ЭЛЕКТРОНИКА

ЛЕКЦИЯ 5

ЛОГИЧЕСКИЕ ОСНОВЫ ЦИФРОВЫХ УСТРОЙСТВ
(2 ЧАС)

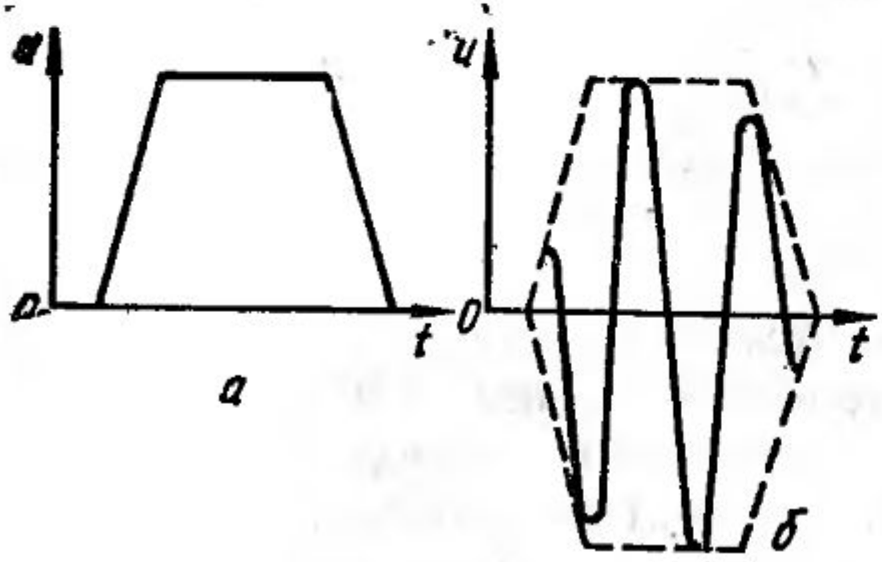
(ОСНОВНЫЕ ЛОГИЧЕСКИЕ ОПЕРАЦИИ И ТАБЛИЦЫ ИСТИННОСТИ; ЭЛЕМЕНТЫ ИЛИ-НЕ И И-НЕ; РЕАЛИЗАЦИЯ СЛОЖНЫХ ЛОГИЧЕСКИХ ФУНКЦИЙ; МИНИМИЗАЦИЯ ЛОГИЧЕСКИХ ФУНКЦИЙ; ЗАПИСЬ ЛОГИЧЕСКИХ ФУНКЦИЙ В УНИВЕРСАЛЬНЫХ БАЗИСАХ; ИНТЕГРАЛЬНЫЕ СХЕМЫ, БАЗОВЫЕ МАТРИЧНЫЕ КРИСТАЛЛЫ И ПЛИС)

Королев Владимир Александро

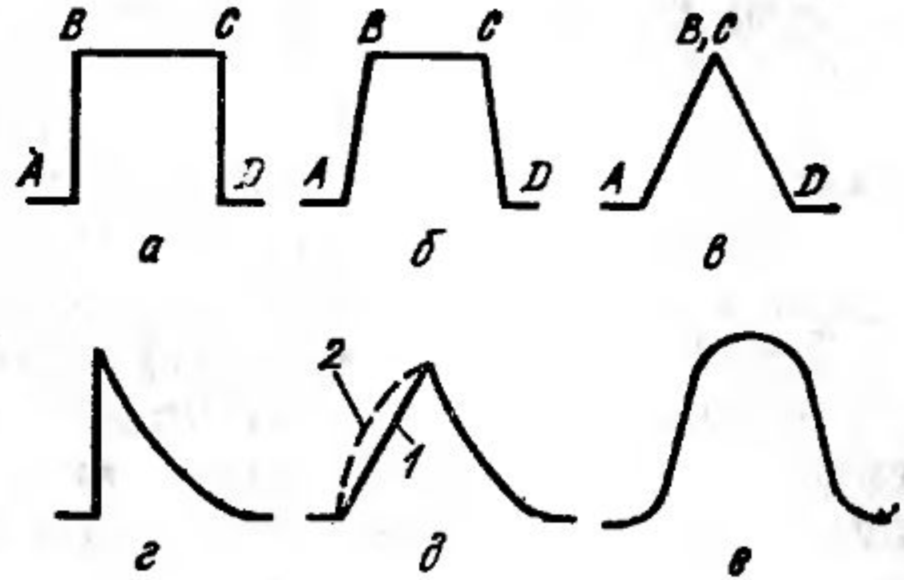
Дискретные электронные устройства (ДЭУ) предназначены для приёма, преобразования и передачи электрических сигналов, полученных путём квантования по времени и/или уровню исходной аналоговой функции *x (t)*.

Квантованием наз. процесс замены непрерывного сигнала его дискретными значениями в отдельных точках

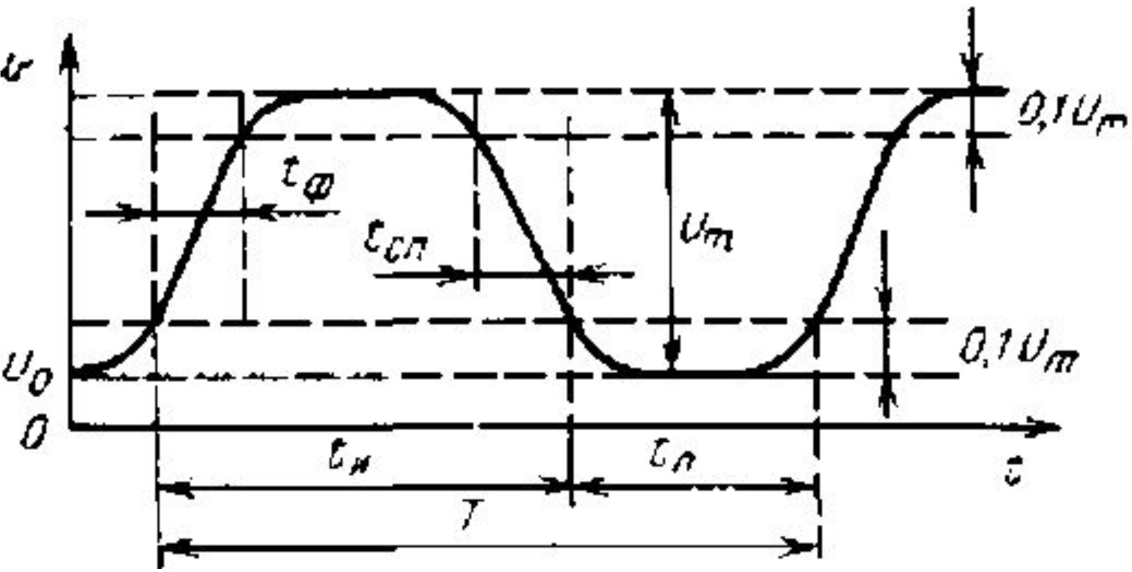
Действующие в них сигналы пропорциональны конечному числу выбранных по определённому закону значений реальной физической величины, отображаемой в виде различных параметров импульсов или перепадов сигнала, но так как информация о её изменении может быть получена только при сравнении двух импульсов, полу- нс такой информации растягивается во времени. Следовательно,, для получения полной информации о конечном во времени физическом процессе необходимо бесконечное число импульсов, т. е. временные масштабы протекания физического процесса и его отображения при помощи импульсов не совпадают. Поэтому в ДЭУ используется только часть информации о реальной физической величине, т. е. процесс представления ин формации сопряжён с частичной ее потерей.

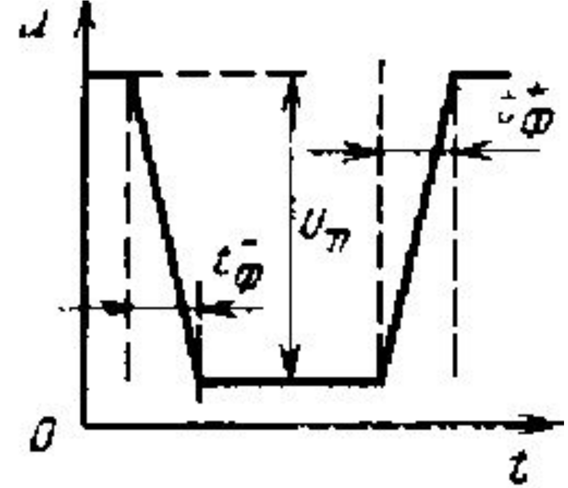


u
0
t
а
u
0
t
б



A
B
C
D
а
A
B
C
D
б
A
B,C
D
в
г
2
1
д
е





Электрическим импульсом наз. кратковременное периодически повторяющееся отклонение напряжения *u(t)* или тока *i(t)* от установившегося значения.

Перепадами напряжения или тока наз. быстрое изменение u(t) или i(t) между двумя постоянными уровнями.

Величина $f=1/T$ наз. частотой следования импульсов;

t_n = ***длительность паузы между импульсами***; $K_з = t_и/T$ — ***коэффициент заполнения импульсов***; у= $T/t_и$— ***скважность импульсов***.

Периодически повторяющиеся перепады напряжения с производными *dufdt* различных знаков (положительные *duldt*>0 и отрицательные *du/fdt*<0 перепады) образуют импульсы прямоугольной формы. В частном случае, когда положительные и отрицательные перепады следуют через равные промежутки времени, напряжение прямоугольной формы называют ***меандром***.

Импульсные признаки, используемые для передачи двоичных кодов

Символ	Амплитудные			Временные			Полярные	Частотные		Фазовые
1			U_1	τ						
0			U_2	τ_1						

Рис. 1.2. Последовательная передача кодовой комбинации видеоимпульсами

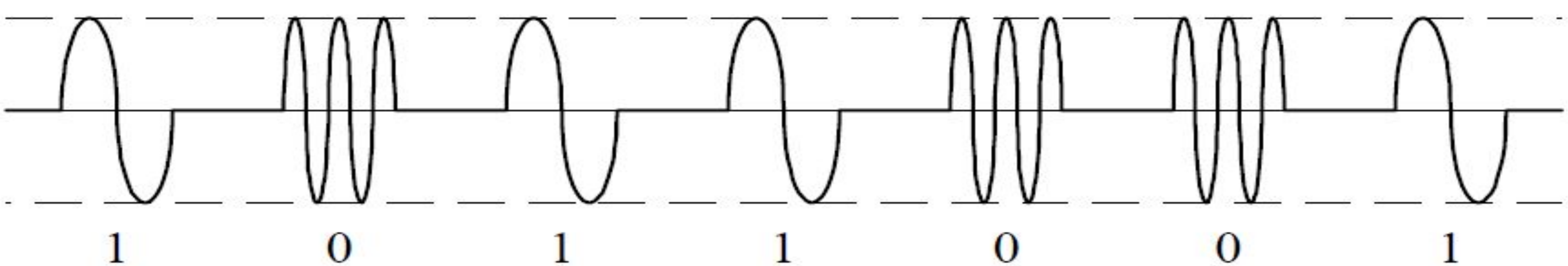
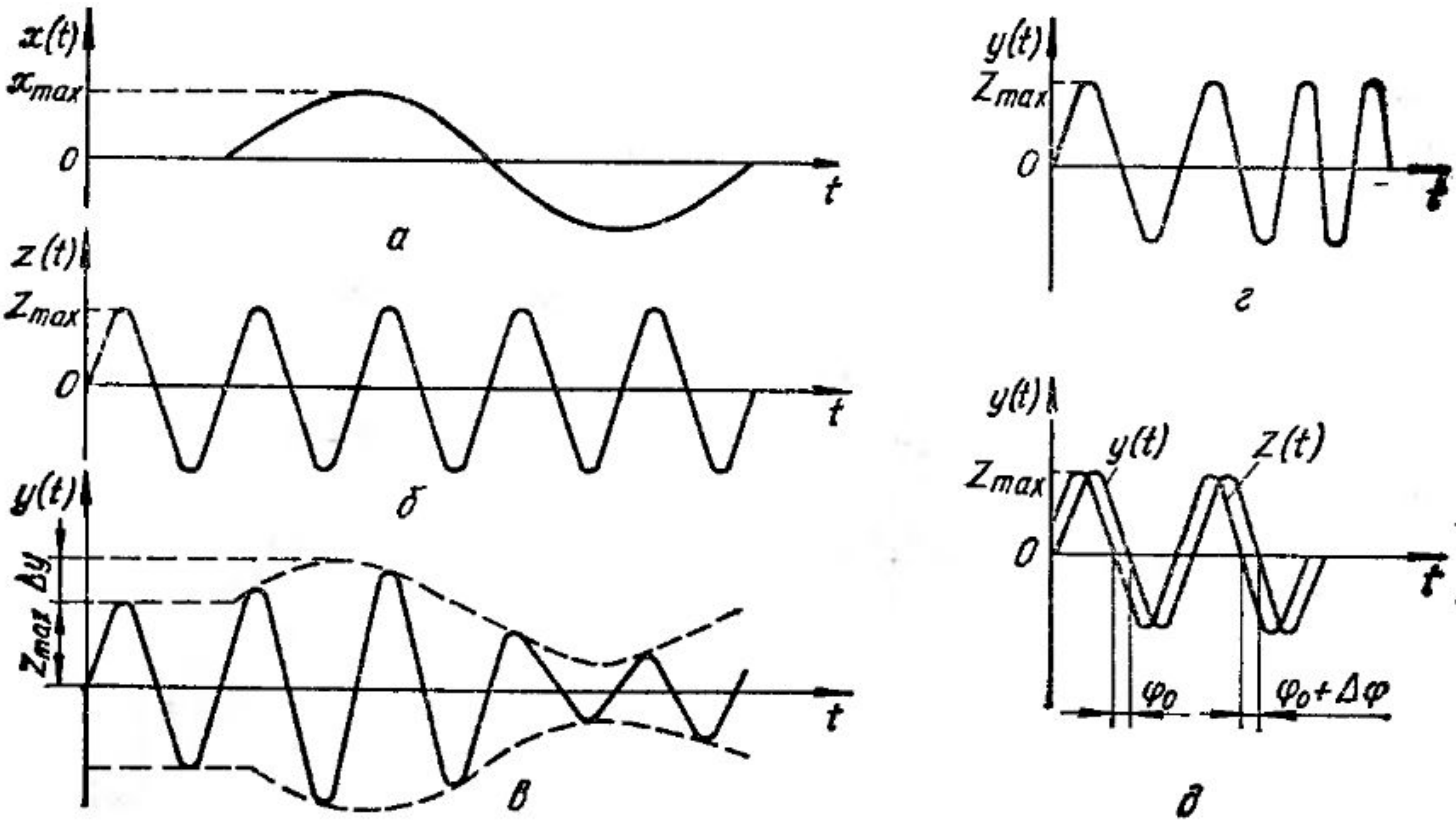


Рис. 1.3. Последовательная передача кодовой комбинации

Параллельная передача кодовых комбинаций

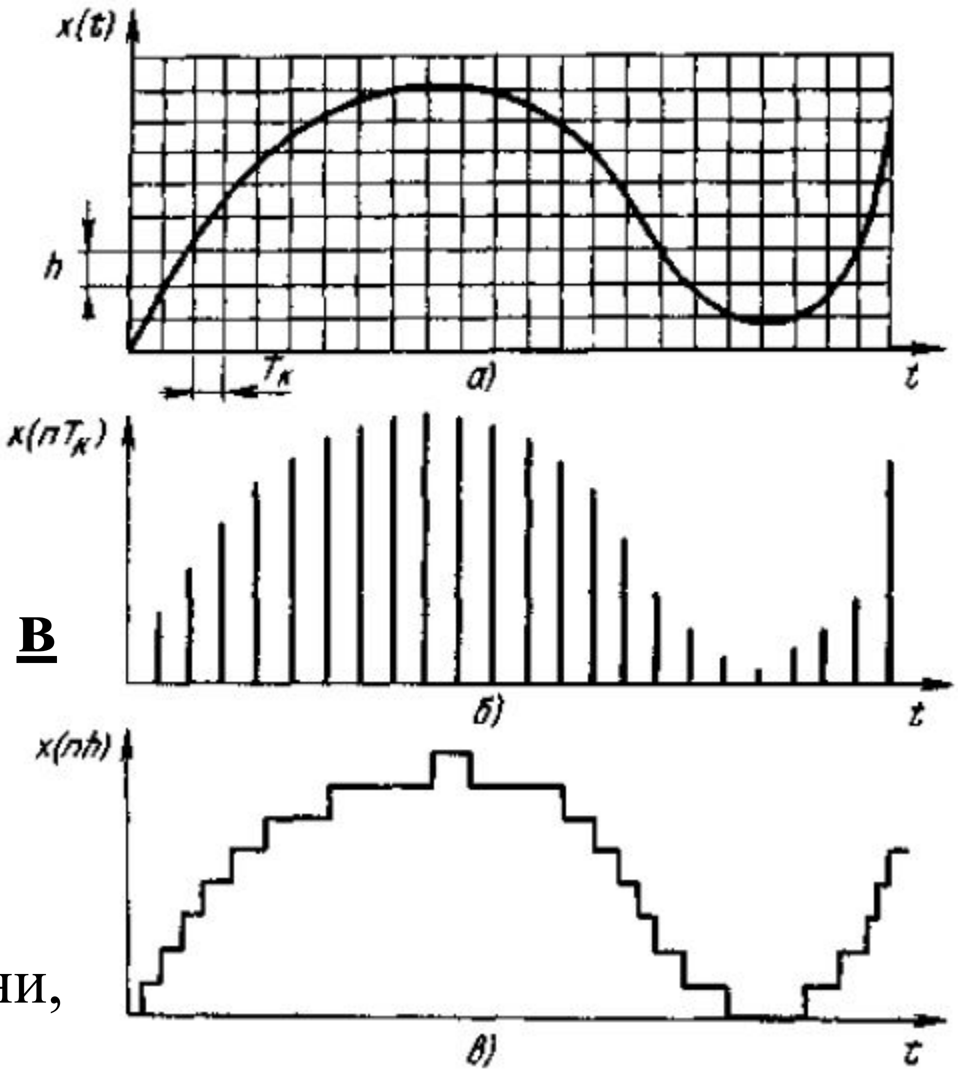
Номер разряда	Частота	Номер кодовой комбинации и время её передачи	
		1-t_1	2-t_2
1	f_1	1	1
2	f_2	0	1
3	f_3	0	1
4	f_4	1	0


x(t)
x_max
0
t
а
z(t)
Z_max
0
t
б
y(t)
Δy
Z_max
t
в
y(t)
Z_max
0
t
г
y(t)
y(t)
Z(t)
Z_max
0
t
φ0
φ0+Δφ
д

По типу квантования сигнала ДЭУ делят на три подкласса: импульсные, релейные и цифровые.

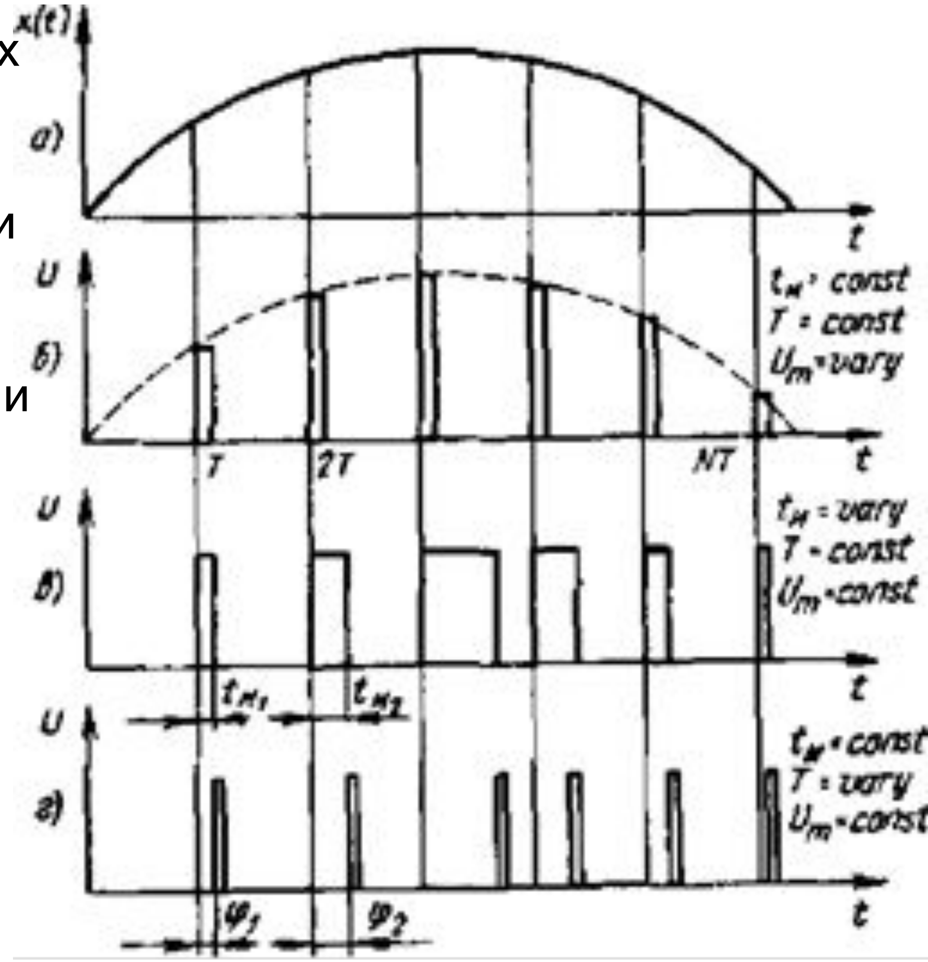
Импульсные электронные устройства **(ИЭУ) реализуют квантование исходного сигнала *x(t)* по времени и преобразуют его в последовательность импульсов, как правило, неизменной частоты.**

В ИЭУ хотя и нарушается непрерывность представления сигналов (информация) во времени, сами значения для выбранных моментов времени точно соответствуют значениям *x(t)*, т. е. непрерывность сигнала по величине сохраняется.



