в противоположных состояниях.

Переключение триггера из одного устойчивого состояния в другое происходит при подаче активных сигналов на входы.

Под действием уровня *R*= 1 элемент А установится в состояние, при котором на его выходе $Q = 0$, следовательно, на инверсном выходе $\vec{Q} = 1$, и, таким образом, триггер устанавливается в состояние 0. Если триггер и прежде, до подачи сигнала *R= 1,* находился в состоянии 0, то его состояние не изменяется. Если же триггер находился в состоянии 1, то при подаче сигнала *R*=1 произойдет переключение элемента А и на его выходе установится уровень *Q*= 0; далее этот уровень, действуя на входе элемента В, переключит его и на выходе элемента В установится уровень $\vec{Q} = 1$, после чего триггер оказывается установленным в состояние 0. Таким образом, при переключении триггера из одного состояния в другое его элементы последовательно переключаются и время переключения равно удвоенному среднему времени задержки распространения сигнала в логическом элементе ИЛИ—НЕ:

$$t_{\text{пер}} = 2t_{\text{зад.р.ср}} . \qquad (3.1)$$

Очевидно, чем меньше $t_{\text{пер}}$, тем в единицу времени удастся произвести большее число переключений триггера, будет выше допустимая частота переключений, иначе говоря, будет выше ***быстродействие*** триггера.

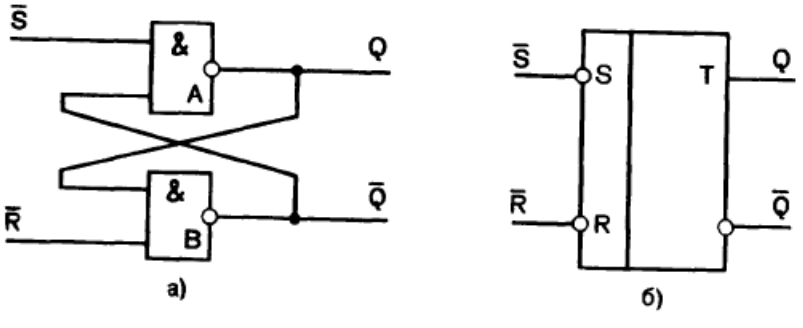
Процесс установки триггера в состояние 1 при подаче на вход S уровня 1 аналогичен описанному.

Одновременная подача активных уровней 1 на оба входа Rи Sне допускается, так как при этом на обоих выходах установится уровень 0, а после снятия со входов активных уровней состояние триггера окажется неопределенным: в силу случайных причин триггер может установиться в состояние 0 либо 1.

На рис. *3.1 б* приведена таблица состояний триггера: Q_0—состояние триггера до подачи на вход активного уровня, звездочкой отмечено состояние, соответствующее запрещенной комбинации уровней на входах: *S*= *R*= 1.

Логическое выражение, определяющее функционирование RS-триггера:

$$Q = S \vee \overline{R} Q_0, \qquad (3.2)$$



3.3

т. е. триггер устанавливается в состояние 1 под действием входного уровня $S = 1$ либо остается в этом состоянии 1, если *R*= 0 и прежнеесостояние триггера Q_0= 1.На рис. *3.1 в* показано условное обозначение асинхронногоRS-триггера.

RS-триггер с инверсными входами.Логическая структура триггера приведена на рис. 3.3а. Отличие от логической структуры рассмотренного выше RS-триггера с прямыми входами состоит лишь в том, что здесь использованы логические элементы И—НЕ.

При этом активным логическим уровнем на входах является уровень 0, пассивным — 1. Для того чтобы активными были, как и в предыдущем триггере, входные сигналы_ $S = 1$ и $R = 1$, будем считать^ что на входы подаются инверсии $\overline{S}$ и $\overline{R}$. Тогда при $S = 1$ (или $R = 1$) $\overline{S} = 0$ (или $\overline{R} = 0$)и на входе триггера будет действовать активный уровень 0. Другое удобство такого обозначения входных величин состоит в том,что триггер с инверсными входами описывается той же таблицей состояний (рис. 3.1 б), что и триггер с прямыми входами.

Рассмотрим устойчивые состояния триггера. Пусть на входах действуют пассивные уровни $S = 0$ и $R = 0$ ($\bar{S} = 1$ и $\bar{R} = 1$). В состоянии 0 триггера $Q = 0$, этот уровень передается на вход элемента В и вызывает на его выходе уровень $\bar{Q} = 1$, , уровень 1 с выхода элемента В подается на вход элемента А, и так как на обоих входах элемента А уровень 1, то на выходе этого элемента *Q*= 0. Аналогично определяется второе устойчивое состояние триггера. _

При подаче активного уровня $\bar{S} = 0$ $(S = 1)$ на выходе элемента А устанавливается уровень *Q*= 1, на выходе элемента В — уровень $\bar{Q} = 0$ и триггер оказывается установленным в состояние 1. При подаче активного уровня $\bar{R} = 0$ $(R = 1)$ триггер устанавливается в состояние 0. Как и в случае триггера с прямыми входами, одновременная подача активных уровней на оба входа не допускается.

На рис. *3.3б* показан способ обозначения RS-триггера с инверсными входами.

Синхронный RS-триггер имеет дополнительный синхронизирующий (тактирующий) вход С. Сигнал на входе С разрешает прием сигналов с информационных входов в заданные временные интервалы. При отсутствии сигнала — информационные входы отключаются и сигналы на этих входах не влияют на состояние триггера. При этом обеспечивается одновременный прием сигналов разными частями схемы в заданные временные отрезки.

На рис. 4.3 показаны логические структуры и условное графическое обозначение ***одноступенчатого (однотактного) синхронного RSтриггера.***

Логическое выражение функционирования синхронного RS- триггера

$$Q = \bar{C}Q_0 \vee C\left(S \vee \bar{R}Q_0\right).$$

Синхронный RS-триггер со статическим управлениемпереключается при наличии на входе С потенциала. При С = 0 триггер сохраняет ранее установленное в нем состояние Q_0.При С = 1 состояние триггера определяется действующими на входах уровнями так же, как и в асинхронном RS-триггере.

Синхронный RS-триггер с динамическим управлениемпереключается приналичии на входе*C* импульса (рис. 4.4).

Условием четкой работы данного триггера является временное совмещение синхросигналов с напряжениями, подаваемыми на входы *R* и *S*. .Прежде чем подавать на входы напряжения новых значений, необходимо дождаться, когда триггер установится в какое-либо состояние.

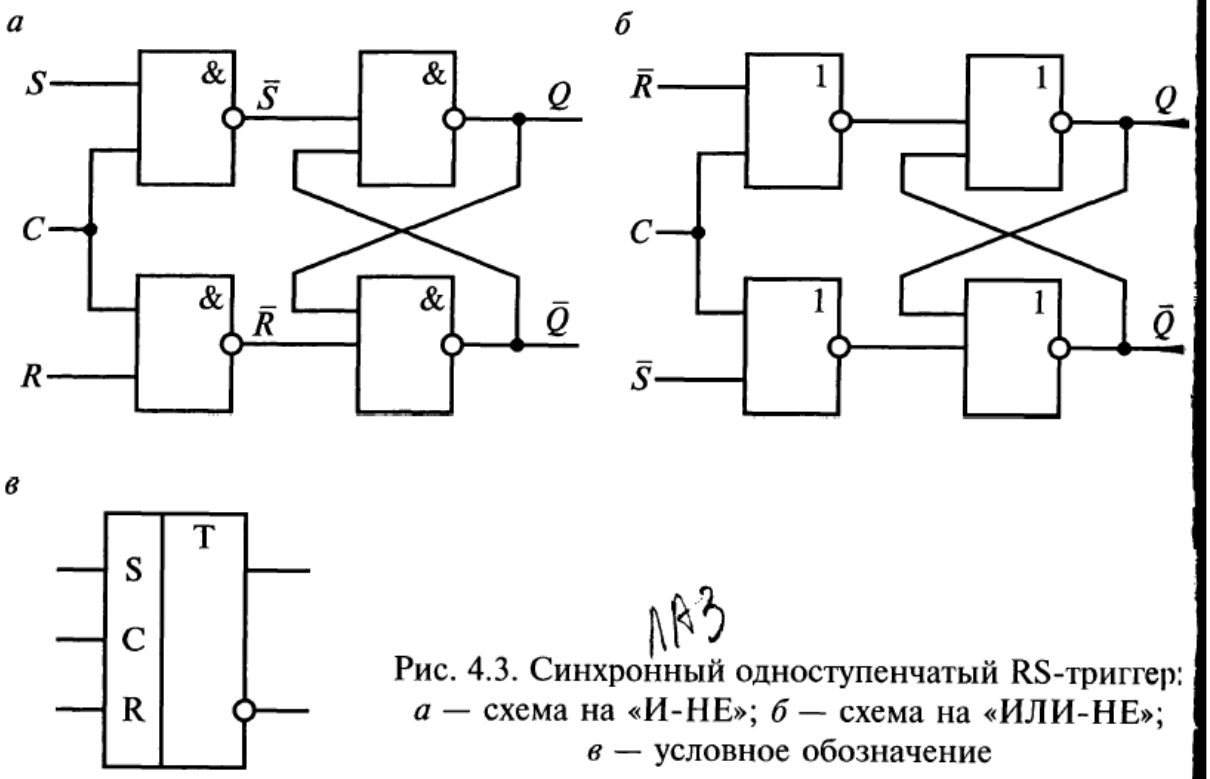


Рис. 4.3. Синхронный одноступенчатый RS-триггер: *а* — схема на «И-НЕ»; *б* — схема на «ИЛИ-НЕ»; *в* — условное обозначение

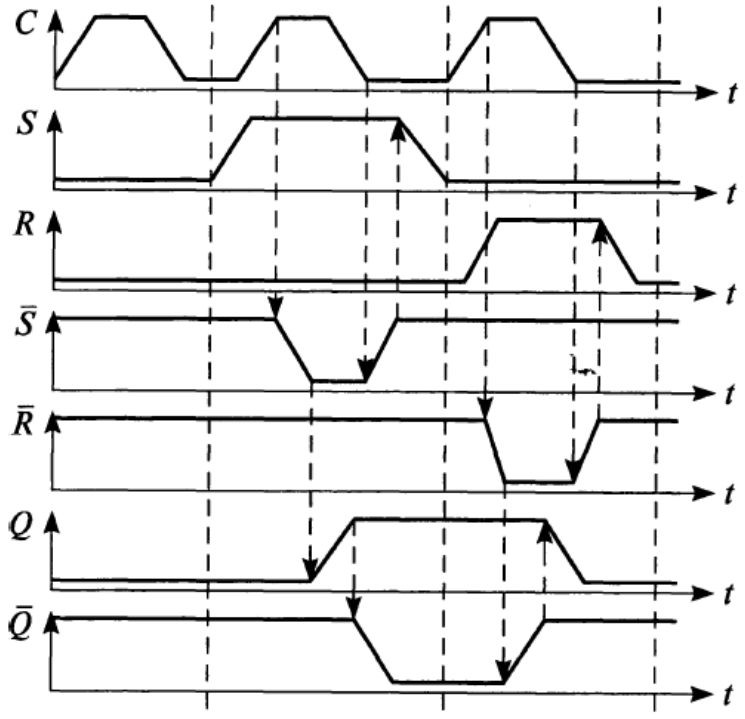


Рис. 4.4. Временна́я диаграмма работы однотактного синхронного RS-триггера с динамическим управлением

Особенность триггеров с ***двухступенчатым*** запоминанием информации состоит в том, что они содержат две триггерные структуры ступени): одна из них образует ведущий триггер, другая —ведомый. Оба триггера функционируют как синхронные триггеры.

Схема на рис. 4.5 состоит из двух синхронных RS-триггеров. При Наличии на входе С высокого уровня DD1 воспринимает информации с входов *S*и *R.*За счет элемента «НЕ» на входе DD2C =0 и информация с выходов DD1 не воздействует на DD2. В момент окончания действия высокого уровня на входе С на выходе «НЕ» появляется сигнал высокого уровня, разрешающий перезапись в DD2 информации из DD1. Таким образом, при *C* = 1 вторая ступеньотключена от первой, а при С = 0 первая не принимает информации с входов *S*и *R.*

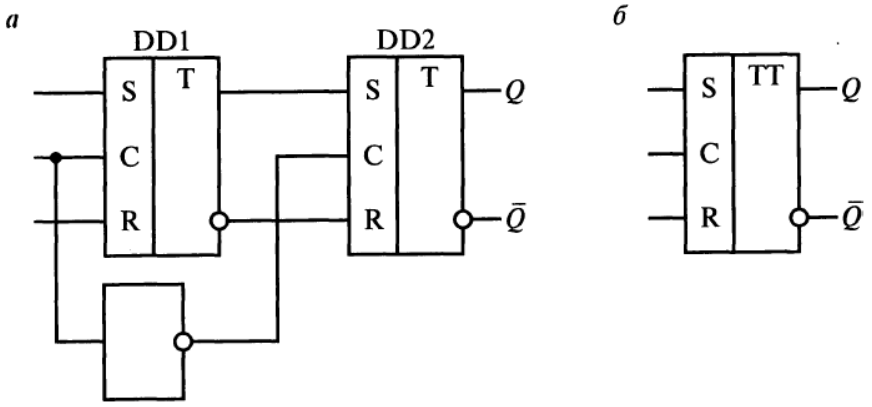


Рис. 4.5. Двухступенчатый синхронный RS-триггер: *а* — схема; *б* — условное обозначение

D-триггер предназначен для задержки выходного сигнала на один рабочий такт. Используется в быстродействующих цифровых устройствах.

Функционирование D-триггера определяется логическим выражением

$$Q = \bar{C}Q_0 \vee CD.$$

При*C* = *1* триггер устанавливается в состояние, определяемое логическим уровнем на входе *D.*При С = 0 он сохраняет ранее установленное состояние Q_0.

На рис. 4.6. представлена логическая структура ***одноступенчатогоD-триггера***, состоящего из RS-триггера с логическими элементами «НЕ» и «И» на входах. При С = 0 на выходах элементов «И»

образуются пассивные для входов асинхронного RS-триггера уровни. При *C*= 1 уровень, поданный на информационный вход *D*, создает активный уровень либо на входе *R*(при *D*= 0), либо на входе *S*(при *D*= 1) асинхронного RS-триггера, и триггер устанавливается в состояние, соответствующее логическому уровню на входе *D*. Таким образом, D-триггер воспринимает информацию со входа *D*при*C* = 1 и хранит ее неопределенное время пока C = 0.

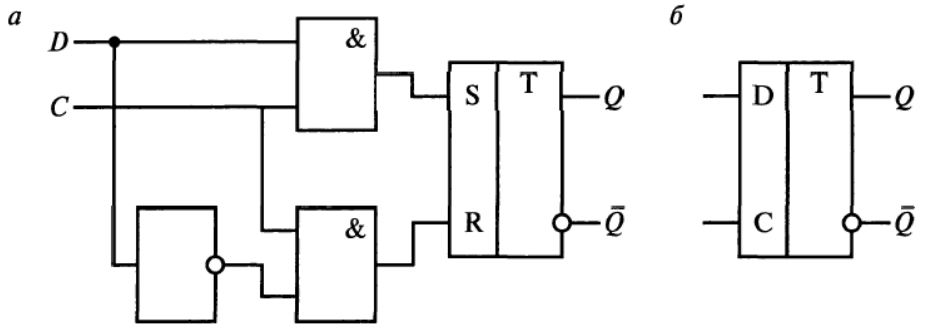


Рис. 4.6. Одноступенчатый D-триггер: *а* — схема; *б* — условное обозначение

Двухступенчатый D-триггер собирают на двух RS-триггерах и двух схемах типа «НЕ» (рис. 4.7). Пусть сигнал на входе появляется в интервале между первым и вторым синхроимпульсами С (рис. 4.8). В момент поступления импульса первый RS-триггер перебросится в единичное состояние. Под действием положительного сигнала по входу синхронизации второго RS-триггера он переключается в единичное состояние в паузе между вторым и третьим импульсами С. Таким образом, если сигнал на входе *D*по-явился между первым и вторым синхросигналами, то выходной сигнал *Q*будет между вторым и третьим. Т.е. осуществляется задержка на один такт.

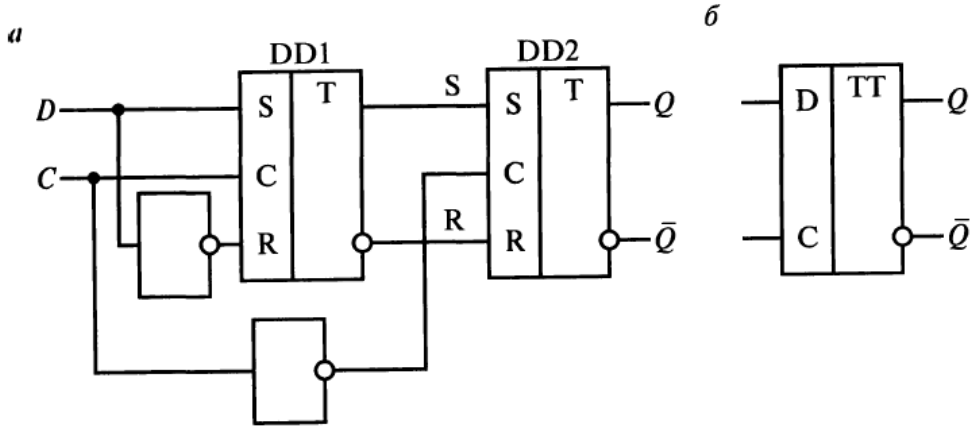


Рис. 4.7. Двухступенчатый D-триггер: *а* — схема; *б* — условное обозначение

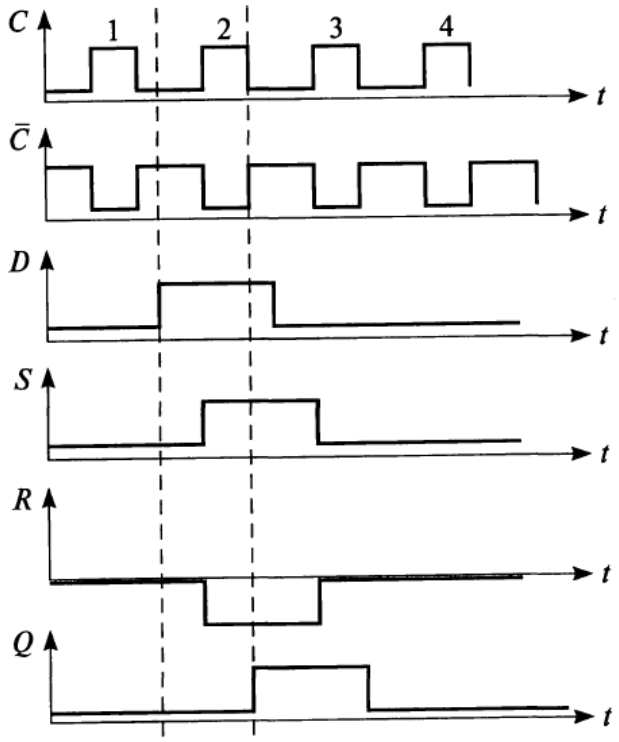


Рис. 4.8. Временна́я диаграмма работы двухступенчатого D-триггера

Если после каждого переключения обеспечить автоматическую смену уровня потенциала на входе D, то с каждым импульсом на C триггер будет менять свое состояние. Для этого вход D соединяют с выходом Q (рис. 4.9).

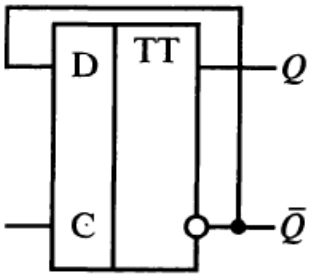


Рис. 4.9. Двухступенчатый D-триггер в счетном режиме

На рис. 4.10, *а* приведена логическая структура ***D-триггера с динамическим управлением***при положительном фронте сигнала. Элементы DD1 и DD2 составляют простейшую выходную триггерную структуру, состояние которой определяет состояние D-триггера. Элементы DD3—DD6 образуют схему, формирующую сигналы Y_1 и Y_2, которыми переключается выходная триггерная структура. На положительном фронте синхронизирующего сигнала на входе*C* выход устанавливается в состояние, соответствующее логическому уровню на входе *D.*Входы S_d и R_d —установочные, сигналами низкого уровня на этих входах триггер устанавливается н состояние соответственно «1» и «0».

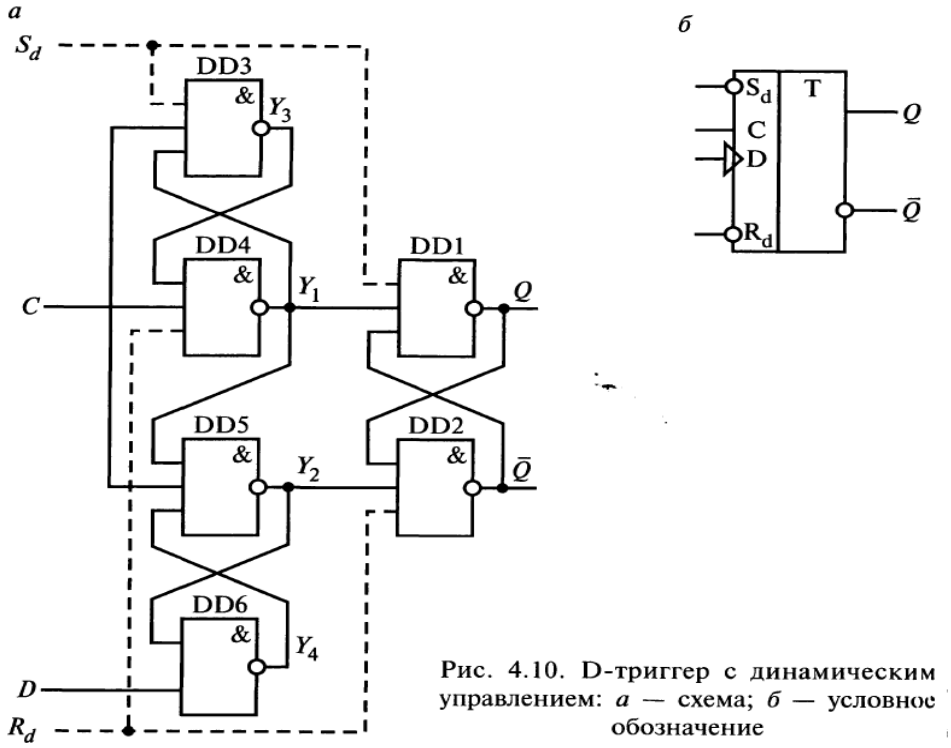


Рис. 4.10. D-триггер с динамическим управлением: *а* — схема; *б* — условное обозначение

JK-триггер.Предпосылкой к разработке JK-триггера было стремлениек устранению запретной ситуации *(R = S*= 1) в RS-триггере. ВходыJK-триггера связаны с выходами так, что, когда появляются два единичных сигнала, один из них блокируется.

Функционирование JK-триггера описано логическим выражением

$$Q = JQ_0 \vee KQ_0.$$

JK-триггер обычно выполняют синхронным путем введения в схему управления цепи тактирующего входа С. При С = 0 триггер находится в режиме хранения. В табл. 4.2 представлены состоянияJK-триггера.

Таблица 4.2

Таблица состояний JK-триггера

J	K	C	Q	Режим работы
0	0	1	Q_0	Хранение
0	1	1	0	Запись «0»
1	0	1	1	Запись «1»
1	1	1	$\bar{Q}_0$	Переключение в противоположное состояние

Принцип работы одноступенчатого JK-триггера реализуют на элементах «И-НЕ» или на элементах «ИЛИ-НЕ» с включенным в цепь RS-триггером(рисю4.11) Убрав штриховые линии (вход С) можно получить асинхронный триггер.

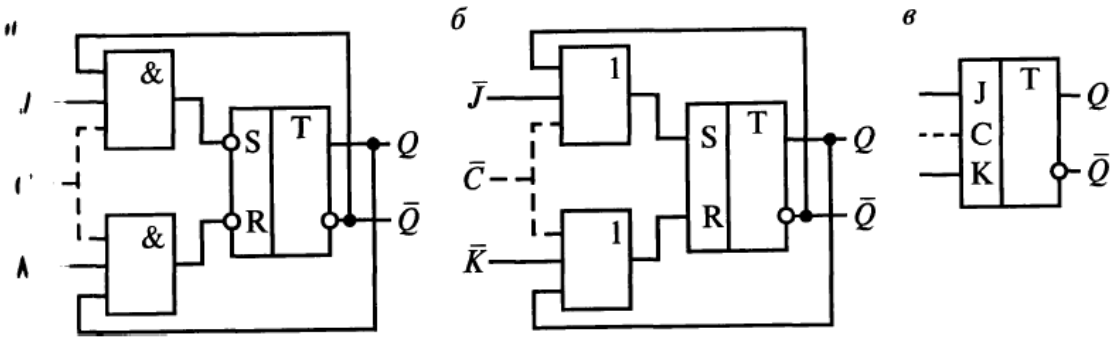


Рис. 4.11. Одноступенчатый JK-триггер: *а* — схема на элементах «И-НЕ»; *б* — схема на элементах «ИЛИ-НЕ»; *в* — условное обозначение

Двухступенчатый JK-триггер строится с использованием двух RS-триггеров (рис. 4.12). Один из RS-триггеров DD1 (ведомый) предназначен для хранения^ текущего состояния ***Q₀.***Снимаемые с

его выходов сигналы Q_0 и $\bar{Q}_0$ совместно с информационными сигналами входов *J*и *K* используются для формирования нового состояния *Q*в другом RS-триггере DD2 (ведущем).

При низком уровне на входе*C* триггер DD1 не реагирует на сигналы входов J и *K*. На вход *C*DD2 при этом подается высокий уровень, и состояние DD1 передается DD2. Оба триггера оказываются в одном и том же состоянии. При переходе на входе *C* к высокому уровню на синхронизирующий вход DD2 через инвертор подается низкий уровень, и логическая связь между триггерами обрывается. DD1 устанавливается в состояние *Q,*определяемое его логическим выражением. Подача на вход*C* вновь низкого уровня приводит к передаче состояния *Q*из DD1 в DD2.

На рис. 4.13 построена схема ***JK-триггера с динамическим управлением по фронту***. При действии положительного фронта сигнала на синхронизирующем входе C триггер устанавливается в состояние «1» при J = 1 и в состояние «0» при*K* = 1.

JK-триггер является универсальным,так как на его базе можно построить схемы различным типов триггеров (рис.4.14)благодаря разлиному включению и использованию входов.

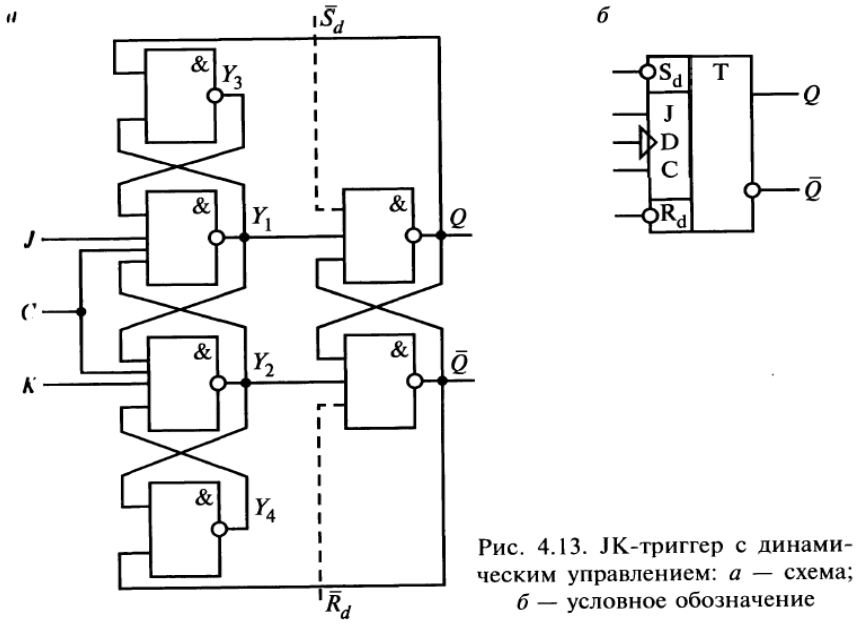


Рис. 4.13. JK-триггер с динамическим управлением: *а* — схема; *б* — условное обозначение

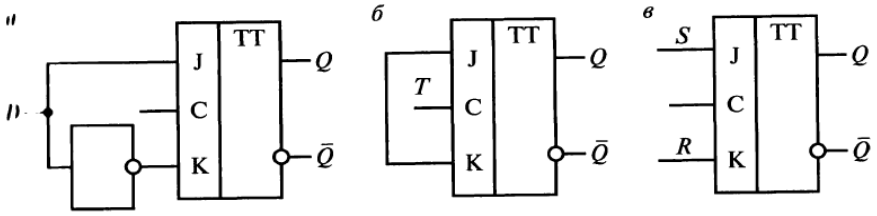


Рис. 4.14. Типы триггеров на базе JK-триггера: *а* — D-триггер; *б* — T-триггер; *в* — RS-триггер

Т-триггер представлен на рис. 4.15. При положительном фронте импульса, поступающего на вход Т, ведущий DD1 устанавливается и состояние, противоположное состоянию ведомого DD2, при отрицательном фронте входного импульса происходит передача сигнала, соответствующего состоянию DD1, в DD2.

Т-триггер (рис. 4.16) можно построить с помощью JK-триггера или D-триггера.

В табл. 4.3 представлены параметры интегральных микросхем триггеров.

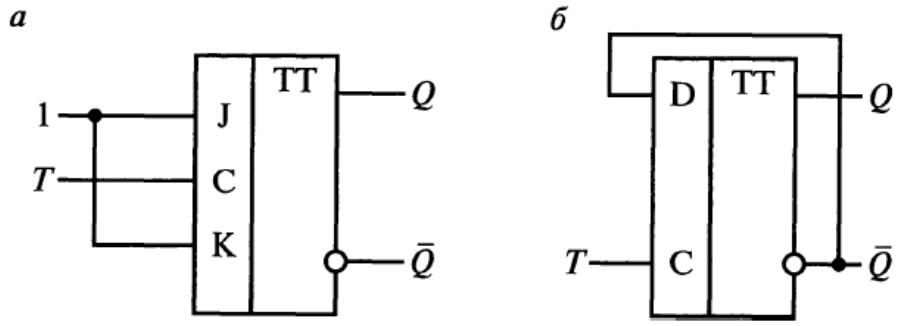


Рис. 4.16. Т-триггер: *а* — на базе JK-триггера; *б* — на базе D-триггера

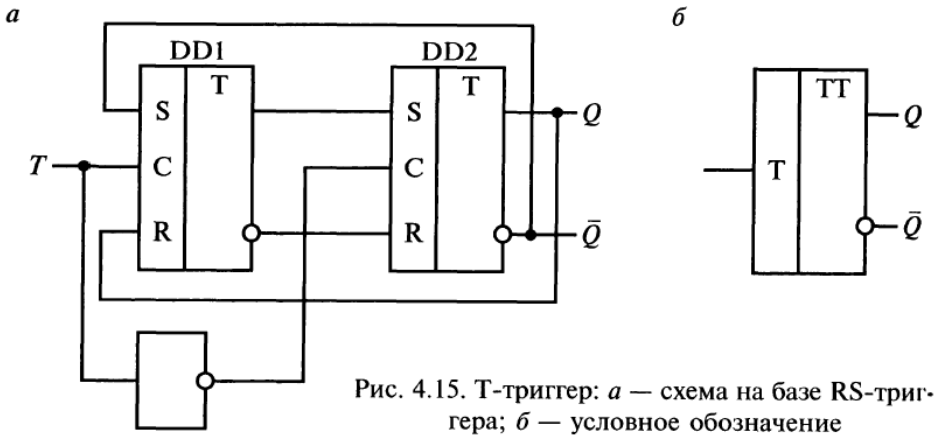


Рис. 4.15. Т-триггер: *а* — схема на базе RS-триггера; *б* — условное обозначение

Лекция 8Счетчики

Счетчик **(СТ)** — это ПЦУ, предназначенное для подсчета импульсов, поступающих на его вход, и фиксации этого числа в виде кода. СТ применяются для формирования адреса ячеек запоминающих устройств, счета количества циклов выполнения операций, запоминания кода в аналого-цифровых преобразователях.

СТ содержит входную логику, управляющую его работой, и выходную логику, которая используется для указания окончания счета или формирования сигнала переноса *P* (рис. 4.21). Для приведения СТ в начальное состояние используется сигнал сброса, поступающий на вход *R.*Параллельный код для предварительной установки счетчика поступает на входы $S_0 - S_n$. .Сигнал разрешения параллельной загрузки*E* останавливает счет и позволяет подготовленным на входах $S_0 - S_n$. данным загрузиться в счетчик в момент прихода очередного тактового импульса. СТ считает тактовые импульсы, поступающие на вход С, если присутствует сигнал разрешения счета на входе V. Выходными сигналами являются сигналы, снимаемые с выходов отдельных разрядов Q_1-Q_n, сигнал окончания счета или сигнал переноса Р.