Квантование аналогового сигнала $x(t)$ (a) по времени ($б$) и уровню ($в$)

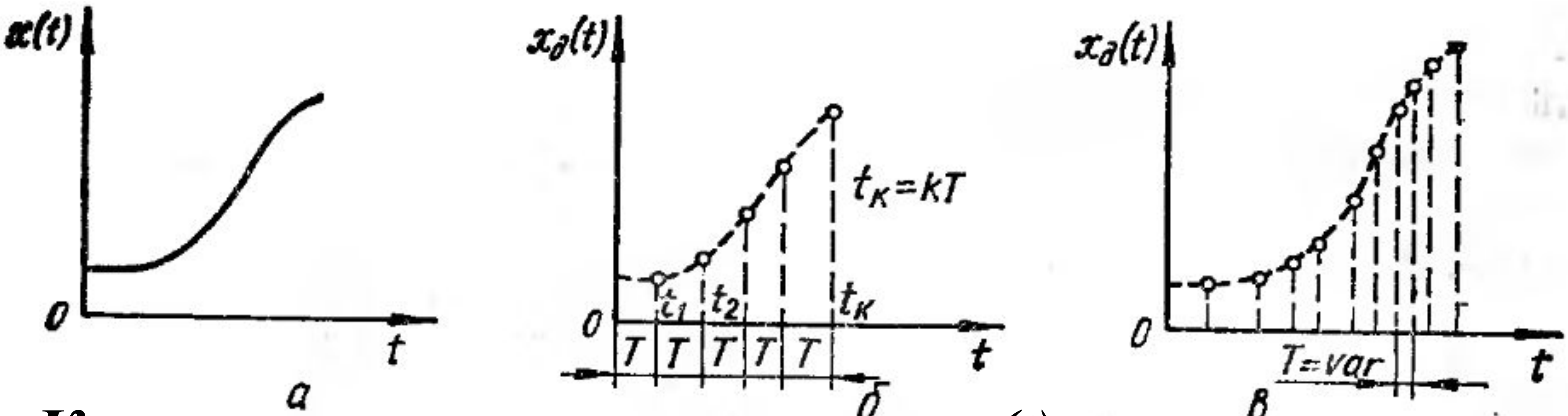
Импульсные электронные устройства реализуют квантование исходного сигнала x(t)) по времени и преобразуют его в последовательность импульсов, как правило, неизменной частоты. В ИЭУ хотя и нарушается непрерывность представления сигналов (информация) во времени, сами значения для выбранных моментов времени точно соответствуют значениям x(t), т. е. непрерывность сигнала по величине сохраняется.



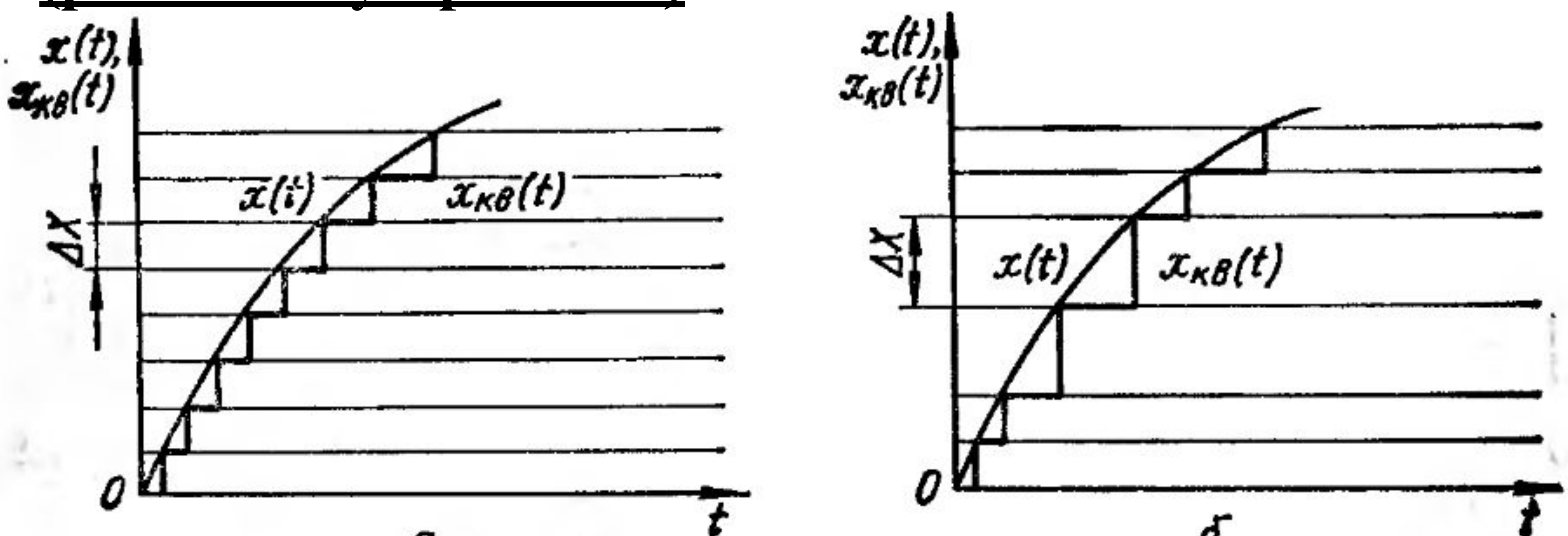
Виды импульсной модуляции

а — изменение исходной аналоговой величины; б — последовательность амплнтудно-модулирояаияых Ь - импульсов: в — последовательность широтно-модулированных импульсов; г — последовательность фазо-модулированных импульсов

Квантование исходного сигнала *x(t)* по времени (импульсные устройства)



Квантование исходного сигнала *x(t)* по уровню (релейные устройства)

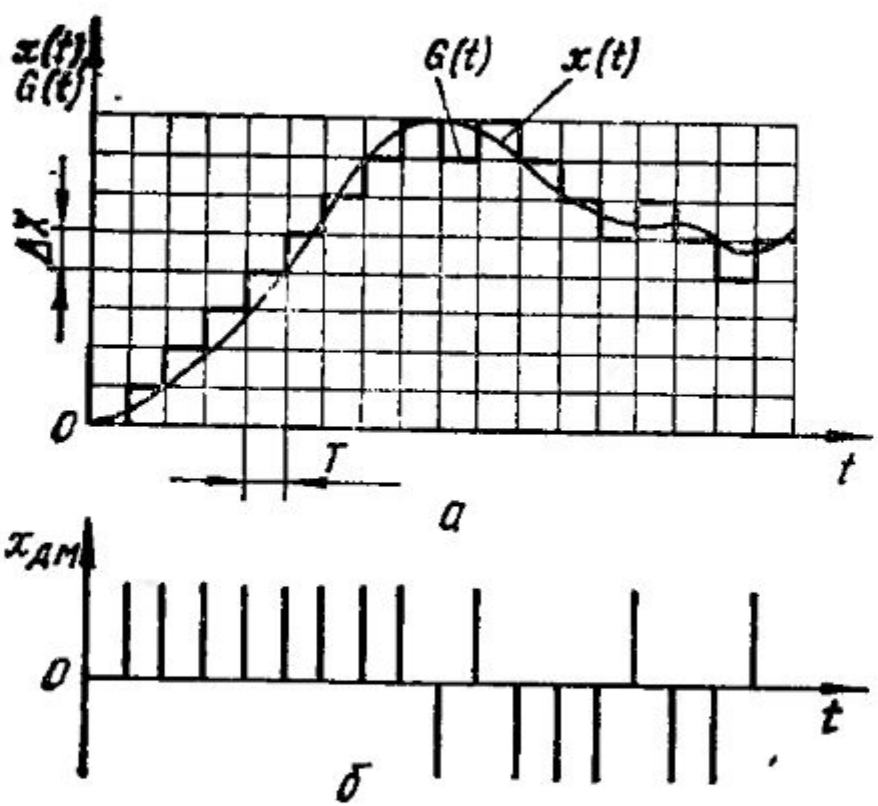


Цифровые электронные устройства (ЦЭУ) реализуют квантование исходного сигнала x(/) как по времени, так и по величине. Поэтому в фиксированные моменты времени такие сигналы только приближенно соответствуют значениям Очевидно, чем больше дискретных значений, которые может принимать сигнал, т. е. чем больше уровней дискретизации, тем точнее соответствует дискретный сигнал аналоговому. Однако в любом случае мы имеем дело **с** конечным числом его значений. Таким образом, в дискретном сигнале нарушена непрерывность представления информации как по величине, так и во времени.

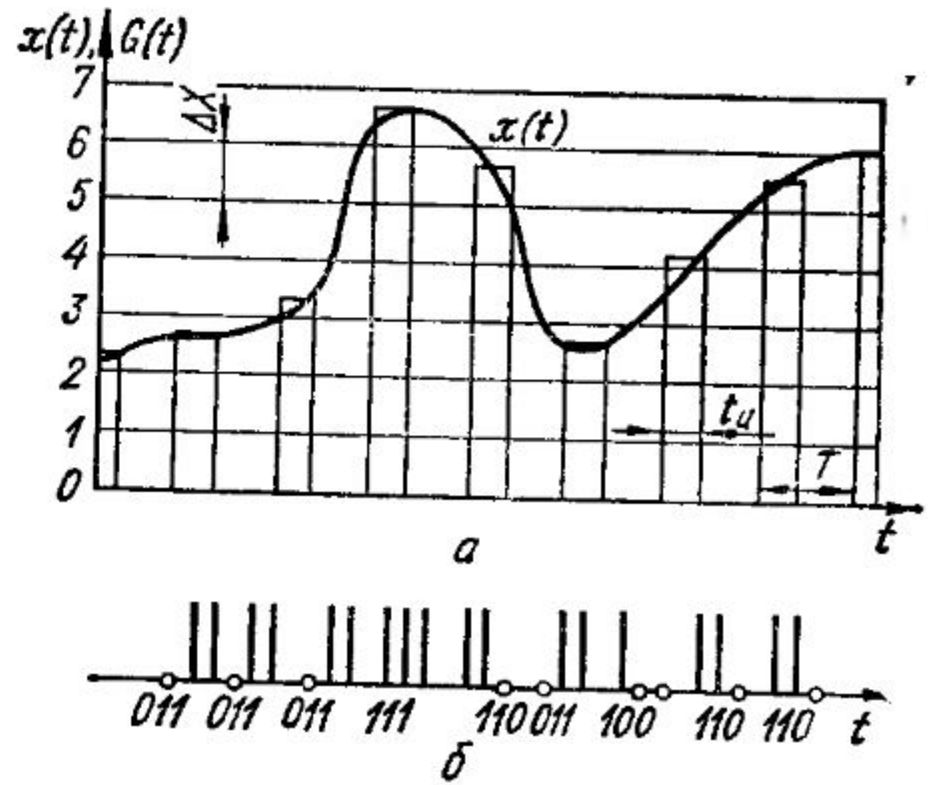
В свою очередь, конечному числу дискретных значений исходной физической величины можно поставить в соответствие некоторое число. Процесс замены дискретных уровней сигнала последовательностью чисел носит название кодирования, а совокупность полученных чисел называется кодом сигнала. Таким образом, процесс непосредственного преобразования и передачи сигналов можно заменить процессом преобразования и передачи кодов, поставленных в соответствие исходным сигналам.

Устройства, занимающиеся формированием, преобразованием и передачей кодов, поставленных в соответствие реальным значениям физических переменных, называют *цифровыми устройствами.* Передача кодов, каждый из которых, как правило, представляется некоторой последовательностью однотипных импульсов, требует некоторого времени. Очевидно, что это время больше времени, необходимого для передачи той же информации в импульсной и 1ем более непрерывной системах. Поэтому при прочих pamtux условиях количество информации, передаваемой цифровым способом, минимально.

Квантование исходного сигнала *x(t)* по времени и по уровню



Дельта-квантование



Импульсно-кодовое квантование

Релейные электронные устройства (РЭУ) реализуют квантование исходного сигнала *x(t)* по уровню и Преобразуют его в ступенчатую функцию, высота каждой из ступенек которой пропорциональна некоторой наперед заданной величине/t (см. рис. 1.4,в). Изменение уровня сигнала происходит в произвольные моменты времени, определяемые только заданными уровнями *nh* и величиной *x(t),* Поэтому аналогично с ИЭУ в моменты формирования ступенек сигнал РЭУ точно отражает значение исходной *x(t)*. Следовательно, при дискретизации представления по величине в РЭУ сохраняется непрерывность отображения информации во времени.

Основная область применения РЭУ связана не с преобразованием информации, а с преобразованием энергии, т. е. с силовой электроникой По сравнению с ИЭУ они, как правило, проще (отсутствует импульсный модулятор) и обладают большим быстро действием.

Достоинства ЦЭУ: высокая помехоустойчивость; высокая на дежность; возможность длительного хранения информации без ее потери; экономическая эффективность, обусловленная высокой 1ехнологичностью и повторяемостью устройств; энергетическая эффективность, а также совместимость с интегральной технологией.

Недостатки ЦЭУ: малое быстродействие; малая точность.

Однако меньшее быстродействие цифровых устройств с лихвой окупается возможностью унификации самих цифровых элементов, что позволяет с помощью их большого количества успешно решать вопросы повышения точности и быстродействия ЦЭУ.

Минимально возможный объем, который может занимать ЭУ, к конечном счете определяется количеством теплоты, выделяемой в этом объеме. Поэтому использование дискретных ^методов обработки информации позволяет реализовать ДЭУ в значительно меньшем объеме, чем в случае аналоговой информации.

Ранее мы отметили, что способность реализации сложных алгоритмов обработки информации в минимальных объемах с минимальными затратами и высокой надежностью работы является основной причиной повсеместного использования электронных >стройств Сказанному в полной мере отвечают цифровые электронные устройства, которые, несмотря на меньшие быстродействие и точность по сравнению с другими рассмотренными типами ЭУ, получают в настоящее время все большее распространение.

***Цифровыми* наз. устройства формирования, преобразования и передачи кодовых слов.**

Кодом наз. систему символов представления информации, удобную для обработки, хранения и передачи (число в десятичной или двоичной системе счисления).

В цифровой технике для записи кодовых символов, или просто кода, используют две цифры: 0 и 1 (сигналы с двумя уровнями напряжения: высоким и низким).

Современные устройства цифровой обработки информации используют: числа и логические переменные.

Числа - количественные характеристике процесса, объекта, системы, над ними можно производить арифметические действия.

Логические переменные определяют состояние системы или принадлежность её к определённому классу состояний.

Цифровые методы передачи информации по сравнению с другими имеют ряд преимуществ. Главными из них являются следующие:
1) приём сигнала сводится не к измерению, а к обнаружению 1 или 0;
2) сообщения в цифровой форме легко обрабатываются, запоминаются, коммутируются и регистрируются;
3) возможна многократная передача без накопления ошибок;
4) применение помехоустойчивого кодирования позволяет значительно увеличить достоверность передачи телемеханических сообщений;
5) упрощаются требования, предъявляемые к радиолиниям в отношении калибровки эталонных уровней;
6) улучшается использование канала связи в случае применения специальных кодов, статистически согласованных с передаваемыми сообщениями.

Под кодированием в широком смысле понимается переход от одного способа задания информации к другому, допускающий восстановление исходной информации. Теория кодирования получила большое развитие, начиная с 40-х годов XX века после работ К.Шеннона.

В данном конспекте большое внимание уделено теоретическим основам построения кодовых комбинаций, а также преобразованию кода передаваемой и обрабатываемой информации с сохранением его числового эквивалента. Преобразование может осуществляться программным или аппаратным способом.

Целями кодирования сообщений обычно являются:
1) передача по общему каналу связи нескольких или многих сообщений для кодового разделения сигналов;
2) повышение помехоустойчивости и достоверности передачи сообщений;
3) более экономное использование полосы частот канала связи, т.е. уменьшение избыточности;
4) уменьшение стоимости передачи и хранения сообщений;
5) обеспечение скрытности передачи и хранения информации;
6) преобразование любой информации независимо от ее происхождения и назначения в единую систему символов;
7) приведение исходных символов в соответствие с характеристиками канала связи.

Существующие системы счисления подразделяются на ***позици онные и непозиционные.*** В непозиционных системах значение кон кретной цифры постоянно у не зависит от ее расположения в записи числа.

$$X_q = x_{n-1}q^{n-1} + x_{n-2}q^{n-2} + \dots + x_0q^0 + x_{-1}q^{-1} + \dots + x_{-m}q^{-m}, \quad (14.1)$$

где x_i — разрядный коэффициент ($x_i = 0 \dots q-1$); q^i — весовой коэффициент.

Число q наз. основанием системы счисления, может быть как целым, так и дробным. Если в выражении (14.1) отбросить весовые коэффициенты q' и соответствующие знаки сложения, то получим сокращенную запись числа, носящую название **q-ичного кода** числа X,,. Номер позиции цифры x_i называют его разрядом. Разряды с положительными степенями q образуют целую часть числа Xq, с отрицательными степенями — дробную. Цифры x_{n-1} и x_{-m} соответственно являются старшим и младшим разрядами числа.

$$F(10) = \sum_{i=0}^{n-1} A_i 10^i \qquad F(2) = \sum_{i=0}^{n-1} A_i 2^i$$

Количество различных чисел, которое может быть записано в позиционной системе счисления с основанием q при заданном числе разрядов: $N = q^{n+m}$

Количество разрядов, необходимое для записи в позиционной системе счисления с основанием q некоторого числа X, можно определить из следующих соображений. Для записи числа X в системе с основанием q должно выполняться условие $X_q \leqslant q^{n+m} - 1$

Тогда

$$n + m \geqslant \log_q (X_q + 1)$$

В цифровой технике нашли применение только позиционные системы счисления.

$$F(10) = \sum_{i=0}^{n-1} A_i 10^i \qquad F(2) = \sum_{i=0}^{n-1} A_i 2^i$$

Двоичная система счисления, как и десятичная, относится к позиционным системам и является системой с основанием 2.

В десятичной системе число A, имеющее n-разрядную целую часть и m-разрядную дробную часть, представляется суммой

$$A = a_{n-1}\cdot 10^{n-1} + a_{n-2}\cdot 10^{n-2} + \ldots + a_i 10^i + \ldots + a_0 10^0 +$$
$$+ a_{-1}\cdot 10^{-1} + a_{-2} 10^{-2} + \ldots + a_{-m} 10^{-m},$$

где a_i — десятичная цифра от 0 до 9, а основанием системы счисления является число 10.

Например, число 236,75 в десятичной системе в соответствии с этим уравнением можно записать

$$236,75 = 2\cdot 10^2 + 3\cdot 10^1 + 6\cdot 10^0 + 7\cdot 10^{-1} + 5\cdot 10^{-2}.$$

Аналогично в двоичной системе число B можно представить в виде суммы

$$B = b_{n-1}\cdot 2^{n-1} + b_{n-2}\cdot 2^{n-2} + \ldots + b_i 2^i + \ldots + b_0\cdot 2^0 +$$
$$+ b_{-1}\cdot 2^{-1} + b_{-2} 2^{-2} + \ldots + b_{-m}\cdot 2^{-m},$$

где b — двоичные цифры 0 и 1, а основанием системы счисления является число 2 (в десятичном виде).

Например, то же число 236,75 в двоичном коде запишется $236,75 = 1\cdot 2^7 + 1\cdot 2^6 + 1\cdot 2^5 + 0\cdot 2^4 + 1\cdot 2^3 + 1\cdot 2^2 + 0\cdot 2^1 + 0\cdot 2^0 +$
$+ 1\cdot 2^{-1} + 1\cdot 2^{-2}$. <u>1110110011</u>

***Число символов в кодовом слове цифрового устройства фиксировано**, т.е. кодовые слова имеют одинаковую длину.*

Если кодовое слово имеет n символов (разрядов), то из них можно составить N = 2^n комбинаций кодовых слов. Например, в 32-разрядном вычислительном устройстве можно закодировать 2^{32} = 4 296 967 298 слов.

Для оценки количества цифровой информации используют *бит* и *байт* (1 байт = 8 бит).

Функционирование цифровых устройств можно представить следующим образом:

- посредством генератора тактовых импульсов производится **синхронизация начала** выполнения отдельных операций преобразования входного кодового слова и отводится **время выполнения** команды (в течение одного или нескольких периодов тактовых импульсов);
- после активизации начала операции осуществляется **преобразование** всех входных кодовых слов (логических нулей и единиц) в требуемые выходные кодовые слова;
- выходные кодовые слова отправляются **на хранение** в память цифрового устройства и/или во внешние устройства **для выполнения** определённых действий.

Переход от системы счисления с большим основанием к системе счисления с меньшим основанием выполняется с соблюдением следующих правил:

а) целая часть исходного числа делится на основание новой системы счисления;

б) дробная часть исходного числа умножается на основание новой системы счисления. <u>Преобразуем число 25,12 в двоично-десятичную систему</u>

Решение. 1. Преобразуем целую часть:

$$25:2=12+1\ (X_0=1)$$

$$12:2=6+0\ (X_1=0)$$

$$6:2=3+0\ (X_2=0)$$

$$3:2=1+1\ (X_3=1)$$

$$1:2=0+1\ (X_4=1)$$

Запись целой части двоичного числа X_2 производится с последнего результата деления, т. е. $25_{10}=11001_2$.

2. Преобразуем пробную часть:

$$0{,}12\cdot 2=0+0{,}24\ (X_{-1}=0)$$

$$0{,}24\cdot 2=0+0{,}48\ (X_{-2}=0)$$

$$0{,}48\cdot 2=0+0{,}96\ (X_{-3}=0)$$

$$0{,}96\cdot 2=1+0{,}92\ (X_{-4}=1)$$

$$0{,}92\cdot 2=1+0{,}84\ (X_{-5}=1)$$

Запись дробной части двоичного числа производится с первого результат умножения, т. е. $0{,}12_{10}=0{,}0001_2$.

В качестве математического аппарата для функций и аргументов, принимающих только два значения — 0 и 1, используется двоичная (булева) алгебра — алгебра логики.
Логическими (булевыми, двоичными) переменными (аргументами, высказываниями) в двоичной алгебре называются величины, которые независимо от их конкретной физической сущности могут принимать только два значения — 0 и 1.

Десятичное число	Двоичное число	Десятичное число	Двоичное число
0	0	8	1000
1	1	9	1001
2	10	10	1010
3	11	11	1011
4	100	12	1100
5	101	13	1101
6	110	14	1110
7	111	15	1111

Арифметические устройства (сумматоры, умножители) предназначены для выполнения арифметических операций над бинарными кодовыми словами. Числа (кодовые слова) в цифровых устройствах обычно представляют в позиционной двоичной системе счисления, осуществляемой по следующему правилу:

$$A=(a_1a_2...a_n)=a_1\cdot 2^{n-1}+a_2\cdot 2^{n-2}+...+a_n\cdot 2^0, \quad (5.1)$$

где $a_1, a_2, \ldots, a_n$ — весовые коэффициенты, принимающие значения 1 и 0; n — число разрядов в коде. Например, $26_{(10)} = 11010_{(2)}$, $n = 5$.

Натуральный ряд чисел в различных системах счисления

Десятичная	Шестнадцатеричная	Восьмеричная	Двоичная	Десятичная	Шестнадцатеричная	Восьмеричная	Двоичная
1	2	3	4	1	2	3	4
0	0	0	0	11	*B*	13	1011
1	1	1	1	12	*C*	14	1100
2	2	2	10	13	*D*	15	1101
3	3	3	11	14	*E*	16	1110
4	4	4	100	15	*F*	17	1111
5	5	5	101	16	10	20	10000
6	6	6	110	17	11	21	10001
7	7	7	111	18	12	22	10010
8	8	10	1000	19	13	23	10011
9	9	11	1001	20	14	24	10100
10	A	12	1010	21	15	25	10101

Десятичный код	Двоичный код $2^3\ 2^2\ 2^1\ 2^0$	Код Грея
0	0 0 0 0	0 0 0 0
1	0 0 0 1	0 0 0 1
2	0 0 1 0	0 0 1 1
3	0 0 1 1	0 0 1 0
4	0 1 0 0	0 1 1 0
5	0 1 0 1	0 1 1 1
6	0 1 1 0	0 1 0 1
7	0 1 1 1	0 1 0 0
8	1 0 0 0	1 1 0 0
9	1 0 0 1	1 1 0 1
10	1 0 1 0	1 1 1 1
11	1 0 1 1	1 1 1 0
12	1 1 0 0	1 0 1 0
13	1 1 0 1	1 0 1 1
14	1 1 1 0	1 0 0 1
15	1 1 1 1	1 0 0 0

$$q_i = 2^{i-1}$$

8-4-2-1

$$q_i = 2^i - 1$$

1, 3, 7, 15, 31,….

Запись кодовых комбинаций десятичных чисел от 0 до 15 различными кодами

Десятичный	8-4-2-1 на все сочетания	2-4-2-1 (Айкена)	4-2-2-1	5-1-2-1	Код Грея 15-7-3-1	Джонсона	Единично-десятичный неравномерный
1	2	3	4	5	6	7	8
0	0000	0000	0000	0000	0000	00000	25 →11 11111
1	0001	0001	0001	0001	0001	00001	14 → 1 1111
2	0010	0010	0010	0010	0011	00011	Единично-десятичный равномерный
3	0011	0011	0101	0011	0010	00111	
4	0100	0100	0110	0111	0110	01111	
5	0101	1011	1001	1000	0111	11111	8а
6	0110	1100	1010	1001	0101	11110	
7	0111	1101	1101	1010	0100	11100	25 →0000000110000011111
8	1000	1110	1110	1011	1100	11000	14 →0000000010000001111
9	1001	1111	1111	1111	1101	10000	Унитарный 12 - разрядный
10	1010	10000	10000	10000	1111	100000	
11	1011	10001	10001	10001	1110	100001	
12	1100	10010	10010	10010	1010	100011	8б
13	1101	10011	10101	10011	1011	100111	12 →111111111111
14	1110	10100	10110	10111	1001	101111	11 →011111111111
15	1111	11011	11001	11000	1000	111111	8 →000011111111

Самые современные и мощные микропроцессоры (компьютеры) из перечня арифметических операций способны выполнять только операцию сложения, то есть все их действия сводятся к суммированию.

Основной арифметической операцией, которая используется в цифровой технике, является **сложение** двоичных чисел, а к нему приводятся другие — вычитание, умножение, деление.

Двоичные числа складываются так же, как и десятичные: $0_2 + 0_2 = 0_2$; $0_2 + 1_2 = 1_2$; $1_2 + 0_2 = 1_2$; $1_2 + 1_2 = 10_2$. Для «удобства» ЦВМ, в последнем случае, записывается 0 от 10, а 1 оставляется в «уме машины» для *переноса* в первый разряд. Последнее сложение записывается и читается так: «1 + 1 = 0 плюс *перенос* 1». При сложении многоразрядных чисел эта перенесённая единица находит своё место. **Вычитание**

Положим, что из 1010_2 надо вычесть 0111_2, что равносильно $10_{10} — 7_{10} = 3_{10}$).

Алгоритм вычисления таков: сначала двоичное вычитаемое число *прямого* кода $[A]_n = 0111_2$ записывается в форме *обратного* кода $[A]_д = 1000_2$ (в обратном коде все 1 прямого кода заменяются на 0, а 0 — на 1). Результат обратного кода складывается с уменьшаемым, то есть $1010_2 + [A]_д = 1010_2 + 1000_2 = \boldsymbol{\mathit{1}}0010_2$ и получают промежуточное число $\boldsymbol{\mathit{1}}0010_2$. После этого производится перенос ***1*** из высшего разряда (отмечен жирным курсивом) промежуточного числа, и она складывается с содержимым младшего разряда, то есть $0010_2 + \boldsymbol{\mathit{1}}_2 = 0011_2$.

Заметив, что произведённый перенос *1* называется *циклическим переносом,* резюмируем, что полученное число 0011_2, равное 3_{10}, и есть искомый результат вычитания. Изложенный алгоритм вычитания не удобен для человека, однако, он «удобен» для ЦВМ.

Операции умножения и деления также приводятся к сложению

Операции над кодовыми словами, представленными в виде электрических сигналов, **в цифровом устройстве могут выполняться следующими двумя способами:**

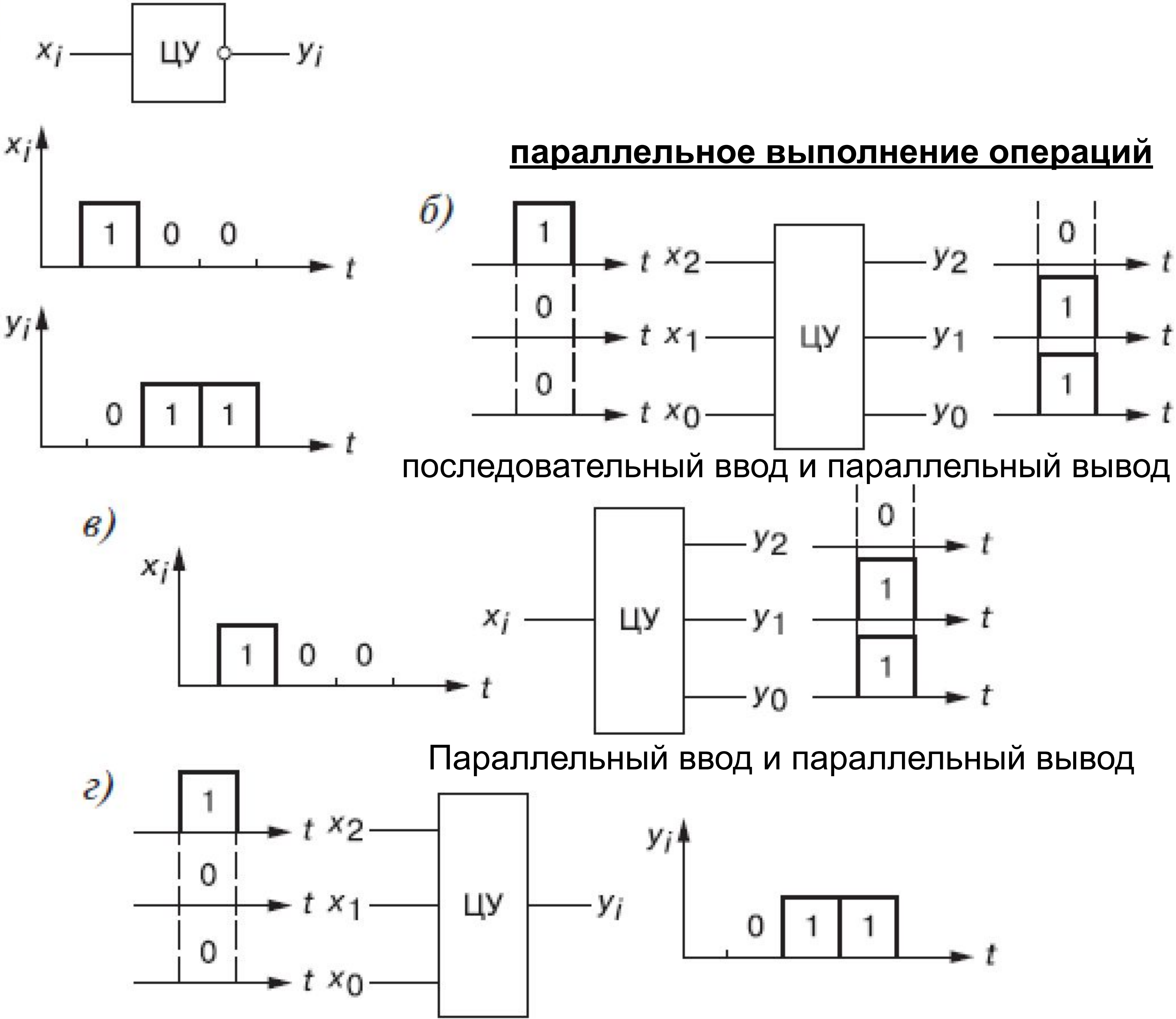
- **последовательное** (поразрядное, побитовое) выполнение операций, при котором символы 1 и 0 кодового слова поступают последовательно по времени на единственный вход цифрового устройства и по завершении операции последовательно символ за символом выводятся из него. На рис. 5.1, *а* показано выполнение операции цифровым устройством ЦУ (инвертором) над трехразрядным входным словом $x_2x_1x_0$ = 100, при котором биты выходного слова $y_2y_1y_0$ = 011 принимают противоположные значения;
- **параллельное выполнение операций**, при котором символы 1 и 0 кодового слова поступают одновременно на три входа ЦУ и по завершении операции одновременно выводятся из него (рис. 5.1, *б*).

В ряде случаев используют комбинированные способы обработки информации: с последовательным вводом и параллельным выводом (рис. 5.1, *в*) и с параллельным вводом и последовательным выводом (рис. 5.1, *г*) последовательное (поразрядное, побитовое) выполнение операций

параллельное выполнение операций

последовательный ввод и параллельный вывод

Параллельный ввод и параллельный вывод



ОСНОВНЫЕ ПОЛОЖЕНИЯ АЛГЕБРЫ ЛОГИКИ

В отличие от аналоговых электронных устройств, в цифровых устройствах (ЦУ) входные и выходные сигналы могут принимать ограниченное количество состояний.

Работа любого логического устройства подчиняется законам формальной логики. Для описания алгоритмов работы цифровых устройств необходим соответствующий математический аппарат. Такой аппарат для решения задач формальной логики в середине прошлого века разработал ирландский математик Д. Буль.

Булева алгебра (алгебра логики) —математическая система, оперирующая двумя понятиями: событие истинно и событие ложно. Естественно ассоциировать эти понятия с цифрами, используемыми в двоичной системе счисления. Далее будем их называть соответственно логическими единицей (лог. 1) и нулем (лог. 0).

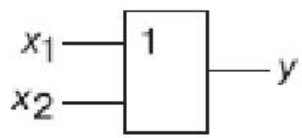
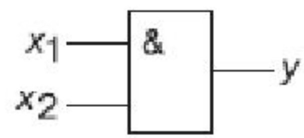
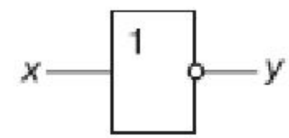
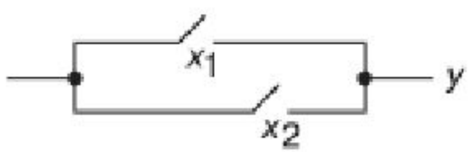
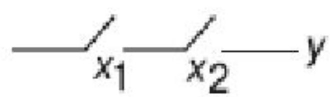
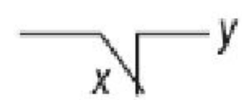
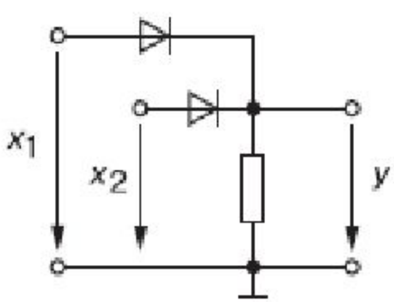
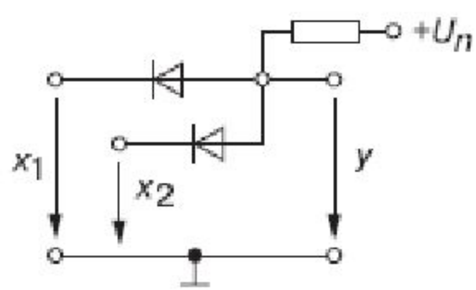
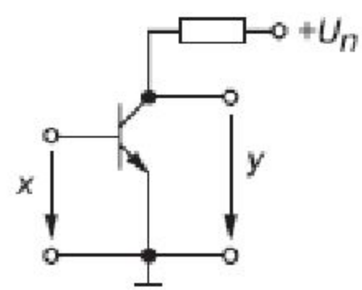
В соответствии с логическим соглашением (ГОСТ 2.743-82), в зависимости от конкретной физической реализации элементов ЦУ, более положительному значению физической величины, "Н" - уровень, соответствует состояние "логическая 1", а менее положительному значению ,"L - уровень" - "логический 0". Такое соглашение называется **положительной** логикой. Обратное соотношение называется отрицательной логикой. ГОСТ 19480 - 89 дает наименования, определения и условные обозначения основных параметров и характеристик цифровых микросхем.

Для логических переменных, принимающих только два значения, существуют 4 основных операции. Операция логическое "И" (AND) конъюнкция или логическое умножение (* или /\). Операция логическое "ИЛИ" (OR), дизъюнкция или логическое сложение (+ или \ . Операция логическое "НЕ" (NOT) инверсия или отрицание, обозначается чертой над логическим выражением или " ~ ". Операция эквивалентности - "=" .

Аксиомы алгебры логики

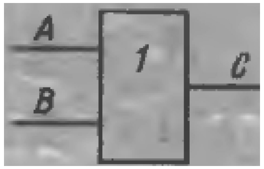
(1)	0 + 0 = 0		1 * 1 = 1	(1')
(2)	1 + 1 = 1		0 * 0 = 0	(2')
(3)	1 + 0 = 0 + 1 = 1		0 * 1 = 1 * 0 = 0	(3')
(4)	~1 = 0		~0 = 1	(4')

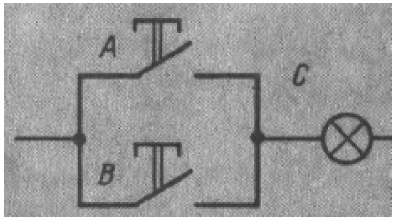
Формы отображения основных логических функций

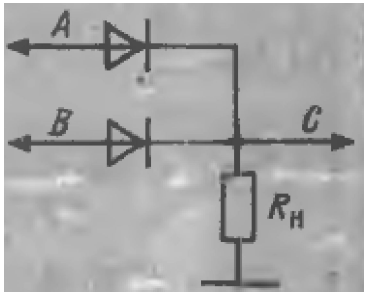
Наименование функции →	Дизъюнкция	Конъюнкция	Инверсия
Символическая	$\vee$ или $+$	$\wedge$ или $\cdot$	$\overline{x}$
Буквенная	ИЛИ	И	НЕ
Условная графическая			
Аналитическая	$y = x_1 \vee x_2 = x_1 + x_2$	$y = x_1 \wedge x_2 = x_1 x_2$	$y = \overline{x}$
Табличная (истинности)	x_1 x_2 y 0 0 0 0 1 1 1 0 1 1 1 1	x_1 x_2 y 0 1 0 0 1 0 1 0 0 1 1 1	x y 0 1 1 0
Контактная			
Схемо-техническая			

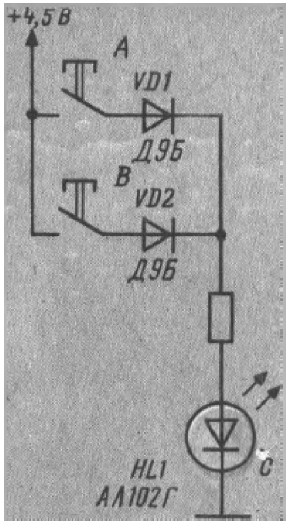
Логический элемент <u>**ИЛИ**</u> моделирует операцию **логического сложения**, или, как ее еще называют, операцию **дизъюнкции**. Алгебраически эта операция записывается следующим образом: $A+B=C$ или $A \lor B=C$. Буквами А и В обозначены простые высказывания, или двоичные переменные, буквой *С* — сложное высказывание, или переключательная функция. Последнее название показывает, что функция зависит от переключений переменных А и В. Если простые высказывания соединены союзом «или», то сложное высказывание истинно, если истинно хотя бы одно из простых высказываний. Соответственно, *С* должно равняться 1, если А или В равны 1 по отдельности или одновременно. Зависимость между двоичными переменными А и В и переключательной функцией *С* может быть задана в виде таблицы истинности, в ней написаны условия истинности сложного высказывания в зависимости от истинности простых высказываний.