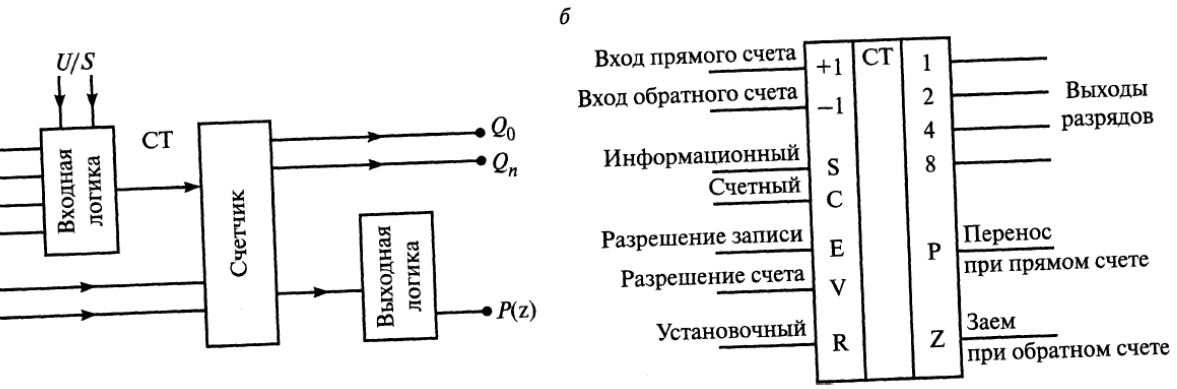


Рис. 4.21. Счетчик: *а* — обобщенная схема; *б* — условное обозначение

СТ строятся на Т-триггерах (JK- или D-триггерах в счетном режиме).

Счетчики делятся:

1. По целевому назначению (по направлению счета):

суммирующие (прямое направление) — с подачей на вход очередного импульса состояние СТ увеличивается на единицу (+1);

вычитающие (обратное направление) — с подачей на вход очередного импульса состояние СТ уменьшается на единицу (—1);

реверсивные — счет осуществляется в прямом и обратном направлениях.

2. По способу организации счета:

асинхронные — сигнал от разряда к разряду передается естественным путем в различные интервалы времени;

синхронные — сигнал от разряда к разряду передается принудительным путем с помощью тактовых сигналов.

3. По способу кодирования:

двоичное — двоичные (СТ2);

двоично-десятичное — декадные (СТ10 или $СТ^{2}/_{10}$);

одинарное — кольцевые СТ;

унитарное — СТ Джонсона.

4. По способу организации цепей переноса между разрядами:

с последовательным переносом;

с параллельным (сквозным) переносом;

с параллельно-последовательным (групповым) переносом.

Основные характеристики СТ:

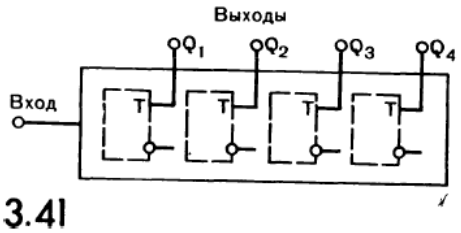
модуль счета — число устойчивых состояний в цикле (предельное число входных сигналов, которое может сосчитать СТ) $M = 2^n$, где n — разрядность СТ (количество триггеров в схеме);

разрешающая способность — минимально допустимый период следований входных сигналов, при котором обеспечивается надежная работа СТ;

время регистрации – интервал времени между моментами поступления входного сигнала и окончания самого длиннного переходного процесса в СТ;

емкость – максимальное число единичных сигналов, которое может быть зафиксировано на СТ.

Для представления чисел в счетчике могут использоваться двоичная или десятичная системы счисления. При использовании двоичной системы состояния триггеров и соответствующие им уровни на прямых выходах триггеров определяют цифры двоичных разрядов числа. Если для регистрации двоичного числа в счетчике используется n триггеров, то максимальное значение числа, до которого может вестись счет, $N = 2^n — 1$. Так, при n = 4 $N = 15$. На рис. 3.41 показана последовательность чисел, а в табл. 3.15 — соответствующие им состояния триггеров в процессе счета при $n = 4$.



3.41

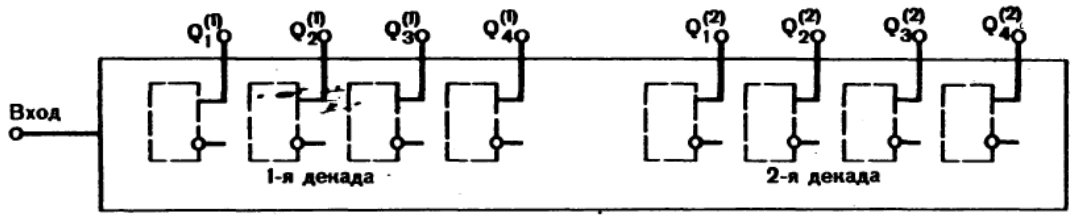
ТАБЛИЦА 3.15

Число в счетчике	Состояние триггеров			
	Q_4	Q_3	Q_2	Q_1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	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

При использовании десятичной системы цифры разрядов десятичного числа в счетчике представляются в четырех- разрядной двоичной форме, т. е. используется двоично-кодированная десятичная система счисления. Таким образом, для представления цифр каждого разряда десятичного числа требуется четыре триггера, и еслиЧисло десятичных разрядов k, то число триггеров, необходимое для регистрации чисел в счетчике, равно $4k$, а максимальное значение чисел $N = 10^k - 1$. В табл. 3.16 показана последовательность состояний триггеров в двухразрядном десятичном счетчике, приведенном на рис. 3.42.

ТАБЛИЦА 3.16

Число в счетчике	Состояние триггеров							
	Q_4^2	Q_3^2	Q_2^2	Q_1^2	Q_4^1	Q_3^1	Q_2^1	Q_1^1
0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	1
2	0	0	0	0	0	0	1	0
.	.	.	.	.	.	.	.	.
9	0	0	0	0	1	0	0	1
10	0	0	0	1	0	0	0	0
11	0	0	0	1	0	0	0	1
.	.	.	.	.	.	.	.	.
99	1	0	0	1	1	0	0	1



3.42

Двоичные счетчики. Асинхронный суммирующий СТ с последовательным переносом на JK-триггерах (рис. 4.22) имеет следующие особенности: в каждом триггере входы J иК объединены, на эти входы подан высокий уровень, синхронизирующий вход С является счетным входом триггера; сигнал с прямого выхода триггера каждого разряда поступает на счетный вход С триггера следующего, более старшего, разряда, на счетный вход триггера первого разряда подаются входные импульсы.Переключение каждого тригге-ра происходит по спаду импульса.

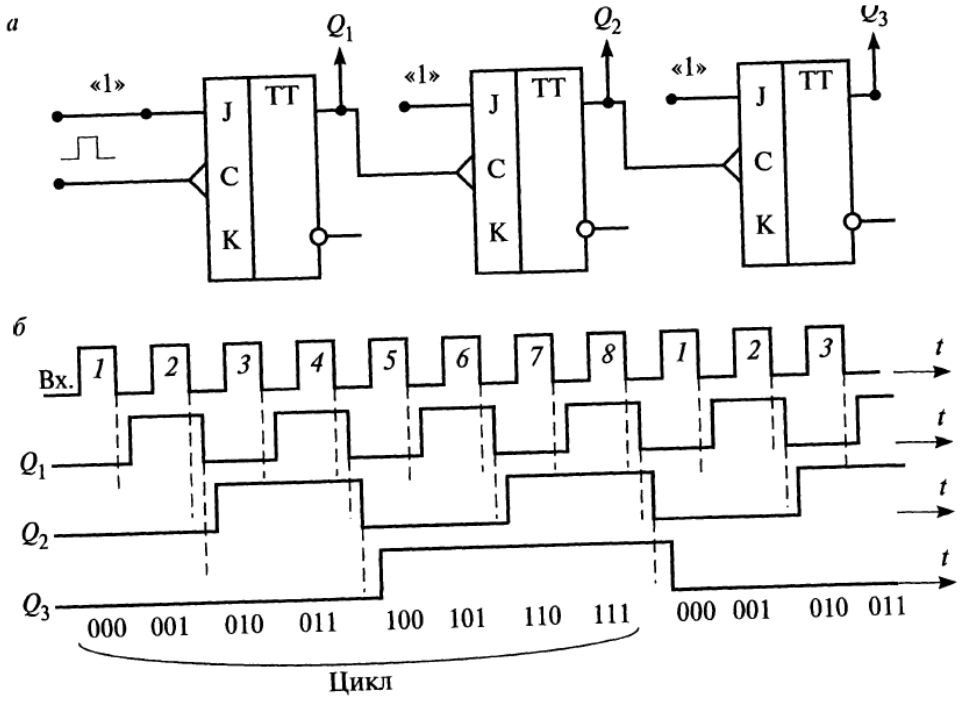


Рис. 4.22. Двоичный суммирующий асинхронный трехразрядный СТ с последовательным переносом на JK-триггерах: *а* — логическая схема; *б* — временна́я диаграмма

С каждым входным импульсом число в счетчике увеличивается на единицу. Такое нарастание числа происходит до тех пор, пока после $(2^n$ — 1)-го входного импульса (n — число разрядов в счет чике) в счетчике не устанавливается двоичное число Далее с приходом 2^n-го импульса в счетчике устанавливается исходное со

стояние 00...0, после чего счет ведется сначала. Таким образом, при непрерывной подаче на вход импульсов счетчик циклически с периодом 2^n входных импульсов устанавливается в исходное состояние

При построении ***асинхронного вычитающего СТ с последовательным переносом на JK-триггерах*** достаточно изменить в схеме рис. 4.22 соединение триггеров: инверсный выход первого триггера соединить со счетным входом второго триггера и т.д. Выходной сигнал следует также снимать с прямых выходов триггеров. В этом случае при поступлении сигнала сброса *R*на всех выходах схемы установятся высокие уровни, а при поступлении счетных импульсов на вход С триггеры схемы будут изменять свои состояния, описываемые последовательно убывающими двоичными числами.

Асинхронный суммирующий СТ с последовательным переносом на D-триггерах (рис. 4.23) переключается по фронту импульса. На счетный вход триггера первого разряда подаются входные импульсы. Сигнал с инверсного выхода триггера каждого разряда поступает на информационный вход *D*этого же триггера и на счетный вход*C* следующего триггера более старшего разряда. Выходной двоичный код фиксируется на прямых выходах триггеров.

Для построения ***асинхронного вычитающего счетчика с последовательным переносом на D-триггерах*** необходимо поменять в схеме рис. 4.23 инверсные выходы триггеров на прямые выходы.

Недостатком асинхронных СТ является то, что в ходе переключения младшие разряды СТ принимают уже новые состояния, в то время как старшие еще находятся в прежнем, т.е. при смене одного числа другим СТ проходит ряд промежуточных состояний, каждое из которых может быть принято за двоичный код числа прошедших на входе импульсов. Также данные СТ имеют сравнительно низкое быстродействие. При монтаже схем асинхронных СТ необходимо в качестве первого триггера использовать самый быстродействующий, так как он должен иметь большую частоту переключения, чем последующие.

Синхронный суммирующий СТ с параллельным переносом на JK-триггерах (рис.4.24) с М=16 состоит из триггеров DD1 –DD4 и двух элементов «И». Тактовые импульсы поступают на входы С одновременно. DD1 переключается каждым счетным импульсом, так как на его входы J и K постоянно подается «1». Остальные триггеры переключаются импульсами при следующих условиях: DD2- при Q_1=1; DD3- при Q_1=1, Q_2=1; DD4- при Q_1=1; Q_2=1; Q_3=1.

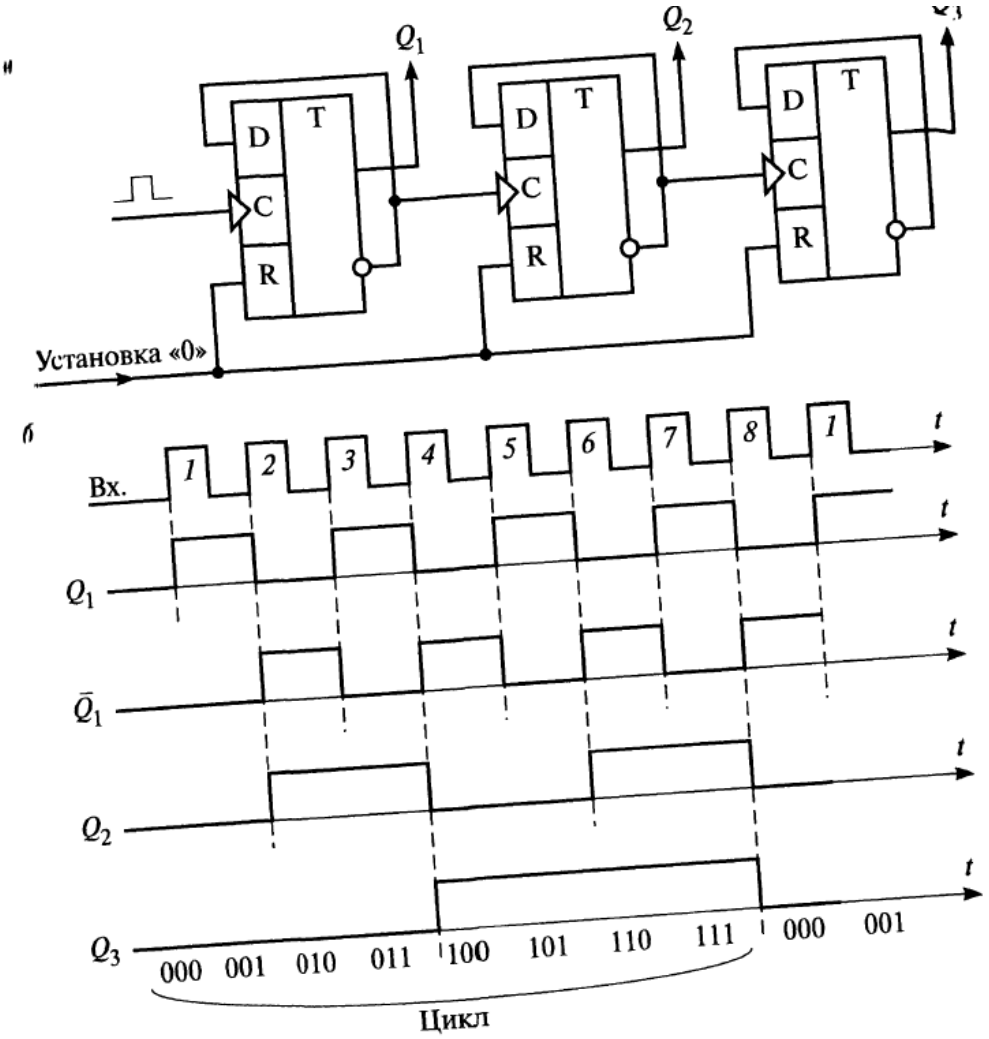


Рис. 4.23. Двоичный суммирующий асинхронный трехразрядный СТ с последовательным переносом на D-триггерах: *а* — логическая схема; *б* — временна́я диаграмма

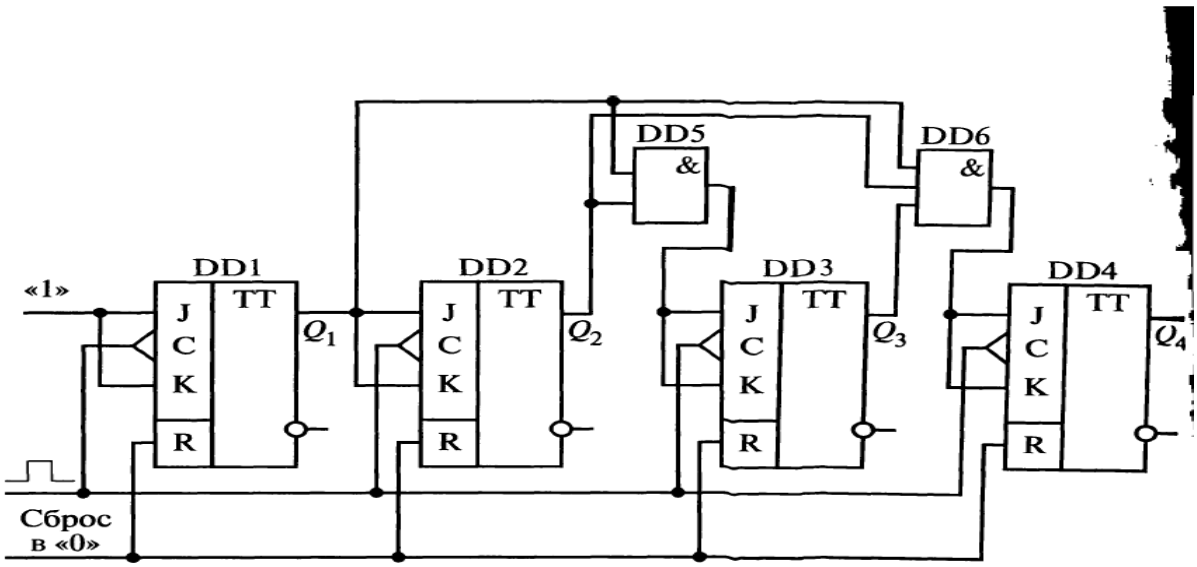


Рис. 4.24. Схема двоичного суммирующего синхронного СТ на JK-триггерах

Реверсивный счетчик допускает в процессе работы переключение из режима суммирования в режим вычитания и наоборот. Схема на рис. 4.25 содержит две цепи передачи переносов, одна из которых соответствует схеме суммирующего счетчика, другая — схеме вычитающего счетчика. Управляющие сигналы I_1 и I_2 включают в работу одну или другую цепь. При $I_1 = 1$ и $I_2 = 0$ оказываются закрытыми элементы DD2.1—DD2.3 и, следовательно, отключена цепь передачи переносов режима вычитания; СТ работает в режиме суммирования. При $I_1 = 0$ и $I_2 = 1$ закрыты элементы DD1.1—DD1.3 и отключена цепь передачи переносов режима суммирования; СТ работает в режиме вычитания.

Кольцевые счетчики. Простейшая схема выполняется на базе регистра сдвига (см. рис. 4.19) путем замыкания прямого выхода с его входом. В таком СТ единица, записанная в регистр на первом такте, с выхода Q_nснова попадает на его вход, и далее весь цикл повторяется. Модуль счета кольцевого СГ $M = n$. Для его увеличения можно или увеличивать количество триггеров в кольце, или включать СТ последовательно. Так, например, СТ на 10 импульсов ($M = 10$) можно реализовать последовательным соединением одного счетного триггера и кольцевого СТ из пяти триггеров.

Основным недостатком кольцевых СТ является их низкая помехозащищенность. Например, если пол действиемпомехи исчезнет записанная в СТ единица, то все триггеры окажутся в нулевом состоянии и СТ работать не сможет. Для устранения подоб- ! ных сбоев используются различные способы коррекции состояния СТ. Схема с автоматической коррекцией состояния приведена на рис. 4.26. В этой схеме независимо от того, в каком состоянии пос-

ле включения окажутся триггеры, после четырех тактовых импульсов на входе С установится требуемое выходное состояние 1000.

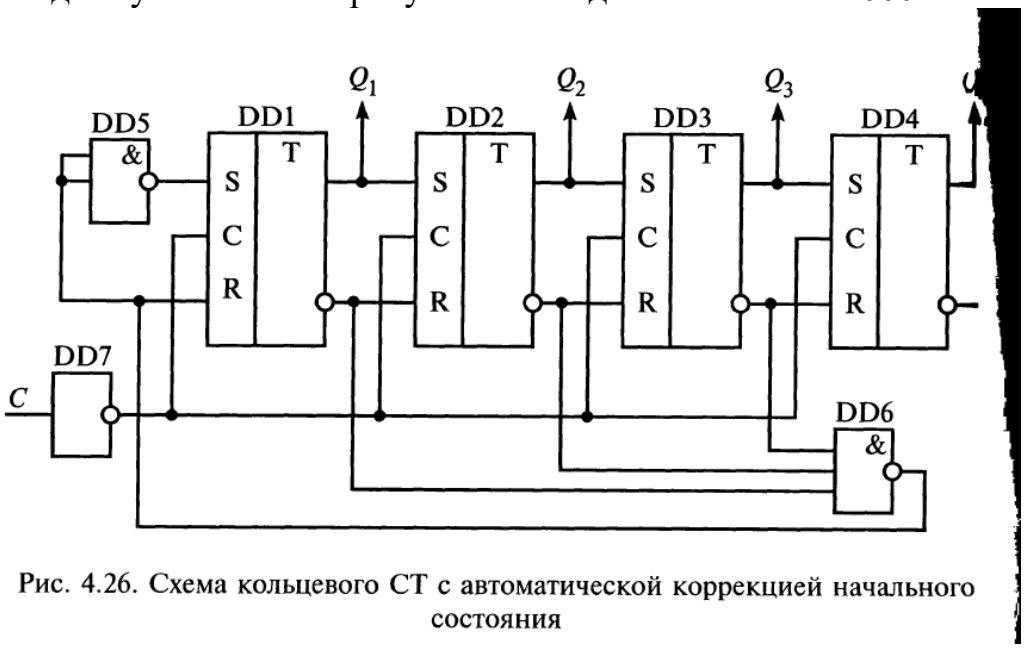


Рис. 4.26. Схема кольцевого СТ с автоматической коррекцией начального состояния

Лекция 9Регистры

Регистр — это ПЦУ, предназначенное для хранения одного многоразрядного числа, сдвига хранимого в регистре числа на определенное число разрядов влево или вправо, преобразования числа из последовательной формы в параллельную и обратно. При этом число должно быть представлено с двоичным представлением цифр разрядов.

Регистр строится в виде набора триггеров, каждый из которых служит для хранения цифр определенного числа. Разрядность регистра (число информационных входов) соответствует количеству используемых в нем триггеров.

Регистры делятся:

1. По виду выполняемых операций:
   для приема и передачи информации —регистры памяти;
   для сдвига информации — регистры сдвига.
2. По способу приема и передачи информации:
   параллельные — предназначенное для хранения число подается одновременно всеми разрядами;
   последовательные — ввод числа производится путем последовательной во времени подачи цифр отдельных разрядов (обычно начиная с цифры младшего разряда);
   смешанные — ввод числа осуществляется в параллельно-последовательной форме.

Регистры памяти. Пусть на вход параллельного регистра поступает двоичный код числа. Прием такого числа может производиться

в регистр, построенный с использованием синхронных RS-триггеров(рис. 4.17). Прием информации в DD1—DDnосуществляется подачей на входы X_1, X_2, X_n сигналов, соответствующих логическим переменным при поступлении синхроимпульса на вход С.Предварительно перед занесением в регистр кода числа необходимо его сбросить в «0».

Регистр памяти можно построить с использованием синхронныхD-триггеров. В схеме регистра при высоком уровне на входе С триггеры устанавливаются в состояния, определяемые действующими на входах *D*цифрами разрядов.

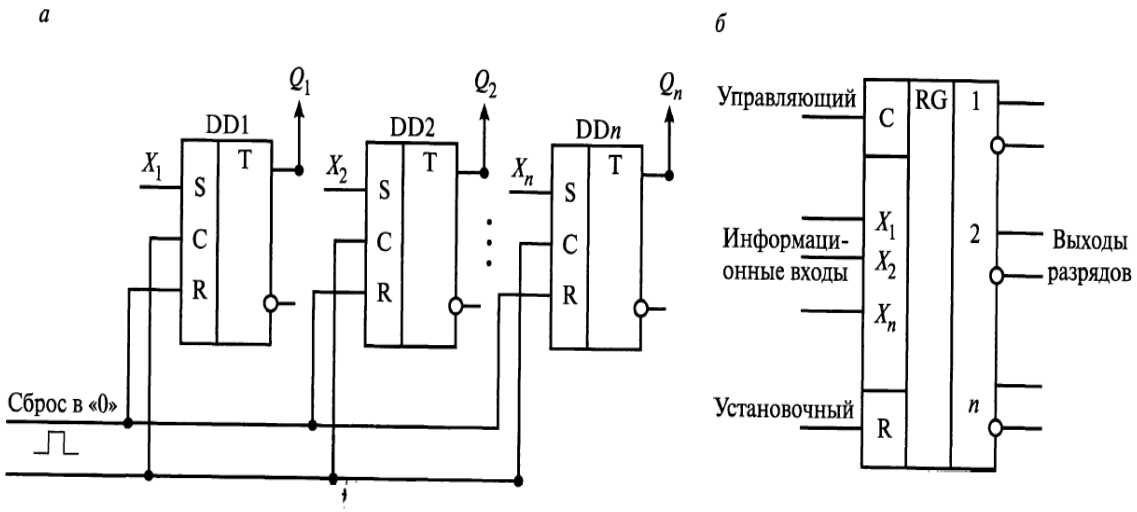


Рис. 4.17. Параллельный регистр для хранения *n*-разрядного кода числа: *а* — логическая схема; *б* — условное обозначение

Сдвигающие регистры применяют в качестве запоминающих устройств, преобразователей последовательного кода в параллельный,

устройств задержки и счетчиков импульсов.

При сдвиге происходит смещение двоичного кода числа (влево RG← или вправо: RG→) на заданное число разрядов. На рис. 4.18 показана схема регистра сдвига на D-триггерах и временная диаграмма работы. Каждый D-триггер является двухступенчатым синхронным. Сигнал «сдвиг» подается на входы синхронизации. Под его действием на первые ступени D-триггеров переписываются значения состояний вторых ступеней триггеров предыдущих разрядов. В паузе между синхросигналами осуществляется изменение состояния ступеней триггеров.

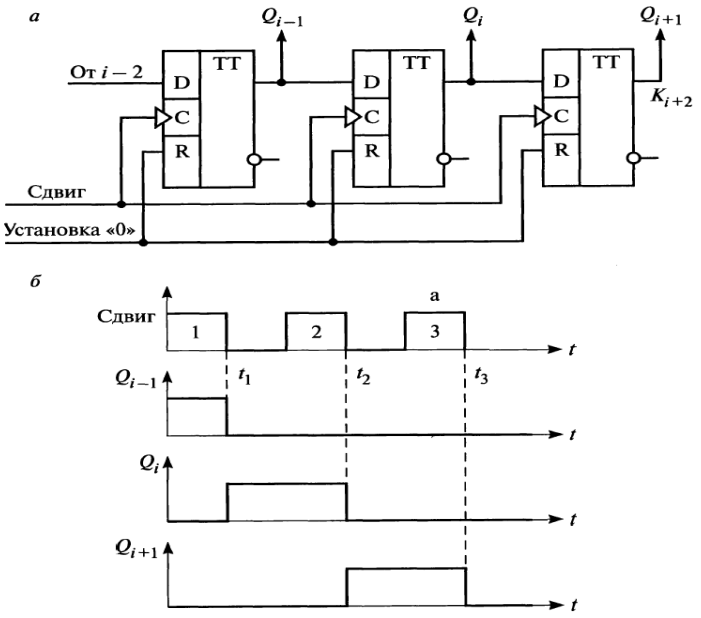


Рис. 4.18. Регистр сдвига на D-триггерах: *а* — логическая схема; *б* — временна́я диаграмма

Рассмотрим передачу кода 100, находящегося на выходах Q_{i-1}, Q_i и Q_{i+1} (рис. 4.18, *б)*. Пусть остальные разряды регистра слева и справа хранят «0». За время действия синхросигнала «сдвиг» переключаются первые вспомогательные ступени триггеров регистра. При этом состояние выходов не изменяется. В интервале времени между t_1 и фронтом второго синхросигнала «1» из вспомогательного триггера i-го разряда передается в основной триггер i-го разряда. Сброс основного триггера i-го разряда в «0» происходит и следующем такте при передаче «1» в разряд $i+1$. Последний третий импульс сдвига в момент t_3 устанавливает на всех выходах регистра «0». «1» оказывается продвинутой из $(i-1)$-го разряда и $(i+2)$-й. Таким образом осуществляется сдвиг вправо.

Сдвиговые регистры можно реализовать на JK-триггерах или RS-триггерах (рис. 4.19). Если при поступлении первого тактового импульса на входах $X1$ и $X2$ установлены сигналы $X1 = X2 = 1$, которые затем снимаются по приходу второго тактового импульса, то в результате в DD1 будет записан сигнал $Q_1 = 1.$. С приходом второго тактового импульса в DD1, $Q_1 = 0,$ а на выходе DD2 появится $Q_2 = 1,$, который перед этим был на выходе DD1. При поступлении последующих тактовых импульсов единичный сигнал перемещается последовательно в DD3 и DD4, после чего все триггеры устанавливаются в нулевое состояние.

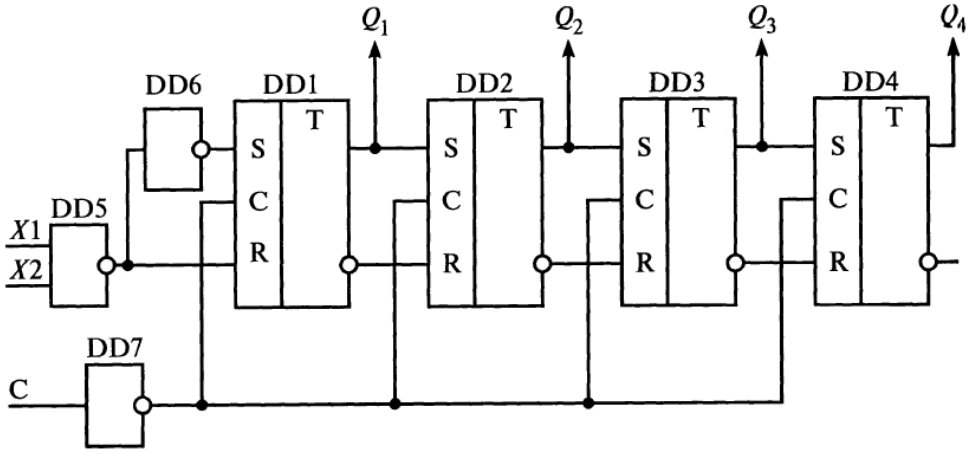


Рис. 4.19. Схема сдвигающего регистра на RS-триггерах

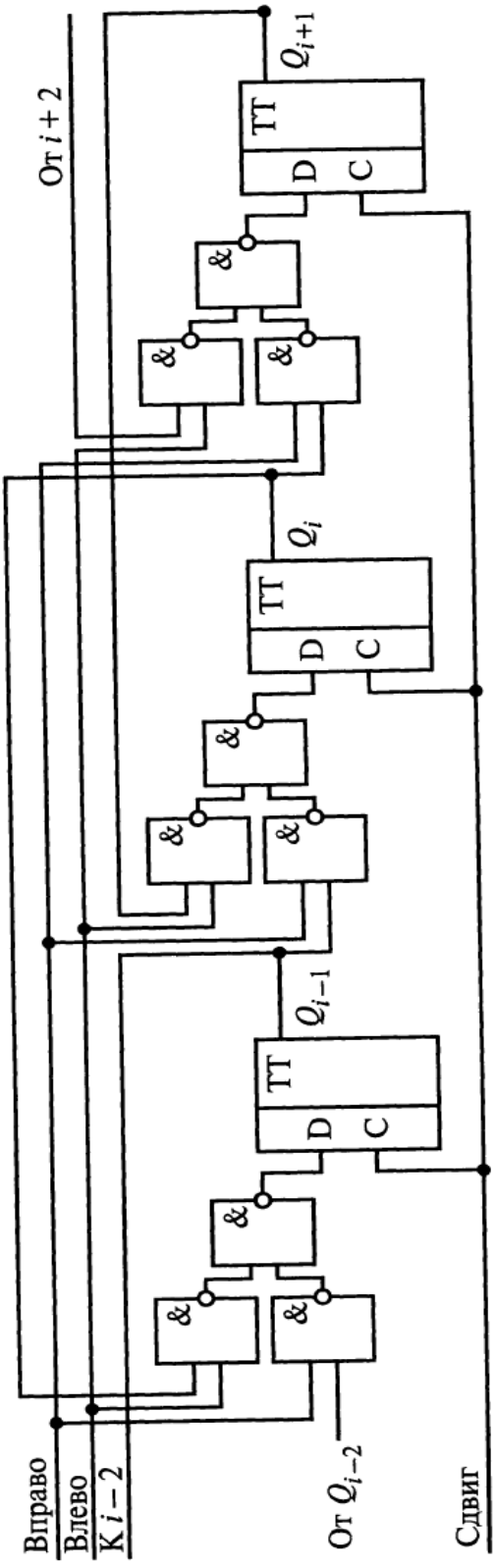


Рис. 4.20. Схема реверсивного регистра сдвига на D-триггерах

Для всех регистров сдвига характерны следующие положения: предварительно устанавливается исходное состояние и в первый триггер вводится единица; для регистра *n* триггеров после поступления*n* входных тактовых импульсов первоначально введенная единица выводится, вследствие чего прямые выходы всех регистров оказываются в нулевом состоянии.

Лекция 10Шифраторы и дешифраторы

Шифраторы. Назначение, УГО, схемная реализация, маркировка

Определения. Шифратор (называемый также *кодером)* — устройство, осуществляющее преобразование десятичных чисел в двоичную систему счисления.