A	B	C
0	0	0
0	1	1
1	0	1
1	1	1

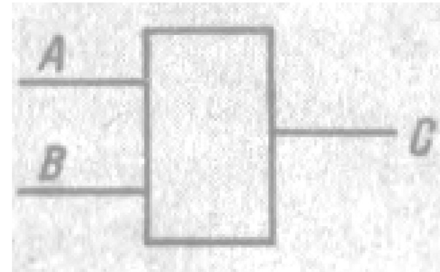




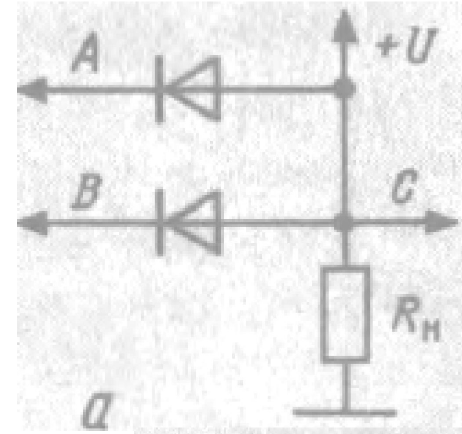


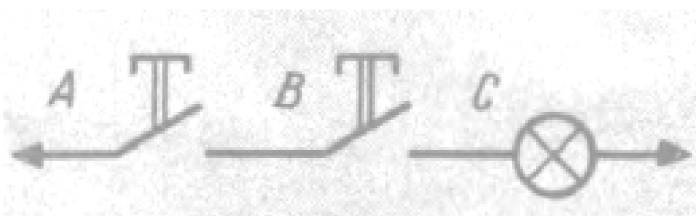


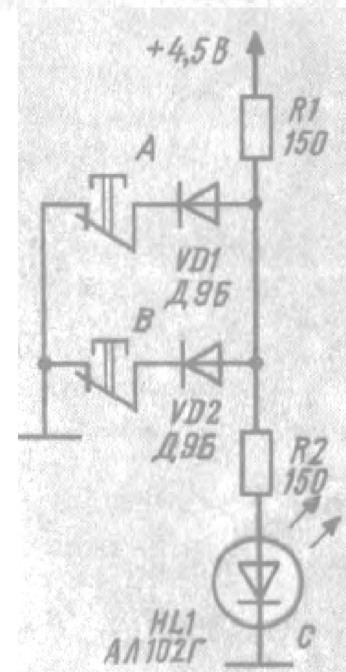
Логический элемент **И** выполняет операцию **логического умножения**, или **конъюнкции**. Алгебраически эта операция записывается следующим образом: $C=A*B$ или $C=A \wedge B$, при этом C= 1 только в том случае, если А и В одновременно равны 1. Эти правила можно записать в виде следующей таблицы:



A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

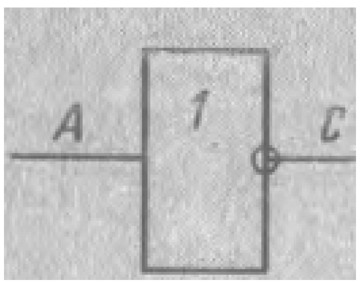




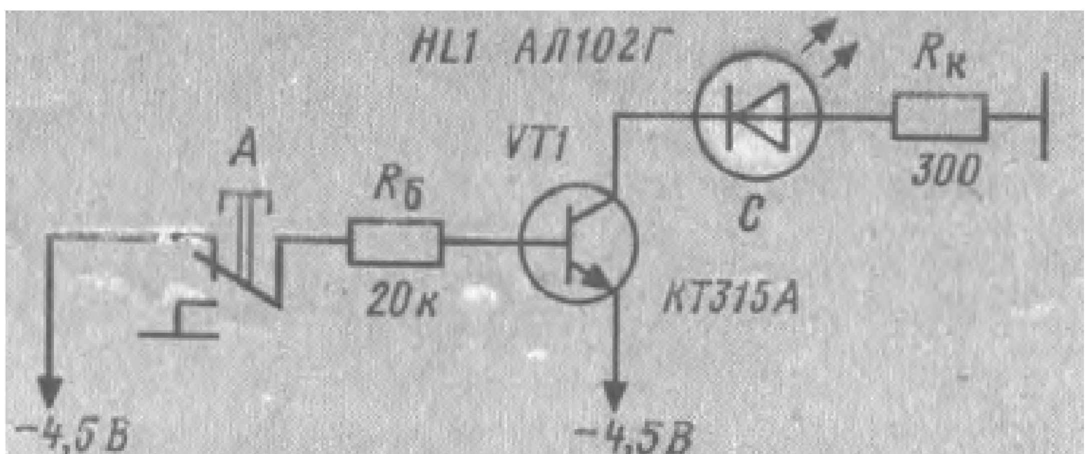


Сравнив таблицы истинности логических элементов И и ИЛИ, легко заметить, что из одной таблицы легко получить другую, если заменить единицы нулями и нули единицами.

Логический элемент **НЕ** выполняет операцию отрицания, или | **инверсии**, алгебраически она записывается следующим образом: I $C=A$, при этом на выходе будет сигнал 1, если на входе имеется I сигнал 0 и, наоборот, выходной сигнал равен 0 при входном сигна|ле 1. Работа элемента НЕ записывается в виде следующей таблицы:

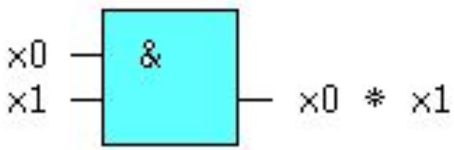


A	C
0	1
1	0

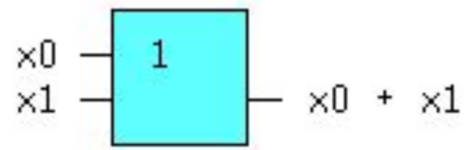


1.3 УСЛОВНЫЕ ОБОЗНАЧЕНИЯ ЛОГИЧЕСКИХ ФУНКЦИЙ НА СХЕМАХ

Функция F1 "И" (конъюнкция)

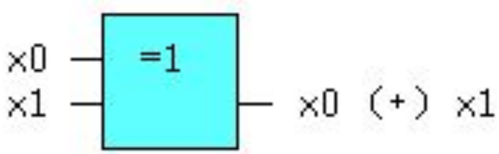


Функция F7 "ИЛИ" (дизъюнкция)

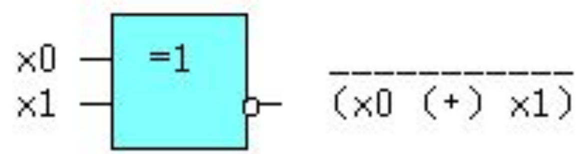


Функция F6 "ИСКЛЮЧАЮЩЕЕ ИЛИ", называемая также для двух аргументов ф-ей "НЕ-РАВНОЗНАЧНОСТИ" или "СУММА ПО МОДУЛЮ ДВА" , имеет следующее схемное обозначение.Логическая операция (+) называется суммой по модулю два.

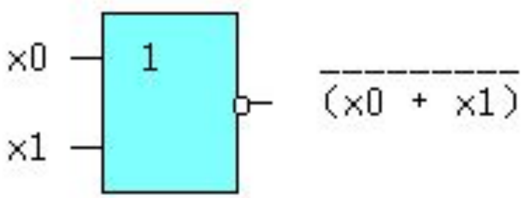
Функция F6 "ИСКЛЮЧАЮЩЕЕ ИЛИ



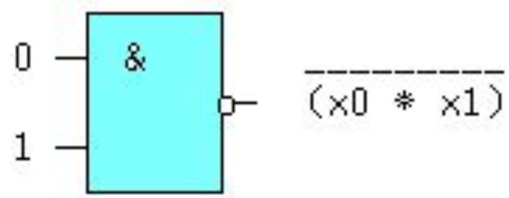
Функция F9 "ИСКЛЮЧАЮЩЕЕ ИЛИ - НЕ"



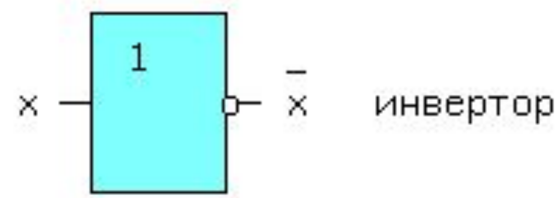
Функция F8 "ИЛИ - НЕ"



Функция F14 "И - НЕ" и



Функция F12 "НЕ" (инверсия)



Функция "И" равна единице, если равны единице ВСЕ ее аргументы. Функция "ИЛИ" равна единице, если равен единице ХОТЯ БЫ один аргумент. Функция "ИСКЛЮЧАЮЩЕЕ ИЛИ" (XOR) равна единице, если равен единице ТОЛЬКО один ее аргумент.

Есть мужчины (М) и женщины (Ж)
у МЖ и ЖМ могут быть дети,
у ММ и ЖЖ нет!
Для двух значений результатом исключающего или будет единица, если начальные значения разные, и ноль, если одинкаовые:

Элементарный набор логических функций

Входные переменные		Элементарные логические функции и их номера															
x_1	x_0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

В булевой алгебре особое место занимают функции двух переменных. Имея набор функций двух переменных, можно на основании принципа суперпозиции образовать переключательную функцию любого числа переменных

		Обозначение
0	Константа нуль	0
1	Конъюнкция, логическое умножение, произведение, совпадение, И	x_1x_0, $x_1\wedge x_0$, $x_1\&x_0$
2	Запрет по x_0, ЗАПРЕТ	$x_1\bar{x}_0$
3	Переменная x_1	x_1
4	Запрет по x_1, ЗАПРЕТ	$\bar{x}_1x_0$
5	Переменная x_0	x_0
6	Неравнозначность, сумма по модулю 2, исключающее ИЛИ	$x_1\oplus x_0$, $x_1+x_0 \pmod 2$
7	Дизъюнкция, логическое сложение, сумма, ИЛИ	x_1+x_0, $x_1\vee x_0$
8	Отрицание дизъюнкции, функция Пирса, ИЛИ — НЕ	$\overline{x_1+x_0}$, $x_1\overline{\vee}x_0$, $x_1\downarrow x_0$
9	Равнозначность, эквивалентность	$x_1\equiv x_0$
10	Отрицание x_0	$\bar{x}_0$
11	Импликация по x_0	$x_0\to x_1$, $x_1+\bar{x}_0$
12	Отрицание x_1	$\bar{x}_1$
13	Импликация по x_1	$x_1\to x_0$, $\bar{x}_1+x_0$
14	Отрицание конъюнкции, функция Шеффера, И — НЕ	$\overline{x_1x_0}$, $x_1\overline{\wedge}x_0$, x_1/x_0
15	Константа единица	1

№ функции	Название функции	Обозначение
0	Константа нуль	0
1	Конъюнкция, логическое умножение, произведение, совпадение, И	$x_1 x_0,\ x_1 \wedge x_0,\ x_1 \& x_0$
2	Запрет по x_0, ЗАПРЕТ	$x_1 \bar{x}_0$
3	Переменная x_1	x_1
4	Запрет по x_1, ЗАПРЕТ	$\bar{x}_1 x_0$
5	Переменная x_0	x_0
6	Неравнозначность, сумма по модулю 2, исключающее ИЛИ	$x_1 \oplus x_0$, $x_1 + x_0 (\mathrm{mod}\ 2)$
7	Дизъюнкция, логическое сложение, сумма, ИЛИ	$x_1 + x_0,\ x_1 \vee x_0$
8	Отрицание дизъюнкции, функция Пирса, ИЛИ — НЕ	$\overline{x_1 + x_0},\ x_1 \overline{\vee} x_0$, $x_1 \downarrow x_0$
9	Равнозначность, эквивалентность	$x_1 \equiv x_0$
10	Отрицание x_0	$\bar{x}_0$
11	Импликация по x_0	$x_0 \to x_1,\ x_1 + \bar{x}_0$
12	Отрицание x_1	$\bar{x}_1$
13	Импликация по x_1	$x_1 \to x_0,\ \bar{x}_1 + x_0$
14	Отрицание конъюнкции, функция Шеффера, И — НЕ	$\overline{x_1 x_0},\ x_1 \overline{\wedge} x_0$, x_1 / x_0
15	Константа единица	1

1. $x + 0 = x$;

2. $x \cdot 0 = 0$;

3. $x + 1 = 1$;

4. $x \cdot 1 = 1$;

5. $x \cdot x = x$;

6. $x + x = x$;

7. $x + \bar{x} = 1$;

8. $x \cdot \bar{x} = 0$;

9. $\bar{\bar{x}} = x$;

10. $x_1 + x_2 = x_2 + x_1$;

11. $x_1 x_2 = x_2 \cdot x_1$;

12. $(x_1 + x_2) + x_3 = x_1 + (x_2 + x_3)$;

13. $(x_1 x_2) x_3 = x_1 (x_2 x_3)$;

14. $x_1 (x_2 + x_3) = x_1 x_2 + x_1 x_3$;

15. $x_1 + x_2 x_3 = (x_1 + x_2)(x_1 + x_3)$;

16. $x_1 x_2 + x_1 \bar{x}_2 = x_1$; $(x_1 + x_2)(x_1 + \bar{x}_2) = x_1$;

17. $x_1 + x_1 x_2 = x_1$; $x_1 (x_1 + x_2) = x_1$;

18. $\overline{x_1 + x_2} = \bar{x}_1 \bar{x}_2$; $\overline{x_1 x_2} = \bar{x}_1 + \bar{x}_2$.

Ф-лы 1-9 представляют собой *тождества*, в справедливости которых легко убедиться прямой подстановкой 0 и $x = 1$. Соотношения 10 и 11 иллюстрируют *переместительный*, а 12 и 13 — *сочетательный* законы.

Соотношения 16 называют *правилом склеивания*, а соотношения 17 — *правилом поглощения*.

В преобразованиях логических выражений важную роль играют формулы 14-18. Формулы де Моргана (18), как отмечалось, используют для того, чтобы перейти от логического произведения к логической сумме и обратно.

Доказательство истинности приведенных законов получают путем подстановки всех комбинаций переменных x_t (причем левая и правая части уравнений должны быть тождественны) или путем алгебраических преобразований на основе тех же законов.

Например, для ***правила склеивания***

$$x_1 + x_1 x_2 = x_1(1 + x_2) = x_1 \cdot 1 = x_1.$$

универсальным логическим операциям (устройствам) относят две разновидности базовых элементов:

функцию Пирса, $y = x_1 \downarrow x_2 = \overline{x_1 + x_2};$

обозначаемую символически вертикальной стрелкой (стрел ка Пирса) и отображающую операцию **ИЛИ-НЕ**. Этой операции соответствует столбец y_8 в таблице 5.4. Для простейшей функции двух переменных x_1 и x_2 функция y = 1 тогда и только тогда, когда $x_1 = x_2 = 0$:

функцию Шеффера, обозначаемую символически вертикальной черточкой $y = x_1 | x_2 = \overline{x_1 x_2}.$

(штрих Шеффера) и отображающую операцию **И-НЕ.** Этой операции соответствует столбец y_{12} в таблице 5.4. Для простейшей функции двух переменны x_1 и x_2 функция y = 0 тогда и только тогда, когда $x_1 = x_2 = 1$:

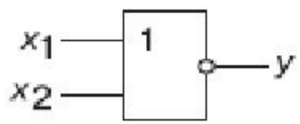
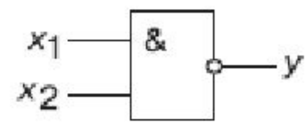
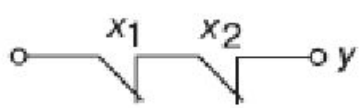
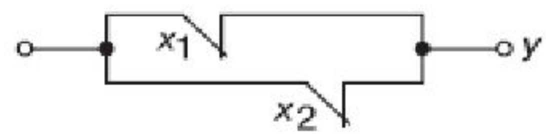
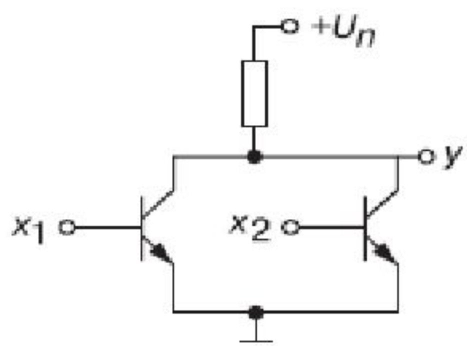
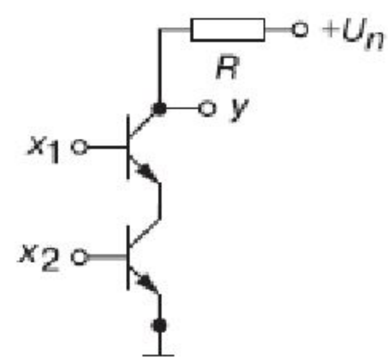
Сопоставляя таблицы истинности для операций ИЛИ и И, можно обосновать некоторые соотношения булевой алгебры, имеющие большое практическое значение. Например, **принцип дуальности булевой алгебры записывается в виде двух следующих положений**:

- если $x_1 + x_2 = y$, то $\bar{x}_1 + \bar{x}_2 = \bar{y}$;
- если $x_1 x_2 = y$, то $\bar{x}_1 \bar{x}_2 = \bar{y}$.

Из этих соотношений вытекает **теорема (правило) де Моргана*****: инверсия выражения может быть представлена тем же выражением без инверсии с изменением всех знаков конъюнкции на знаки дизъюнкции, знаков дизъюнкции на знаки конъюнкции и инверсией всех аргументов***, т. е.

$$\overline{x_1 + x_2} = \bar{x}_1 \bar{x}_2; \quad \overline{x_1 x_2} = \bar{x}_1 + \bar{x}_2$$

Формы отображения базовых логических функций

Наименование функции →	Функция Пирса	Функция Шеффера
Символическая	↓	|
Буквенная	ИЛИ-НЕ	И-НЕ
Условная графическая		
Аналитическая	$y = x_1 \downarrow x_2$	$y = x_1 \mid x_2$
Табличная (истинности)	x_1 x_2 y 0 0 1 0 1 0 1 0 0 1 1 0	x_1 x_2 y 0 1 1 0 1 1 1 0 1 1 1 0
Контактная		
Схемо-техническая		

2И—НЕ (штрих Шеффера)	$x_1 \mid x_0$	$\overline{x_1 x_0}$; $\overline{x_1}+\overline{x_0}$; $\overline{x_1 \wedge x_0}$; $\overline{x_1} \vee \overline{x_0}$
2ИЛИ—НЕ (стрелка Пирса)	$x_1 \downarrow x_0$	$\overline{x_1 + x_0}$; $\overline{x_1}\,\overline{x_0}$; $\overline{x_1 \vee x_0}$; $\overline{x_1} \wedge \overline{x_0}$

Булева функция, зависящая от n аргументов, называется «n-местной и является ***полностью определенной***, если указаны ее значения для всех двоичных наборов значений ее аргументов. Число таких наборов зависит от числа переменных n и равно 2^n.

Если булева функция определена не на всех наборах, то она является неполностью определенной или ***недоопределенной***.

Некоторые наборы двоичных переменных физически не могут быть реализованы в проектируемых устройствах в силу накладываемых проектировщиком ограничений. Например, при проектировании устройства фиксации прохождения детали, которое срабатывает от двух датчиков положения, каждый из которых кодируется одной переменной x_t, при условии, что расстояние между ними больше длины детали, исключено одновременное срабатывание двух датчиков. Тогда значение функции $F(x_1 x_2)$ при $x_1 = 1$, x_2 — 1 называется ***безразличным (неопределенным, факультативным)*** и обозначается знаком -. *При формализации булевых функций, содержащих безразличные состояния, их все же доопределяют значениями 0 или 1. Поскольку физически в реальном устройстве такие состояния не реализуются, значение доопределенной функции зависит только от проектировщика и поставленного критерия, например максимальной надежности разрабатываемого устройства, его простоты.*

Булевые (переключательные) функции бывают комбинационными и временными.

Комбинационными наз. функции, значение которых однозначно определяется значениями их аргументов. Комбинационные функции иногда называют ***функциями без памяти***, подчеркивая отсутствие в них свойства запоминания информации. Это означает, что после того, как изменение аргументов прекращается, тот факт, что они имели другое, чем в данный момент, значение уже не может влиять на формирование значения переключательной функции. Комбинационная функция «забывает» старые аргументы и может реагировать только на значения новых.

Схемы, реализующие комбинационные функции, называются комбинационными (КС).