Пусть в шифраторе имеется *m* входов, последовательно пронумерованных десятичными числами (0, 1, 2, 3, ..., *m* — 1) и *n* выходов. Подача сигнала на один из входов приводит к появлению на выходах n-разрядного двоичного числа, соответствующего номеру возбужденного входа.

Очевидно, трудно строить шифраторы с очень большим числом входовm, поэтому они используются для преобразования в двоичную систему счисления относительно небольших десятичных чисел.

Шифраторы широко используются в разнообразных устройствах ввода информации в цифровые системы. Такие устройства ввода могут снабжаться клавиатурой, каждая клавиша которой связана с определенным входом шифратора. При нажатии выбранной клавиши подается сигнал на определенный вход шифратора, и на его выходе возникает двоичное число, соответствующее выгравированному на клавише символу.

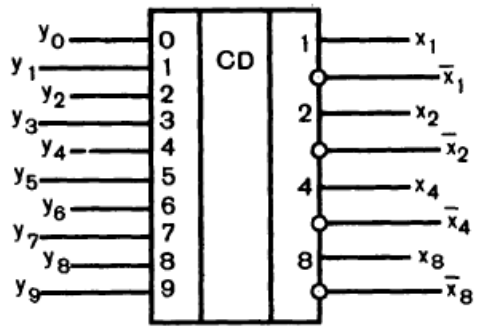
Для обратного преобразования двоичных чисел в небольшие по значению десятичные числа используются *дешифраторы* (называемые также *декодерами).* Входы дешифратора предназначаются для подачи двоичных чисел, выходы последовательно нумеруются десятичными числами. При подаче на входы двоичного числа появляется сигнал на определенном выходе, номер которого соответствует входному числу.

Дешифраторы имеют широкое применение. В частности, они используются в устройствах, печатающих на бумаге выводимые из цифрового устройства числа или текст. В таких устройствах двоичное число, поступая на вход дешифратора, вызывает появление сигнала на определенном его выходе. С помощью этого сигнала производится печать символа, соответствующего входному двоичному числу.

В самих цифровых устройствах часто возникает необходимость преобразования числовой информации из одной двоичной формы в другую (из одного

двоичного кода в другой). Примером такого преобразования может служить преобразование чисел из двоичного *кода 8421*, в котором выполняются арифметические операции, в двоичный *код 2 из 5* для передачи по линии связи. Эта задача выполняется устройствами, называемыми*преобразователями кодов.*

Шифраторы. Пусть требуется преобразовать десятичные числа 0, 1,2, ..., 9 в двоичное представление в *коде 8421.*



3.18

ТАБЛИЦА3.2

Десятичное число	Двоичный код 8421			
	x_8	x_4	x_2	x_1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	C	1

На рис. 3.18 приведено символическое изображение такого шифратора. Символ CDобразован из букв, входящих в английское слово CODER. Слева показаны 10 входов, обозначенных десятичными цифрами 0, 1, ..., 9. В зависимости от значения поступающего на вход десятичного числа, происходит подача уровня логической 1 на вход, обозначенный соответствующей цифрой. Справа показаны выходы шифратора; цифрами 1, 2, 4, 8 обозначены весовые коэффициенты двоичных разрядов, соответствующих отдельным выходам. Для каждого двоичного разряда предусмотрены два выхода — прямой и инверсный. На прямом выходе формируется логический уровень, соответствующий цифре двоичного разряда, на инверсном выходе — ее инверсия. (Наличие таких парафазных выходов не обязательно в шифраторе.)

Из приведенного в табл. 3.2 соответствия десятичного и двоичного кодов следует, что переменная x_1 на выходной шине 1 имеет уровень логической 1, если имеет этот уровень одна из входных переменных y_1, y_3, y_5, y_7, y_9. Следовательно, $x_1 = y_1 \vee y_3 \vee y_5 \vee y_7 \vee y_9$.

Для инверсного выхода логическое выражение имеет вид $\bar{x}_1 = y_0 \vee y_2 \vee y_4 \vee y_6 \vee y_8$.

Для остальных выходов

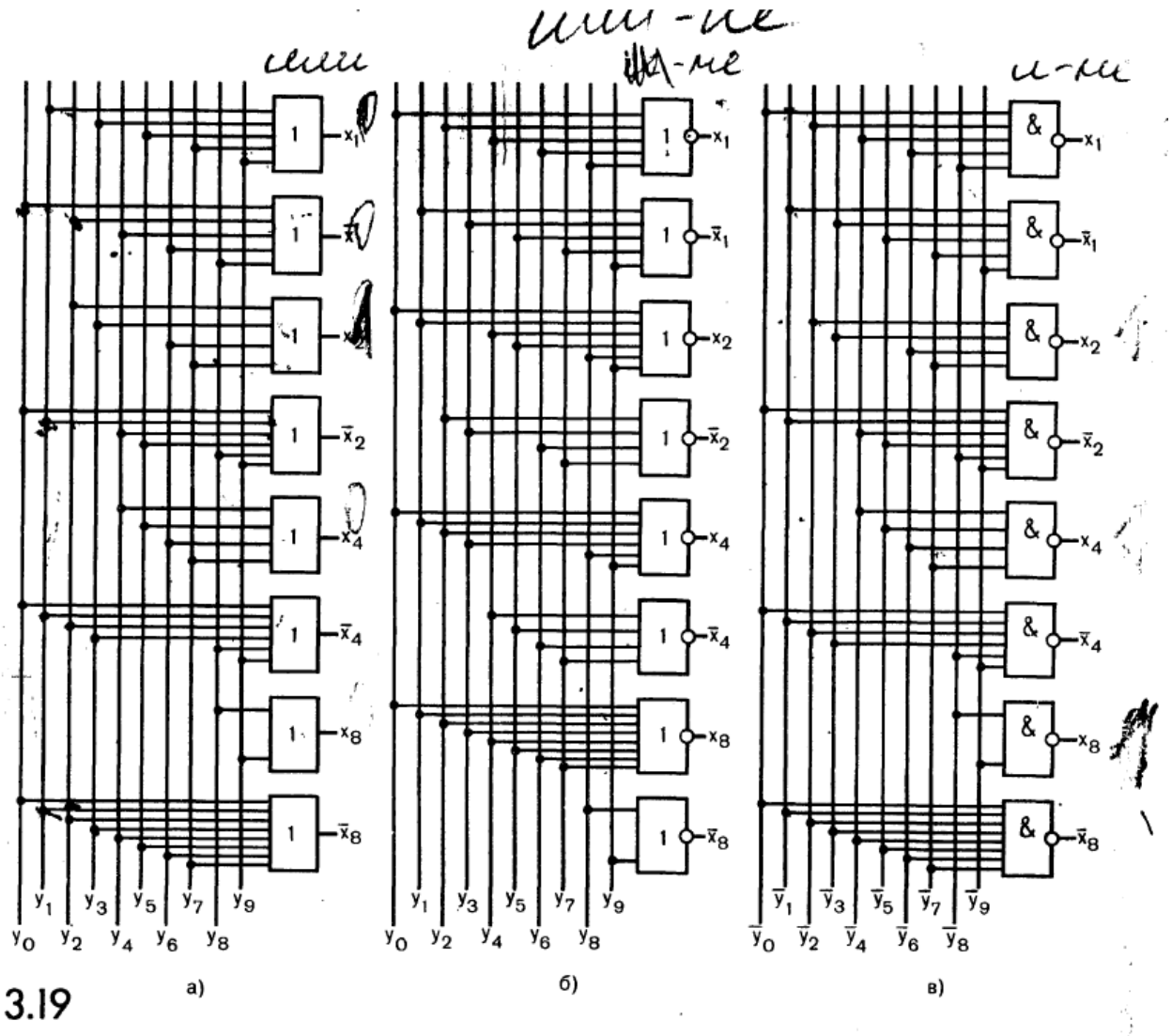
$$x_2 = y_2 \vee y_3 \vee y_6 \vee y_7;$$
$$\bar{x}_2 = y_0 \vee y_1 \vee y_4 \vee y_5 \vee y_8 \vee y_9;$$
$$x_4 = y_4 \vee y_5 \vee y_6 \vee y_7;$$

$$\bar{x}_4 = y_0 \vee y_1 \vee y_2 \vee y_3 \vee y_8 \vee y_9;$$
$$x_8 = y_8 \vee y_9;$$
$$\bar{x}_8 = y_0 \vee y_1 \vee y_2 \vee y_3 \vee y_4 \vee y_5 \vee y_6 \vee y_7.$$

Этой системе логических выражений соответствует схема на рис. 3.19*а*. На рис. 3.19*б* изображена схема шифратора на элементах ИЛИ—НЕ. При выполнении шифратора на элементах И—НЕ следует пользоваться следующей системой логических выражений:

$$x_1 = \overline{\overline{y_1 \vee y_3 \vee y_5 \vee y_7 \vee y_9}} = \overline{\bar{y}_1 \cdot \bar{y}_3 \cdot \bar{y}_5 \cdot \bar{y}_7 \cdot \bar{y}_9} = \bar{y}_1/\bar{y}_3/\bar{y}_5/\bar{y}_7/\bar{y}_9;$$

$$\bar{x}_1 = \bar{y}_0/\bar{y}_2/\bar{y}_4/\bar{y}_6/\bar{y}_8;$$
$$x_2 = \bar{y}_2/\bar{y}_3/\bar{y}_6/\bar{y}_7;$$
$$\bar{x}_2 = \bar{y}_0/\bar{y}_1/\bar{y}_4/\bar{y}_5/\bar{y}_8/\bar{y}_9;$$
$$x_4 = \bar{y}_4/\bar{y}_5/\bar{y}_6/\bar{y}_7;$$
$$\bar{x}_4 = \bar{y}_0/\bar{y}_1/\bar{y}_2/\bar{y}_3/\bar{y}_8/\bar{y}_9;$$
$$x_8 = \bar{y}_8/\bar{y}_9;$$
$$\bar{x}_8 = \bar{y}_0/\bar{y}_1/\bar{y}_2/\bar{y}_3/\bar{y}_4/\bar{y}_5/\bar{y}_6/\bar{y}_7.$$



3.19

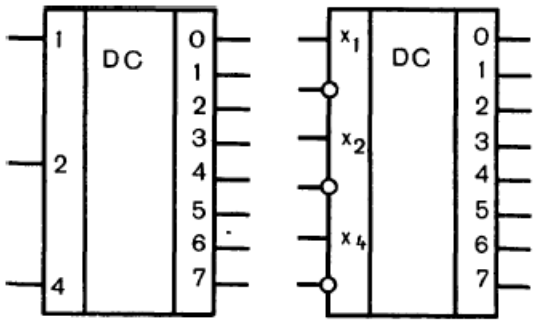
В этом случае требуется формирование инверсии входных переменных y_0, y_1 ..., y_9. Схема шифратора,выполненная на элементах И-НЕ, приведена на рис. 3.19.

Изложенным способом могут быть построены шифраторы, выполняющие преобразование десятичных чисел в двоичное представление с использованием любого двоичного кода.

Задание

1. Постройте шифратор для преобразования десятичных чисел 0, 1, 2, ... 15 в двоичный *код 8421*.

Дешифраторы. Назначение, УГО, схемная реализация, маркировка.



3.20 а) б)

На рис. 3.20 а приведено символическое изображение дешифратора. Символ DCобразован из букв английского слова DEKODER. Слева показаны входы, на которых отмечены весовые коэффициенты двоичного кода. Справа выходы, пронумерованные десятичными числами, соответствующими отдельным комбинациям входного двоичного кода. На каждом выходе образуется уровень логической 1 при строго определенной комбинации входного кода.

Дешифратор может иметь парафазные входы для подачи наряду с входными переменными их инверсий, как показано на рис. 3.206.

По способу построения различают *линейные, прямоугольные* и *пирамидальные дешифраторы.*

Л и н е й н ы й д е ш и ф р а т о р . Рассмотрим построение дешифратора, осуществляющего преобразование, заданное табл. 3.3.

Значения входных переменных определяются следующими логическими выражениями:

$$
\begin{aligned}
y_0 &= \overline{x}_8 \cdot \overline{x}_4 \cdot \overline{x}_2 \cdot \overline{x}_1;\\
y_1 &= \overline{x}_8 \cdot \overline{x}_4 \cdot \overline{x}_2 \cdot x_1;\\
y_2 &= \overline{x}_8 \cdot \overline{x}_4 \cdot x_2 \cdot \overline{x}_1;\\
y_3 &= \overline{x}_8 \cdot \overline{x}_4 \cdot x_2 \cdot x_1;\\
y_4 &= \overline{x}_8 \cdot x_4 \cdot \overline{x}_2 \cdot \overline{x}_1;\\
y_5 &= \overline{x}_8 \cdot x_4 \cdot \overline{x}_2 \cdot x_1;\\
y_6 &= \overline{x}_8 \cdot x_4 \cdot x_2 \cdot \overline{x}_1;\\
y_7 &= \overline{x}_8 \cdot x_4 \cdot x_2 \cdot x_1;\\
y_8 &= x_8 \cdot \overline{x}_4 \cdot \overline{x}_2 \cdot \overline{x}_1;\\
y_9 &= x_8 \cdot \overline{x}_4 \cdot \overline{x}_2 \cdot x_1.
\end{aligned}
\qquad (3.10)
$$

$$
\begin{aligned}
\overline{y}_0 &= \overline{x}_8 \cdot \overline{x}_4 \cdot \overline{x}_2 \cdot \overline{x}_1 = \overline{\overline{x}_8/\overline{x}_4/\overline{x}_2/\overline{x}_1};\\
y_1 &= \overline{x}_8/\overline{x}_4/\overline{x}_2/x_1;\\
y_2 &= \overline{x}_8/\overline{x}_4/x_2/\overline{x}_1;\\
\overline{y}_3 &= \overline{x}_8/\overline{x}_4/x_2/x_1;\\
\overline{y}_4 &= \overline{x}_8/x_4/\overline{x}_2/\overline{x}_1;\\
\overline{y}_5 &= \overline{x}_8/x_4/\overline{x}_2/x_1;\\
\overline{y}_6 &= \overline{x}_8/x_4/x_2/\overline{x}_1;\\
\overline{y}_7 &= \overline{x}_8/x_4/x_2/x_1;\\
\overline{y}_8 &= x_8/\overline{x}_4/\overline{x}_2/\overline{x}_1;\\
\overline{y}_9 &= x_8/\overline{x}_4/\overline{x}_2/x_1.
\end{aligned}
\qquad (3.11)
$$

В линейном дешифраторе выходные переменные формируются по (3.10) либо (3.11). При выполнении дешифратора на элементах И—НЕ пользуются (3.11), получая инверсии выходных функций. В этом случае каждой комбинации входного кода будет соответствовать уровень логического 0 на строго определенном выходе, на остальных выходах устанавливается уровень логической 1. На рис. 3.21 показана структура дешифратора, построенного на элементах И—НЕ, и его символическое изображение.

У схемы имеются особенности, характерные для дешифраторов в интегральном исполнении:

- для уменьшения числа входов формирование инверсий входных переменных производится в самом дешифраторе;
- подключенные непосредственно ко входам дополнительные инверторы уменьшают нагрузку со стороны дешифратора на его входные цепи.

Дешифратор с 16 выходами для дешифрирования всех возможных комбинаций четырехразрядного двоичного *кода 8421* можно построить из двух рассмотренных дешифраторов с 10 выходами. На рис. 3.22 показана структура такого дешифратора. В каждом из дешифраторов используется по 8 выходов, которые и образуют требуемые 16 выходов $(y_0, y_1, \ldots, y_{15})$.

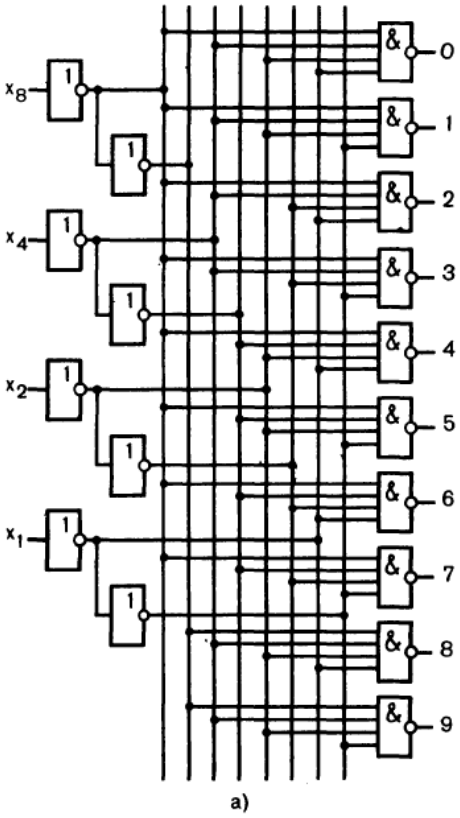
Прямоугольный дешифратор. Рассмотрим принцип построения прямоугольного дешифратора на примере дешифратора с 4 входами и 16 выходами.

Разобьем входные переменные x_8, x_4, x_2, x_1 на две группы по две переменные в каждой: x_8, x_4 и x_2, x_1. Каждую пару переменных используем в качестве входных переменных отдельного линейного дешифратора на четыре выхода, как показано на рис. 3.23а. Выходные переменные линейных дешифраторов определяются следующими логическими выражениями:

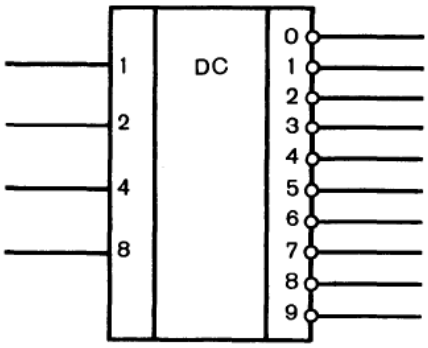
$$y'_0 = \overline{x}_8 \cdot \overline{x}_4;\ y'_3 = x_8 \cdot x_4;\ y''_2 = x_2 \cdot \overline{x}_1;$$
$$y'_1 = \overline{x}_8 \cdot x_4;\ y''_0 = \overline{x}_2 \cdot \overline{x}_1;\ y''_3 = x_2 \cdot x_1.$$
$$y'_2 = x_8 \cdot \overline{x}_4;\ y''_1 = \overline{x}_2 \cdot x_1;$$

ТАБЛИЦА 3.3

Входной код 8421				Номер выхода
x_8	x_4	x_2	x_1	
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9

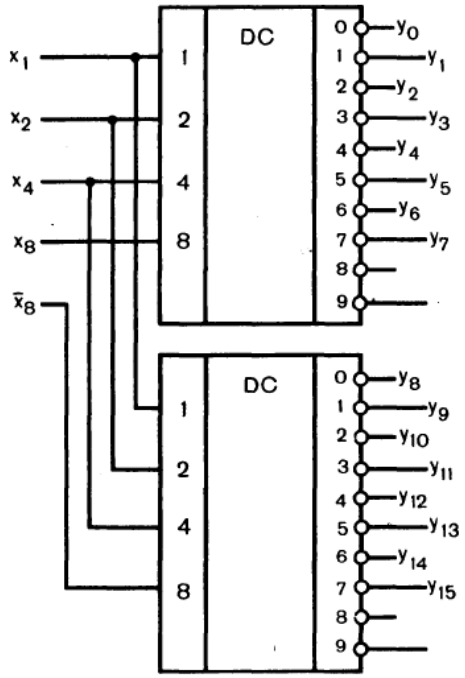


а)



б)

3.21



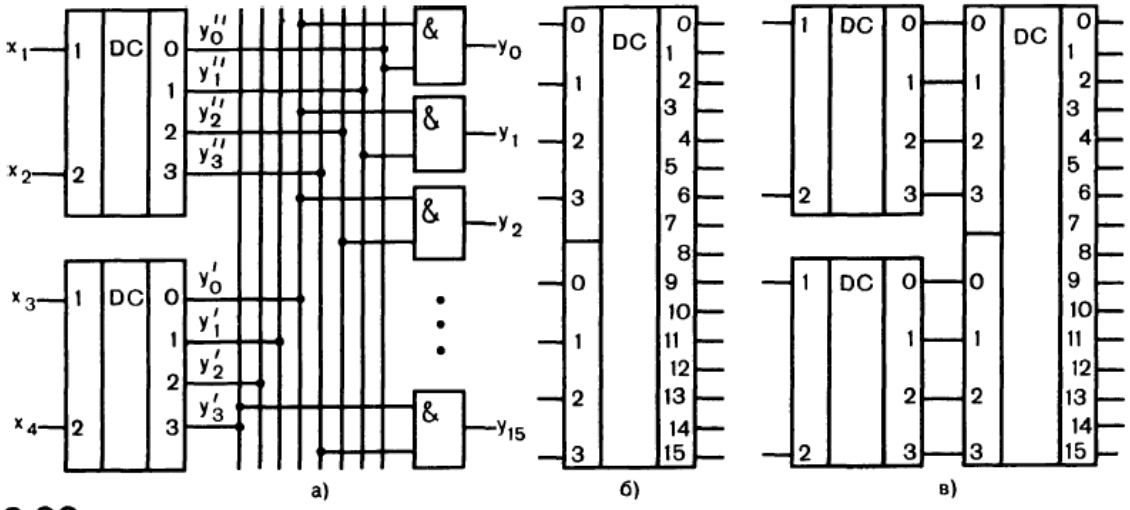
3.22

Эти дешифраторы выполняют функции первой ступени дешифратора.

Выходные переменные $y_0, y_1, \ldots, y_{15}$ прямоугольного дешифратора можно представить логическими выражениями, используя в них в качестве аргументов выходные переменные $y'_0, \ldots, y'_3$ и $y''_0, \ldots, y''_3$ линейных дешифраторов:

$$y_0 = \overline{x}_8 \cdot \overline{x}_4 \cdot \overline{x}_2 \cdot \overline{x}_1 = y'_0 \cdot y''_0;$$
$$y_1 = \overline{x}_8 \cdot \overline{x}_4 \cdot \overline{x}_2 \cdot x_1 = y'_0 \cdot y''_1;$$
$$y_2 = \overline{x}_8 \cdot \overline{x}_4 \cdot x_2 \cdot \overline{x}_1 = y'_0 \cdot y''_2;$$
$$\cdots\cdots\cdots\cdots$$
$$y_{15} = x_8 \cdot x_4 \cdot x_2 \cdot x_1 = y'_3 \cdot y''_3.$$

Эти логические операции выполняются в отдельном дешифраторе второй ступени, называемом *матричным* и состоящем из двухвходовых элементов. На рис. 3.23 б показано символическое изображение матричного дешифратора, где помеченные десятичными числами две группы входов служат для подключения к выходам двух предварительных ступеней дешифрации. На рис. 3.23 в представлена структура прямоугольного дешифратора с использованием символов линейного и матричного дешифраторов.



3.23

Задание

2. С использованием элементов ИЛИ—НЕ постройте линейный дешифратор, осуществляющий преобразование двоичных чисел в *коде 8421* в десятичные числа 0, 1, 2, ..., 9.

3. На элементах ИЛИ—НЕ постройте прямоугольный дешифратор, осуществляющий преобразование двоичных чисел в *коде 8421* в десятичные числа 0, 1, 2, ..., 15.

Лекция 11Преобразователи кодов

Преобразователи кодов, назначение и УГО.

Форма представления десятичных чисел, при которой каждая цифра числа предварительно представляется в двоичной форме, называется двоично-кодированной десятичной системой (табл. 3.1).

Таблица 3.1

Представление цифр в двоично-кодированных десятичных системах

Десятичные цифры	Двоично-кодированные десятичные системы (коды)					
	8421	2421	7421	С избытком 3	3а + 2	2 из 5
0	0000	0000	0000	0011	00010	11000
1	0001	0001	0001	0100	00101	01100
2	0010	0010	0010	0101	01000	00110
3	0011	0011	0011	0110	01011	00011
4	0100	0100	0100	0111	01110	10001
5	0101	1011	0101	1000	10001	10100
6	0110	1100	0110	1001	10100	01010
7	0111	1101	1000	1010	10111	00101
8	1000	1110	1001	1011	11010	10010
9	1001	1111	1010	1100	11101	01001

Примечание. Код 8421 (название кода составлено из весовых коэффициентов разрядов двоичного числа) — естественное представление десятичных цифр в двоичной системе счисления. Код 7421 интересен тем, что любая кодовая комбинация содержит не более двух единиц. В коде «2 из 5» все кодовые комбинации содержат точно две единицы. Это свойство используется для обнаружения ошибочных комбинаций. В коде 2421 и коде «с избытком 3» кодовая комбинация, соответствующая любой из десятичных цифр, предстанляет собой инверсию комбинации, соответствующей ее дополнению до дешпн. Код 3а + 2 имеет полезное свойство: любая пара кодовых комбинаций отличается не менее чем в двух разрядах, что позволяет обнаруживать ошибочные комбинации (ошибка, изменяющая цифру одного разряда любой из кодов комбинаций, приводит к так называемой запрещенной комбинации, не и пользуемой для представления десятичных цифр в этом коде).

В ЦУ возникает необходимость преобразования (перекодирования) числовой информации из одного кода в другой. ИМС, выполняющие эти операции, называются *преобразователями кодов.*

Для преобразования кодов можно пользоваться двумя методами:

- методом, основанным на преобразовании исходного двоичного кода в десятичный и последующего преобразования десятичного представления в требуемый двоичный код;

- методом использования логического устройства комбинационного типа, непосредственно реализующего данное преобразование.

Первый метод структурно реализуется соединением дешифратора и шифратора и удобен в случаях, когда можно использовать стандартные дешифраторы и шифраторы в интегральном исполнении.

Рассмотрим подробнее второй метод на конкретных примерах преобразования двоичных кодов.

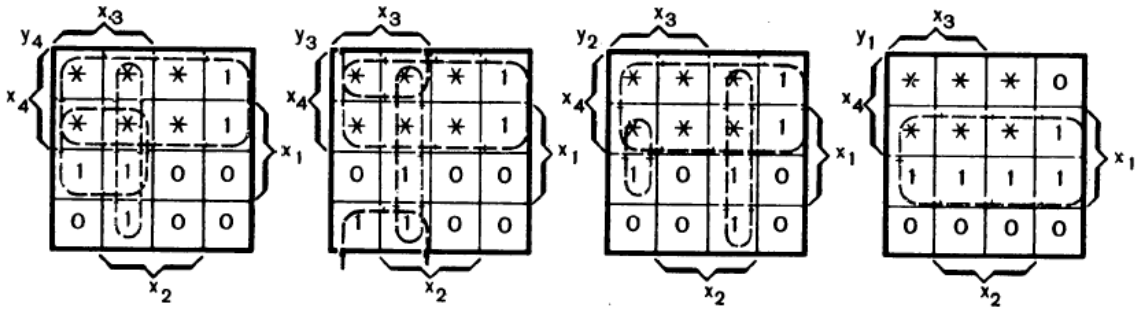
П р е о б р а з о в а н и е *кода 8421* в *код 2421.* Обозначим переменные, соответствующие отдельным разрядам *кода 8421*,— x_4, x_3, x_2, x_1 то же для *кода 2421* — y_4, y_3, y_2, y_1. В табл. 3.4 приведено соответствие комбинаций обоих кодов.

ТАБЛИЦА 3.4

Код 8421				Код 2421			
x_4	x_3	x_2	x_1	y_4	y_3	y_2	y_1
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	1
0	1	0	0	0	1	0	0
0	1	0	1	1	0	1	1
0	1	1	0	1	1	0	0
0	1	1	1	1	1	0	1
1	0	0	0	1	1	1	0
1	0	0	1	1	1	1	1

Каждая из переменных y_4, y_3, y_2, y_1.может рассматриваться функцией аргументов x_4, x_3, x_2, x_1 и следовательно,может быть представлена через эти аргументы соответствующим логическим выражением. Для получения указанных логических выражений представим переменные y_4, y_3, y_2, y_1. Таблицами истинности в форме таблицы Вейча (табл.3.5).

ТАБЛИЦА 3.5



Получим минимальную форму логических выражений, представленных через операции И, ИЛИ, НЕ и через операцию И—НЕ

$$y_4 = x_4 \vee x_3x_2 \vee x_3x_1;$$
$$y_3 = x_4 \vee x_3x_2 \vee x_3\overline{x}_1;$$
$$y_2 = x_4 \vee \overline{x}_3x_2 \vee x_3\overline{x}_2x_1;$$
$$y_1 = x_1.$$

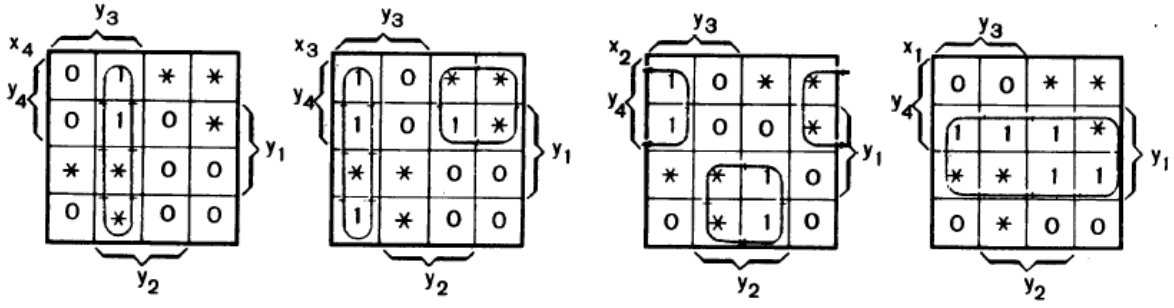
$$y_4 = \overline{x}_4/(x_3/x_2)/(x_3/x_1);$$
$$y_3 = \overline{x}_4/(x_3/x_2)/(x_3/\overline{x}_1);$$
$$y_2 = \overline{x}_4/(\overline{x}_3/x_2)/(x_3/\overline{x}_2/x_1);$$
$$y_1 = x_1.$$

На рис. 3.24 приведена логическая структура преобразователя кодов, построенная на элементах И—НЕ с использованием полученных логических выражений.

П р е о б р а з о в а н и е *кода 2421* в *код 8421*. Для реализации данного преобразования (обратного по отношению к рассмотренному выше) требуется получить логические выражения для переменных x_4, x_3, x_2. X_1, используя в качестве аргументов переменные y_4, y_3, y_2, y_1.

Таблицы Вейча для переменных x_4, x_3, x_2, x_1 представлены табл. 3.6.

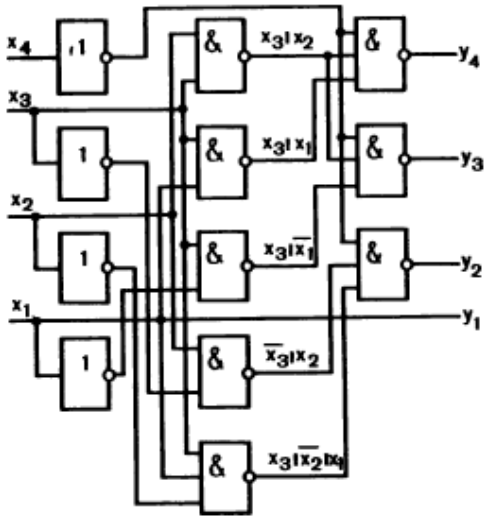
ТАБЛИЦА 3.6



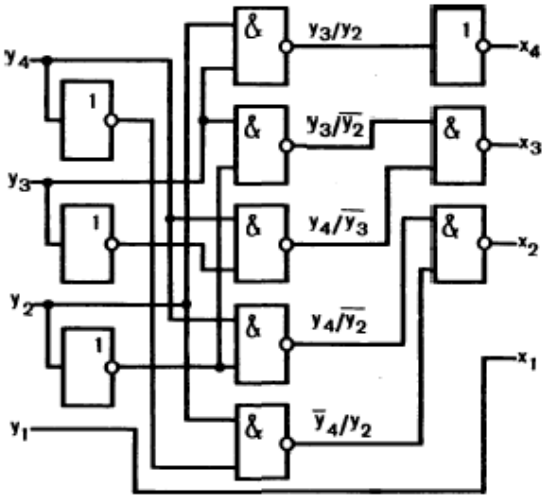
Логические выражения для переменных x_4, x_3, x_2, x_1:

$$x_4 = y_3 y_2; \qquad x_4 = \overline{y_3 / y_2};$$
$$x_3 = y_3 \overline{y}_2 \vee y_4 \overline{y}_3; \qquad x_3 = (y_3 / \overline{y}_2) / (y_4 / y_3);$$
$$x_2 = y_4 \overline{y}_2 \vee \overline{y}_4 y_2; \qquad x_2 = (y_4 / \overline{y}_2) / (\overline{y}_4 / y_2);$$
$$x_1 = y_1. \qquad x_1 = y_1.$$

Логическая структура преобразователя приведена на рис. 3.25.



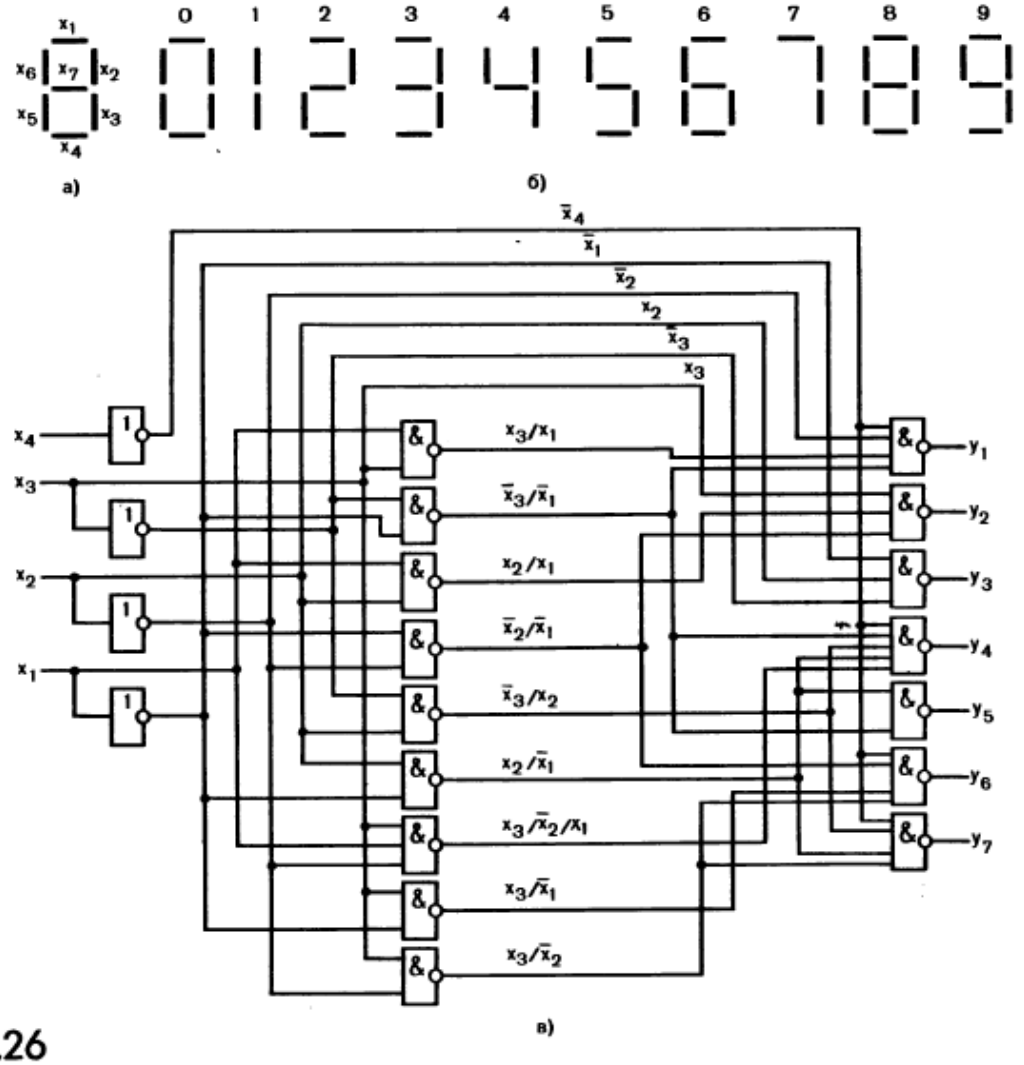
3.24



3.25

+Построить самой схему проще согласно методичке.

Преобразователь для цифровой индикации.



3.26

Один из способов цифровой индикации состоит в следующем.

Имеется семь элементов, расположенных так, как показано на рис. 3.26*а*. Каждый элемент может светиться либо не светиться, в зависимости от значения соответствующей логической переменной, управляющей свечением данного элемента. Вызывая свечение элементов в определенных комбинациях, можно получить изображение десятичных цифр 0, 1, ..., 9 (рис. 3.26*б*).

ТАБЛИЦА 3.7

Десятичная цифра	Двоичный код 8421				Переменные, управляющие элементами индикатора						
	x_4	x_3	x_2	x_1	y_1	y_2	y_3	y_4	y_5	y_6	y_7
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	1	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	1	0	1	1

ТАБЛИЦА 3.8

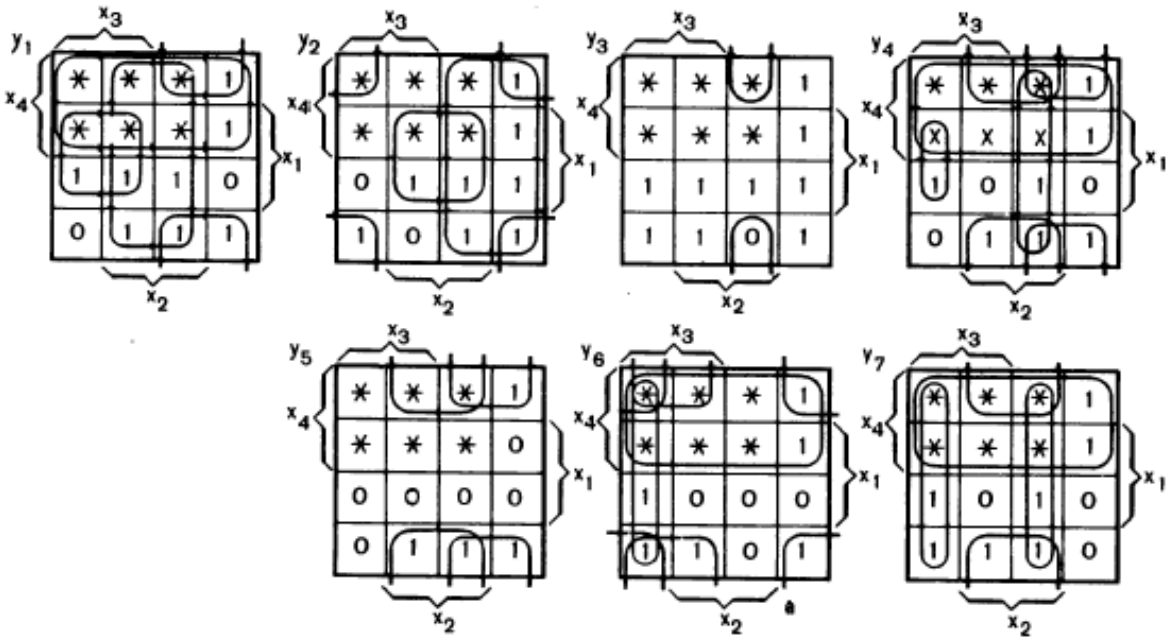


Таблица истинности для этих переменных представлена в табл. 3.7.

Таблицы истинности для отдельных переменных в форме таблиц Вейча представлены в табл. 3.8. Логические выражения переменных:

$$y_1 = x_4 \vee x_2 \vee x_3x_1 \vee \overline{x}_3\overline{x}_1; \qquad y_1 = \overline{x}_4/\overline{x}_2/(x_3/x_1)/(\overline{x}_3/\overline{x}_1);$$

$$y_2 = \overline{x}_3 \vee x_2x_1 \vee \overline{x}_2\overline{x}_1; \qquad y_2 = x_3/(x_2/x_1)/(\overline{x}_2/\overline{x}_1);$$

$$y_3 = x_3 \vee \overline{x}_2 \vee x_1; \qquad y_3 = \overline{x}_3/x_2/\overline{x}_1;$$

$$y_4 = x_4 \vee \overline{x}_3x_2 \vee x_2\overline{x}_1 \vee \overline{x}_3\overline{x}_1 \vee \vee x_3\overline{x}_2x_1; \qquad y_4 = \overline{x}_4/(\overline{x}_3/x_2)/(x_2/\overline{x}_1)/(\overline{x}_3/\overline{x}_1)/ /(x_3/\overline{x}_2/x_1);$$

$$y_5 = \overline{x}_3\overline{x}_1 \vee x_2\overline{x}_1; \qquad y_5 = (\overline{x}_3/\overline{x}_1)/(x_2/\overline{x}_1);$$

$$y_6 = x_4 \vee x_3\overline{x}_1 \vee \overline{x}_2\overline{x}_1 \vee x_3\overline{x}_2; \qquad y_6 = \overline{x}_4/(x_3/\overline{x}_1)/(\overline{x}_2/\overline{x}_1)/(x_3/\overline{x}_2);$$

$$y_7 = x_4 \vee \overline{x}_3x_2 \vee x_3\overline{x}_2 \vee x_2\overline{x}_1. \qquad y_7 = \overline{x}_4/(\overline{x}_3/x_2)/(x_3/\overline{x}_2)/(x_2/\overline{x}_1).$$

Логическая структура преобразователя приведена на рис. 3.26*в*.

Задание

4. Постройте преобразователи двоичного *кода 8421* в следующие виды кодов (см. табл. 3.1):

а) *2 из 5*; б) *с избытком 3*; в) *3а + 2*; г) *7421*.

5. Постройте преобразователи, осуществляющие преобразование приведенных ниже видов кода в двоичный *код 8421*:

а) *2 из 5*; б) *с избытком 3*; в) *3а + 2*; г) *7421*.

Лекция 12Мультиплексоры и демультиплексоры

Мультиплексоры, назначение, маркировка, УГО

Назначение и принцип работы мультиплексора. Мультиплексор имеет несколько информационных входов $(D_0, D_1, \ldots)$, адресные входы $(A_0,$, вход для подачи стробирующего сигнала С и один выход Q. На рис. 3.27*а* показано символическое изображение мультиплексора с четырьмя информационными входами.

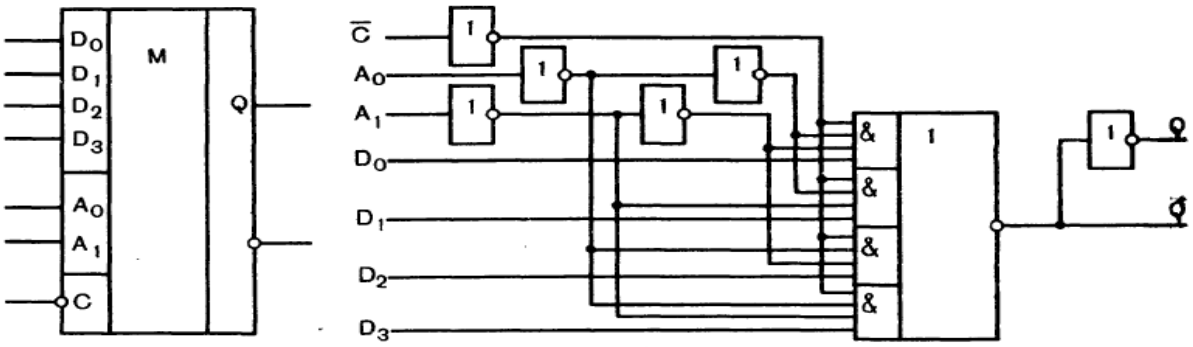
Каждому из информационных входов мультиплексора присваивается номер, называемый адресом. При подаче стробирующего сигнала на вход С мультиплексор выбирает один из входов адрес которого задается двоичным кодом на адресных входах, и подключает его к выходу.

Таким образом, подавая на адресные входы адреса различных информационных входов, можно передавать цифровые сигналы с этих входов на выход Q. Очевидно, число информационных входов $n_{\text{инф}}$ и число адресных входов $n_{адр}$ связаны соотношением $n_{\text{инф}} = 2^{n_{адр}}$.

Функционирование мультиплексора определяется табл. 3.9

ТАБЛИЦА 3.9

Адресные входы		Стробирующий сигнал	Выход
A_1	A_0	C	Q
X	X	0	0
0	0	1	D_0
0	1	1	D_1
1	0	1	D_2
1	1	1	D_3



3.27

При отсутствии стробирующего сигнала (C=0) связь между информационными входами и выходом отсутствует (Q = 0). При подаче стробирующего сигнала (C=1) на выход передается логический уровень того из информационных входов D_i, номер которого*i* в двоичной форме задан на адресных входах. Так,при задании адреса $A_1A_0 = 11_2 = 3_{10}$ на выход Q будет передаваться сигнал информационного входа с адресом 3_{10}, т. е. D_3.

По этой таблице можно записать следующее логическое выражение для выхода Q:

$$Q = (D_0 \cdot \overline{A}_1 \cdot \overline{A}_0 \vee D_1 \cdot \overline{A}_1 \cdot A_0 \vee D_2 \cdot A_1 \overline{A}_0 \vee \vee D_3 (A_1 \cdot A_0) \cdot C. \qquad (3.12)$$

Построенная по этому выражению принципиальная схема мультиплексора показана на рис. 3.27 *б*.

Мультиплексоры обычно компонуются с четырьмя двухвходовыми, двумя четырехвходовыми или одним восьмивходовым в одном корпусе.

В тех случаях, когда требуется передавать на выходы многоразрид ные входные данные в параллельной форме, используется параллельное включение мультиплексоров по числу разрядов передаваемых данных.

Использование мультиплексоров для синтеза комбинационных устройств. Мультиплексоры могут быть использованы для синтеза логических функций. При этом может быть достигнуто значительное сокращение числа используемых в схеме элементов (корпусов интегральных схем).

Логическое выражение мультиплексора (3.12) содержит члены со всеми комбинациями адресных переменных. Следовательно, если требуется

синтезировать функцию трех переменных $f(x_1, x_2, x_3)$, то две из этих переменных (например, x_1, x_2) могут быть поданы на адресные входы A_1 и A_0, а третья x_3 — на информационный вход.

Например, пусть требуется синтезировать функцию, заданную табл. 3.10. Логическое выражение функции) $f(x_1, x_2, x_3) = x_1x_2x_3 \vee x_1\bar{x}_2x_3 \vee \bar{x}_1x_2x_3 \vee \bar{x}_1\bar{x}_2\bar{x}_3$.

ТАБЛИЦА 3.10

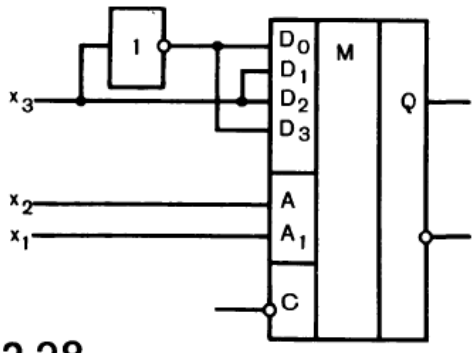
x_1 \ x_2				
	1	0	1	0
	0	1	0	1

(x_3)

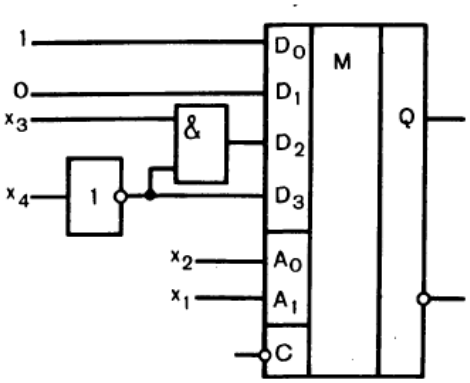
ТАБЛИЦА 3.11

Адресные переменные		Информационные входы	Выход Q
x_1	x_2		
0	0	$D_0 = \bar{x}_3$	$\bar{x}_1\bar{x}_2\bar{x}_3$
0	1	$D_1 = x_3$	$\bar{x}_1x_2x_3$
1	0	$D_2 = x_3$	$x_1\bar{x}_2x_3$
1	1	$D_3 = \bar{x}_3$	$x_1x_2\bar{x}_3$

Рассматривая переменные x_1, x_2 в качестве адресных переменных, получим табл. 3.11, из которой видно, что мультиплексор на выходе Q реализует заданную логическую функцию. Принципиальная схема показана на рис. 3.28



3.28



3.29

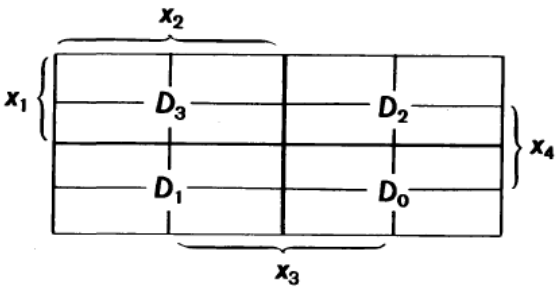
При синтезе комбинационных схем мультиплексоры могут быть использованы совместно с элементами некоторого базиса. Если число адресных входов мультиплексора $n_{адр}$, то из общего числа n переменных функции $n_{адр}$ могут быть поданы на адресные входы. Тогда на информационные входы мультиплексора подаются функции n —$n_{адр}$ переменных.

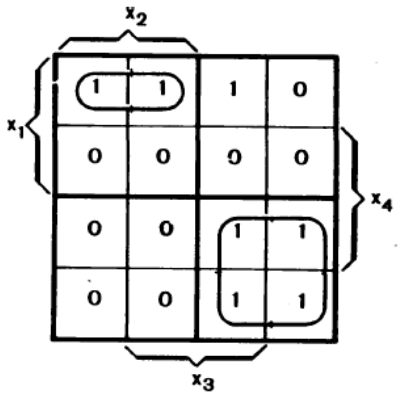
Пусть, например, требуется синтезировать логическую функцию четырех переменных с использованием четырехвходовогомультиплексора. Если адресными переменными являются x_1x_2, то на информационные входы мультиплексора должны подаваться функции переменных x_3 и x_4, определяемые показанными в табл. 3.12 областями таблицы Вейча. Внутри каждой очерченной для информационных входов области таблицы Вейча проводится минимизация обычными методами, после чего строятся схемы, формирующие подаваемые на информационные входы мультиплексора функции.

Покажем этот прием на реализации функции, заданной табл. 3.13.

ТАБЛИЦА 3.13

ТАБЛИЦА 3.12





При подаче переменных x_1 и x_2 на адресные входы мультиплексорана его информационные входы должны подаваться $D_0 = 1;\ D_1 = 0;$ $D_2 = x_3\bar{x}_4;\ D_3 = \bar{x}_4.$

Реализующая заданную функцию схема показана на рис. 3.29.

Следует иметь в виду, что, синтезируя логическое устройство с использованием мультиплексора, необходимо также построить вариант схемы без использования мультиплексора. Затем сравнением полученных вариантов определить, какой из вариантов оказывается лучшим по числу используемых в схеме корпусов интегральных схем.

Пример. Построить логическую схему мультиплексора 4 → 1.

1. По заданному числу входных информационных каналов определим число адресных входов m, используя формулу

$$n = 2^m \Rightarrow 4 = 2^2 \Rightarrow m = 2,$$

где m — число адресных входов;

n — число информационных входов MUX.

Таблица 3.10

Таблица истинности

A_1	A_0	Y
0	0	X_0
0	1	X_1
1	0	X_2
1	1	X_3

2. Составим ТИ состояния входов выходов MUX (табл. 3.10).

3. Используя ТИ, получим выражение для его выходной функции:

$$Y = X_0\left(\bar{A}_0\bar{A}_1\right) + X_1\left(A_0\bar{A}_1\right) + X_2\left(\bar{A}_0A_1\right) + X_3\left(A_0A_1\right).$$

4. По уравнению п. 3 построим схему MUX в простом базисе (рис. 3.12).

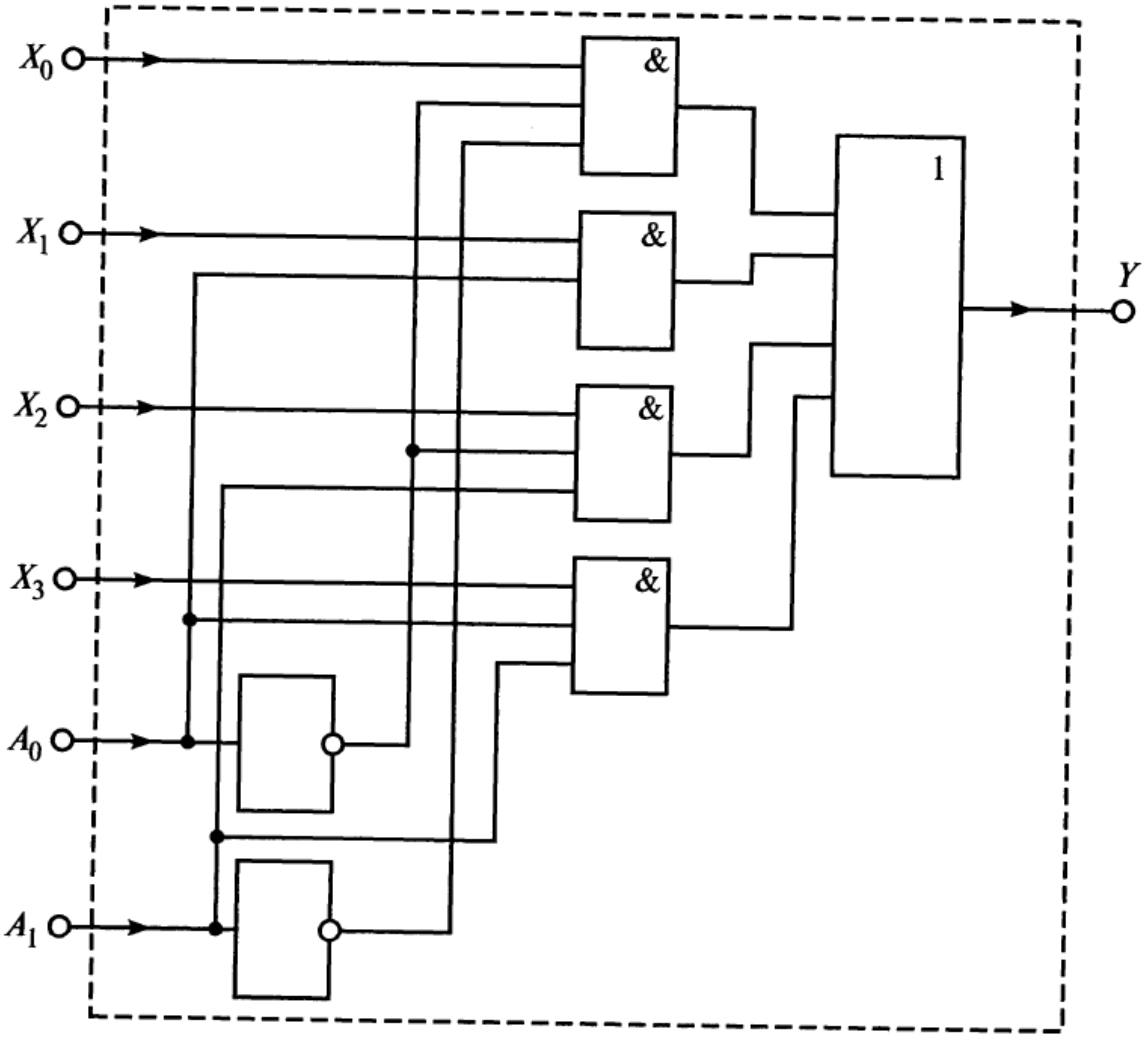


Рис. 3.12. Схема MUX 4 → 1

Для расширения числа входных линий используют *каскадирование МУХ*. На рис. 3.13 (табл. 3.11) показана схема MUX 16 → 1, выполненная на MUX 8 → 1. При значении адресного сигнала $A_3 = 0$ включается микросхема DD1, а при значении $A_3 = 1$ — DD2. При включении DD1 на общий выход поступает один из информационных сигналов X_0—X_7, подключенных к входам DD1. При включении DD2 на общий выход поступают сигналы X_8—X_{15}.

Таблица 3.11

Таблица — спецификация к схеме

Обозначение на схеме	ИМС	Количество	Коэффициент использования
DD1, DD2	К155КП7	2	1/1
DD3	К155ЛН1	1	1/6

Пример. Задан адресный код 1010_2 (см. рис. 3.13). Он разбивается на младшие разряды ($A_0 = 0$, $A_1 = 1$, $A_2 = 0$), которые поступают на адресные входы DD1, DD2 и выбирают в них X_2 канал. Старший разряд ($A_3 = 1$) поступает

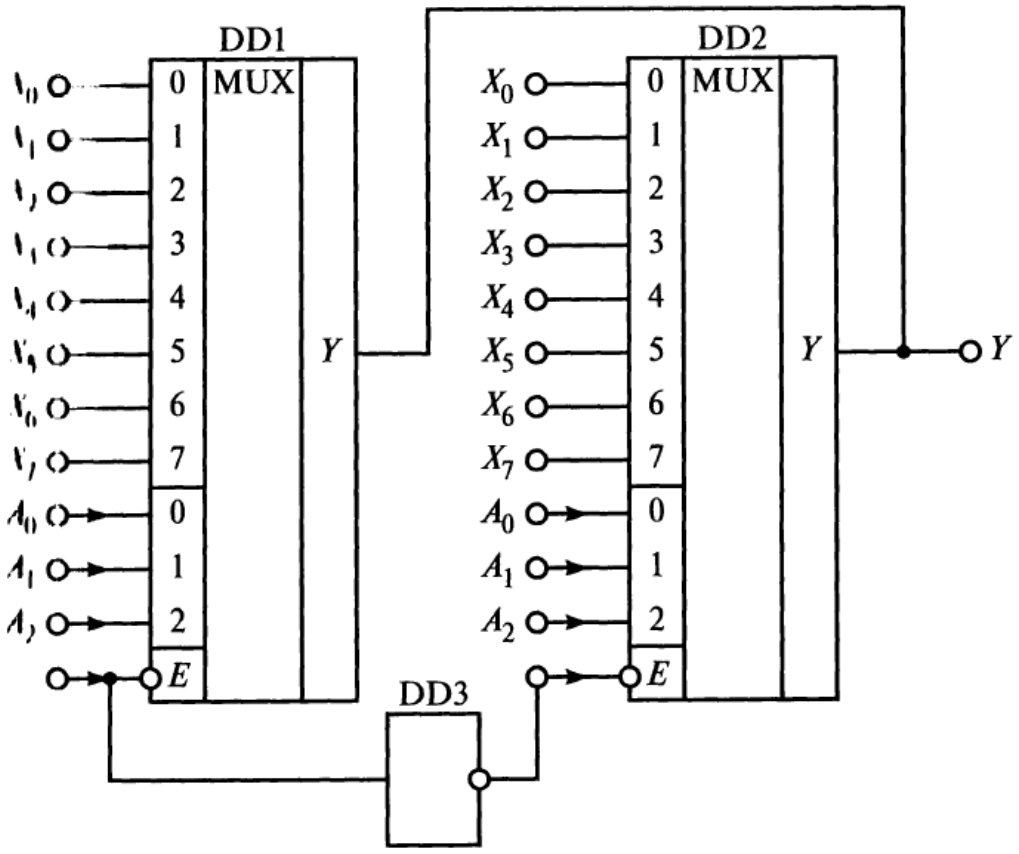


Рис. 3.13. Схема каскадного MUX (16 → 1)

на стробирующий инверсный вход DD1, запрещая его работу, и на вход инвертораDD3, где инвертируется и включает схему DD2. Полученные выходные сигналы DD1, DD2 поступают на DD4. Таким образом информация на выход *У*поступает с 10-го канала.

Алгоритм построения схемы каскадного MUXна базисных ИМС:

1. По заданному числу входных информационных каналов определить число адресных входов *m*, используя формулу $n = 2^m$.
2. В соответствии с найденным числом адресных входов записать адресный код и разбить его на старшие и младшие разряды.
3. Определить количество базовых ИМС *N,* на которые подаются младшие разряды адресных сигналов, используя формулу

$$N = n/n^*,$$

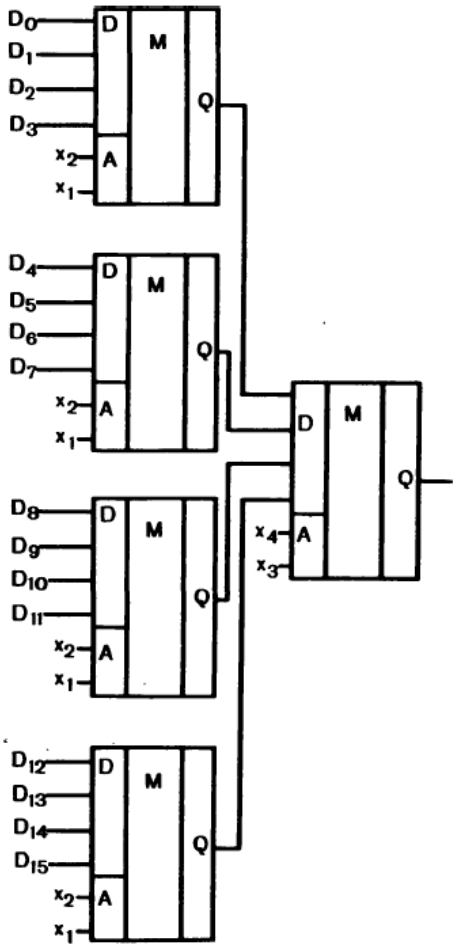
где *n* — число заданных информационных каналов;

*n** — число информационных каналов в базовой ИМС.

4. Продумать и соединить ИМС в соответствии с логикой построения.

Мультиплексорное дерево. Максимальное число входов мультиплексоров, выполненных в виде интегральных схем, равно восьми. Если требуется построить мультиплексорное устройство с большим числом входов, можно объединить мультиплексоры в схему так называемого дерева. Такое мультиплексорное дерево, построенное на четырехвходовых мультиплексорах, показано на рис. 3.30. Схема состоит из четырех мультиплексоров первого уровня с адресными переменными $x_1 x_2$ и мультиплексора второго уровня с адресными переменными $x_3 x_4$. Мультиплексорное устройство имеет 16 входов, разбитых на четверки, подключенные к отдельным мультиплексорам первого уровня. Мультиплексор второго уровня, подключая к общему выходу устройства выходы отдельных мультиплексоров первого уровня, переключает четверки входов. Внутри же четверки требуемый вход выбирается мультиплексором первого уровня. По такой схеме, используя восьмивходовые

мультиплексоры, можно построить мультиплексорное устройство, имеющее 64 входа.



3.30

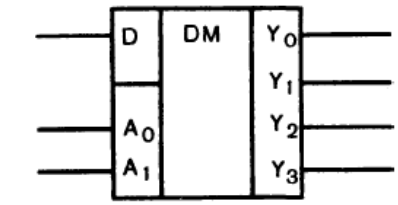
В первом и втором уровнях мультиплексорного дерева можно использовать мультиплексоры с разным числом входов. Если в первом уровне такого дерева используются мультиплексоры с числом адресных переменных $n_{адр1}$, а во втором—с числом переменных $n_{адр2}$, то общее число входов мультиплексорного дерева будет равно $n_{инф} = 2^{n_{адр1} + n_{адр2}}$, а число мультиплексоров в схеме составит $2^{n_{адр2}} + 1$.

Демультиплексоры, назначение, маркировка, УГО

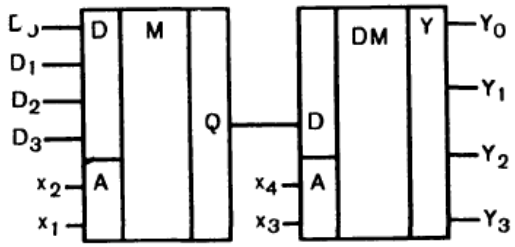
Демультиплексор – функциональный узел,который обеспечивает передачу цифровой информации,поступающей по одной линии, на несколько выходных линий. Выбор выходной линии осуществляется при помощи сигналов,поступающих на адресные входы. DMосуществляет обратные преобразования мультиплексора.

Демультиплексор является устройством, которое имеет один вход и несколько выходов. Вход подключается к выходу, имеющему заданный адрес. На рис. 3.31 показано символическое изображение демультиплексора с

четырьмя выходами. Функционирование этого демультиплексора определяется табл. 3.14.



3.31



3.32

Объединяя мультиплексор с демультиплексором, можно построить устройство, в котором по заданным адресам один из входов подключается к одному из выходов (рис. 3.32).

Таким образом, может быть выполнена любая комбинация соединений входов с выходами.

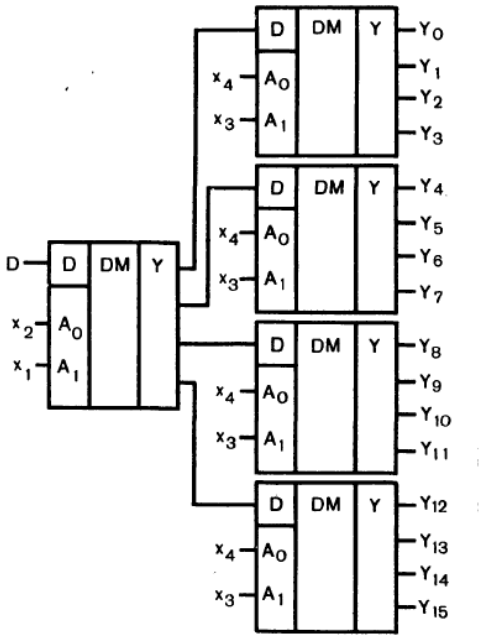
ТАБЛИЦА 3.14

Адресные входы		Выходы			
A_1	A_0	Y_0	Y_1	Y_2	Y_3
0	0	*D*	0	0	0
0	1	0	*D*	0	0
1	0	0	0	*D*	0
1	1	0	0	0	*D*

Например, при комбинации значений адресных переменных $x_1 = 1$, $x_2 = 0$, $x_3 = 0$, $x_4 = 0$ вход D, окажется подключенным к выходу Y_0.

Использование демультиплексораможет существенно упростить логическое устройство, имеющее несколько выходов, на которых формируются различные логические функции одних и тех же переменных.