Временными (функциями с памятью) наз. функции, значения которых определяются как значениями аргументов в данный момент времени, так и другими параметрами, прежде всего временем, поэтому при одних и тех же значениях аргументов значение временной функции может быть разным.

$$F = f(x_1, x_2, \ldots, x_n, t).$$

Временные функции делят на:

-- ***временные булевы функции*** (ВБФ) типа

Значение этой функции при одних и тех же значениях аргументов зависит от момента времени, т. е. в различные моменты времени реализуются различные комбинационные булевы функции;

-- ***рекуррентные булевы функции первого рода (РБФ-1)*** типа

$$F = f(x_{1t}, x_{2t}, \ldots, x_{nt}, F_{t-1}, F_{t-2}, \ldots, F_{t-r}).$$

--рекуррентные булевы функции второго рода (РБФ-2) типа

$$F = f[x_{1t}, x_{2t}, \ldots, x_{nt}, x_{1(t-1)}, x_{2(t-1)}, \ldots, x_{n(t-1)}; x_{1(t-k)}, x_{2(t-k)}, \ldots, x_{n(t-k)}].$$

Рассмотрим некоторое логическое устройство, на входе которого присутствует некоторый *n*-разрядный двоичный код $x_{n-1} \ldots x_1 x_0$, а на выходе соответственно **m**-разрядный двоичный код $z_{n-1} \ldots z_1 z_0$ (рис.). Для того чтобы описать поведение этой схемы, необходимо определить зависимость каждой из выходных переменных от входного двоичного кода

Зависимость выходных переменных, выраженная через совокупность входных переменных с помощью операций алгебры логики, носит название функции алгебры логики (ФАЛ) или переключательной функцией. Задать ФАЛ это значит определить значения выходных переменных для всех возможных комбинаций входных. Очевидно, что для n-разрядного двоичного кода i … существует 2^N различных значений.

Функция называется полностью определенной, если заданы 2^N ее значений. Если часть значений функции не задана, то она называется частично определенной или недоопределенной.

Иногда известно, что по условиям работы устройства появление некоторых входных кодов невозможно, и поэтому значения ФАЛ на этих кодах не задаются. При этом возникают так называемые *факультативные* или необязательные значения функции, которые могут задаваться произвольными. Входные коды, для которых ФАЛ имеет факультативные значения, наз. *запрещенными*.

Устройства, поведение которых описывается при помощи ФАЛ, называют *логическими*.

Обобщенная схема логического устройства

Способы записи ФАЛ

- Словестное представление;
- Таблица истинности
- Алгебраическое выражение

№	a	b	c	y
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1

Дизъюнктивная нормальная форма (ДНФ) —логическая сумма элементарных логических произведений (дизъюнкция элементарных конъюнкций), в каждое из которых аргумент или его отрицание входит не более одного раза. Математическое выражение логической функции в совершенной ДНФ (СДНФ) получают из таблицы истинности: для каждого набора аргументов, на котором функция равна 1, записывают элементарные произведения переменных, причем переменные, значения которых равны нулю, записывают с инверсией. Полученные произведения, наз. конституентами единицы или минтермами, суммируют.

$$y(a,b,c) = \overline{a}bc + a\overline{b}c + ab\overline{c} + abc$$

сокращение записи СДНФ $y(a,b,c) = \sum(3,5,6,7)$

<u>Конъюнкция, включающая в себя полный набор переменных, на котором функция равна единице, наз. конституентой единицы (минтермом), а запись функции в виде суммы конституент единицы (т. е. дизъюнкция конституент единицы) называется совершенной дизъюнктивной нормальной формой (СДНФ).</u>

x_3x_2 \ x_1x_0	00	01	10	11
00	0	1	0	0
01	1	1	1	0
10	1	0	0	0
11	0	1	1	0

•**Совершенной конъюнктивной нормальной формой (СКНФ) называют логическое произведение элементарных сумм (конъюнкция элементарных дизъюнкций), в каждую из которых аргумент или его отрицание входят один раз. Для каждого набора аргументов таблицы истинности, на котором функция *y* равна 0, составляют элементарную сумму, причем переменные, значение которых равно 1, записывают с отрицанием. Полученные суммы, называемые *конституентами нуля* или *макстермами*, объединяют операцией логического умножения.**

$$y(a,b,c)=(a+b+c)(a+b+\overline{c})(a+b+c)(\overline{a}+b+c)$$

№	a	b	c	y
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1

сокращения записи СКНФ

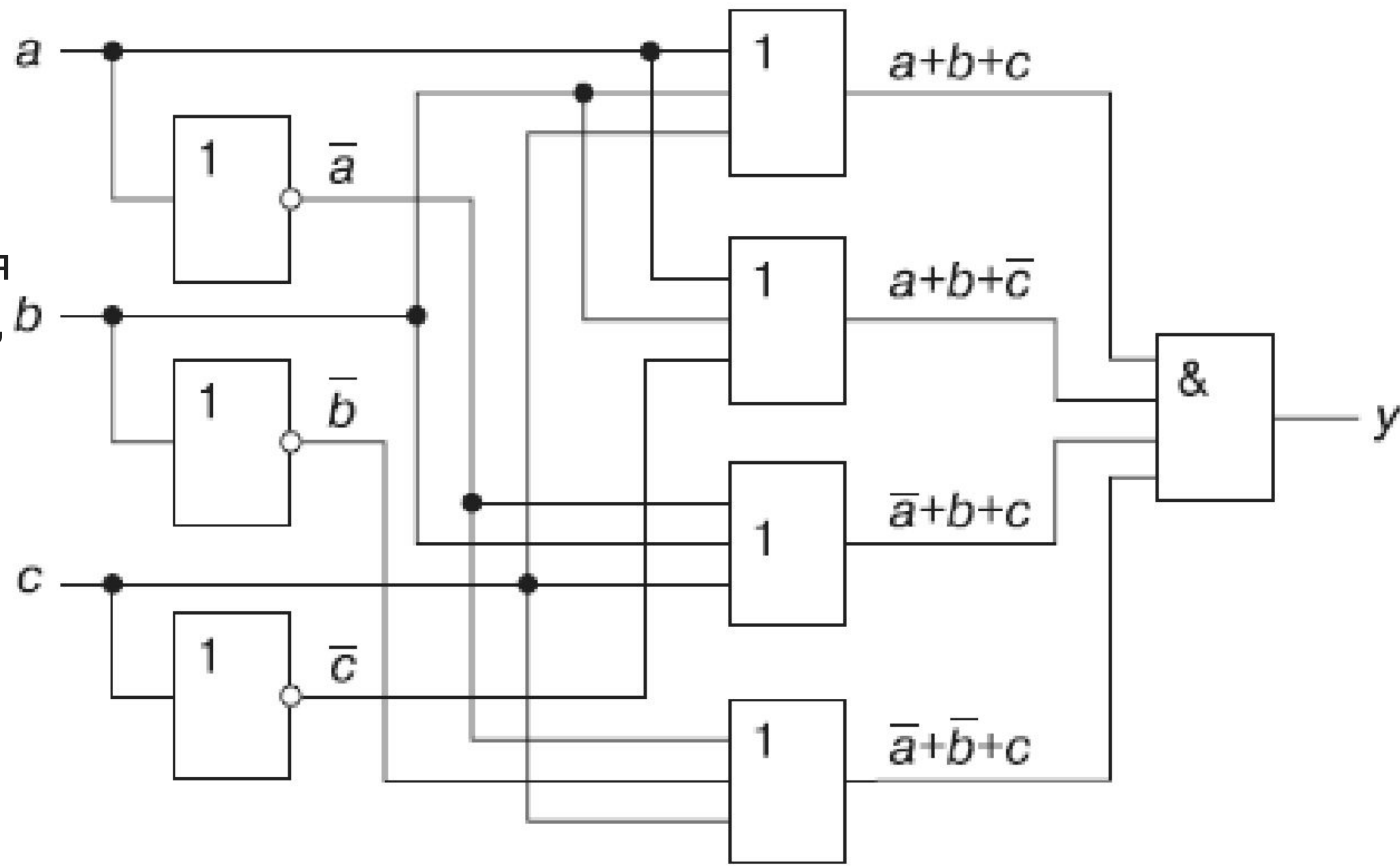
$$y(a,b,c)=\prod(0,1,2,4)$$

Для построения логической схемы необходимо логические элементы, предназначенные для выполнения логических операций, располагать, начиная от входа, в порядке, указанном в булевом выражении.

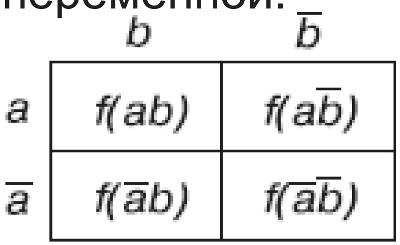
$$y = (a + b + c)(a + b + \overline{c})(\overline{a} + b + c)(\overline{a} + \overline{b} + c)$$

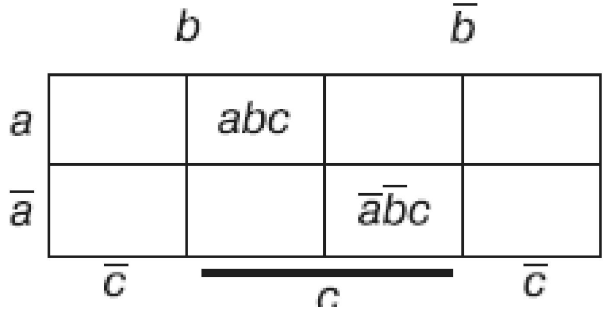
Слева располагаем входы *a*, *b* и *c* с ответвлениями на три инвертора, затем четыре элемента ИЛИ и, наконец, элемент И на выходе

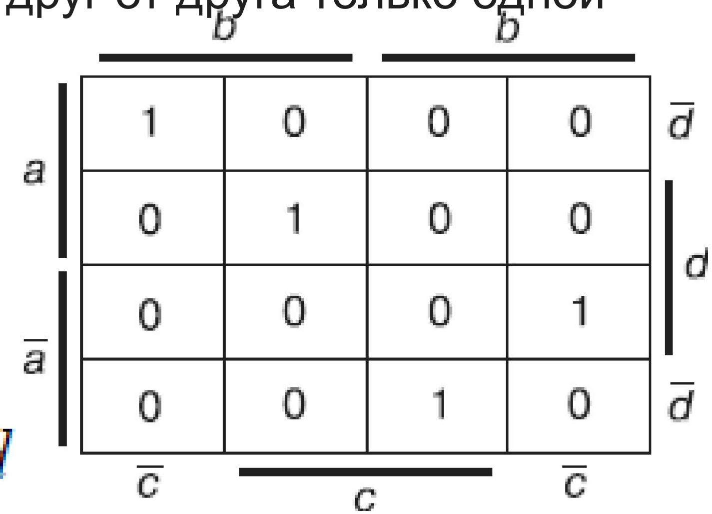
Любую логическую функцию можно реализовать по выражениям, представленным в виде СДНФ или СКНФ. Но полученная таким образом схема, не оптимальна с точки зрения её практической реализации: громоздка, содержит много элементов, и возникают трудности в обеспечении её высокой надёжности.



Минимизация логической функции - это минимизация стоимости её технической реализации, уменьшение количества элементарных логических элементов, использование только однородных базовых элементов (типа И-НЕ, ИЛИ-НЕ и др.)
Для интерпретации любых логических функций и их минимизации широко используют диаграммы Венна и карты Карно, базирующиеся на табличном представлении логических функций с числом переменных, не превышающих 4…5.
Карта Карно-Вейча — графическое представление всех минтермов (2^n) для данного числа переменных (*n*).
Каждый минтерм изображается в виде клетки, расположенной так, что минтермы, находящиеся в соседних клетках, отличаются друг от друга только одной переменной.







$$y = ab\bar{c}\bar{d} + abcd + \bar{a}\bar{b}c\bar{d} + \bar{a}\bar{b}\bar{c}d$$

Множество клеток позволяет отобразить все наборы аргументов, а карту Карно можно рассматривать как упорядоченное представление подмножеств. Так, в верхней строке рис. 5.5, *б* и во втором столбце имеем пересечение аргументов *a*, *b* и *c*, в нижней строке и третьем столбце пересечение аргументов, $\bar{b}$ и *c* и т. д.

Основу минимизации логических функций с помощью карт Вейча (Карно) составляет следующее: два минтерма, находящиеся в соседних клетках карты, могут быть заменены одной конъюнкцией, содержащей на одну переменную меньше. Если соседними являются две пары минтермов, то такая группа из четырех минтермов может быть заменена конъюнкцией, которая содержит на две переменные меньше.

В основе методов минимизации лежат поиск и склеивание соседних конъюнкций.

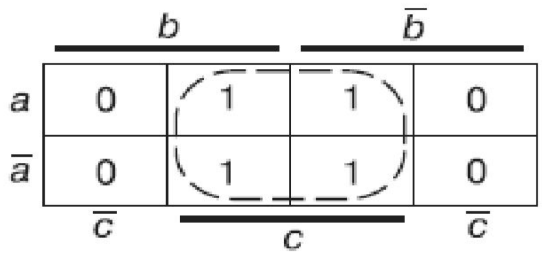
Соседними называются две одинакового ранга конъюнкции, являющиеся логическими произведениями одних и тех же переменных, если только одна какая-либо переменная входит в одну из конъюнкций с отрицанием, а в другую — без него.

Принцип склеивания соседних конъюнкций можно проиллюстрировать следующим приме

$$F = x_3x_2x_1x_0 + x_3\overline{x_2}x_1x_0 = x_3x_1x_0(x_2 + \overline{x_2}) = x_3x_1x_0 \cdot 1 = x_3x_1x_0.$$

Минимизация ФАЛ

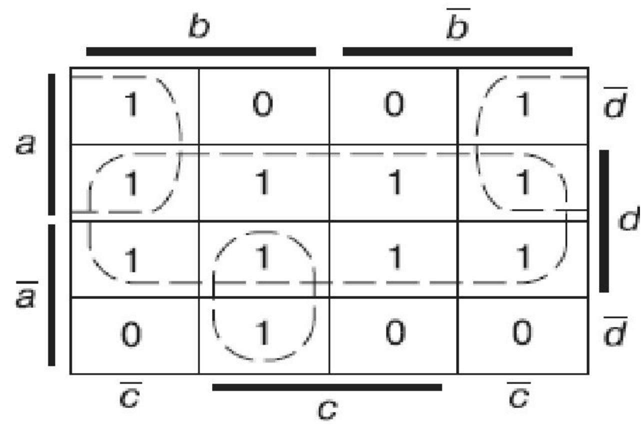
$y = abc + a\bar{b}c + \bar{a}bc + \bar{a}\bar{b}c$



$y = c$

а)

Минимизация ФАЛ по заданной таблице истинности



$y = d + \bar{a}bc + a\bar{c}\bar{d}$

Карты Вейча для двух, трех и четырех переменных следует рассматривать как плоскости, полученные из поверхностей торов, поделенных соответственно на 4, 8 и 16 клеток (причем сначала тор разрезан и выпрямлен в цилиндр, а затем этот цилиндр разрезан по образующей и развернут в плоскость).

При минимизации функции следует помнить, что одна и та же клетка карты может входить в несколько групп и что соседними клетками являются не только клетки, расположенные рядом по горизонтали и вертикали, но и клетки на противоположных границах карты.

3	1
2	0

6	7	3	2
4	5	1	0

12	14	6	4
13	15	7	5
9	11	3	1
8	10	2	0

25	29	13	9	24	28	12	8
27	31	15	11	26	30	14	10
19	23	7	3	18	22	6	2
17	21	5	1	16	20	4	0

51	59	27	19	49	57	25	17
55	63	31	23	53	61	29	21
39	47	15	7	37	45	13	5
35	43	11	3	33	41	9	1
50	58	26	18	48	56	24	16
54	62	30	22	52	60	28	20
38	46	14	6	36	44	12	4
34	42	10	2	32	40	8	0

102	118	54	38	103	119	55	39	99	115	51	35	98	114	50	34
110	126	62	46	111	127	63	47	107	123	59	43	106	122	58	42
78	94	30	14	79	95	31	15	75	91	27	11	74	90	26	10
70	86	22	6	71	87	23	7	67	83	19	3	66	82	18	2
100	116	52	36	101	117	53	37	97	113	49	33	96	112	48	32
108	124	60	44	109	125	61	45	105	121	57	41	104	120	56	40
76	92	28	12	77	93	29	13	73	89	25	9	72	88	24	8
68	84	20	4	69	85	21	5	65	81	17	1	64	80	16	0

Минимизация ФАЛ выполняется по следующему алгоритму:

• на карте Вейча ФАЛ выделяют ***прямоугольные области, объединяющие клетки с выбранным значением функции «лог 1» или «лог. 0». Каждая область должна содержать 2k клеток, где k - целое число (0, 1, 2, 4, 8, …).*** Выделенные области могут пересекаться, т. е. одна клетка может входить в несколько различных областей;

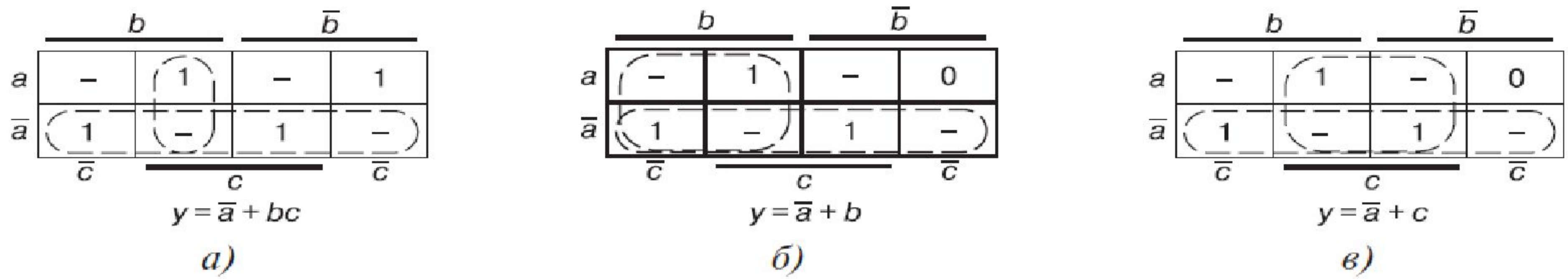
• каждая из выделенных областей описывается произведением переменных, которые для этой области остаются неизменными. Каждое произведение должно содержать n – k переменных;

• ***из полученного множества выбирают минимальное число максимально больших областей, включающих все клетки с выбранным значением ФАЛ;***

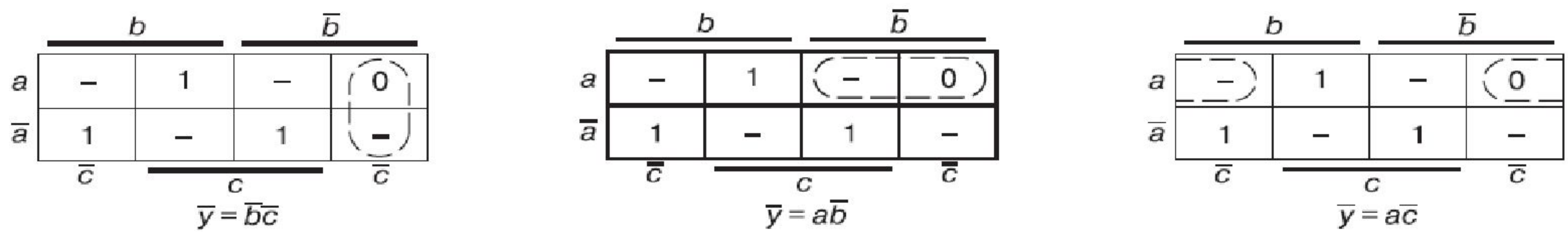
• ***логически суммируют выбранные произведения. Полученное выражение является минимальной дизъюнктивной ФАЛ.***

<u>Для получения минимально простой технической реализации целесообразно проводить минимизацию как единичных, так и нулевых значений функции</u>, и из полученных минимальных форм выбрать простейшую.
На практике часто встречаются логические функции, часть значений которых не задана, т. е. эти значения могут быть произвольными. Такие ФАЛ называют *недоопределенными.* При различном доопределении ФАЛ могут быть получены различные минимальные формы. При доопределении ФАЛ необходимо стремиться к тому, чтобы на карте Карно было выделено минимальное число максимально больших областей.