Заметим, что если на вход демультиплексора подавать константуD=1, то на выбранном в соответствии с заданным адресом выходе будет логическая 1, на остальных выходах — логический 0. При этом по выполняемой функции демультиплексор превращается в дешифратор.



3.33

При необходимости иметь большое число выходов может быть построено *демулътиплексорное дерево*.. На рис. 3.33 показано такое дерево,построенное на демультиплексорах с четырьмя выходами. Демультиплексор первого уровня подключает вход Dк определенной четверке выходов, демультиплексоры второго уровня выбирают нужный выход в четверке, куда и передается сигнал с входа D.

Входной сигнал X поступает на вход коммутатора и через него передается на выходы Y_0—Y_n. Адресные сигналы A_0—A_m имеют то же назначение, что и у MUX. Сигнал стробированияE разрешает передачу входного сигнала через коммутатор (рис. 3.15).

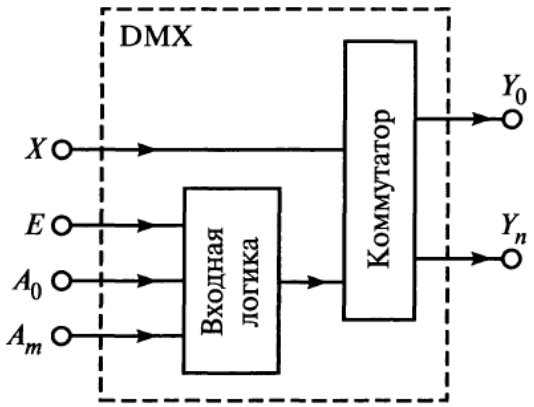


Рис. 3.15. Демультиплексор: *а* — обобщенная схема; *б* — условное обозначение

Для обозначения коммутационных возможностей DMXпользуются записью $1 \to n,$, где n — число выходов DMX. Они могут быть полными и неполными.

Пример. Построить логическую схему DMX $1 \to 4.$

1.По заданному числу входных информационных каналов определим число адресных входов m, используя формулу

$$n = 2^m \Rightarrow 4=2^2 \Rightarrow m = 2,$$

где m — число адресных входов;

n — число информационных входов DMX.

2.Составим ТИ (табл. 3.13) состояния входов выходов DMX.

Таблица 3.13

Таблица истинности

A_1	A_0	Y_0	Y_1	Y_2	Y_3
0	0	X	0	0	0
0	1	0	X	0	0
1	0	0	0	X	0
1	1	0	0	0	X

3.Используя данные ТИ, получим выражение для выходных сигналов DMX:

$$Y_0 = X\left(\overline{A}_0 \cdot \overline{A}_1\right) = \overline{\overline{X} + A_0 + A_1};\ Y_1 = X\left(A_0 \cdot \overline{A}_1\right) = \overline{\overline{X} + \overline{A}_0 + A_1};$$
$$Y_2 = X\left(\overline{A}_0 \cdot A_1\right) = \overline{\overline{X} + A_0 + \overline{A}_1};\ Y_3 = X\left(A \cdot A_1\right) = \overline{\overline{X} + \overline{A}_0 + \overline{A}_1}.$$

4.Построим схему на элементах «И» (рис.3.16)используя первую часть выходных уравнений п.3 и на элементах «И-НЕ» по второй части уравнений п.3 (рис.3.17)

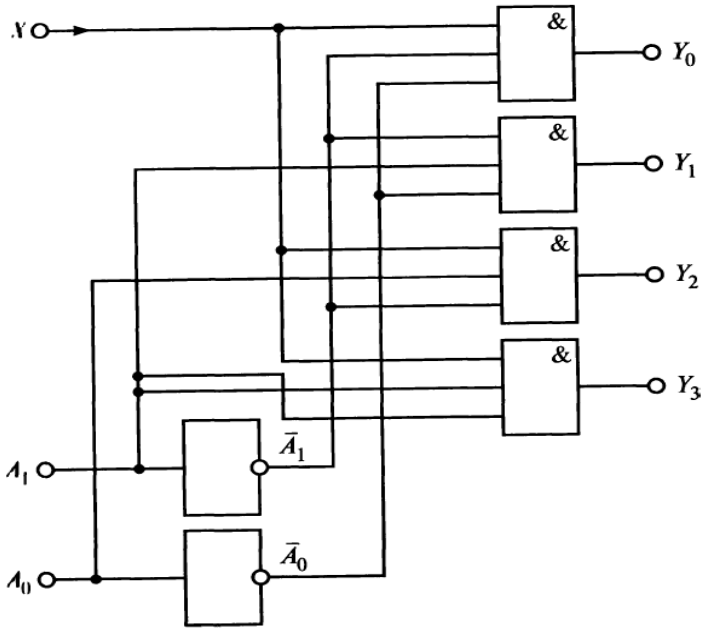


Рис. 3.16. Схема демультиплексора на элементах «И»

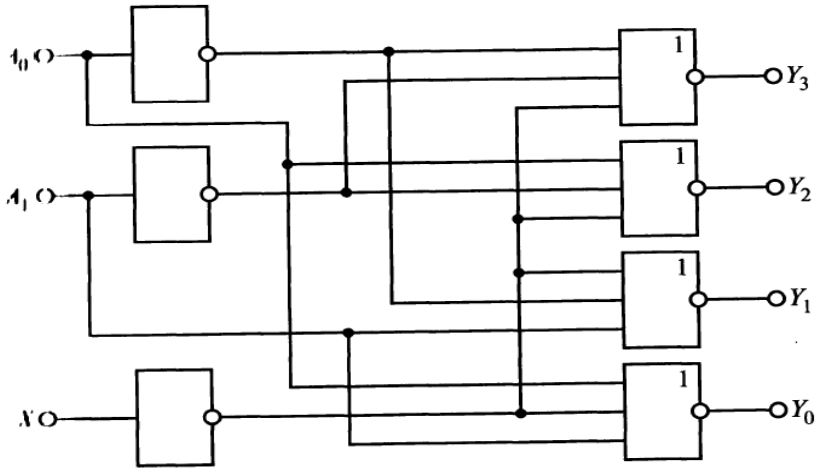
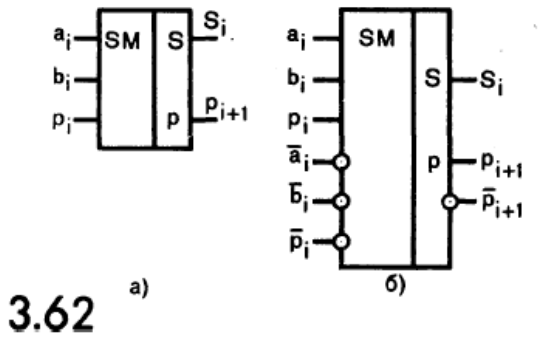


Рис. 3.17. Схема демультиплексора на элементах «ИЛИ-НЕ»

Лекция 13Комбинационные двоичные сумматоры

Сумматоры, назначение, классификация, УГО

Одноразрядный двоичный сумматор. Из рассмотренного в § 3.2 принципа сложения многоразрядных двоичных чисел следует, что в каждом из разрядов производятся однотипные действия: определяется цифра суммы путем сложения по модулю 2 цифр слагаемых и поступающего в данный разряд переноса и формируется перенос, передаваемый в следующий разряд. Эти действия реализуются так называемым ***одноразрядным двоичным сумматором.*** Символическое изображение такого сумматора показано на рис. 3.62 а. Одноразрядный сумматор имеет три входа для подачи цифр разрядов слагаемых a_i, b_i и переноса p_i; на выходах формируются сумма s_i и перенос p_{i+1}, предназначенный для передачи в следующий разряд. **В** одноразрядном сумматоре могут предусматриваться входы для подачи как прямых a_i, b_i, p_i, ,так и инверсных значений $\overline{a}_i$, $\overline{b}_i$, $\overline{p}_i$ входных переменных, а также выходы, на которых формируются инверсные значения выходных переменных. Пример символа такого одноразрядного сумматора приведен на рис. 3.62 б.



3.62

ТАБЛИЦА 3.30

Входы			Выходы	
Слагаемые		Перенос	Сумма	Перенос
a_i	b_i	p_i	s_i	p_{i+1}
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1

В табл. 3.30 показано функционирование одноразрядного сумматора. Пользуясь этой таблицей истинности, запишем логические выражения для выходных величин s_i и p_{i+1} в различных базисах:
в базисе И — Н Е

$$s_i = \overline{\bar{a}_i \cdot b_i \cdot \bar{p}_i \vee a_i \cdot \bar{b}_i \cdot \bar{p}_i \vee \bar{a}_i \cdot \bar{b}_i \cdot p_i \vee a_i \cdot b_i \cdot p_i} =$$
$$= \overline{(\bar{a}_i \cdot b_i \cdot \bar{p}_i) \cdot (a_i \cdot \bar{b}_i \cdot \bar{p}_i) \cdot (\bar{a}_i \cdot \bar{b}_i \cdot p_i) \cdot (a_i \cdot b_i \cdot p_i)} =$$
$$= (\bar{a}_i / b_i / \bar{p}_i) / (a_i / \bar{b}_i / \bar{p}_i) / (\bar{a}_i / \bar{b}_i / p_i) / (a_i / b_i / p_i);$$

$$p_{i+1} = a_i \cdot b_i \cdot \bar{p}_i \vee \bar{a}_i \cdot b_i \cdot p_i \vee a_i \cdot \bar{b}_i \cdot p_i \vee a_i \cdot b_i \cdot p_i = a_i \cdot b_i \vee a_i \cdot p_i \vee b_i \cdot p_i =$$
$$= a_i \cdot b_i \vee (a_i \vee b_i) \cdot p_i = \overline{(a_i b_i) \cdot (a_i \vee b_i) \cdot p_i} = \overline{(a_i \cdot b_i) \cdot (\bar{a}_i \cdot \bar{b}_i) \cdot p_i} =$$
$$= (a_i / b_i) / (\bar{a}_i / \bar{b}_i) / p_i;$$

в базисе ИЛИ—НЕ

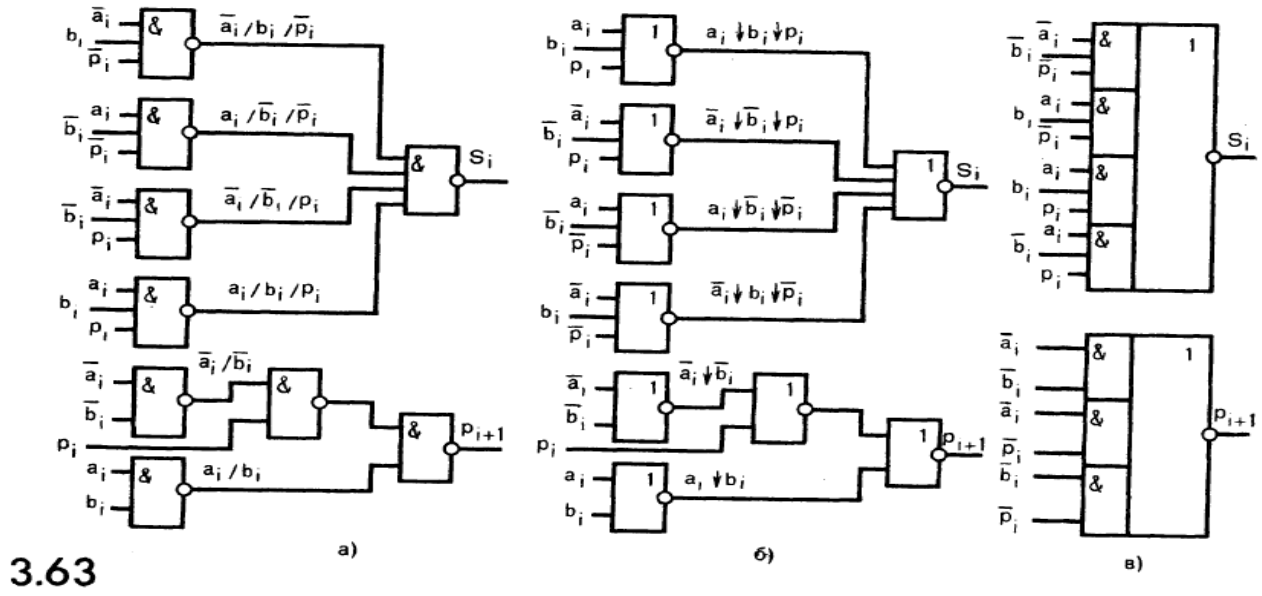
$$s_i = \overline{(a_i \vee b_i \vee p_i)(\bar{a}_i \vee \bar{b}_i \vee p_i)(a_i \vee \bar{b}_i \vee \bar{p}_i)(\bar{a}_i \vee b_i \vee \bar{p}_i)} =$$
$$\overline{(a_i \vee b_i \vee p_i) \vee (\bar{a}_i \vee \bar{b}_i \vee p_i) \vee (a_i \vee \bar{b}_i \vee \bar{p}_i) \vee (\bar{a}_i \vee b_i \vee \bar{p}_i)} =$$
$$= (a_i \downarrow b_i \downarrow p_i) \downarrow (\bar{a}_i \downarrow \bar{b}_i \downarrow p_i) \downarrow (a_i \downarrow \bar{b}_i \downarrow \bar{p}_i) \downarrow (\bar{a}_i \downarrow b_i \downarrow \bar{p}_i);$$

$$p_{i+1} = (a_i \vee b_i \vee p_i)(a_i \vee \bar{b}_i \vee p_i)(\bar{a}_i \vee b_i \vee p_i)(a_i \vee b_i \bar{p}_i) =$$
$$= (a_i \vee p_i)(b_i \vee p_i)\overline{(a_i \vee b_i)} = \overline{(a_i b_i \vee b_i p_i \vee a_i p_i \vee p_i)(a_i \vee b_i)} =$$
$$= (a_i b_i \vee p_i)(a_i \vee b_i) = (a_i b_i \vee p_i) \vee (a_i \vee b_i) = \overline{(\bar{a}_i \vee \bar{b}_i) \vee p_i) \vee (a_i \vee b_i)} =$$
$$= ((\bar{a}_i \downarrow \bar{b}_i) \downarrow p_i) \downarrow (a_i \downarrow b_i);$$

в базисе И—ИЛИ—НЕ

$$s_i = \overline{\bar{a}_i \bar{b}_i \bar{p}_i \vee a_i b_i \bar{p}_i \vee \bar{a}_i b_i p_i \vee a_i \bar{b}_i p_i}; \quad p_{i+1} = \overline{\bar{a}_i \bar{b}_i \vee \bar{a}_i \bar{p}_i \vee \bar{b}_i \bar{p}_i}.$$

На рис. 3.63 приведены схемы сумматоров, построенные с использованием полученных выше логических выражений.

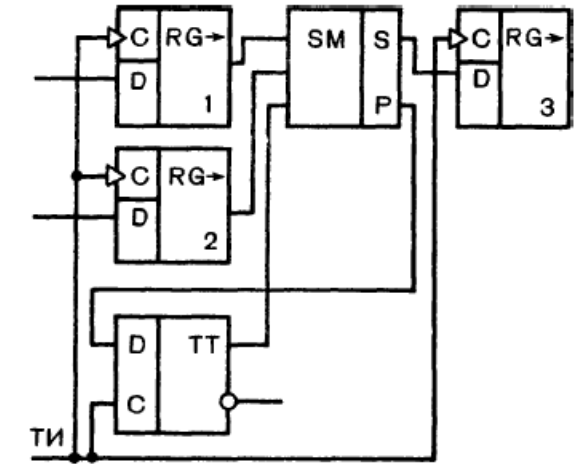


3.63

Многоразрядные двоичные сумматоры. В зависимости от способа ввода кодов слагаемых сумматоры делятся на два типа: ***последовательного*** и ***параллельного действия.*** В сумматоры первого типа коды чисел вводятся в последовательной форме, т. е. разряд за разрядом (младшим разрядом вперед), в сумматоры второго типа каждое из слагаемых подается в параллельной форме, т. е. одновременно всеми разрядами.

Сумматор последовательного действия (рис. 3.64). Он состоит из одноразрядного сумматора, выход p_{i+1} которого соединен с входом p_i через D-триггер. Изображенные на рисунке регистры сдвига не входят непосредственно в схему сумматора, они служат для подачи на входы сумматора разрядов слагаемых (регистры RG-1 и RG-2) и приема выдаваемых сумматором разрядов суммы (регистр RG-3).

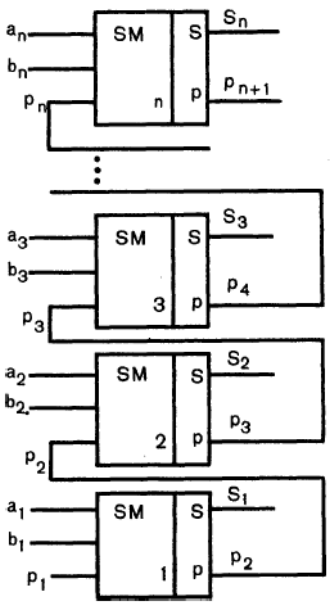
Операция суммирования во всех разрядах слагаемых осуществляется с помощью одного и того же одноразрядного сумматора. Возможность такого построения сумматора достигается за счет того, что слагаемые поступают в последовательной форме.



3.64

С первым тактовым импульсом на входы сумматора поступают из регистров RG-1 и RG-2 цифры первого разряда слагаемых a_1 и b_1, из D-триггера на вход P_i подается логический 0. Суммируя поданные на входы цифры, одноразрядный сумматор формирует первый разряд суммы S_1, выдаваемый на вход регистра RG-3, й перенос P_2, принимаемый в D-триггер. Второй тактовый импульс осуществляет в регистрах сдвиг на разряд вправо; на входы одноразрядного сумматора при этом подаются цифры второго разряда слагаемых a_2, b_2 и перенос P_2; получающаяся цифра второго разряда суммы вдвигается в регистр RG-3, перенос P_3 принимается в триггер и т. д. По такой схеме невозможно осуществление циклического переноса (т. е. прибавление единицы переноса из старшего разряда в младший разряд суммы). В связи с этим для представления отрицательных чисел можно пользоваться лишь дополнительным кодом.Очевидное достоинство сумматора последовательного действия заключается в малом объеме оборудования, требуемого для его построения. Однако связанная с этим необходимость в последовательной обработке разрядов приводит к низкому быстродействию.

Сумматор параллельного действия. Сумматор параллельного действия состоит из отдельных разрядов, каждый из которых содержит одноразрядный сумматор, как показано на рис. 3.65.



3.65

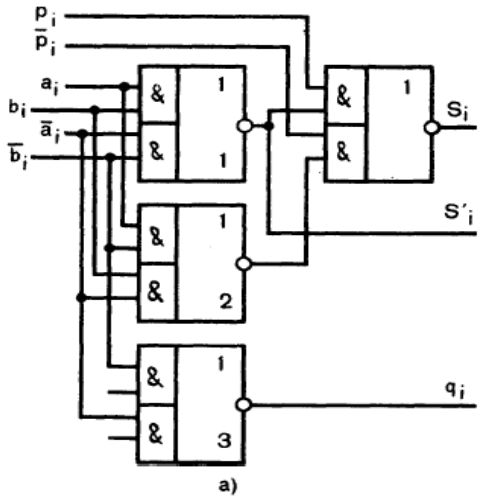
При подаче слагаемых цифры их разрядов поступают на соответствующие одноразрядные сумматоры. Каждый из одноразрядных сумматоров формирует на своих выходах цифру соответствующего разряда

суммы и перенос, передаваемый на вход одноразрядного сумматора следующего (более старшего) разряда. Включение шины циклического переноса, передающего перенос из старшего разряда младший, требуется в случае, когда для представления отрицательных чисел используется обратный код.

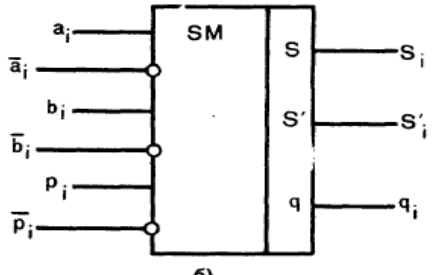
в

Повышение скорости работы параллельных сумматоров. Для обеспечения высокой скорости работы параллельных сумматоров последние должны строиться на элементах, обладающих высоким быстродействием. Трудности в достижении высокого быстродействия в сумматоре, построенном по схеме рис. 3.65, связаны с тем, что процесс распространения переносов в нем носит последовательный характер. Импульс переноса в каждом разряде формируется после того, как будет сформирован и передан импульс переноса из предыдущего разряда.

В наиболее неблагоприятном случае возникший в младшем разряде перенос может последовательно вызывать переносы во всех остальных разрядах. При этом время передачи переносов окажется $\tau = n\tau_1$, где τ_1 —задержка распространения переноса в одном разряде.



а)



б)

3.66

Уменьшение τ достигается следующими приемами.

1. При построении схем одноразряядных сумматоров стремятся к уменьшению числа элементов в цепи между входом на

который поступаетимпульс переноса p_i, и выходом, на котором формируется переда-
ваемый в следующий разряд импульс переноса p_{i+1}. Реализация этого принципа видна в схемах сумматоров на рис. 3.63, в которых цепь от p_i к p_{i+1} содержит не более двух логических элементов.

2. В цепях от p_i к p_{i+1} применяют элементы с повышенным быстродействием.

3. Схемы сумматоров следует строить таким образом, чтобы сигналы с выхода каждого логического элемента в цепи от p_i к p_{i+1} поступали на возможно меньшее число других логических элементов, так как присоединение дополнительного элемента к той или иной точке цепи переносов, как правило, приводит к увеличению паразитной емкости, удлинению фронтов сигналов и, следовательно, к увеличению задержки распространения и снижению быстродействия сумматора.

4. Применяют цепи параллельной передачи переносов.

Лекция 14Цифровые компараторы

Компараторы, принцип работы, УГО

Компаратор — это КЦУ, предназначенное для определения равенства двоичных чисел(рис. 3.23). Два числа *a* и *b*равны при равенстве цифр в одноименных разрядах: $a_i = b_i$.

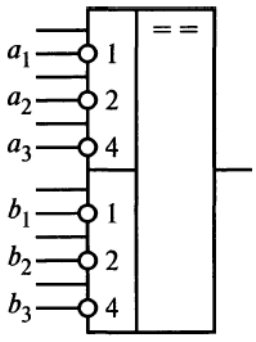


Рис. 3.23. Условное графическое обозначение компаратора

Равенство $a_i = b_i$имеет место при $a_i = 1$, $b_i = 1$ или при $a_i = 0$, $b_i = 0$. Поэтому логическая функция, выражающая это равенство, равна единице,

если единице равно произведение этих цифр или произведение их инверсных значений. Это разрешимо с помощью функции «Равнозначность»:

$$Y = a_i \cdot b_i \vee \overline{a}_i \cdot \overline{b}_i.$$

Так как числа равны при равенстве цифр в первых, во вторых и в *n*-х разрядах, то логическая функция, выражающая равенство двух чисел — это логическая функция функционирования компаратора, имеющая вид

$$Y = \left(a_1 \cdot b_1 \vee \overline{a}_1 \cdot \overline{b}_1\right) \cdot \left(a_2 \cdot b_2 \vee \overline{a}_2 \cdot \overline{b}_2\right) \cdot \ldots \cdot \left(a_n \cdot b_n \vee \overline{a}_n \cdot \overline{b}_n\right).$$

Для построения цифрового компаратора необходимо преобразовать функцию *Y*к базису {И-НЕ}, используя закон двойного отрицания и теорему де Моргана:

$$Y = \overline{\overline{Y}} = \overline{\left(a_1 \cdot b_1 \vee \overline{a}_1 \cdot \overline{b}_1\right) \cdot \left(a_2 \cdot b_2 \vee \overline{a}_2 \cdot \overline{b}_2\right) \cdot \ldots \cdot \left(a_n \cdot b_n \vee \overline{a}_n \cdot \overline{b}_n\right)} =$$

$$= \overline{\overline{a_1 \cdot b_1} \cdot \overline{\overline{a}_1 \cdot \overline{b}_1} \cdot \ldots \cdot \overline{a_n \cdot b_n} \cdot \overline{\overline{a}_n \cdot \overline{b}_n}}.$$

Схема, реализующая это выражение, строится на логических элементах «И-НЕ». Если необходимо, чтобы при равенстве кодов на выходе компаратора была «1», то к выходу схемы следует присоединить инвертор.

Пример. Построить логическую схему цифрового компаратора для сравнения чисел $a = 101$, $b = 111$.

1. Заданные двоичные числа a и b поразрядно будут иметь вид:

$$a = a_3 = 1,\ a_2 = 0,\ a_1 = 1;\ b = b_3 = 1,\ b_2 = 1,\ b_1 = 1.$$

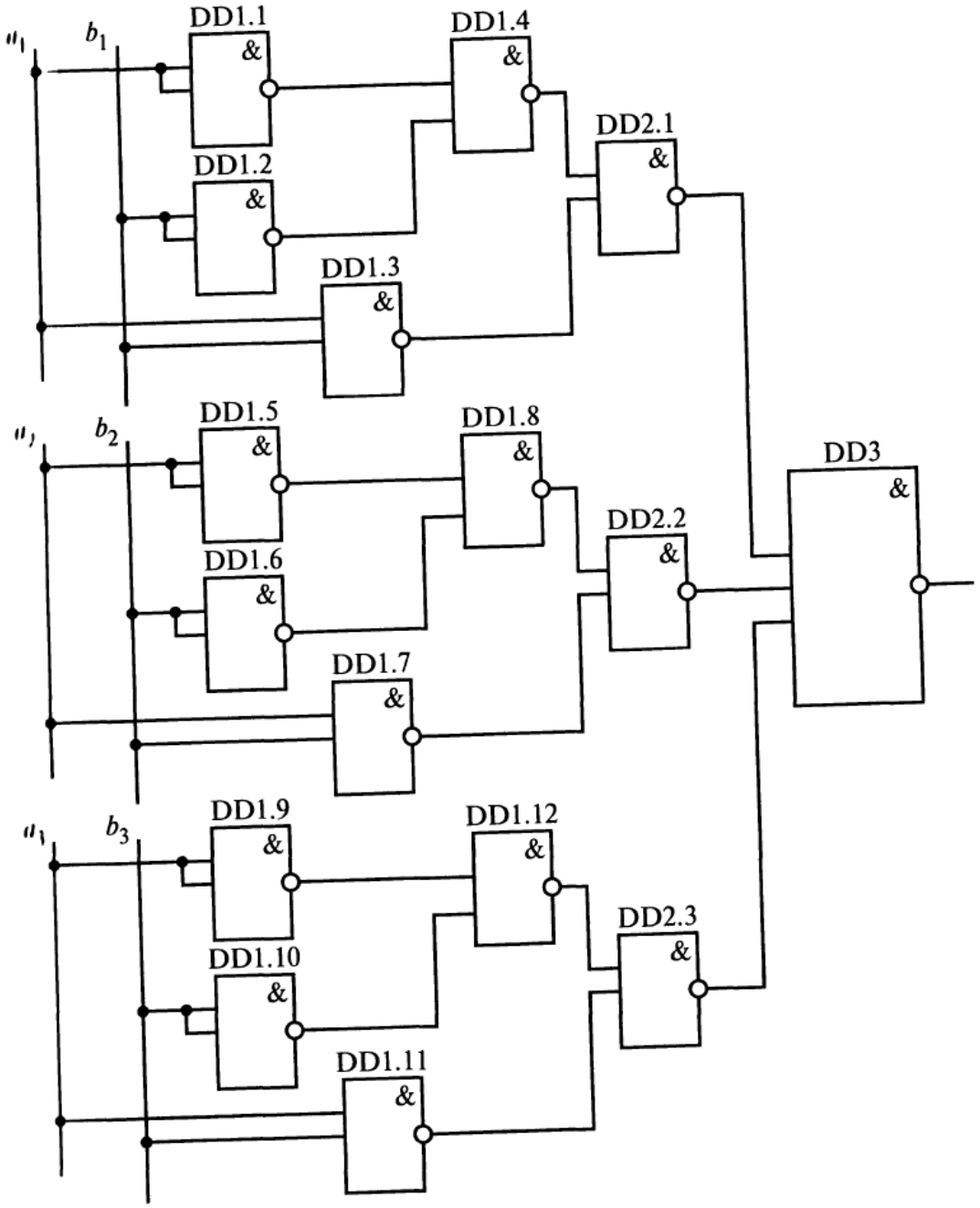


Рис. 3.24. Схема компаратора

2. Логическая функция, реализующая функционирование компаратора, будет иметь вид

$$Y = \left(a_1 \cdot b_1 \vee \bar{a}_1 \cdot \bar{b}_1\right) \cdot \left(a_2 \cdot b_2 \vee \bar{a} \cdot_2 \bar{b}_2\right) \cdot \left(a_3 \cdot b_3 \vee \bar{a}_3 \cdot \bar{b}_3\right).$$

3. Функция п. 2 в базисе {И-НЕ} примет вид

$$Y = \bar{\bar{Y}} = \overline{\overline{\left(a_1 \cdot b_1 \vee \bar{a}_1 \cdot \bar{b}_1\right) \cdot \left(a_2 \cdot b_2 \vee \bar{a}_2 \cdot \bar{b}_2\right) \cdot \left(a_3 \cdot b_3 \vee \bar{a}_3 \cdot \bar{b}_3\right)}} =$$

$$= \overline{\overline{a_1 \cdot b_1} \cdot \overline{\bar{a}_1 \cdot \bar{b}_1} \cdot \ldots \cdot \overline{a_3 \cdot b_3} \cdot \overline{\bar{a}_3 \cdot \bar{b}_3}}.$$

4. Построим схему компаратора (рис. 3.24) (табл. 3.19) на основе функции п. 3.

Таблица 3.19

Таблица — спецификация к схеме

Обозначение в схеме	ИМС	Количество	Коэффициент использования
DD1.1—DD1.12	К155ЛА3	3	4/4
DD2.1—DD2.3	К155ЛА3	1	3/4
DD3	К155ЛА4	1	1/3

Подадим на информационные входы схемы заданные значения разрядов чисел, найдем значение на выходе $Y = 1$. Это свидетельствует о том, что равенства чисел между двоичными числами a и b нет.

Придумать задание для студентов с компоратором!!!!!!!

Задание 5. Собрать схему компаратора, зная таблицу истинности.

A	B	A<B	A=B	A>B
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

Привести логические выражения, отображающие работу компаратора.

Лекция 15Классификация и параметры запоминающих устройств(ЗУ)

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.

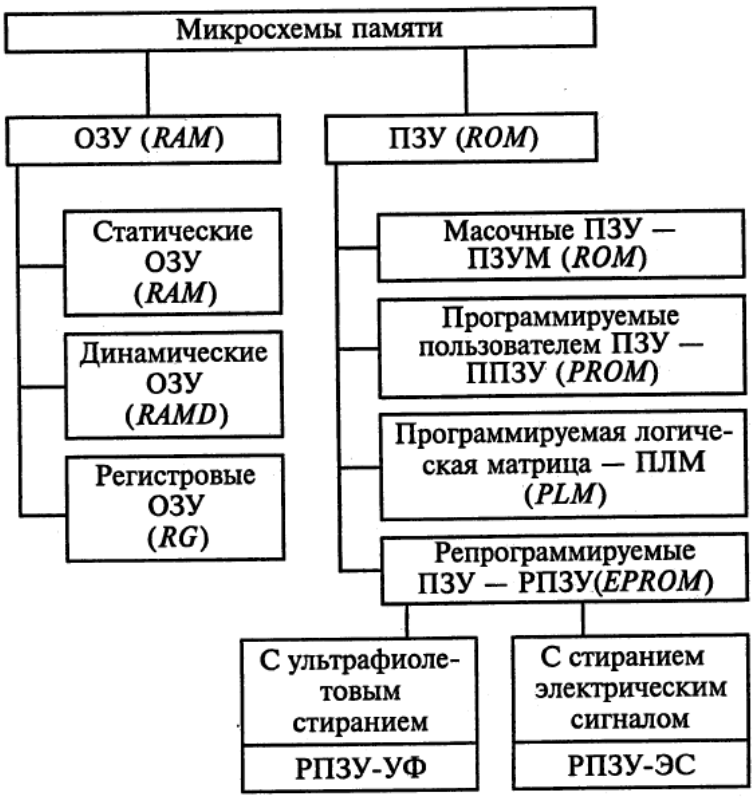


Рис. 18.1

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, $И^2Л$, 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.