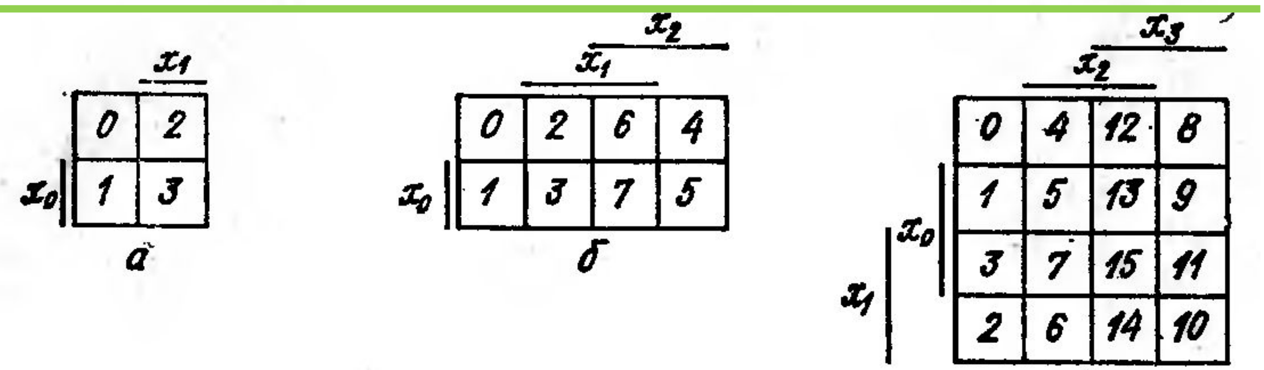
Доопределение ФАЛ по “1”



Доопределение ФАЛ по “0”



Карты Карно



Запись логических функций в универсальных базисах ИЛИ-НЕ и И-НЕ производится в такой последовательности:

- заданная логическая функция минимизируется в базисе ИЛИ, И, НЕ;
- над полученным выражением логической функции ставят двойное отрицание и с помощью правила де Моргана осуществляют переход в универсальный базис ИЛИ-НЕ или И-НЕ;
- при преобразовании логической функции используют следующие выражения:

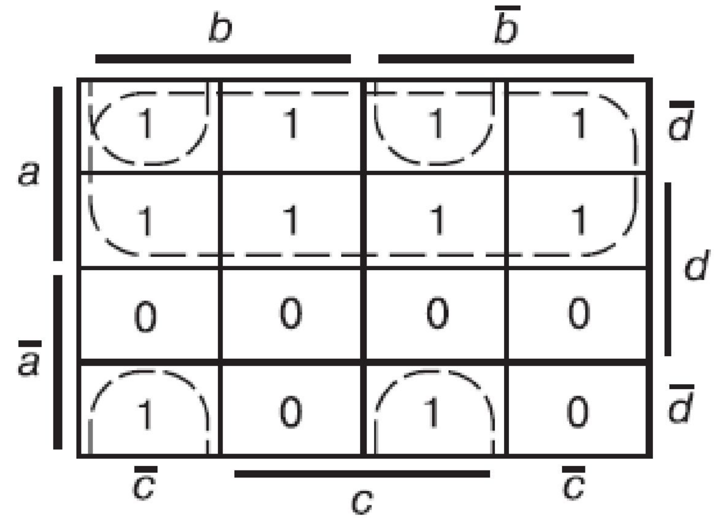
в базисе И-НЕ: $a\overline{b} = a(\overline{ab})$; $\overline{a} = \overline{aa}$; $\overline{a} = \overline{a \cdot 1}$; $\overline{a}b + a\overline{b} = \overline{\overline{[a(\overline{ab})]}\,\overline{[(\overline{ab})b]}}$;

в базисе ИЛИ-НЕ:

$$a + \overline{b} = a + (\overline{a + b}); \ \overline{a} = \overline{a + a}; \ \overline{a} = \overline{a + 0}; \ a\overline{b} + \overline{a}b = \overline{\overline{a + (\overline{a + b})} + \overline{(\overline{a + b}) + b}}.$$

При построении функциональных схем на элементах Шеффера логическую функцию представляют в минимальной КНФ, а при построении функциональных схем на элементах Пирса — в минимальной ДНФ. В этих случаях функциональные схемы содержат минимальное количество элементов и более просты при построении.

Запишем логическую функцию $y=a\bar{d}+a\bar{c}d+abd+b\bar{c}\bar{d}+\bar{b}c\bar{d}$ в базисе И-НЕ и ИЛИ-НЕ в минимальных ДНФ и КНФ.



Вычерчиваем карту Вейча (Карно) для четырех переменных *a*, *b*, *c* и *d* и отметим в ней единицей (1) минтермы, содержащие конъюнкции, входящие в заданную фукцию. В результате склеивания минтермов в карте Карно, для которых заданная функция *y* = 1, получим для выходной функции в минималь

$$y_{ДНФ}=a+b\bar{c}\bar{d}+\bar{b}c\bar{d}$$

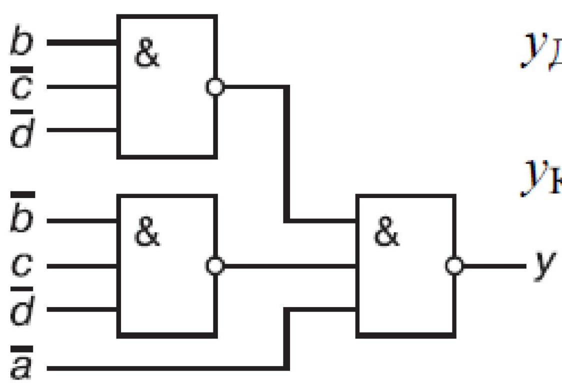
в результате склеивания минтермов, для которых функция *y* = 0, получим выражение для исходной функции в минимальной КНФ:

$$y_{КНФ}=\overline{\bar{a}d+\bar{a}bc+\bar{a}\bar{b}\bar{c}}=(a+\bar{d})(a+\bar{b}+\bar{c})(a+b+c).$$

Для записи логической функции *y*(*a*,*b*,*c*,*d*) в базисе И-НЕ применим к правой части полученных выражений двойное отрицание:

$$y_{ДНФ}=\overline{\overline{a+b\bar{c}\bar{d}+\bar{b}c\bar{d}}}=\overline{\bar{a}(\overline{b\bar{c}\bar{d}})(\overline{\bar{b}c\bar{d}})};$$

$$y_{КНФ}=\overline{\overline{(a+\bar{d})(a+\bar{b}+\bar{c})(a+b+c)}}=\overline{(\overline{\bar{a}d})(\overline{\bar{a}bc})(\overline{\bar{a}\bar{b}\bar{c}})}.$$



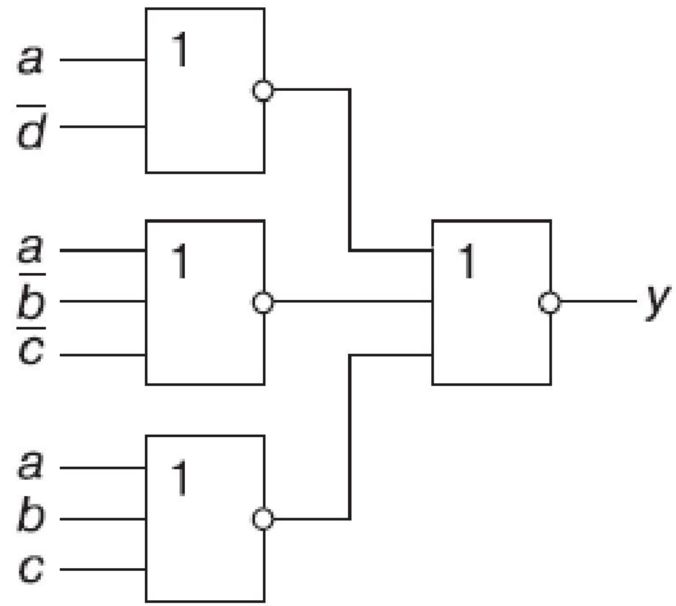
$$y_{ДНФ}=\overline{\bar{a}(\overline{b\bar{c}\bar{d}})(\overline{\bar{b}c\bar{d}})}$$

Для записи логической функции $y(a,b,c,d)$ в базисе ИЛИ-НЕ применим также к правой части выражений (5.11) и (5.12) двойное отрицание. После преобразований получим:

$$y_{\text{ДНФ}} = \overline{\overline{a + b\bar{c}\bar{d} + \bar{b}c\bar{d}}} = \overline{a + \overline{(\bar{b} + c + d)} + \overline{(b + \bar{c} + d)}}; \qquad (5.15)$$

$$y_{\text{КНФ}} = \overline{\overline{(a+\bar{d})(a+\bar{b}+\bar{c})(a+b+c)}} = \overline{\overline{(a+\bar{d})} + \overline{(a+\bar{b}+\bar{c})} + \overline{(a+b+c)}}. \qquad (5.16)$$

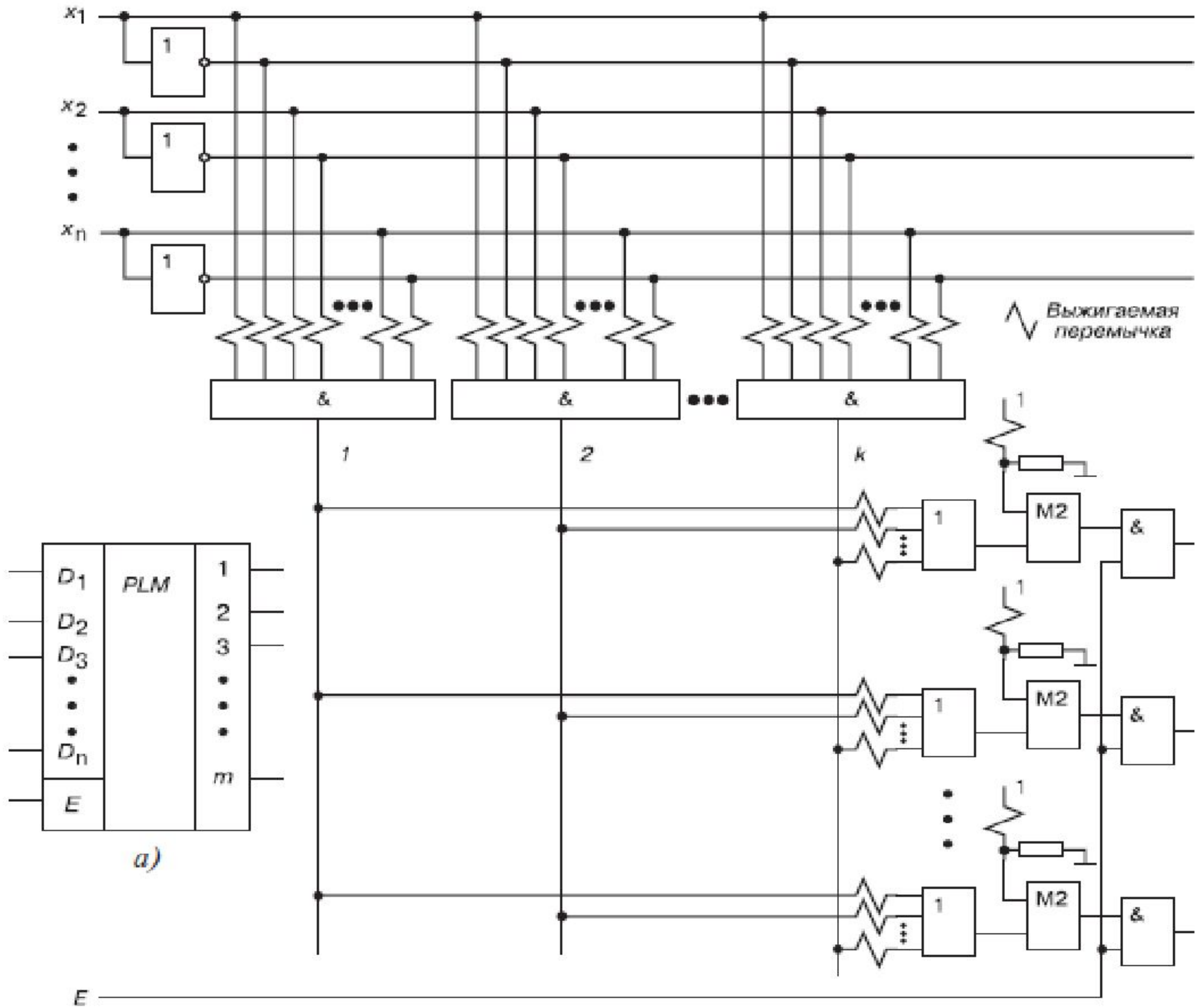
Из анализа выражений (5.15) и (5.16) следует, что функциональные схемы, реализующие эти выражения, будут содержать одинаковое количество элементов Пирса. На



$$y_{КНФ} = \overline{\overline{(a+\bar{d})} + \overline{(a+\bar{b}+\bar{c})} + \overline{(a+b+c)}}$$

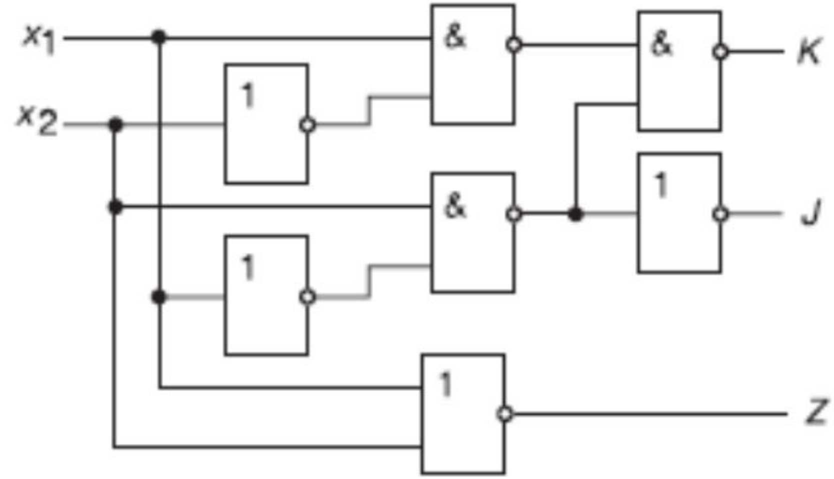
Синтез логических функций и произвольных кодовых преобразований в настоящее время выполняют посредством *программируемых логических матриц* (ПЛМ) средней и даже большой интеграции.

Программируема я логическая матрица имеет n входов (x_1, x_2, , xn), m выходов (y_1, y_2, , y_m), k элементов И, выходы которых образуют k вертикальных шин, и m элементов ИЛИ, выходы которых подключены к сумматорам по модулю 2 (M_2), выполняющим роль управляемых инверторов. Выводы этих инверторов являются вы ходами самой ПЛМ. Каждый элемент И имеет $2n$ входов, которыми он связан со все ми шинами входных сигналов и с шинами их инверсий



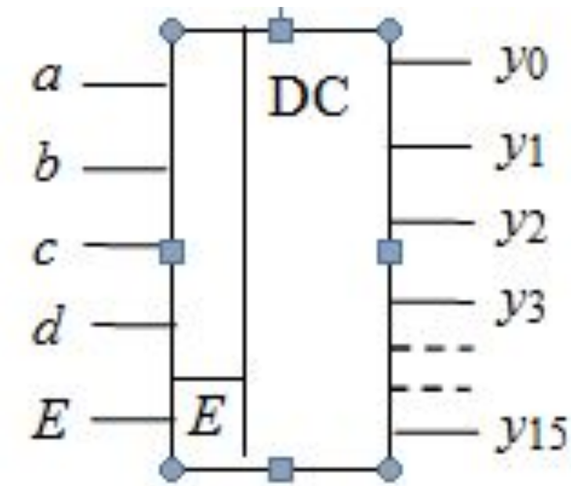
В линии связи включены специальные перемычки, обозначенные на рис. 5.10, *б* короткими зигзагами. Эти перемычки выполняются из нихрома, кристаллического кремния и других материалов или в виде специальных *pn*переходов так, чтобы их можно было разрушать ("выжигать"), оставляя лишь те связи, которые нужны потре бителю ПЛМ. Причём разрушение ненужных легкоплавких перемычек может осуще ствлять и пользователь, подавая на соответствующие выводы корпуса ПЛМ импульсы тока определенной амплитуды и длительности.

Элементы ИЛИ, так же, как и элементы И, имеют на входах выжигаемые пере мычки, с помощью которых они подключены ко всем вертикальным шинам. После выжигания ненужных перемычек на этих входах элементов ИЛИ обеспечивается уровень логического нуля. Аналогичным образом программируют отсутствие или выполнение инвертирования выходов ИЛИ, соответственно пережигая или оставляя перемычки на верхних на рис. 5.10, *б* входах элементов М2.



X		$Z=\bar{x}_1\bar{x}_2$	$J=x_1\bar{x}_2$	$K=\bar{x}_1x_2+x_1\bar{x}_2$
x_1	x_2			
0	0	1	0	0
0	1	0	0	1
1	0	0	1	1
1	1	0	0	0

<u>Дешифратор</u> **<u>(DC) или</u>** ***<u>декодер</u>*** – комбинационная схема с *n* входами и $m = 2^n$ выходами ($m > n$), преобразующая двоичный входной *n*-код (кодовое слово) в унитарный. На одном из *m* выходов дешифратора появляется логическая 1, а именно на том, номер которого соответствует поданному на вход двоичному коду. ***<u>На выходе дешифратора формируется функция, представляющая собой конституенту единицы (минтерм) n переменных</u>***



Дешифраторы часто имеют *разрешающий* (управляющий, стробирующий) вход *E*. При *E* = 1 дешифратор функционирует как обычно, при *E* = 0 на всех выходах устанавливается 0 независимо от поступающего кода адреса. Дешифраторы широко используют во многих устройствах, в том числе в качестве преобразователей двоичного кода в десятичный.

x_2	x_1	x_0	F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

$F_0 = \bar{x}_2\bar{x}_1\bar{x}_0;\quad F_3 = \bar{x}_2 x_1 x_0;$

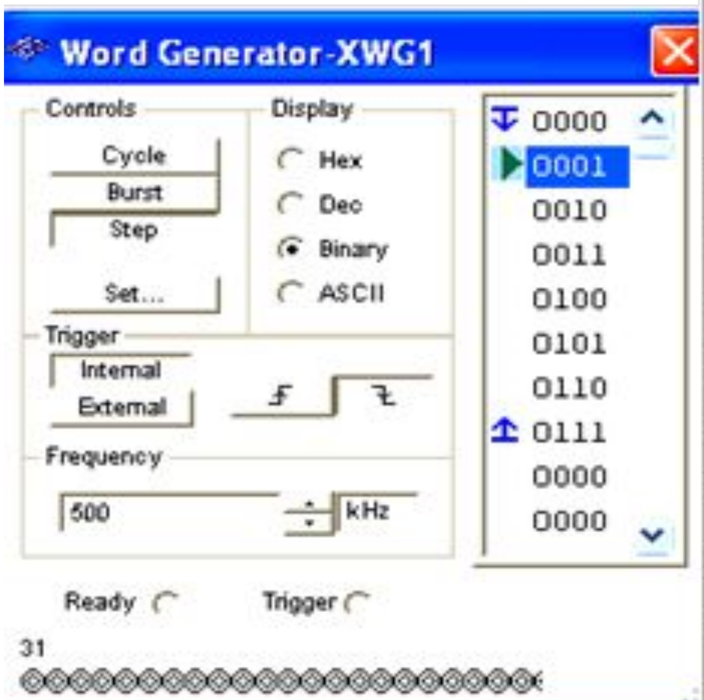
$F_1 = \bar{x}_2\bar{x}_1 x_0;\quad F_4 = x_2\bar{x}_1\bar{x}_0;$

$F_2 = \bar{x}_2 x_1\bar{x}_0;\quad F_5 = x_2\bar{x}_1 x_0;$

$F_6 = x_2 x_1\bar{x}_0;$

$F_7 = x_2 x_1 x_0.$

Каждый выход полного дешифратора реализует конъюнкцию входных переменных (код адреса) или их инверсий: при наборе $y_0 = 1$, при $y_7 = 1$, при *abcd* (1111) $y_{15} = 1$ и т. д.



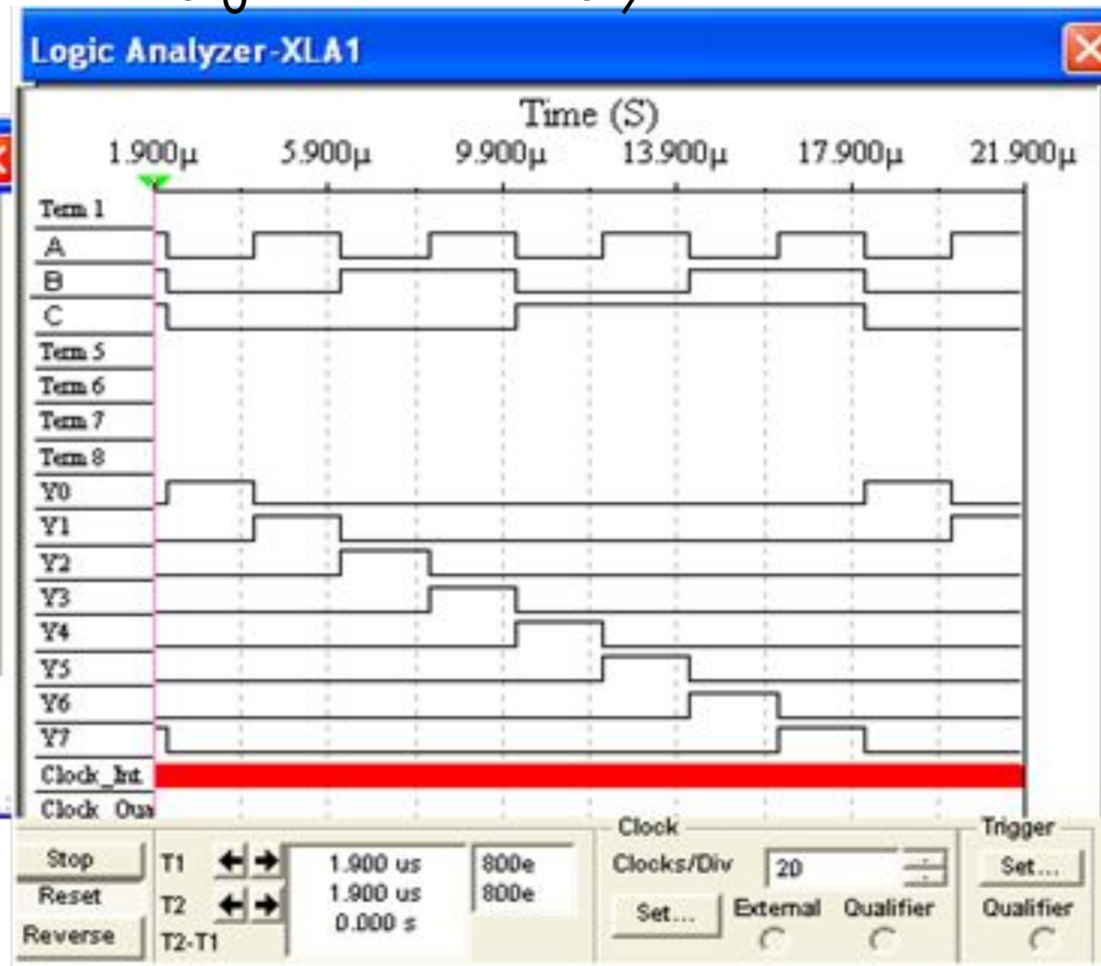
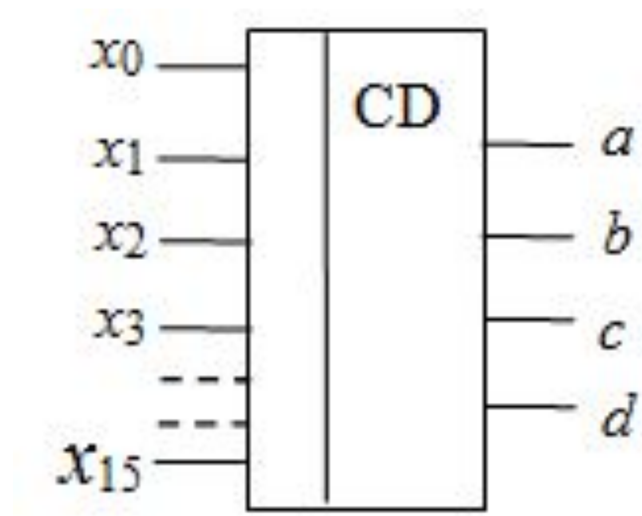


Рис. 30.7

Дешифраторы могут быть неполными, реализующими $m < 2^n$ минтермов. Такие дешифраторы используются, например, для преобразования двоично-десятичного кода в код, предназначенный для управления десятичным индикатором (дешифраторы 4 X 10).

Шифратор (CD) или кодер выполняет функцию, обратную функции дешифратора. Классический шифратор имеет n входов и m выходов (m < n), и при подаче сигнала 1 на один из входов (и не более) на выходе кодера появляется двоичный код номера возбужденного выхода. Число входов и выходов такого шифратора связано соотношением n = 2m

Области использования шифраторов − отображение в виде двоичного кода номера нажатой кнопки или положения многопозиционного переклю чателя, а также номера устройства, подавшего сигнал на обслуживание в микропроцессорных системах



x_9	x_8	x_7	x_6	x_5	x_4	x_3	x_2	x_1	x_0	Q_3	Q_2	Q_1	Q_0
0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0	0	0	1
0	0	0	0	0	0	0	1	0	0	0	0	1	0
0	0	0	0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	0	1	0	0	0	0	0	1	0	0
0	0	0	0	1	0	0	0	0	0	0	1	0	1
0	0	0	1	0	0	0	0	0	0	0	1	1	0
0	0	1	0	0	0	0	0	0	0	0	1	1	1
0	1	0	0	0	0	0	0	0	0	1	0	0	0
1	0	0	0	0	0	0	0	0	0	1	0	0	1

$$Q_3 = x_8 + x_9,$$

$$Q_2 = x_4 + x_5 + x_6 + x_7,$$

$$Q_1 = x_2 + x_3 + x_6 + x_7,$$

$$Q_0 = x_1 + x_3 + x_5 + x_7 + x$$

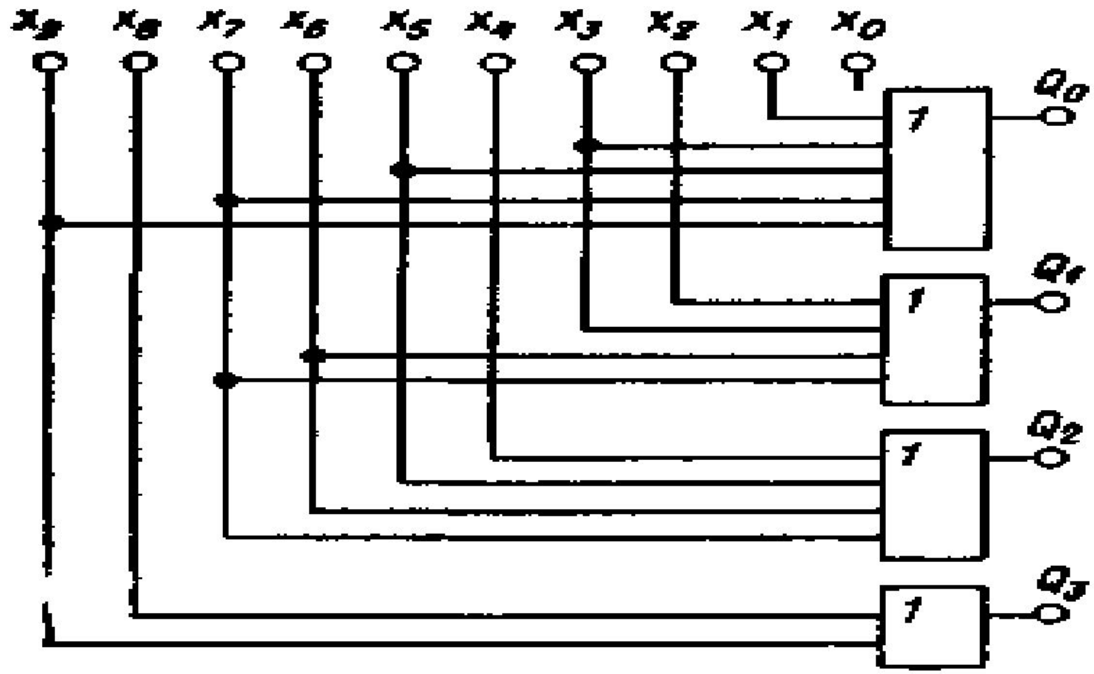
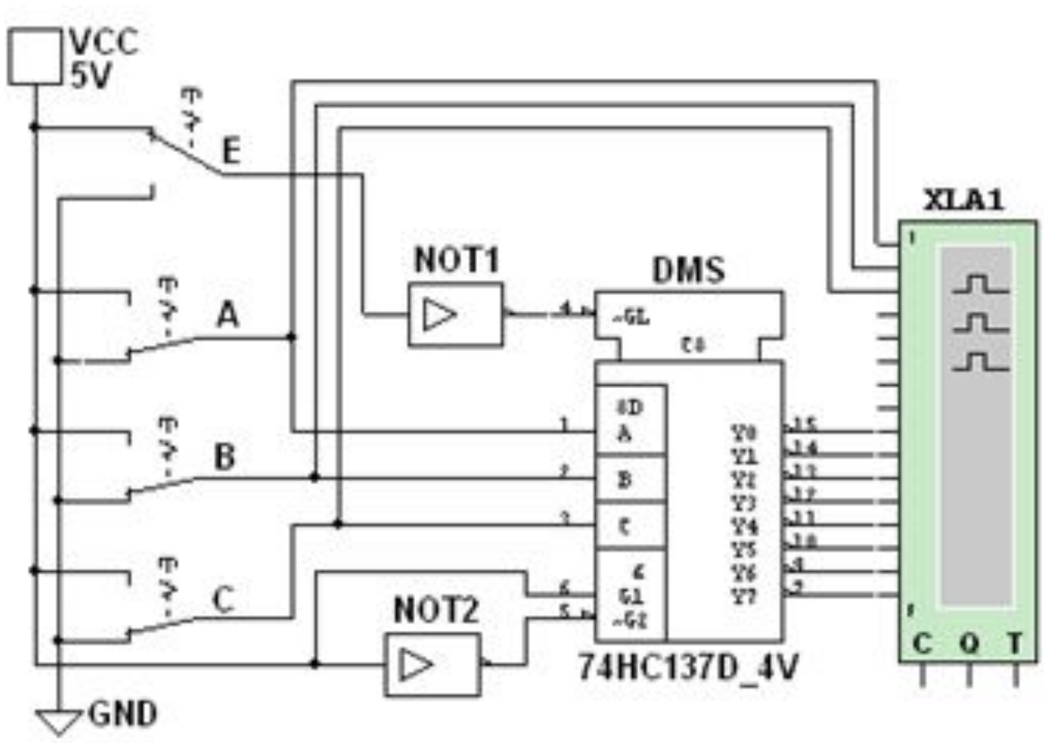


Рис. 16.9. Логическая схема шифратора десятичных чисел


VCC
5V
E
A
B
C
NOT1
NOT2
DMS
74HC137D_4V
XLA1
C Q T
GND

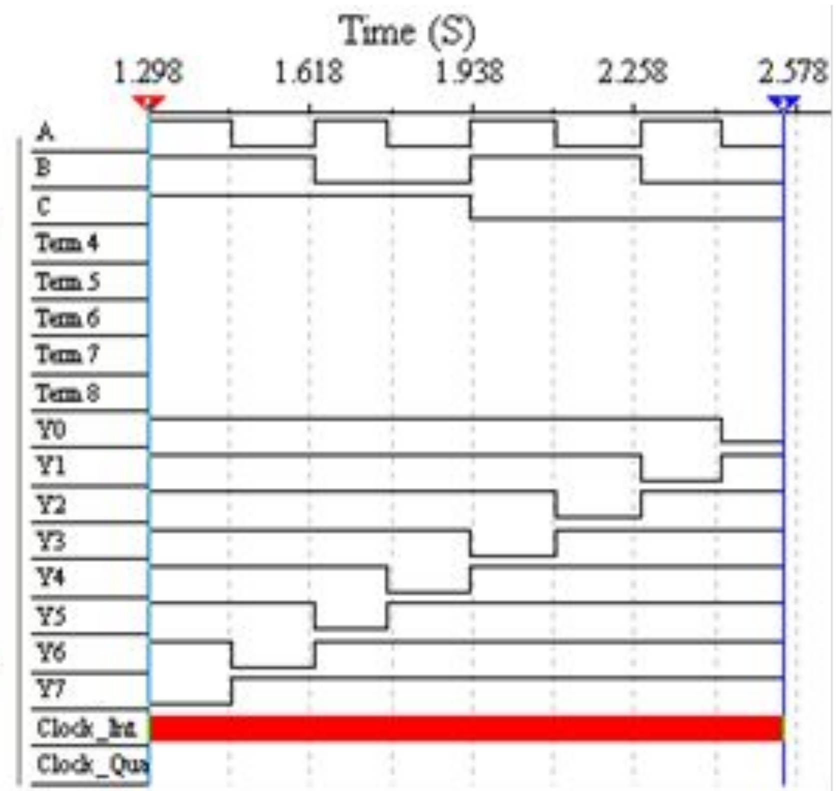

Time (S)
1.298
1.618
1.938
2.258
2.578
A
B
C
Term 4
Term 5
Term 6
Term 7
Term 8
Y0
Y1
Y2
Y3
Y4
Y5
Y6
Y7
Clock_Int
Clock_Qua

Рис. 30.9

x_9	x_8	x_7	x_6	x_5	x_4	x_3	x_2	x_1	x_0	Q_3	Q_2	Q_1	Q_0
0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0	0	0	1
0	0	0	0	0	0	0	1	0	0	0	0	1	0
0	0	0	0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	0	1	0	0	0	0	0	1	0	0
0	0	0	0	1	0	0	0	0	0	0	1	0	1
0	0	0	1	0	0	0	0	0	0	0	1	1	0
0	0	1	0	0	0	0	0	0	0	0	1	1	1
0	1	0	0	0	0	0	0	0	0	1	0	0	0
1	0	0	0	0	0	0	0	0	0	1	0	0	1

$$x_0 = \overline{Q}_3\overline{Q}_2\overline{Q}_1\overline{Q}_0,$$

$$x_1 = \overline{Q}_3\overline{Q}_2\overline{Q}_1 Q_0,$$

$$x_2 = \overline{Q}_3\overline{Q}_2 Q_1\overline{Q}_0,$$

. . . .

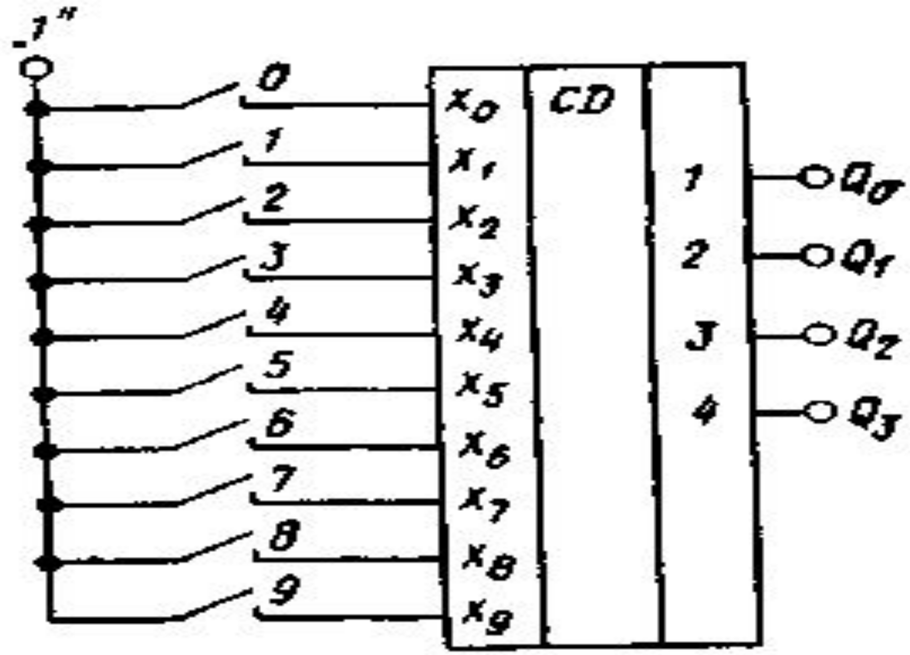
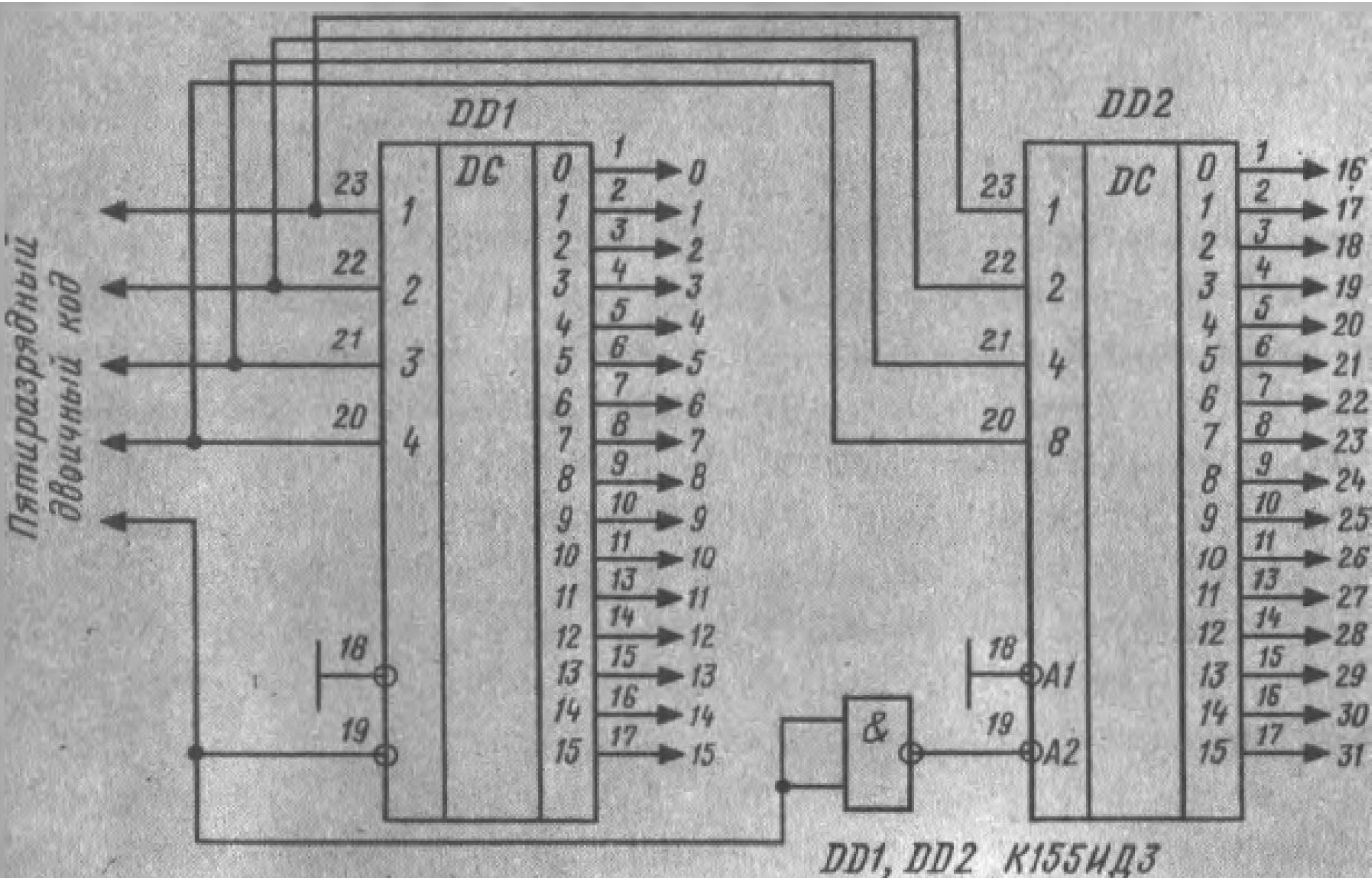


Рис 16.10. Устройство ввода информации с клавиатуры


DD1
DD2
DC
DC
Пятиразрядный двоичный код
DD1, DD2 К155ИД3

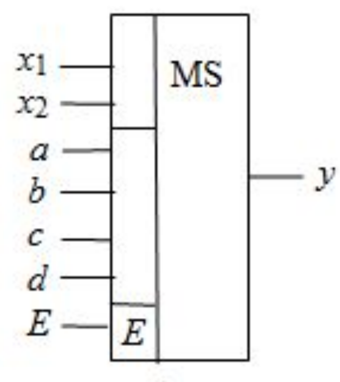
Мультиплексор (MS) – это функциональный узел, осуществляющий подключение (*коммутацию*) одного из нескольких входов к выходу *y*. На выход такого устройства передаётся логический уровень того информационного разряда, номер которого в двоичном коде задан на адресных входах x_1 и x_2

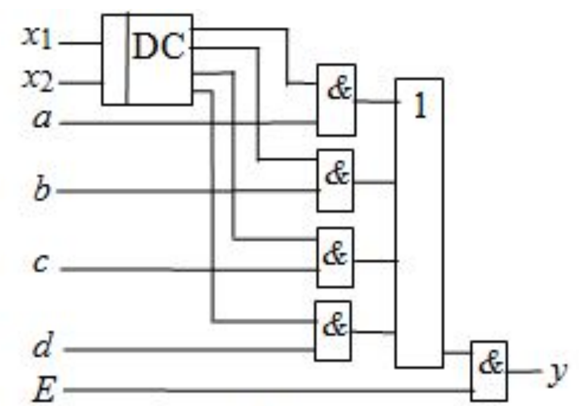
При $x_1 = 0$ и $x_2 = 0$, $y = a$; при $x_1 = 0$ и $x_2 = 1$, $y = b$; при $x_1 = 1$ и $x_2 = 0$, $y = c$ и при $x_1 = 1$ и $x_2 = 1$, $y = d$.
Функционирование мультиплексора описывается выражением
Вход *E* – разрешающий: при *E* = 1 мультиплексор работает как обычно, при *E* = 0 выход узла находится в неактивном состоянии, мультиплексор заперт.