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 mn-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-

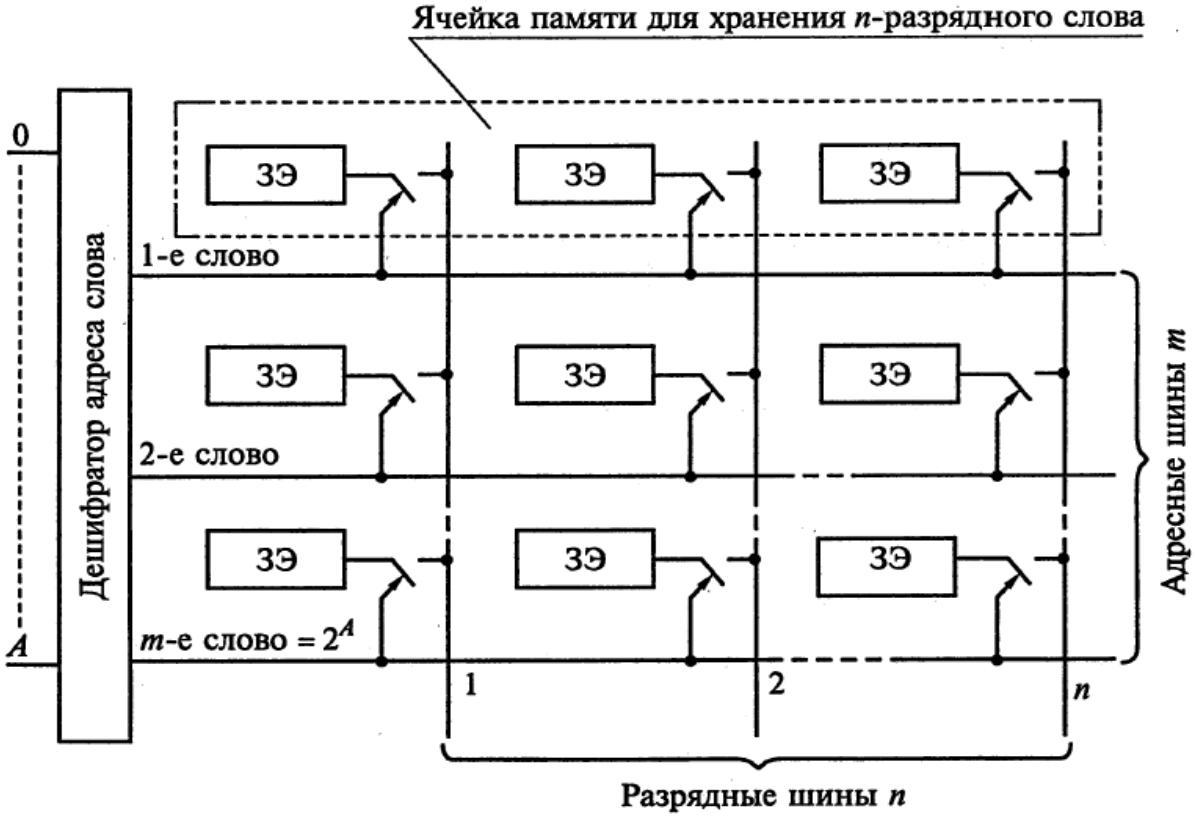


Рис. 18.2

щью разомкнутых ключей доступа ЗЭ 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 $m \times n = 2^A \times n = 2^5 \times 8 = 32 \times 8 = 256$ ЗЭ 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

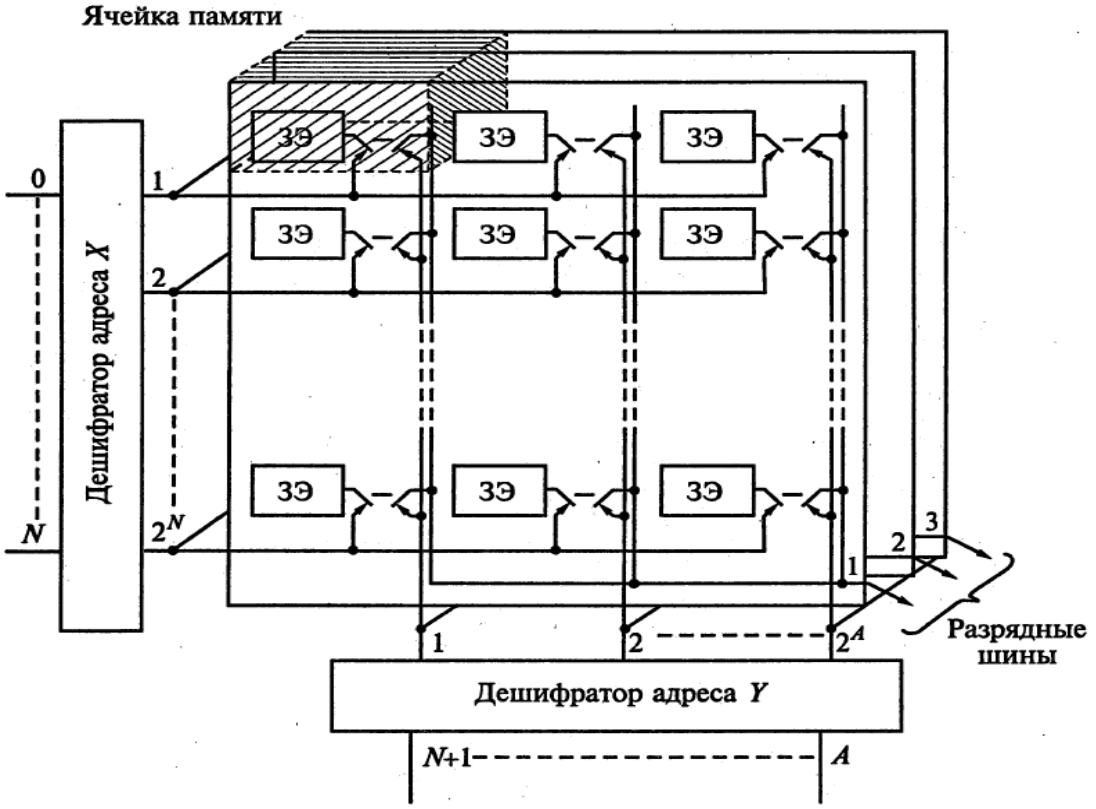


Рис. 18.3

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 ($8X \times 8Y$).

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

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-

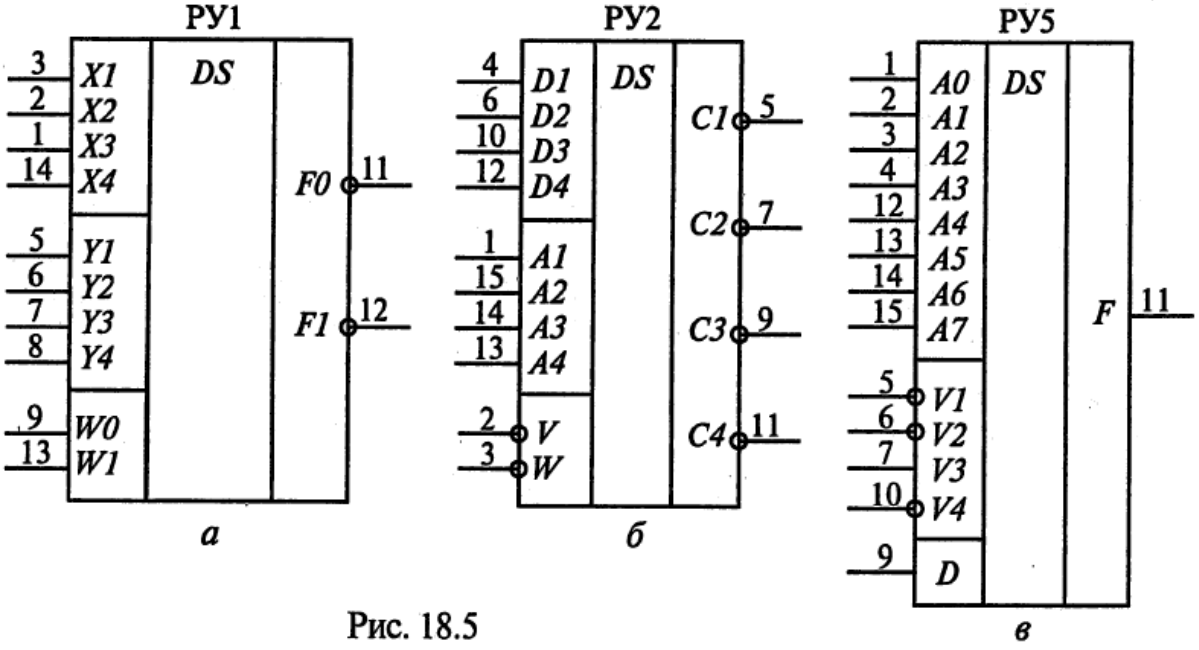


Рис. 18.5

чество адресных шин *X* и *Y*, т.е. емкость ЗУ равна (4 × 4) × 1 = = 16 бит × 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

$$(m \times n) = 2^A \times 1 = 2^8 \times 1 = 256 \times 1 = 256 \text{ бит}.$$

Лекция 16Оперативные запоминающие устройства

Динамическая оперативная память (DRAM – DynamicRandomAccessMemory) – энергозависимая полупроводниковая память с произвольным доступом.

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

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

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

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

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

Каждая ячейка состоит **из одного конденсатора (С)** и **трех транзисторов.Транзистор VT1** разрешает или запрещает запись новых данных или регенерацию ячейки.

Транзистор VT3выполняет роль ключа, удерживающего конденсатор от разряда и разрешающего или запрещающего чтение данных из ячейки памяти.

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

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

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

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

Команды передаются в **блок управления**, который организует работу остальных блоков и периодическую регенерацию ячеек памяти.

Адрес преобразуется в две составляющие – адрес строки и адрес столбца, и передается в соответствующие дешифраторы.

Дешифратор адреса строки определяет, с какой строки надо провести чтение или запись, и выдает на эту строку напряжение.

Дешифратор адреса столбца при чтении данных определяет, какие из считанных бит данных были запрошены и должны быть выданы в шину памяти. При записи данных дешифратор определяет, в какие столбцы надо подать команды записи.

Блок работы с данными определяет, какие данные, в какую ячейку памяти требуется записать, и выдает соответствующие биты данных для записи в эти ячейки.

Блоки регенерации определяют:

- когда происходит чтение данных и надо провести регенерацию ячейки, из которой данные были считаны;
- когда происходит запись данных, а, следовательно, регенерацию ячейки производить не надо.

Буфер данных сохраняет всю считанную строку матрицы, так как при чтении всегда считывается вся строка целиком, и позволяет потом выбрать из считанной строки требуемые биты данных.

Рассмотрим принцип работы динамической памяти на примере структурной схемы, приведенной на рисунке 1. Рассматривать будем работу с первой ячейкой (M11).

1.1. Работа динамической памяти в состоянии покоя.

И так, первое что мы рассмотрим – этот состояние покоя, когда к памяти отсутствуют обращения, и она не в стадии регенерации данных.

DRAM – память энергозависимая, поэтому работа с ней возможна только при подаче питания. На схеме подаваемое на плату питание обозначено, как

Uп. Подаваемое питание распределяется между всеми столбцами матрицы памяти с помощью резисторов R.

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

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

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

Ток по линии AE также попадает на «Блок регенерации 1», а именно, на нижний вход элемента L3 (логическое «И»), то есть на нижний вход элемента L3 подается логическая единица.

Рассмотрим, как в этом случае будет работать блок регенерации.

Так как от контроллера памяти нет никаких сигналов, то на входе элемента L1 (логическое «НЕ») будет логический ноль, а, соответственно, на выходе – логическая «1». Таким образом, на верхнем входе элемента L3 (логическое «И») будет логическая единица.

Имея на входах элемента L3 (логическое «И») две логические единицы, на выходе получим так же логическую единицу.

На выходе элемента L2 (логическое «И») будет логический ноль, так как на обоих его входах напряжение отсутствует, так как от контроллера памяти нет никаких команд и данных.

В результате, на входах элемента L4 (логическое «ИЛИ-НЕ») будет логический ноль и логическая единица, а, соответственно, на его выходе будет логический ноль, то есть напряжение будет отсутствовать. Так как напряжение отсутствует, то ни один конденсатор первого столбца матрицы памяти подзаряжен не будет. (*Хотя, даже если бы напряжение и присутствовало, все равно подзарядка была бы невозможна, так как транзисторы подзарядки (доля ячейки M11 – это VT1) были бы закрыты, ведь ни на одну строку матрицы памяти (S1-Sn) напряжение не подается.)*

Точно такая же ситуация будет со всеми столбцами матрицы памяти.

Таким образом, при бездействии памяти конденсаторы не подзаряжаются и хранят тот заряд (а, соответственно, и тот бит данных), который у них был с момента последней подзарядки. Однако долго это продолжаться не может, так как из-за саморазрядки конденсатор, через несколько десятков

миллисекунд, разрядится, и данные будут утеряны. Поэтому необходимо постоянно проводить регенерацию памяти.

1.2. Работа динамической памяти при чтении данных и регенерации.

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

1. Процессор запрашивает порцию данных (размер зависит от разрядности процессора, для 32-разрядного процессора минимальной единицей обмена, обычно, являются 32 бита) и выдает их адрес.

2. Контроллер шины памяти преобразует адрес в номер строки и номер столбца и выдает номер строки в дешифратор адреса строки. Дешифратор адреса строки выдает сигнал в соответствующую строку матриц памяти. Мы договорились, что в примере данные будем читать из первой ячейки памяти. Поэтому дешифратор адреса строки подаст напряжение на первую строку (S1).

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

4. Дальнейшая работа памяти зависит от наличия или отсутствия заряда на конденсаторе. Рассмотрим отдельно два случая, когда на конденсаторе ячейки M11 есть заряд, и когда нет.

4.1 . В начале рассмотрим случай, когда заряд в конденсаторе есть (ячейка памяти содержит бит со значением ноль):

Так как на конденсаторе C ячейки памяти M11 есть заряд, то транзистор VT2 будет открыт, а, соответственно, ток, создаваемый входным напряжением Uп, пойдет по линии AB. В результате, на выходе Q1 первого столбца тока не будет. А это означает, что с ячейки памяти M11 считан ноль. Соответствующая информация о считанном бите с первого столбца будет записана в «Буфер данных».

Для поддержания транзистора VT2 в открытом состоянии и протекания тока по линии AB расходуется заряд конденсатора C. В результате, конденсатор очень быстро разрядится, если не провести его регенерацию.

Так как на выходе Q1 тока нет, то он не будет поступать и в «Блок регенерации 1», а, соответственно, на нижнем входе элемента L3 (логическое «И») будет логический ноль.

Так как мы рассматриваем случай чтения данных, то сигнал записи V1 и данные для записи D1 в «Блок регенерации 1» подаваться не будут. В остальные блоки регенерации соответствующие сигналы D1-Dm и V1-Vm также подаваться не будут.

В результате, на входе элемента L1 (логическое «НЕ») будет логический «0», а на выходе – логическая «1», поэтому на входах элемента L3 (логическое «И») будет логический «0» и логическая «1». Это значит, что на выходе этого элемента будет логический «0».

На выходе логического элемента L2 (логическое «И») будет логический ноль, так как на обоих его входах напряжение отсутствует, так как от контроллера шины памяти отсутствуют команды на запись и данные для записи.

Имея на обоих входах элемента L4 (логическое «ИЛИ-НЕ») логический «0», на его выходе будем иметь логическую «1», то есть с блока регенерации пойдет ток подзарядки конденсатора С. Так как транзистор подзарядки VT1 ячейки памяти М11 открыт, то ток подзарядки беспрепятственно пройдет в конденсатор С. Остальные ячейки памяти первого столбца имеют закрытый конденсатор подзарядки, а, следовательно, подзарядка их конденсаторов происходить не будет.

4.2 . Теперь рассмотрим случай, когда в конденсаторе нет заряда (ячейка памяти хранит бит со значением «1»):

Ток, создаваемый входным напряжением Uп, пойдет по линии AE, так как транзистор VT2 будет закрыт. Следовательно, на входе Q1 «Буфера данных» будет ток, что означает – с ячейки памяти считана единица. Информация о считанном бите с первого столбца будет записана в «Буфер данных».

Так как в конденсаторе заряда не было, то и подзаряжать его надобности нет. Следовательно, с блока регенерации ток пойти не должен.

Так как на выходе Q1 ток есть, то он поступает и в «Блок регенерации». Следовательно, на нижний вход элемента L3 (логическое «И») подается логическая единица.

Так как мы рассматриваем случай чтения данных, то сигнала записи V1 и данных для записи D1 в «Блок регенерации 1» подаваться не будет. Так же в остальные блоки регенерации, соответствующие сигналы D1-Dm и V1-Vm так же подаваться не будут.

Следовательно, на входе элемента L1 (логическое «НЕ») будет логический ноль, а на выходе – логическая «1». Таким образом, на входах элемента L3 (логическое «И») будут две логические единицы. В результате, на выходе получим так же логическую единицу.

На выходе логического элемента L2 (логическое «И») будет логический ноль, так как на обоих его входах напряжение отсутствует, так как от контроллера памяти нет команд на запись и данных для записи.

В результате, на входах элемента L4 (логическое «ИЛИ-НЕ») будет логический ноль и логическая единица, а, соответственно, на его выходе будет логический ноль, то есть напряжение будет отсутствовать. Так как напряжение отсутствует, то ни один из конденсаторов первого столбца матрицы памяти подзаряжаться не будет.

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

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

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

1.3. Работа динамической памяти при записи данных.

Будем рассматривать принцип записи данных в динамическую память на примере записи данных в ячейку памяти M11:

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

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

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

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

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

6. Если в ячейке M11 хранится бит со значением «0» (в конденсаторе есть заряд), то ток, создаваемый входным напряжением Uп, пойдет по линии AB, иначе – по линии AE. Но нам это не важно, так как в ячейку M11 производится запись данных, а не их чтение, поэтому буфер данных будет игнорировать считанное с ячейки значение. А с выхода элемента L3 «Блока регенерации 1» будет всегда идти логический ноль, так как с дешифратора столбцов приходит сигнал (V1) на запись данных в первый столбец.

В результате, на входе элемента L1 будет логическая единица, а на выходе – логический ноль. Соответственно, на верхнем входе элемента L3 мы всегда имеем логический ноль, что означает – независимо от значений на нижнем входе, на выходе элемента L3 будет логический ноль.

На нижнем входе элемента L2 будет логическая единица, так как с дешифратора адреса столбцов выдается сигнал V1, а на верхнем входе будет либо ноль, либо единица, в зависимости от того, какое значение имеет бит записываемой информации.

Если бит имеет значение «1», то на верхнем входе элемента L2 будет «1». Имея две единицы на входе, мы получим на выходе так же логическую единицу. Соответственно, на входах элемента L4 будет получена логическая «1» и логический «0». В результате, на выходе будет логический «0», то есть ток будет отсутствовать, а, соответственно, зарядка конденсатора C идти не будет. Если до этого конденсатор C содержал заряд, то через несколько микросекунд он разрядится, пропуская ток по линии AB. Таким образом в конденсатор C будет записан бит данных «1», соответствующий разряженному состоянию конденсатора.

Если бит имеет значение «0», то на верхнем входе элемента L2 будет «0». Имея на верхнем входе логический ноль, а на нижнем – логическую единицу, на выходе элемента L2 получим логический ноль. В результате, на верхнем и нижнем входах элемента L4 имеем логические нули, что означает – на выходе элемента L4 будет логическая единица, то есть пойдет ток зарядки конденсатора. Таким образом в конденсатор C будет записан бит данных «0», соответствующий заряженному состоянию конденсатора.

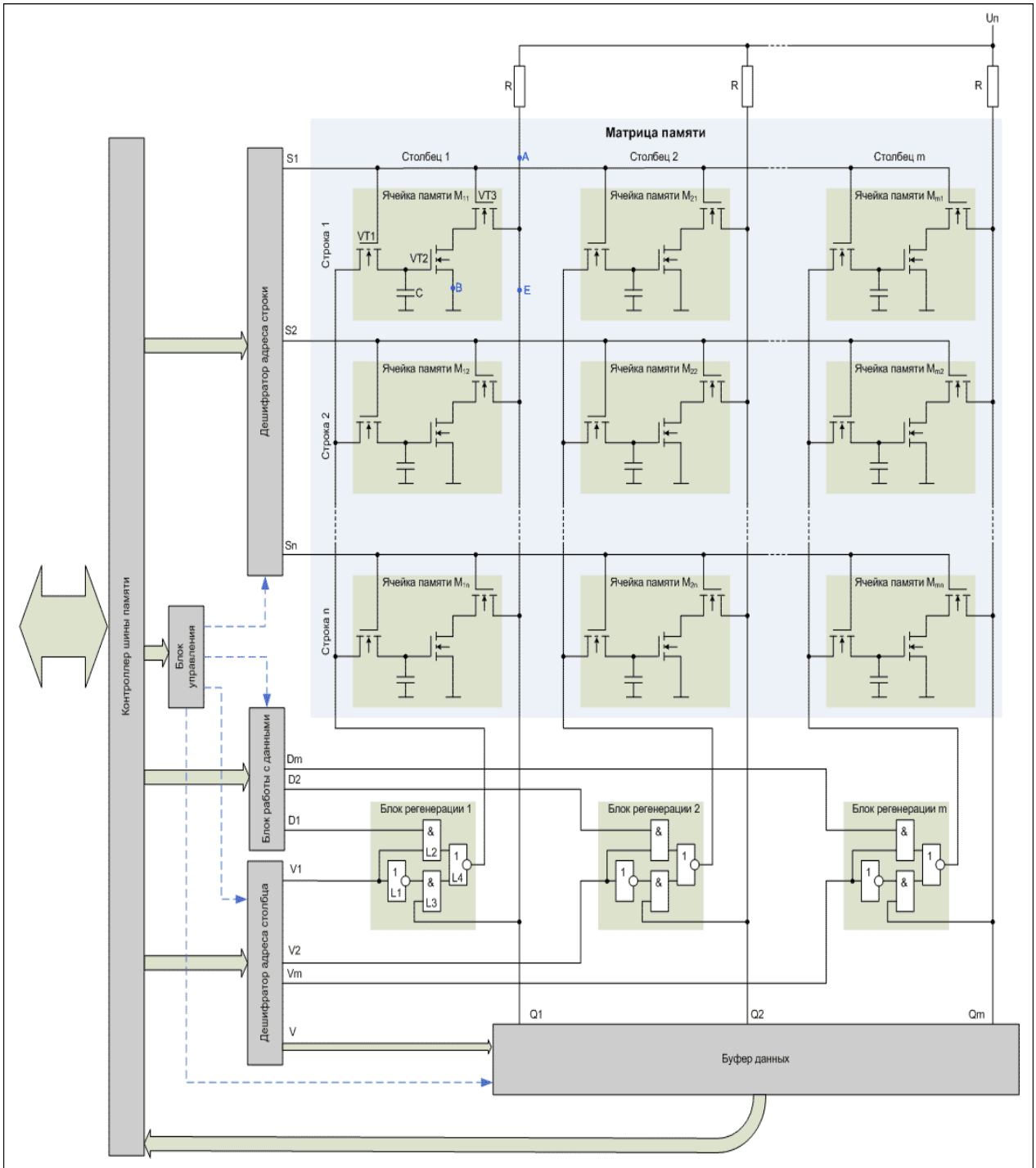


Рисунок 1. Упрощенная структурная схема динамической оперативной памяти

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

.

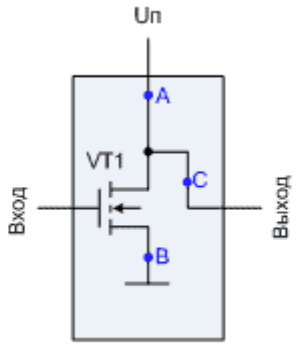


Рисунок 3. Структурная схема инвертера.

На элемент всегда подается питание Uп. В результате, создаваемый ток может пойти либо по линии AB, в этом случае на выходе инвертера ток будет отсутствовать (будет логический ноль), либо – по линии AC, в этом случае на выходе инвертера ток будет присутствовать (будет логическая единица).
По линии AB ток пойдет, если транзистор VT1 будет открыт, а для этого необходимо подать напряжение на вход инвертера.
По линии AC ток пойдет, если транзистор VT1 будет закрыт, а это произойдет при отсутствии напряжении на входе инвертера.
Таким образом, если на вход инвертера подается логическая единица, то на выходе будет логический ноль. И, соответственно, при подаче на вход инвертера логического нуля, на выходе будет поучена логическая единица.

Работа статической памяти.

1. Начнем с **записи данных** в статическую память и рассмотрим случай записи единицы в ячейку M11.
2. В контроллер шины памяти от контроллера памяти, встроенного в северный мост материнской платы или в процессор, приходит адрес ячейки памяти и данные для записи.
3. Адрес ячейки преобразуется на две составляющие – номер строки и номер столбца. Номер строки передается в «Дешифратор адреса строки», откуда на нужную строку подается напряжение.
4. Так как мы рассматриваем запись в ячейку M11, то напряжение с дешифратора адреса строки подается на первую строку S1. В результате, транзисторы VT1, VT2 и VT3 открываются. Аналогичные транзисторы других ячеек памяти, располагающихся в этой строке, также открываются.
5. Через транзистор VT3 первой ячейки и аналогичные транзисторы других ячеек памяти первой строки пойдет ток, соответствующий состоянию

триггеров этих ячеек, в «Буфер данных». Однако «Буфер данных» получаемую информацию будет игнорировать, так как у него нет сигнала от «Блока управления» на сохранение считываемых данных.
6. Параллельно с подачей напряжения на строку матрицы памяти с «Блока работы с данными» будет выдано напряжение, соответствующее записываемым данным, в «Блоки записи 1 - m», а с «Блока дешифровки адреса столбца» на соответствующие столбцы будет выдано разрешение (напряжение, соответствующее логической единице) на запись данных.
7. Блоки записи используются для запрета выдачи тока в линии D и $\overline{D}$ при чтении данных и преобразования из входящих сигналов данных их инвертируемых сигналов для переключения состояния триггеров, в которые необходимо сохранить данные.
В нашем случае, запись проводится в ячейку М11, и записывается единица. Соответственно, с «Блока работы с данными» будет выдана логическая единица в «Блок записи 1», и с «Блока дешифровки адреса столбца» будет выдана логическая единица в «Блок записи 1».
Рассмотрим работу «Блока записи 1» при таких входных сигналах. И так, на входе элемента D.D3 будет логическая единица, а на выходе – логический ноль, так как элемент D.D3 – инвертер (логический элемент «НЕ»). Соответственно, на входах элемента D.D4 (логический элемент «И») будут: логический ноль и логическая единица. В результате, на выходе этого элемента будет логический ноль.
На входах элемента D.D5 (логический элемент «И») будут две логические единицы, в результате, на выходе этого элемента будет логический ноль.
Следовательно, на выходе D1 «Блока записи 1» будет напряжение, соответствующее логическому нулю, а на выходе $\overline{D}1$ будет напряжение, соответствующее логической единице. Эти напряжения будут поданы на все ячейки памяти первого столбца. Однако у всех ячеек, кроме первой, транзисторы, разрешающие запись, закрыты, а, следовательно, подаваемое напряжение попадет только на триггер первой ячейки и переведет его в состояние хранения единицы.
После изменения состояния триггера первой ячейки напряжение с первой строки снимается, и транзисторы VT1, VT2 и VT3 закрываются, запрещая запись и чтение из ячейки.
8. При записи нуля в ячейку памяти все происходит по той же схеме, только с «Блока работы с данными» в «Блок записи 1» будет подано напряжение, соответствующее логическому нулю. Это значит, что на выходе D1 «Блока записи 1» будет напряжение, соответствующее логической единице, а на выходе $\overline{D}1$ будет напряжение, соответствующее логическому нулю. Эти значения напряжений переведут триггер первой ячейки памяти в состояние хранения нуля.
В установленном состоянии триггер первой ячейки останется, пока на него будет подаваться питание Uп.

9. **Чтение записи** происходит еще проще. От контроллера памяти приходит адрес ячеек памяти, с которых требуется считать данные, и команда на чтение.
В результате, адрес преобразуется в номер строки, и на соответствующую строку будет подано напряжение, которое откроет транзисторы разрешения/запрета чтения/записи.
Рассмотрим случай, когда данные считываются из первой ячейки. В этом случае напряжение с «Дешифратора адреса строки» будет подано в первую строку, что приведет к открытию транзисторов VT1, VT2 и VT3 ячейки M11 и всех остальных ячеек первой строки. Ток с триггера первой ячейки, через транзистор VT1, беспрепятственно пройдет в «Буфер данных». То же самое произойдет с остальными ячейками первой строки. Считанные с ячеек памяти первой строки данные сохранятся в «Буфере данных».
10. После того, как информация в «Буфере данных» будет сохранена, «Дешифратор адреса столбцов» выдаст номера столбцов, данные с которых необходимо считать, в «Буфер данных». Соответствующие данные будут переданы из микросхемы памяти в контроллер памяти, располагающийся в материнской плате или непосредственно в процессоре.
Для того чтобы при чтении данных не происходила запись в эти же ячейки, ведь транзисторы, разрешающие запись, открыты, блоки записи выдают в линии D и $\overline{D}$ всех столбцов матрицы памяти напряжение, соответствующее логическому нулю. Как видите, работа статической памяти очень похожа на работу динамической памяти, однако процесс записи и чтения гораздо быстрее, так как не тратится время на заряд и разряд конденсаторов и не требуется регенерация ячеек.

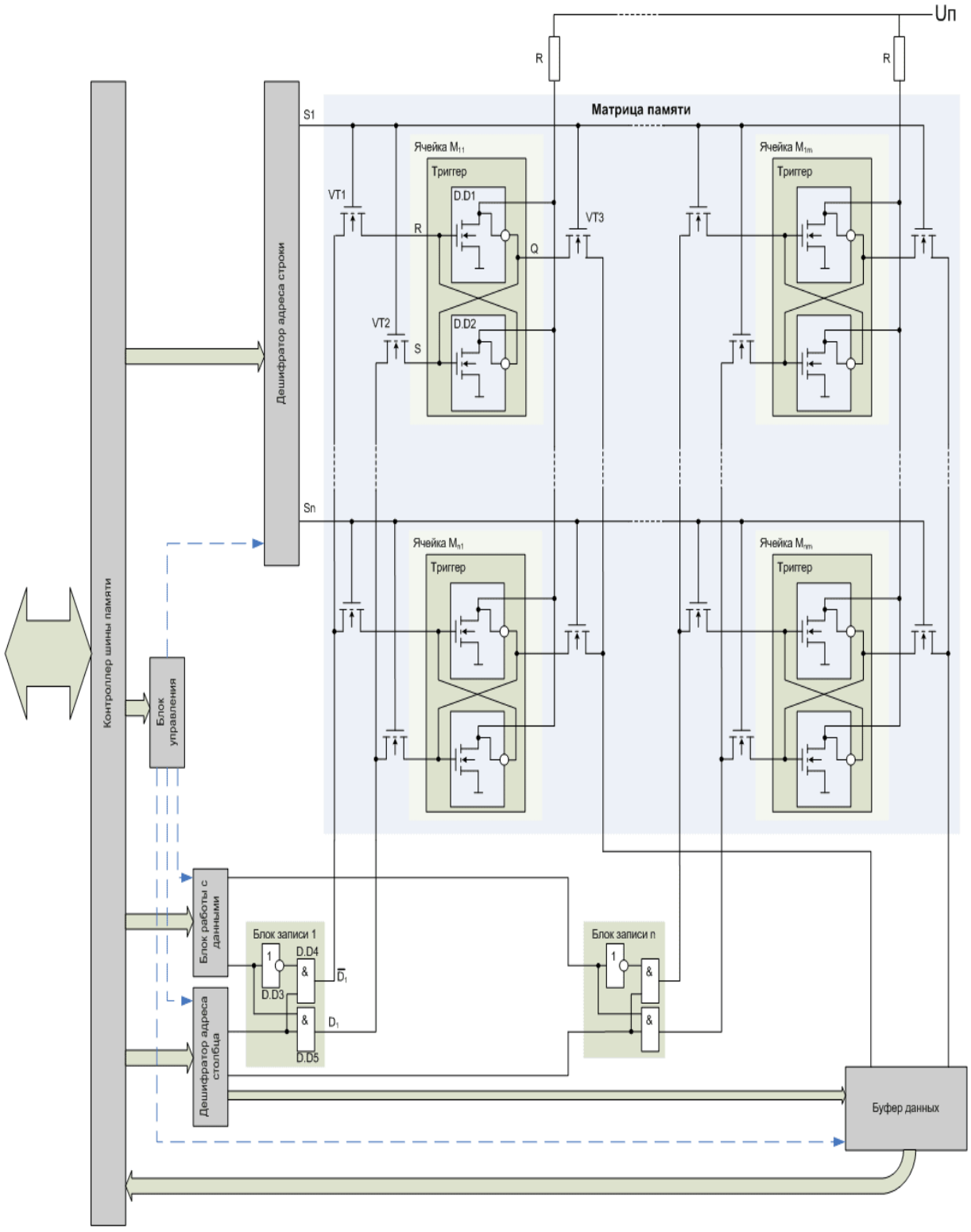


Рисунок 5. Упрощенная структурная схема статической оперативной памяти (SRAM)

Лекция 17Постоянные 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.

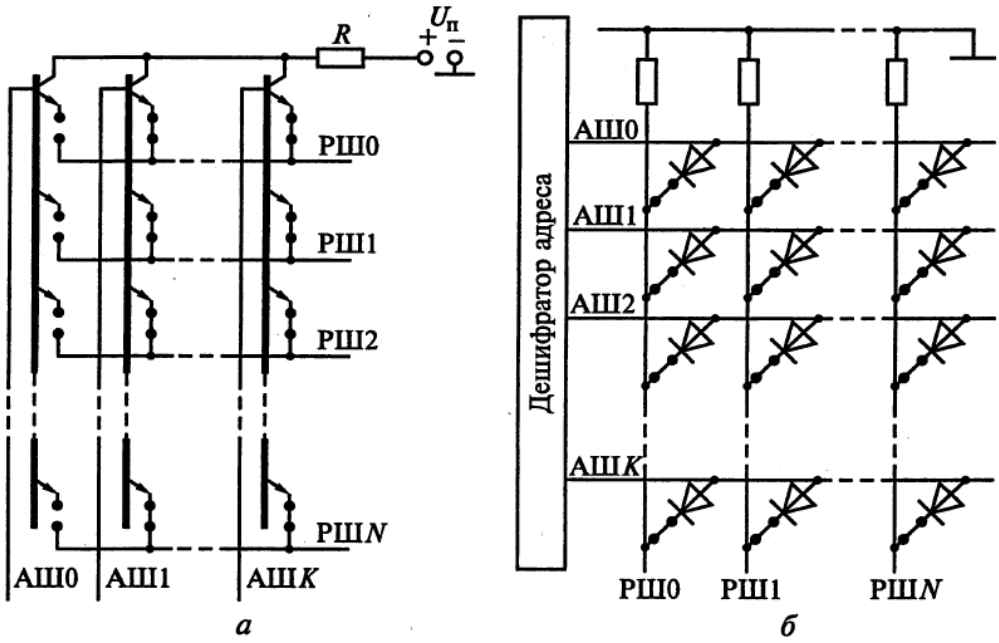


Рис. 19.1

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.

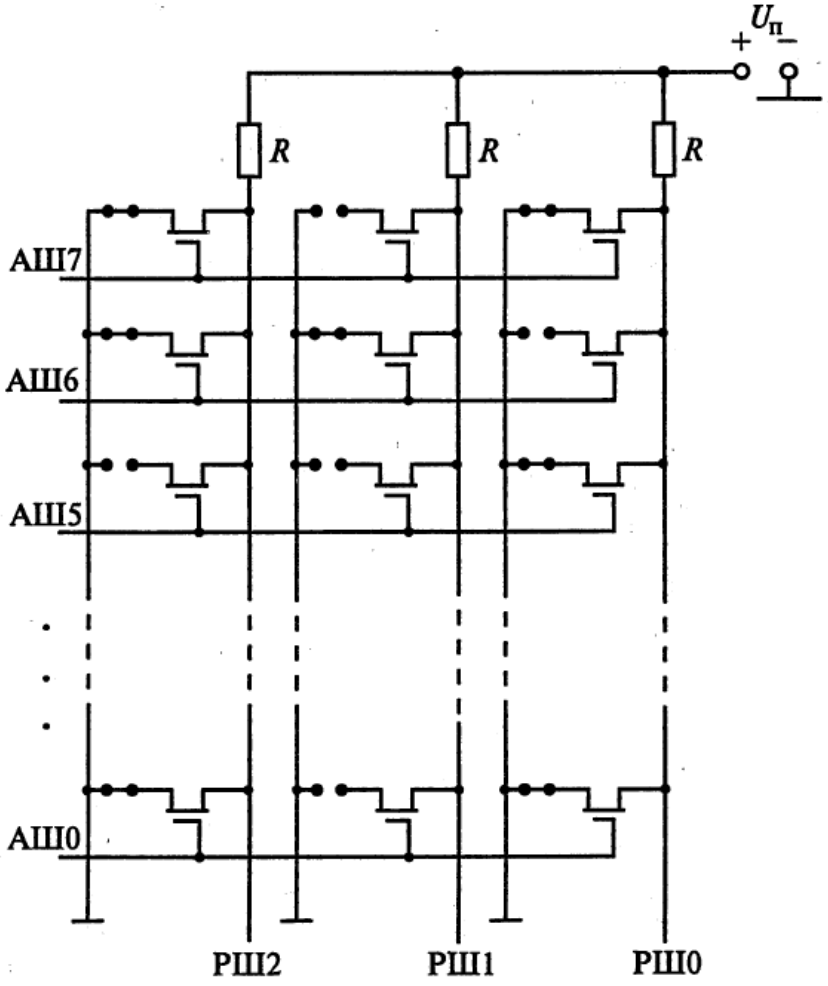


Рис. 19.2

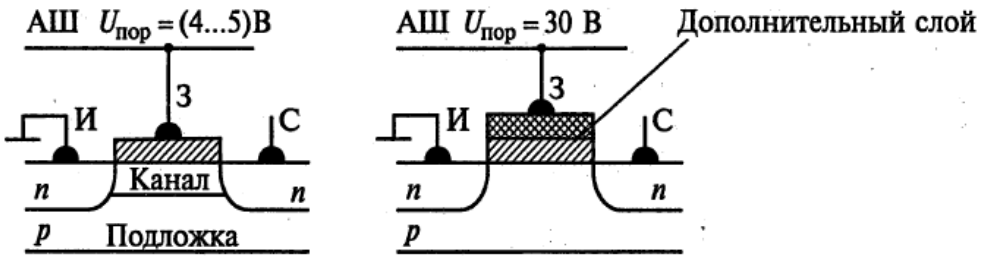


Рис. 19.3

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.

19.3. Однократно программируемые ПЗУ в интегральном исполнении серии КР556

Одним из наиболее распространенных способов введения информации в ППЗУ является принцип плавкого предохранителя,

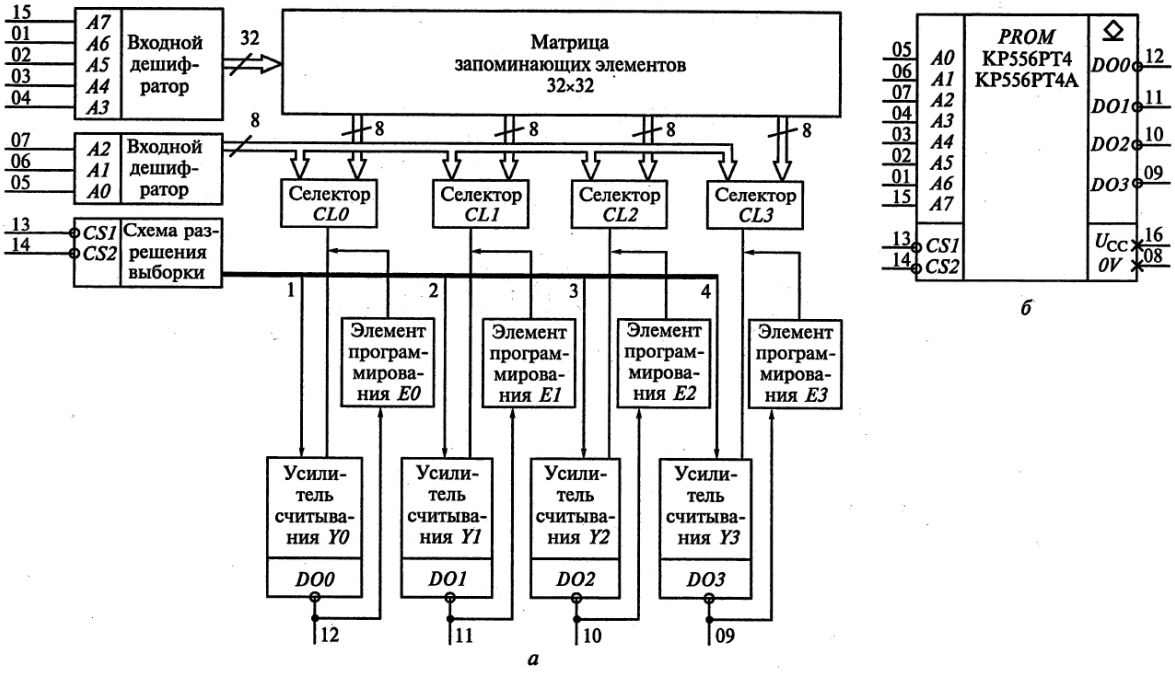
когда для образования необходимой комбинации лог. 0 и лог. 1 производится выжигание дозированным током, например, нихромовых соединений между матричными элементами микросхемы. ППЗУ на биполярных транзисторах программируется следующим образом: все эмиттеры транзисторов соединены с разрядными шинами плавкими 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.

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.


15
01
02
03
04
A7
A6
A5
A4
A3
Входной дешифратор
32
Матрица запоминающих элементов 32×32
07
06
05
A2
A1
A0
Входной дешифратор
8
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
PROM
КР556РТ4
КР556РТ4А
DO0
DO1
DO2
DO3
12
11
10
09
13
14
CS1
CS2
Ucc
0V
16
08
б

Лекция 18 Цифро-аналоговые преобразователи (ЦАП) кода в напряжение

Принципы преобразования

В большинстве случаев получаемый непосредственно от источника информации сигнал представлен в форме непрерывно меняющегося по значению напряжения либо тока (телефонные, телевизионные и другие виды сообщений), для передачи (обработки) которых по линиям используют две формы: аналоговую или цифровую.

Диалоговая форма предусматривает оперирование всеми значениями сигнала, цифровая форма — отдельными его значениями.

И преобразовании сигналов можно выделить следующие процессы:

- дискретизация непрерывных сигналов — определение величины сигнала в момент времени:

$$t_1 = t_0 + T;\ t_2 = t_1 + T;\ \ldots;\ t_n = t_{n-1} + T,$$

гдеT — тактовый интервал времени, достаточно малый, чтобы можно было по дискретным отсчетам восстановить сигнал с требуемой точностью;

$t_0, t_1, \ldots, t_n$ —тактовые моменты времени.

- квантование — замена дискретных значений исходного аналогового напряжения ближайшими к ним уровнями квантования. Каждый уровень имеет порядковый номер и отстоит от другого на величину шага квантования Δ. Различие действительного и квантованного значений напряжения называется ошибкой квантования а (находится в пределах — $\Delta/2 \le \sigma \le \Delta/2$);
- кодирование — представление последовательности чисел двоичным кодом.

Цифро-аналоговый преобразователь (ЦАП) — электронное устройство, предназначенное для преобразования цифровой информации в аналоговую (рис. 6.1). Он используется для формирования сигнала в виде напряжения или тока, функционально связанного с управляющим кодом.

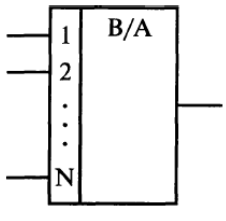


Рис. 6.1. Условное графическое обозначение ЦАП

ЦАП применяются для сопряжения устройств, в качестве узлов обратной связи в аналого-цифровых преобразователях, в устройствах сравнения цифровых величин с аналоговыми,в системах передачи данных, в измерительных приборах и испытательных установках, в синтезаторах напряжения.

ЦАП делятся:

— по принципу действия: с делением напряжения, со сложением токов и со сложением напряжений;

- по виду выходного сигнала: с токовым выходом и выходом по напряжению;

- по полярности выходного сигнала: однополярные и двухполярные;
- по виду источника опорного напряжения: с постоянным опорным напряжением и с изменяющимся опорным напряжением;
- по основным характеристикам: количеству разрядов, быстродействию, точности преобразования, потребляемой мощности;
- по совокупности параметров: общего применения, прецизионные (погрешность нелинейности менее 0,1 %) и быстродействующие (время установления меньше 100 нс).

Основные параметры ЦАП:

Статические параметры:

- разрешающая способность — величина, обратная максимальному количеству градаций выходного сигнала;
- погрешность преобразования: дифференциальная погрешность (максимальное отклонение от линейности для двух смежных значений входного кода) и погрешность нелинейности (максимальное отклонение выходного напряжения от идеальной прямой во всем диапазоне преобразования);
- диапазон значений выходного сигнала;
- характеристики управляющего кода;
- смещение нулевого уровня — выходное напряжение при входном коде, соответствующее нулевому значению.

Динамические параметры:

- время установления выходного сигнала — интервал времени от подачи входного кода до вхождения выходного сигнала в заданные пределы, определяемые погрешностью;
- предельная частота преобразования - наибольшая частота дискретизации, при которой все параметры ЦАП соответствуют заданным значениям;
- динамическая погрешность.

ЦАП с суммированием токов.

В схеме на рис. 6.2 используются n опорных источников тока $I_1, I_2, ..., I_n$. Входной код $b_1, b_2, ..., b_n$, управляет ключами $S_1, S_2, ..., S_n$, которые или подключают источники тока к нагрузке $(b_i = 1)$, или замыкают их накоротко $(b_i = 0)$. Результирующий ток I_Σ равен сумме токов опорных источников,для которых $(b_i = 1)$. Напряжение на выходе равно

$$U_{\text{вых}} = I_\Sigma R_{\text{н}}.$$

Если входной код — двоичный, то результирующий ток равен

$$I_\Sigma = I_0(b_1 \cdot 2^{n-1} + b_2 \cdot 2^{n-2} + \ldots + b_a \cdot 2^0) = I_0 N,$$

где n — число двоичных разрядов входного тока;
N — n-разрядное цифровое слово;
I_0 — опорный ток.

Упрощенная схема ***ЦАП с суммированием напряжений*** (рис. 6.3) состоит из n опорных источников напряжения. $E_1, E_2, \ldots, E_n$. Входной код управляет ключами $S_1, S_2, \ldots, S_n$, которые или подключают соответствующие источники опорного напряжения к нагрузке $(b_i = 1)$, или отключают их $(b_i = 0)$. Результирующее напряжение на выходе равно сумме напряжений включенных опорных источников. Для входного двоичного кода выходное напряжение равно

$$U_\Sigma = U_{\text{вых}} = U_0(b_1 \cdot 2^{n-1} + b_2 \cdot 2^{n-2} + \ldots + b_n \cdot 2^0) = U_0 N,$$

где U_0 — опорное напряжение.

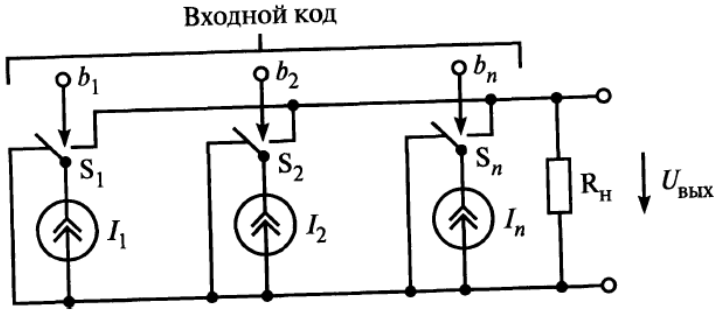


Рис. 6.2. Упрощенная схема ЦАП с суммированием токов

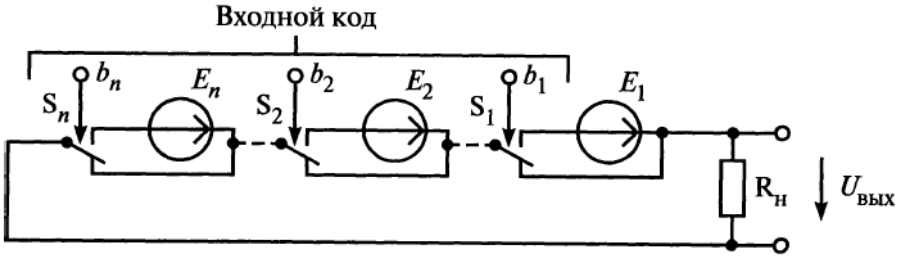


Рис. 6.3. Упрощенная схема ЦАП с суммированием напряжений

Упрощенная схема ***ЦАП с делением опорного напряжения*** (рис. 6.4) содержит один источник опорного напряжения и набор

калиброванных сопротивлений $R_1, R_2, ..., R_n$, с помощью которых напряжение опорного источника можете быть разделено до значения, соответствующего входному коду. Выходное напряжение определяется формулой

$$U_{\text{вых}} = \frac{E_0 R_{\text{н}}}{R_\Sigma + R_n},$$

где R_Σ — результирующее сопротивление, устанавливаемое при помощи ключей $S_1, S_2, ..., S_n$, которые управляются входным кодом.

При $R_n = 0$ эта схема работает как схема со сложением током Практически выполнить $R_n = 0$ можно при помощи операционпо го усилителя с параллельной обратной связью.

Практическая схема ***ЦАП со сложением токов*** выполняется на различных резистивных матрацах и одном источнике опорного напряжения.

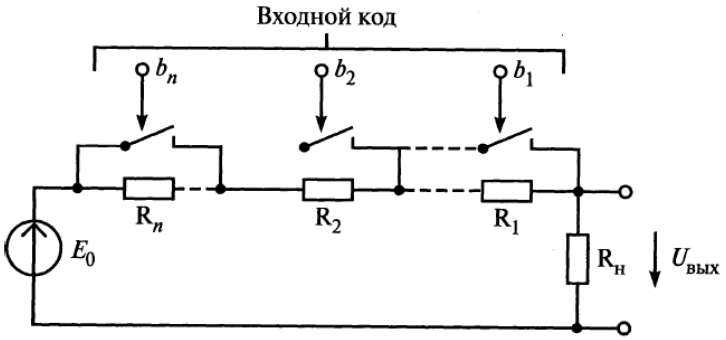


Рис. 6.4. Упрощенная схема ЦАП с делением напряжения

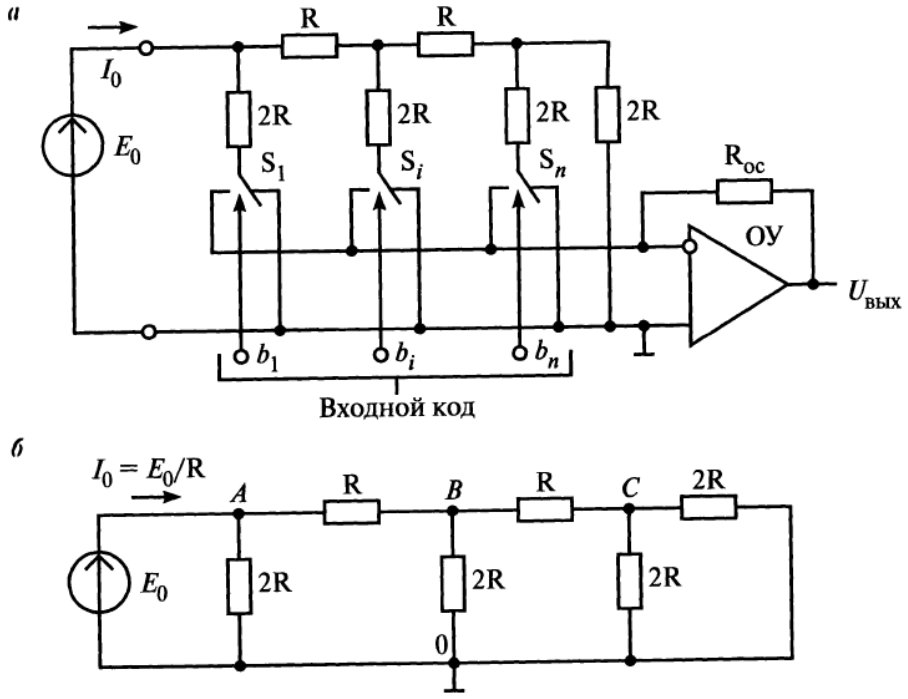


Рис. 6.5. Схема ЦАП: *а* — со сложением токов на резистивной матрице типа R—2R; *б* — структура резистивной матрицы

В схеме на рис. 6.5, *а* используют один источник опорного напряжения и резистивную матрицу типа R—2R. Особенность резистивной матрицы (рис. 6.5, *б)* заключается в том, что при любом положении ключей S_1, S_i, ..., S_n входное сопротивление матрицы всегда равно *R,*а следовательно, ток, втекающий в матрицу,равен $I_0 = E_0/R.$.Далее он последовательно делится в узлах*A, B, C* но двоичному закону. Двоичный закон распределения токов в ветвях резистивной матрицы соблюдается при условии равенства нулю сопротивления нагрузки. Так как нагрузкой резистивной матрицы является операционный усилитель ОУ, охваченный отрицательной обратной связью через сопротивление $R_{ос}$,то его входное сопротивление равно 0 с достаточно высокой точностью.

Напряжение на выходе операционного усилителя составляет

$$U_{\text{вых}} = \frac{E_0 R_{\text{ос}}}{R \cdot 2^n}\left(b_1 \cdot 2^{n-1} + b_2 \cdot 2^{n-2} + \ldots + b_n \cdot 2^0\right) = \frac{E_0 R_{\text{ос}}}{R \cdot 2^n} N,$$

где n — число разрядов преобразователя.

$b_i = 1$, если ключ S_i находится в положении, при котором ток протекает на инвертирующий вход ОУ., $b_i = 0$, если ключ S_i находится в положении, при котором ток протекает в общий вывод.

Максимальное значение выходного напряжения (т.е. напряжение в конечной точке диапазона) имеет место при всех $b_i = 1$ и равно

$$U_{\text{вых}} = \frac{E_0 R_{\text{ос}}(1-2^{-n})}{R \cdot 2^n} = \frac{E_0 R_{\text{ос}}}{R \cdot 2^n} - \Delta,$$

где $\Delta = \frac{E_0 R_{\text{ос}}}{2^n R}$.

Таким образом, выходное напряжение ЦАП зависит не только от входного кода *N,*но и от напряжения E_0 опорного источника если допустить, что напряжение E_0 меняется, то выходное напряжение ЦАП будет пропорционально произведению двух величин: входного кода и напряжения, поданного на вход опорного сигнала. Такие ЦАП обычно называют ***перемножающими.***

В ИМС перемножающих ЦАП источник опорного напряжения отсутствует, но имеется вход для его подключения.

ЦАП со сложением токов, реализуемый на матрице со взвешенными резисторами (рис. 6.6) состоит из матрицы двоично-взвешенных резисторов, сопротивления которых определяются по формуле $R_i = R \cdot 2^{i-n}$, переключателей на каждый разряд, управляемых входными

сигналами, источника опорного напряжения E_0 и сумматора на операционном усилителе ОУ в инвертирующем включении.

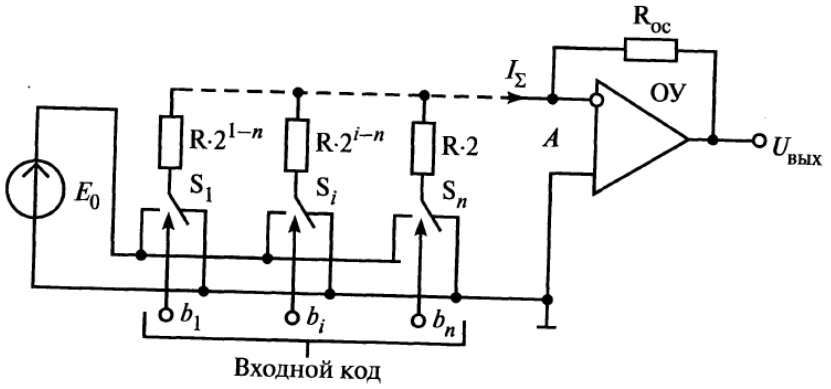


Рис. 6.6. Схема ЦАП со сложением токов на матрице со взвешенными резисторами

Прямой вход ОУ соединен с общим проводом, за счет отрицагельной обратной связи напряжение в суммирующей точкеA равно нулю т.е. резистивная матрица работает в закороченном режиме независимо от состояния переключателей. Когда на цифровые входы ЦАП подан двоичный n-разрядный цифровой код, то каждый цифровой сигнал b_i, управляет переключателем S_i ,обеспечивая подклю-чение резистора с сопротивлением $R_i = R \cdot 2^{i-n}$ к источнику опорного напряжения E_0 или к общему проводу. Если предположить, что внутренние сопротивления источника опорного напряжения и ключей равны нулю, то ток, протекающий в сопротивлении R_i, будет равен

$$I_i = \frac{E_0 b_i}{R \cdot 2^{i-n}} = \begin{cases} 0 & \text{при } b_i = 0; \\ \dfrac{E_0}{R \cdot 2^{i-n}} & \text{при } b_i = 1. \end{cases}$$

Результирующий ток:

$$I_\Sigma = \sum_{i=1}^{n} I_i = \frac{E_0}{R} \sum_{i=1}^{n} b_i \cdot 2^{n-1} = \frac{E_0}{R}\left(b_1 \cdot 2^{n-1} + \ldots + b_n \cdot 2^0\right).$$

Для обеспечения точности и стабильности резистивных матрицприменяется лазерная подгонка резисторов.

Практическая схема ***ЦАП с параллельными делителями напряжения***представлена на рис. 6.7.Каждый делитель состоит из двухсопротивлений R_i и R'_i, сумма значений которых остается постоянной во всем диапазоне преобразования.

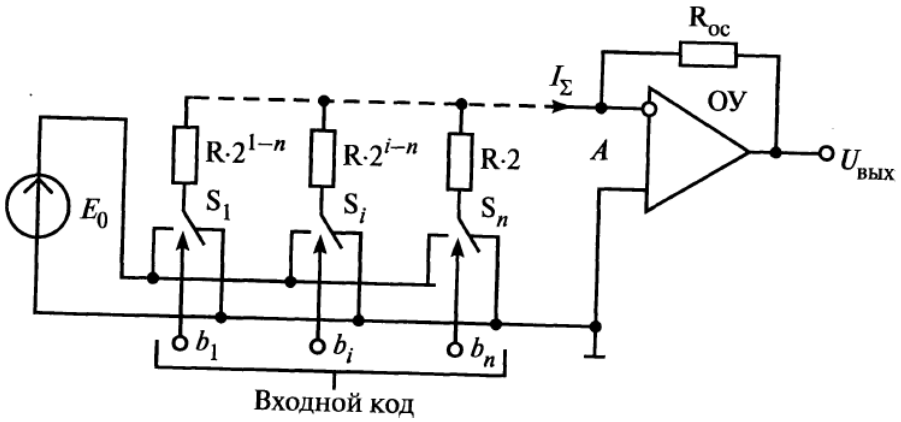


Рис. 6.6. Схема ЦАП со сложением токов на матрице со взвешенными резисторами

Коэффициент передачи каждого звена делителя определяется по формуле

$$\mu = \frac{R_i}{R_i + R_i'}.$$

Такой ЦАП применяют при управлении унитарным кодом и небольшой разрядности ЦАП.

Функциональная схема ЦАП типа К594ПА1 (рис. 6.8) представнмет собой параллельный ЦАП с суммированием токов на комбинированной матрице, которая состоит из взвешенных резисторов и резистивной матрицы R—2R.

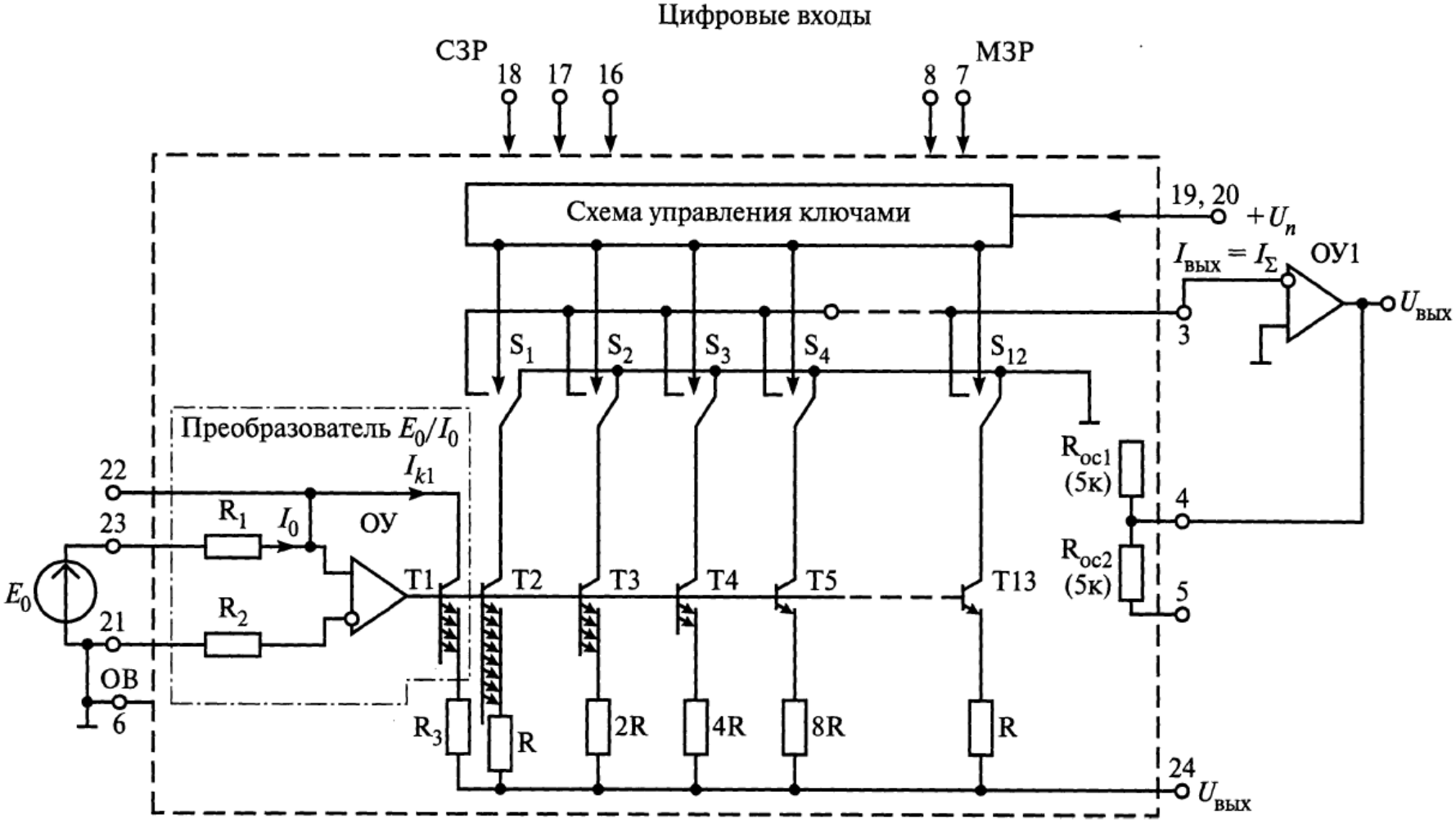


Рис. 6.8. Функциональная схема ЦАП типа К594ПА1

Лекция 19Аналого-цифровые преобразователи (АЦП) информации

Аналого-цифровой преобразователь **(АЦП)** — устройство, предназначенное для преобразования электрических величин в цифровой код. АЦП делятся на последовательные, параллельные и последовательно-параллельные

Последовательный АЦП с единичным приближением(рис. 6.9) со держит генератор импульсов, счетчик, на который поступает импульс при срабатывании триггера от сигнала «Запуск». Так как счетчик соединен с ЦАП, то напряжение на выходе последнего $U_{\text{ЦАП}}$ увеличивается ступенчато. Процесс преобразования заканчивается, когда напряжение $U_{\text{ЦАП}} = U_{\text{вх}}$ и начинается процесс считывания со счетчика выходного кода *N,*представляющего собой цифровой эквивалент входного напряжения в момент окончания преобразования. При повторной подаче импульса «Запуск» на триггер и разрешающего сигнала с компаратора процесс преобразования повторяется. Время преобразования АЦП данного типа является величиной переменной и зависит от уровня входного напряжения. Максимальное время преобразования, соответствующее максимальному входному напряжению,

$$t_{\text{пр.max}} = (2^n - 1)\Delta t_{\text{сч}},$$

где n — число разрядов;
$\Delta t = 1/f_{\text{сч}}$ — период следования счетных импульсов;
$f_{\text{сч}}$ — частота счетных импульсов.

Число разрядов определяется выбранным счетчиком, а время преобразования — частотой четных импульсов.

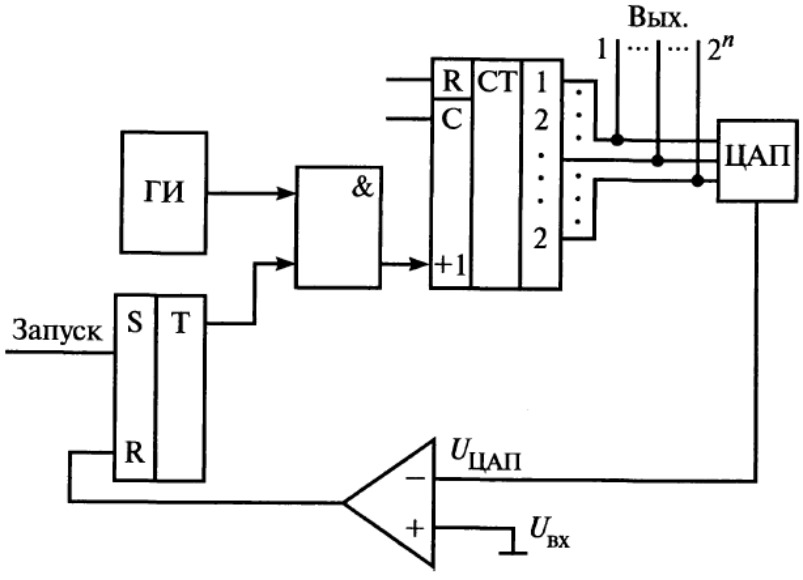


Рис. 6.9. Схема последовательного АЦП с единичным приближением

Минимальное время определяется исходя из минимального периода импульсов, который находят по формуле

$$\Delta t_{max} = t_{сч} + t_{ЦАП} + t_{т} + t_{к} + t_{л},$$

где $t_{сч}$ — максимальное время переходного процесса в счетчике;
$t_{ЦАП}$ — время установления выходного сигнала ЦАП;
$t_{т}$ — время срабатывания управляющегося триггера;
$t_{к}$, $t_{л}$ — время включения компаратора и время срабатывания элемента «И».

Последовательный АЦП с двоично-взвешенным приближением (рис. 6.10) содержит регистры последовательного сравнения, ЦАП и компаратор, сравнивающий входное напряжение с аналоговым на выходе ЦАП. Сравнение производится последовательно начиная с самого старшего разряда регистра сравнения, код которого и преобразуется в аналоговые напряжения в ЦАП. Работа АЦП состоит в следующем: на вход компаратора подаются два напряжения: U_{BX}и $U_{ЦАП}$. При сравнении этих двух напряжений компаратор вырабатывает управляющий сигнал и подает его на логический элемент «И».

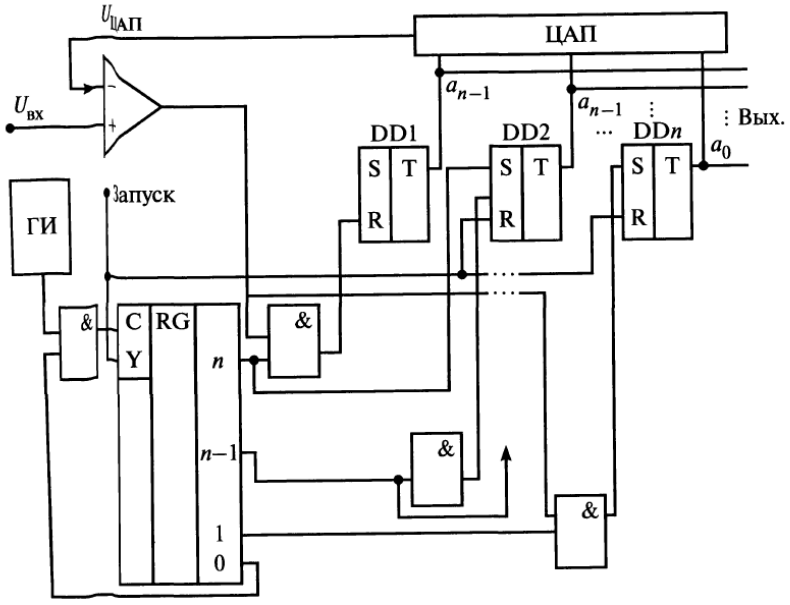


Рис. 6.10. Схема последовательного АЦП с двоично-взвешенным приближением

управляющий работой ЦАП. При поступлении импульса запуски DD1 устанавливается в состояние Q= 1, аDD2—DDnв состоянии Q =0. В начале преобразования компаратор сравнивает входное напряжение U_{BX}с эталонным $U_{э}$,поступающим из ЦАП. Если $U_{BX}>U_{э}$ то на выходе компаратора отсутствует импульс и в старшем разряде ЦАП (на выходе DD1) сохранится «0». При $U_{BX}<U_{э}$компаратор выдает импульс, который, пройдя через элемент «И», устанавливает DD1 в состояние «1». Одновременно

происходит сдвиг в регистре RGи сигнал «1» переходит в *(n* — 1)-й разряд, что обеспечит подачу напряжения $U_э$с ЦАП на компаратор. Аналогично выполняется работа АЦП и во всех остальных тактах.

Таким образом, за *n* тактов преобразуемое напряжение U_{BX} уравновешивается суммой эталонных напряжений, снимаемых с ЦАП. В итоге

за один такт преобразования шестиразрядным АЦП на его выходе получим код 011010, соответствующий входному напряжению.

Последовательные АЦП обладают высоким быстродействием и пригодны для построения многоразрядных преобразователей, благодаря чему они находят широкое применение в автоматизированных системах.

Параллельные АЦП осуществляют повременное квантование сигна набором компараторов, включенных параллельно источнику входного сигнала.

Пороговые уровни компаратора устанавливают резистивным делителем, подключенным к источнику опорного напряжения U_{on} ответствии со шкалой квантования (рис 6.11) Число уровней квантования (число компараторов) определяется выбранным шагом квантоваяΔU и значением входного сигнала. Каждый компаратор срабатывает при определенном напряжении, выдавая при этом на кодирующее устройство соответствующий код. Если,к примеру, входной сигнал находится от2,5 до 3,5ΔU все компараторы К1-К4 устанавливаются в состояние высокого уровня, а К5-К7 в состояние низкого уровня. Для преобразования унитарного кода в двоичный код используется кодирующее устройство.

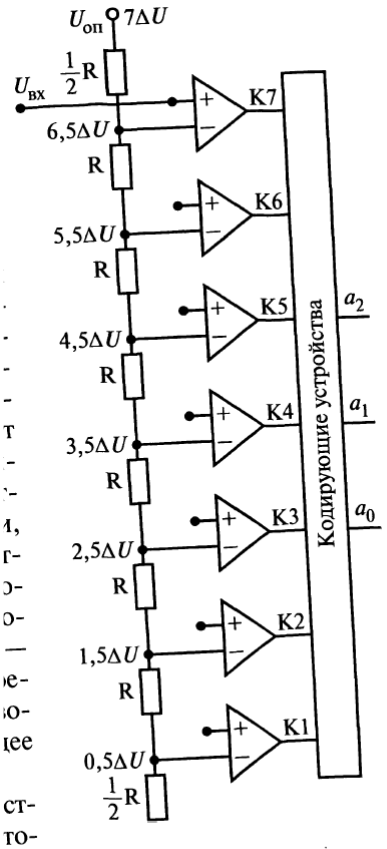


Рис. 6.11. Схема параллельного АЦП

Надежность работы АЦП, его быстродействие и точность зависят от того как подобраны и согласованы выходы и входы компараторов и кодирующего устройства.

,то

Для улучшения работы АЦП их дополняют устройствами стробирования,буферными регистрами или элементами постоянной памяти(рис. 6.12).

Стробируемый сигнал должен поступать одновременно на все компораторы. Во избижание ошибок в порядке срабатывания компараторов АЦП дополняется элементами «И» ,которые выделяют верхний срабатывающий компоратор. В качестве кодирующего устройства используется

ПЗУ,выполненое на диодной матрице и работающее только в режиме «считывание». В схеме элементы «И»

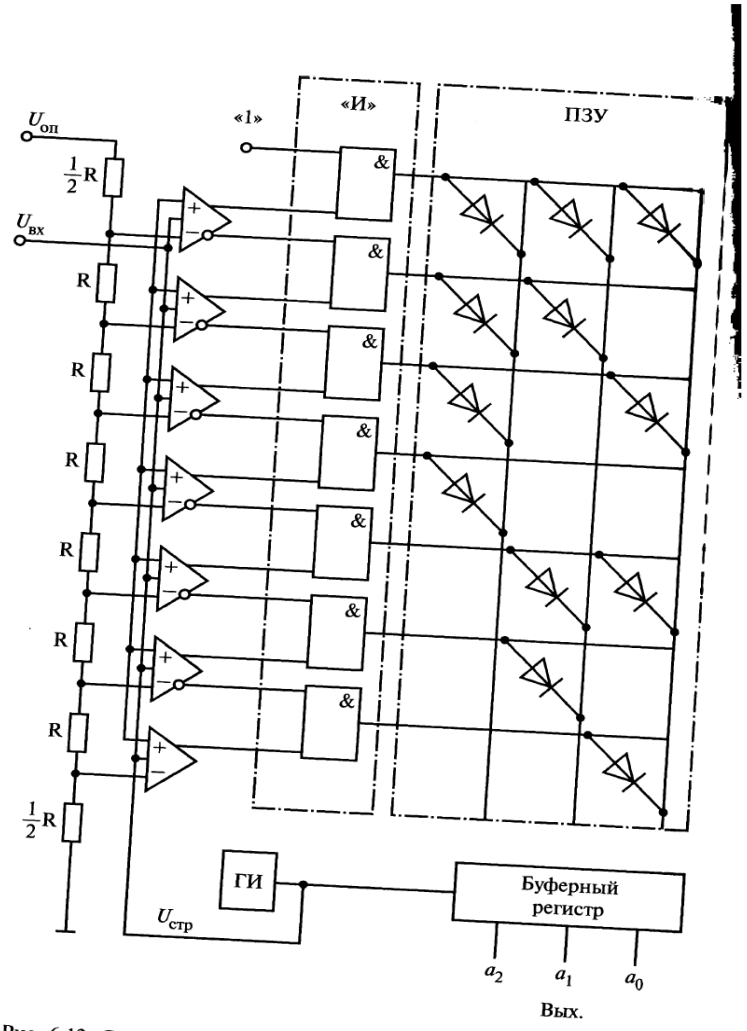


Рис. 6.12. Схема параллельного АЦП с элементами стабилизации

В схеме элементы «И» преобразуют прямые и инверсные выходные сигналы компаратора памятью так, что сигнал высокого уровня остается только на входной шине, соответствующей высшему номеру сработавшего компаратора. Для нормальной работы АЦП необходимо, чтобы за время считывания результатов с выходов компараторов входной сиг- изменялся не более чем на. $\Delta U.$

ПараллельныеАЦП обладают самым высоким быстродействием по сравнению с другими типами АЦП, но содержат большое число компараторов, что сдерживает разработку многоразрядных преобразователей. Кроме того, точность преобразования ограничивается точностью и стабильностью компараторов и резистивных делителей. По данному принципу преобразования построены АЦП типа К1107ПВ1.

АЦП двухтактного интегрирования (рис. 6.13) имеет хорошую линейную характеристику, малые шумы, высокую помехозащищенность и

низкую стоимость. Полный рабочий цикл состоит из двух тактов: первый такт интегрирования входного сигнала и второй такт интегрирования опорного напряжения.

Сигнал «Запуск» воздействует на DD1, включается ключ Кл1. Входной сигнал напряжение U_{BX}поступает на вход интегратора, где он преобразуется по закону

$$U_{\text{инт}} = \frac{1}{RC}\int_{t_1}^{t_2} U_{\text{вх}}(t)\,dt.$$

На вход компаратора подается сигнал интегрирования и нулевое напряжение. Так как в начальный момент времени $(t_1)\ U_{\text{инт}} = 0,$ компаратор срабатывает и переводит DD3 в состояние Q= 1, в результате чего элемент «И» открывается и импульсы с ГИ поступают на DD4. Входной сигнал интегрируется за фиксированный интервал времени

$$T_2 = t_2 - t_1 = 2^n \Delta t_{\text{сч}}.$$

Конец интервала T_2 фиксируется счетчиком, который в момент времени t_2выдает сигнал переполнения, поступающий на DD1, DD2. При этом ключ Кл1 выключается, а ключ Кл2 включается, и начинается второй рабочий такт. Теперь на вход интегратора поступает опорное напряжение U_{on}обратной полярности.

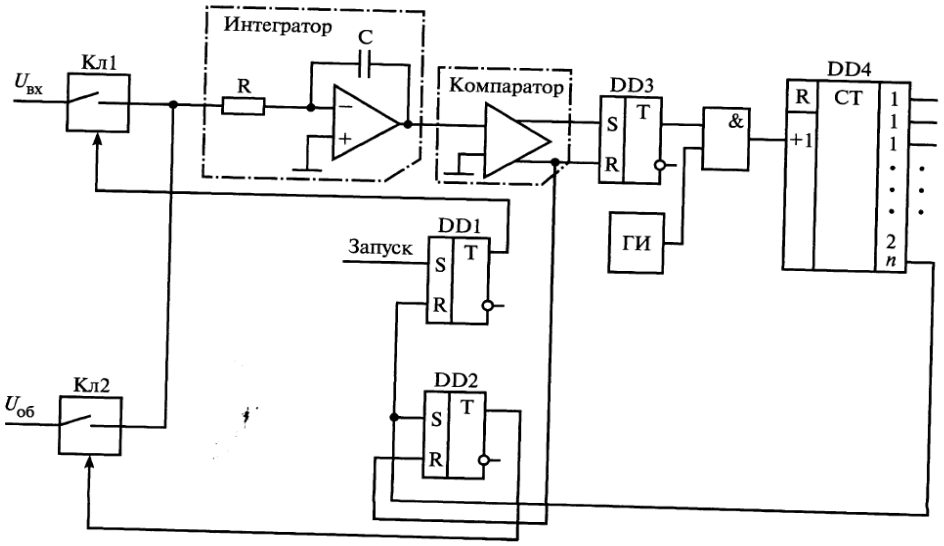


Рис. 6.13. Схема АЦП двухтактного интегрирования

Начиная **с** момента времени t_2счетчик вновь заполняется импульсами от генератора, а напряжение на выходе интегратора изменяется по закону

$$U_{\text{ИНТ}} = U_{\text{инт}}(t_2) - \frac{1}{RC}\int_{t_1}^{t_2} U_{\text{оп}}(t)\,dt.$$

В момент времени напряжение на выходе интегратора становится равным нулю, компаратор возвращается в исходное положение и по инверсному выходу переводит триггер DD2 в нулевое состояние. При этой напряжение U_{on}отключается от интегратора, а сигнал с DD3 запрещает подачу импульсов с ГИ на счетчик, который фиксирует при этом числовой код. $N = T_3/\Delta t_{\text{сч}}$.

Лекция 20

Общие сведения о микропроцессорах и микропроцессорных системах. Два подхода к построению.

Структура процессора и подходы к его построению

В зависимости от области применения разработано несколько основных типов электронно-вычислительных машин (ЭВМ): сверхпроизводительные ЭВМ (системы), ЭВМ общего назначения, мини-ЭВМ, микроЭВМ и микропроцессоры. Они существенно отличающихся по своим характеристикам, структуре аппаратных и программных средств и объему подключаемых к ним периферийных устройств. Первая ЭВМ была создана в 1946 г. Начало выпуска первых микроЭВМ — 1972 г.

Процессор — это функциональная часть ЭВМ, предназначенная для обработки информации в соответствии с заложенной в ЭВМ программой. Синтезируется в виде соединения двух устройств: операционного (выполняет все операции) и управляющего (координирует действия узлов операционного устройства) (рис. 8.1).

Каждое элементарное действие, выполняемое в одном изузлом операционного устройства (ОУ) в течение одного тактового периода, называется ***микрооперацией.*** В определенные тактовые периоды одновременно могут выполняться несколько микроопераций. Такая совокупность одновременно выполняемых микроопераций называется ***микрокомандой***(МК), а весь набор микрокоманд, предназначенный для решения определенной задачи – ***микропрограммой.***

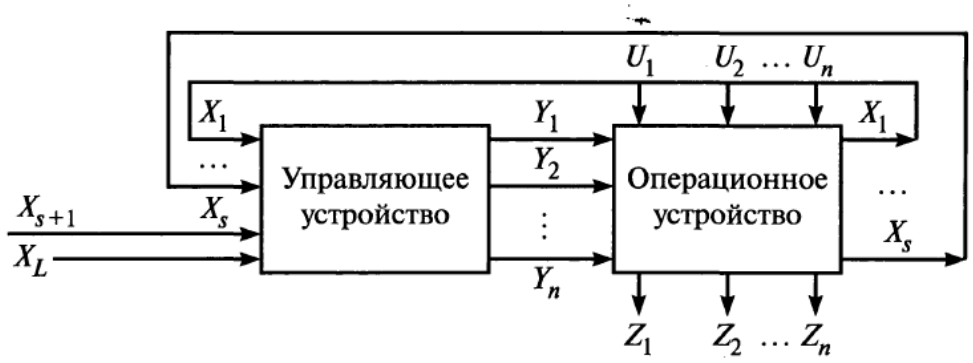


Рис. 8.1. Структурная схема процессора: X_1, ..., X_s — определяют состояние ОУ, X_{s+1}, ..., X_L — внешние сигналы; U_1, U_2, ..., U_n — входные данные; Y_1, Y_2, Y_n — сигналы управления; Z_1, Z_2, ..., Z_n — выходные данные

Так как управляющее устройство (УУ) определяет микропрограмму, то его называют микропрограммным автоматом.

Существует два принципиально разных подхода к проектированию микропрограммного автомата (управляющего устройства): использование принципа схемной логики и использование принципа программируемой логики.

В первом случае в процессе проектирования подбирается некоторый набор цифровых микросхем (обычно малой и средней степени интеграции) и определяется такая схема соединения их выводов,которая обеспечивает требуемое функционирование (рис.8.2).

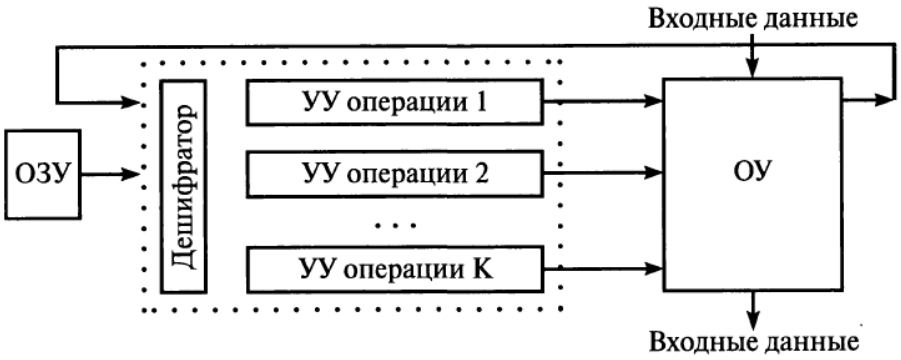


Рис. 8.2. Принцип схемной логики

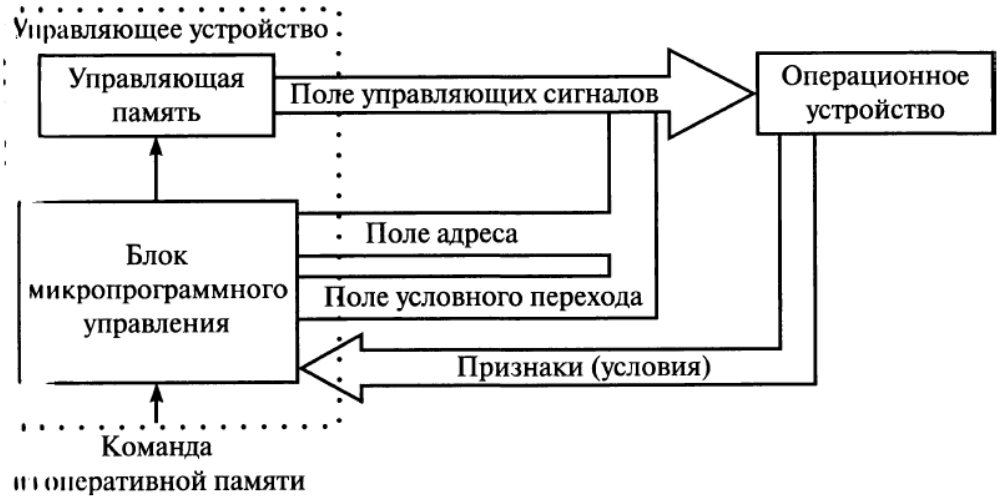


Рис. 8.3. Принцип программируемой логики

Второй подход предполагает построение с использованием од ной или нескольких больших интегральных схемах (БИС) некою рого универсального устройства, в котором требуемое функции нирование (т.е. специализация устройства на выполнение опреде ленных функций) обеспечивается занесением в память устройства определенной программы (или микропрограммы) (рис. 8.3). Функции блока микропрограммного управления (БМУ) сводятся к определению адреса очередной микрокоманды в управляющей памяти (УП). Поступающая из оперативной памяти (ОП) команда содержит адрес первой МК той микропрограммы, которая реализует предусматриваемую командой операцию. Микрокоманда разбивается на две части: одна часть — поле адреса и поле условных переходов (определяет функционирование БМУ при

нахождении адреса очередной МК); другая — поле управляющих сигналов (определяет функционирование ОУ).

Цифровой микропрограммный автомат (МПА) — устройство последовательностного типа, осуществляющее прием, хранение и преобразование дискретной информации по некоторому алгоритму Примером такого автомата является процессор.

МПА под действием входных сигналов принимает состояния и соответствии с набором значений входных сигналов и выдает сигнал, зависящий от внутреннего состояния либо от внутреннего состояния и входных сигналов. Существует два вида МПА: автомат Мили и автомат Мура.

Функционирование автоматов Мили и Мура происходит на трех множествах, это:

- множество возможных входных сигналов

$$x_1, x_2, \ldots, x_n,$$

где n — число возможных входных сигналов;

- множество внутренних состояний

$$a_0, a_1, \ldots, a_k,$$

где a_0 — начальное состояние, в которое перед началом работы устанавливают автомат;

k — число внутренних состояний;

- множество возможных выходных сигналов

$$y_1, y_2, \ldots, y_m,$$

где m — число возможных выходных сигналов.

Работа ***автомата Мили*** определяется

- функцией переходов f:

$$a(t + 1) = f(a(t)); x(t));$$

-функцией выходов φ:

$$y(t) = \varphi(a(t)); x(t)).$$

Особенность ***автомата Мура***заключается в том, что в нем выходной сигнал зависит лишь от внутреннего состояния *a(t)*и не зависит от входного сигнала. Работа автомата Мура определяется:

– функцией переходов f:

$$a(t + 1) = f(a(t));$$

– функцией выходов φ:

$$x(t)), y(t) = \varphi(a(t)).$$

МПА может быть задан словесным алгоритмом, таблицей переходов и выходов, графом состояний и системой уравнений, выражающих зависимости между входами, выходами и внутренними состояниями автомата.

Граф автомата Мили (рис. 8.4, *а)*состоит из узлов (круги), отождествляемых с состояниями автомата. Связи между узлами (стрелки) показывают переходы автомата из одного состояния в другое под воздействием входных сигналов. На каждой связи указывается (через запятую) входной сигнал и формируемый на выходе автомата выходной сигнал.

В табл. 8.1 в клетках, расположенных на пересечении строк текущего состояния автомата $(a_0, ..., a_k)$ со столбцами текущего значения входного сигнала $(x_1, ..., x_n)$, указываются (через дробь) следующее состояние a_j (j изменяется от 0 до k) и значение выходного сигнала y_i (i изменяется от 1 до m).

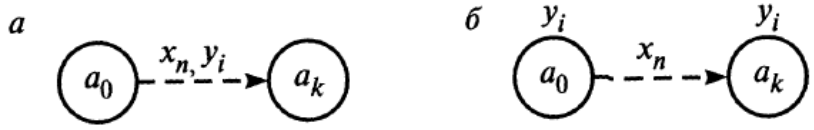


Рис. 8.4. Общий вид графа: *а* — автомат Мили; *б* — автомат Мура

Таблица 8.1

Общий вид таблицы переходов и выходов автомата Мили

Текущее (внутреннее) состояние	Входной сигнал		
	x_1	...	x_n
a_0	a_j/y_i	...	a_j/y_i
...	...	...	...
a_k	a_j/y_i		a_j/y_i

Граф автомата Мура (рис. 8.4, *б)* состоит из узлов (круги), отождествляемых с состояниями автомата, над которыми указывают формируемый на выходе автомата выходной сигнал. Связи между узлами (стрелки) показывают переходы автомата из одною состояния в другое под воздействием входных сигналов. На каждой связи указывается входной сигнал.

Таблица 8.2

Общий вид таблицы переходов и выходов автомата Мура

Текущее (внутреннее) состояние	Выходной сигнал	Входной сигнал		
		x_1	...	x_n
a_0	y_i	a_j	...	a_j
...	...	...	...	...
a_k	y_i	a_j		a_j

В табл. 8.2 в клетках, расположенных на пересечении строк текущего состояния автомата $(a_0, ..., a_k)$ и выходного сигнала y_i со столбцами текущего значения входного сигнала $(x_1, ..., x_n)$, указывается следующее состояние a_j.

Пример. Перейти от графа состояния (рис. 8.5 и рис. 8.6) к таблице переходов и выходов, записать системы множеств.

1. Составим по графу автомата Мили (рис. 8.5) таблицу переходов и выходов (табл. 8.3).

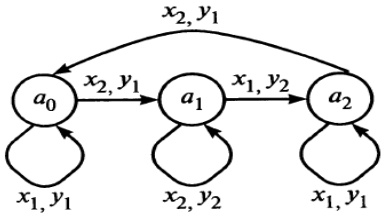


Рис. 8.5. Граф Мили

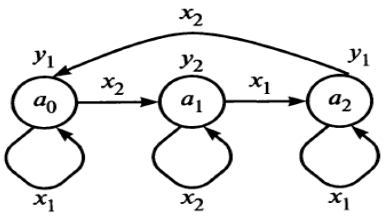


Рис. 8.6. Граф Мура

Таблица 8.3

Таблица переходов и выходов для автомата Мили

Текущее (внутреннее) состояние	Входной сигнал	
	x_1	x_2
a_0	a_0/y_1	a_1/y_1
a_1	a_2/y_2	a_1/y_2
a_2	a_2/y_1	a_0/y_1

2. Запишем систему множеств:

– множество возможных входных сигналов $x_1, x_2, x_2, x_1, x_1, x_2, ...$;

– множество внутренних состояний $a_0, a_0, a_1, a_1, a_2, a_2, a_0, ...$;

– множество возможных выходных сигналов $y_1, y_1, y_2, y_2, y_1, y_1, ...$

3. Составим по графу автомата Мура (рис. 8.6) таблицу переходов и выходов (табл. 8.4).

Таблица 8.4

Таблица переходов и выходов для автомата Мура

Текущее (внутреннее) состояние	Выходной сигнал	Входной сигнал	
		x_1	x_2
a_0	y_1	a_0	a_1
a_1	y_2	a_2	a_1
a_2	y_1	a_2	a_0

4. Запишем систему множеств:
– множество возможных входных сигналов $x_1, x_2, x_2, x_1, x_1, x_2, \ldots$;
– множество внутренних состояний $a_0, a_0, a_1, a_1, a_2, a_2, a_0, \ldots$;
– множество возможных выходных сигналов $y_1, y_1, y_2, y_2, y_1, y_1, \ldots$

Лекция 21

Однокристальные микропроцессоры. Структурная схема и архитектурное построение однокристального микропроцессора.

Микропроцессор(МП) — это функциональная часть вычислительной машины, предназначенная для интерпретации (преобразования) предложений исходной программы в эквивалентную форму двоичных кодов. Физически МП реализуется в виде ИМС высокой степени интеграции. Первый МП был создан фирмой Intelв 1971 г.

МП классифицируются:

1.По назначению:
- универсальные (решают широкий круг задач);
- специализированные (решают конкретные задачи).

2.По числу БИС:
- однокристальные (одна БИС или СБИС);
- многокристальные (несколько БИС).

3.По разрядности:
- с фиксированной разрядностью;
- с наращиваемой разрядностью.

4.По способу управления:
- с микропрограммным управлением;
- с жестким управлением.

Структура МП представлена на рис. 9.1. Основой ее является общая внутренняя шина, соединяющая элементы операционного блока (АЛУ, А, РС, РОН, УС, БА, БД) и элементы управляющего блока (УУ, РК, СТК).

Арифметико-логическое устройство (АЛУ) предназначено для выполнения арифметических операций (сложение с передачей переноса в младший разряд и без учета этого переноса, вычитание с передачей заема в младший разряд и без него), логических операций (конъюнкции, дизъюнкции, неравнозначности, сравнения) и

циклического сдвига.

Устройство управления (УУ) служит для формирования управляющих сигналов (запись, чтение, прерывание и т.д.).

Аккумулятор (А) необходим для обмена информацией с внешними устройствами (т.е. содержимое этого регистра либо может быть выдано на вход МП, либо с входа МП в него может быть принято число от внешнего устройства), при выполнении операций АЛУ в него помещается результат.

Регистр команд (РК) служит для временного хранения кода выполняемой команды.

Регистр адреса (РА) предназначен для фиксации адреса.

Счетчик команд (СТК) необходим для хранения адреса команды. После выборки из оперативной памяти текущей команды содержимое счетчика увеличивается на единицу, и таким образом формируется адрес очередной команды.

Индексный регистр (ИР) предназначен для формирования адресов запоминающих устройств.

Регистр состояния (РС) фиксирует признаки операций АЛУ, состояние МП. Содержимое регистра (флаг) используется для организации переходов.

Регистры общего назначения (РОН) — группа регистров, в которых могут храниться данные или адреса, они образуют внутреннюю (местную) память МП.

Стек (СОП) — сверхоперативная память с определенной (упрощенной) формой адресации. Использует ячейки со старшими адресами, и по мере заполнения стека занимаются ячейки с адресами, последовательно убывающими: принцип «последним вошел — первым вышел»;

Регистр признаков (РП) служит для хранения определенных признаков, выявляемых в числе, которое представляет собой результат выполнения некоторых операций. Пять триггеров этого регистра представлены в табл. 9.1

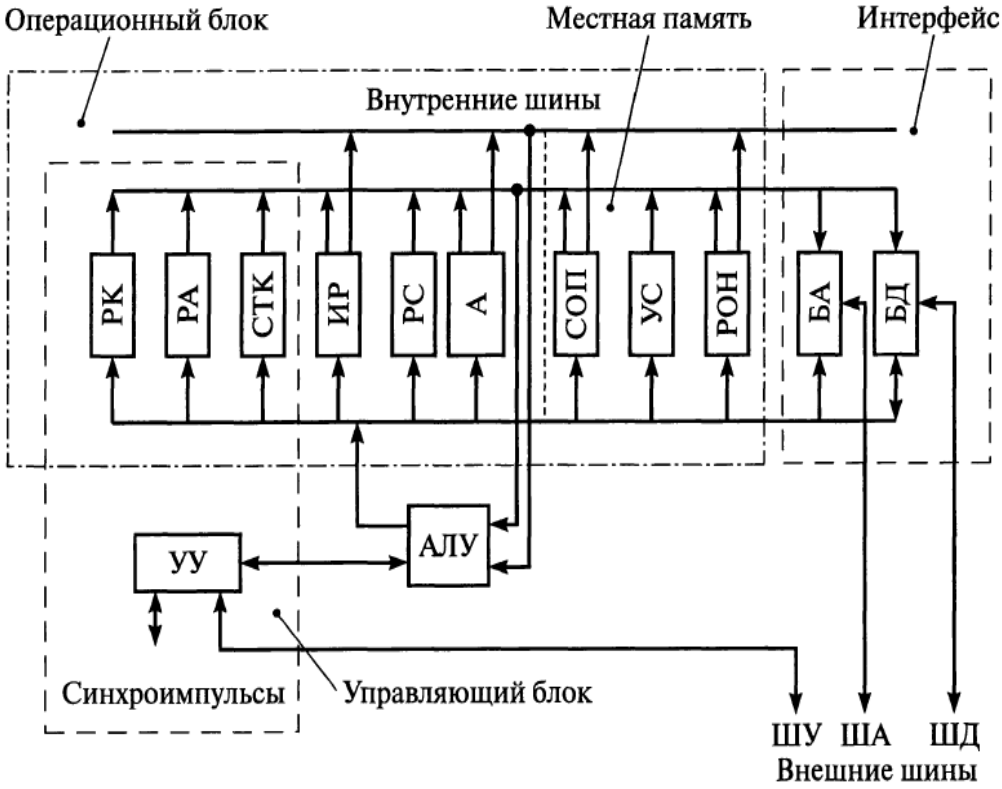


Рис. 9.1. Структурная схема микропроцессора

Таблица 9.1

Обозначение состояния регистра признаков

Триггер	Признак	Условие истинности
Tz	Нуля	Нулевой результат
Ts	Знака	Старший разряд результата равен «1» — число отрицательное; «0» — число положительное
Tc	Переноса	Перенос или заем из старшего разряда
Tv	Дополнительного переноса	Перенос из младшей четверки разрядов
Tp	Четности	Число единиц четно

Указатель стека (УС) предназначен для хранения адреса первой свободной ячейки в стеке. При чтении из стека производится выборка содержимого ячейки по адресу, хранящемуся в УС; при записи в стек вводимое в стек число помещается в ячейку с адресом, на единицу меньшим содержимого УС; одновременно с записью и чтением изменяется содержимое УС: при записи уменьшается, а при чтении увеличивается на единицу.

Буферные регистры (БА, БД) фиксируют признаки операций АЛУ и состояния МП. Содержимое регистров используется для организации переходов внутри программы в соответствии с заданны ми условиями и для временного хранения операнда.

Использованные источники:

1. Фролов, В.А. Цифровая схемотехника часть 1 : учебное пособие / В. А. Фролов. — Москва : ФГБУ ДПО «Учебно методический центр по образованию на железнодорожном транспорте», 2020. — 292 с. — 978-5-907206-18-2. — Текст : электронный // УМЦ ЖДТ : электронная библиотека. — URL: https://umczdt.ru/books/1194/242200/ (дата обращения 10.10.2023). — Режим доступа: по подписке.
2. Фролов, В.А. Цифровая схемотехника часть 2 : учебное пособие / В. А. Фролов. — Москва : ФГБУ ДПО «Учебно методический центр по образованию на железнодорожном транспорте», 2020. — 400 с. — 978-5-907206-19-9. — Текст : электронный // УМЦ ЖДТ : электронная библиотека. — URL: https://umczdt.ru/books/1194/242201/ (дата обращения 10.10.2023). — Режим доступа: по подписке.