<u>Мультиплексором наз. комбинационное логическое устройство, предназначенное для управляемой передачи данных от нескольких источников информации в один выходной канал. Имеет один выход и две группы входов: информационные и адресные.</u>

Типовое применение мультиплексора — передача информации от нескольких разнесенных в пространстве источников (дат чиков) информации на вход одного приемника. Информацию, разнесенную в пространстве, он преобразует к виду с разделением во времени.

Код, подаваемый на адресные входы, определяет, какой из информационных входов в данный момент подключен к выходному выводу. Поскольку л-разрядный двоичный код может принимать 2^n значе ний, то, если число адресных входов мультиплексора равно л, число его информационных входов должно равняться 2^n.

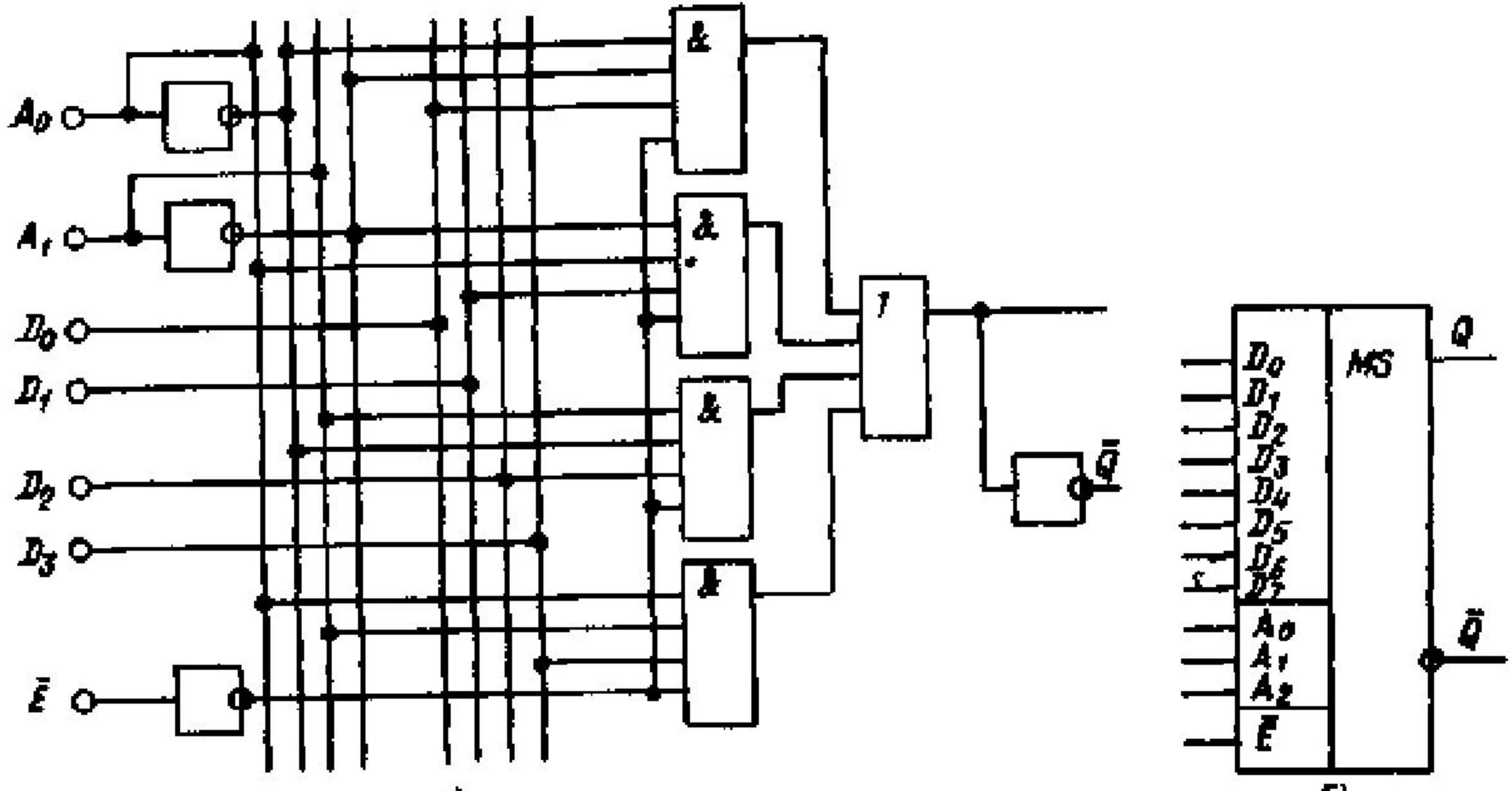




E	A_1	A_0	Q	$\overline{Q}$
1	X	X	0	1
0	0	0	D_0	$\overline{D_0}$
0	0	1	D_1	$\overline{D_1}$
0	1	0	D_2	$\overline{D_2}$
0	1	1	D_3	$\overline{D_3}$

$$Q = D_0\overline{A}_1\overline{A}_0\overline{E} + D_1\overline{A}_1A_0\overline{E} + D_2A_1\overline{A}_0\overline{E} + D_3A_1A_0\overline{E}.$$

Нетрудно заметить, что ФАЛ дешифратора (16.7) отлича
от ФАЛ мультиплексора (16.5) только наличием в последней
полнительного множителя, соответствующего значению сиг
на информационном входе D. Поэтому при $D=1$ демультипле
функционирует как дешифратор. Обратное преобразование де
ратора в демультиплексор требует введения вспомогательных
2И, выполняющих операцию логического умножения между
щим сигналом информационного входа D и соответствующим
гическим произведением адресных сигналов $(Q_3Q_2Q_1Q_0)$. С



а) б)

Рис. 16.4. Логическая схема мультиплексора (*а*) и его условное графическое обозначение (*б*)

Функция алгебры логики, описывающая работу мультиплексора, имеет вид

$$Q = D_0\bar{A}_1\bar{A}_0\bar{E} + D_1\bar{A}_1A_0\bar{E} + D_2A_1\bar{A}_0\bar{E} + D_3A_1A_0\bar{E}. \qquad (16.4)$$

Демультиплексор (DMS) выполняет функцию, обратную функции мультиплексора, т. е. производит коммутацию одного входного сигнала на 2^n выходов, где n – число адресных входов x_i. Он осуществляет преобразование информации из последовательной формы (последовательно-параллельной) в параллельную. Демультиплексор имеет один информационный вход D и несколько выходов, причем вход подключается к выходу y_i, имеющему заданный адрес. условное графическое обозначение демультиплексора, имеющего четыре выхода, закон функционирования которого задан (табл. 30.1). Пользуясь табл. 30.1, запишем переключательные функции для выхода устройства:

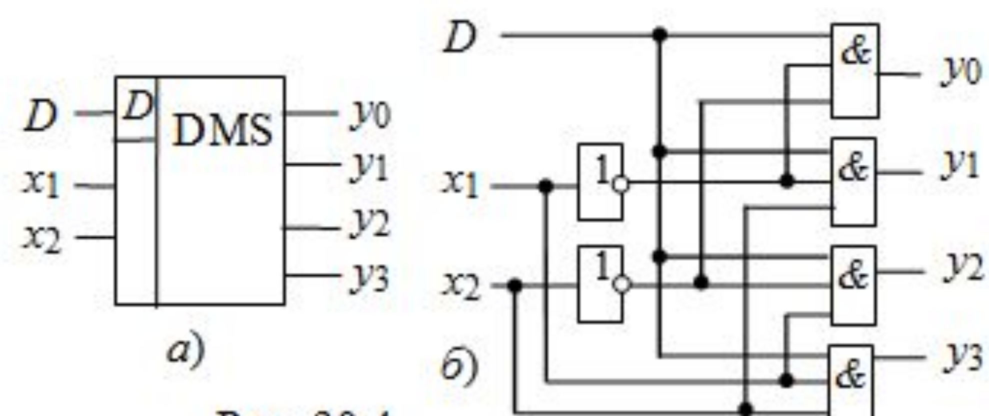


Рис. 30.4

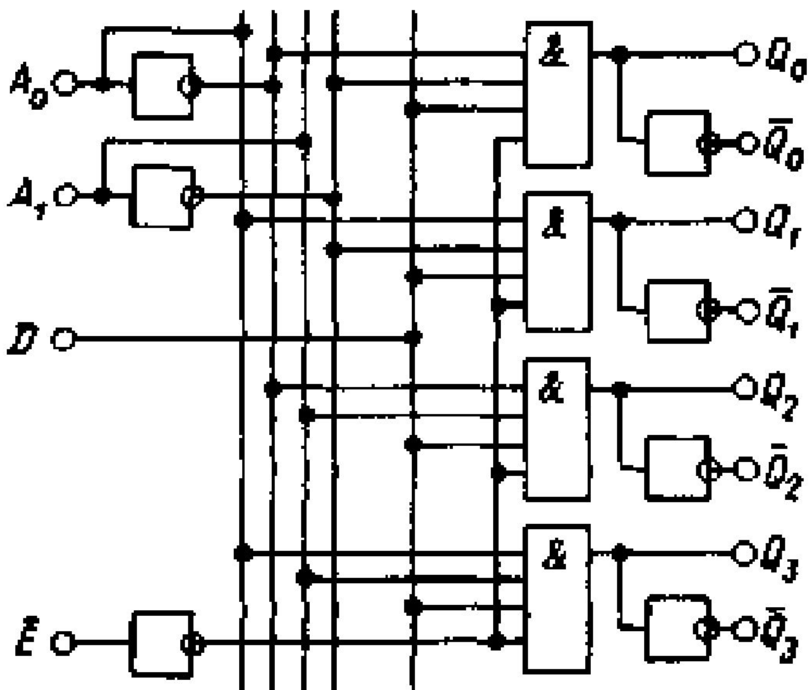
Таблица 30.1

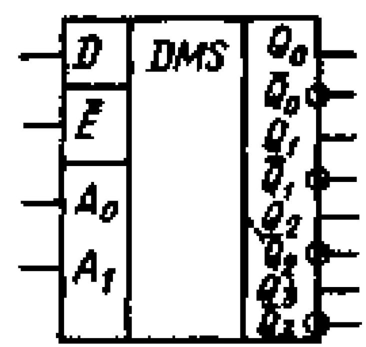
D	x_1	x_2	y_3	y_2	y_1	y_0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

E	A_1	A_0	Q_0	Q_1	Q_2	Q_3
1	X	X	0	0	0	0
0	0	0	D	0	0	0
0	0	1	0	D	0	0
0	1	0	0	0	D	0
0	1	1	0	0	0	D

Данной таблице соответствует следующая система ФАЛ:

$$Q_0 = D\bar{A}_1\bar{A}_0\bar{E} = \bar{D} \downarrow A_1 \downarrow A_0 \downarrow E,$$
$$Q_1 = D\bar{A}_1A_0\bar{E} = \bar{D} \downarrow A_1 \downarrow \bar{A}_0 \downarrow E, \quad (16$$
$$Q_2 = DA_1\bar{A}_0\bar{E} = \bar{D} \downarrow \bar{A}_1 \downarrow A_0E.$$
$$Q_3 = DA_1A_0\bar{E} = \bar{D} \downarrow \bar{A}_1 \downarrow \bar{A}_0E.$$





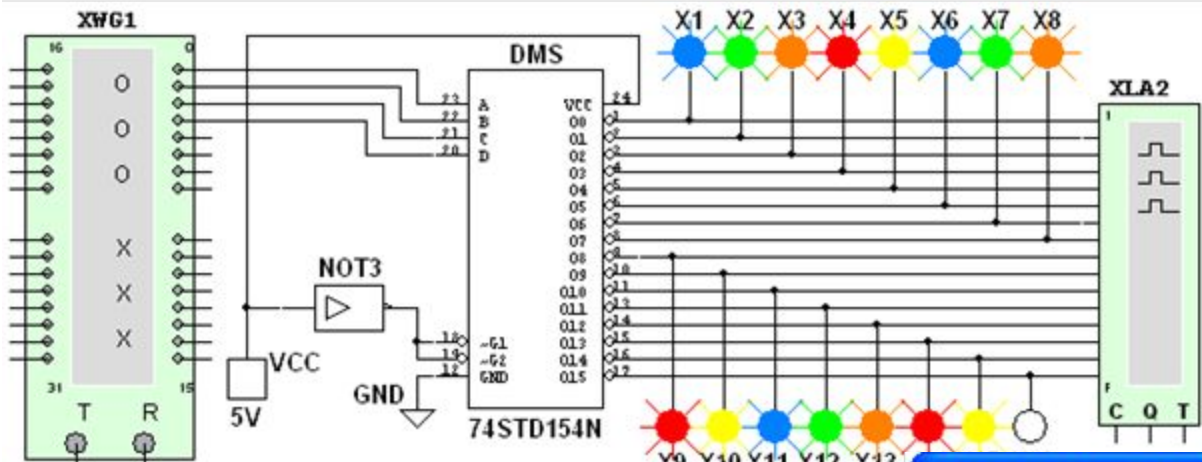
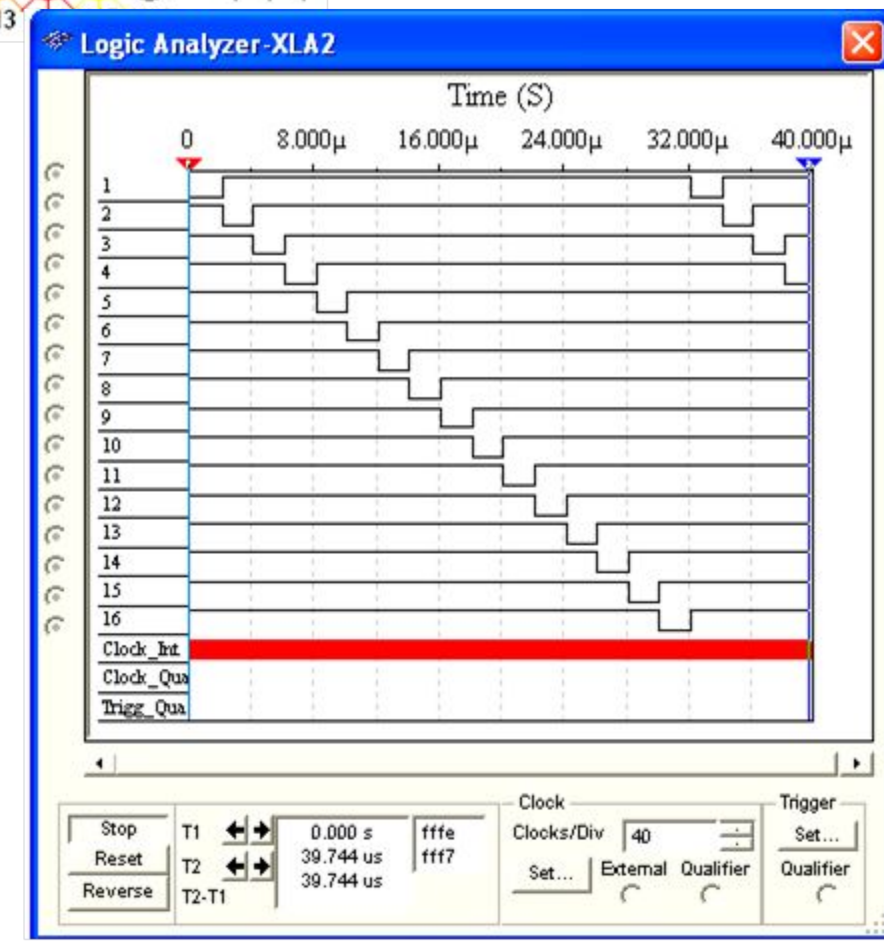
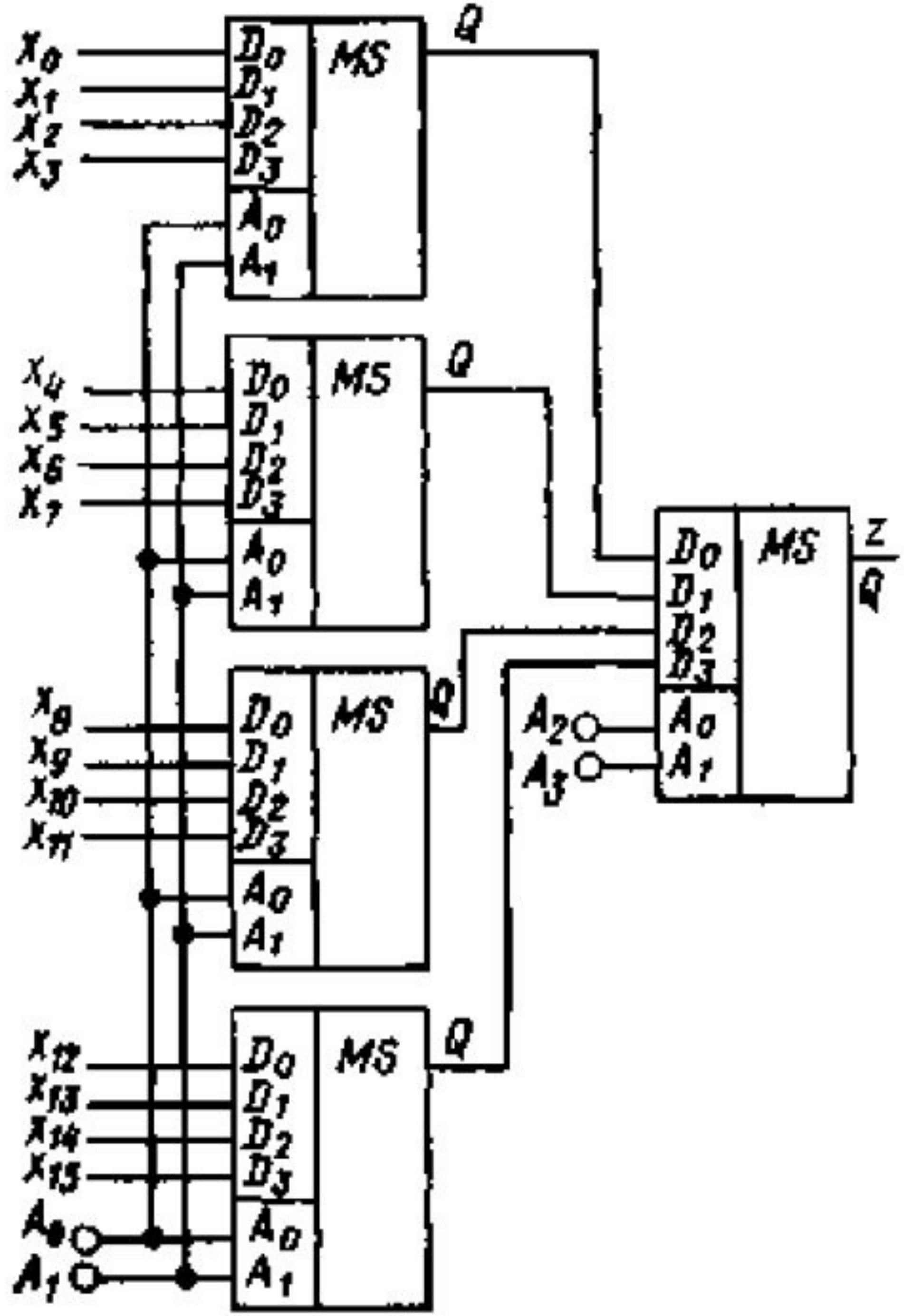

XWG1
DMS
74STD154N
NOT3
VCC
5V
GND
XLA2
X1 X2 X3 X4 X5 X6 X7 X8
X9 X10 X11 X12 X13
A
B
C
D
VCC
~G1
~G2
GND
T R
C Q T

Рис. 30.10


Logic Analyzer-XLA2
Time (S)
0 8.000µ 16.000µ 24.000µ 32.000µ 40.000µ
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
Clock_Int
Clock_Qua
Trigg_Qua
Stop
Reset
Reverse
T1
T2
T2-T1
0.000 s
39.744 us
39.744 us
fffe
fff7
Clock
Clocks/Div
40
Set...
External
Qualifier
Trigger
Set...
Qualifier

Если общее число выходов разрабатываемого устройства превышает имеющиеся в выпускаемых интегральных микросхемах, то используют параллельное подключение нескольких схем - ***<u>мультиплексорное дерево</u>***



демультиплексорное дерево, построенное на мультиплексорах с четырьмя выходами. Объединяя мультиплексор с демультиплексором, получают комбинационное устройство, в котором по заданным адресам один из входов подключается к одному из его выходов

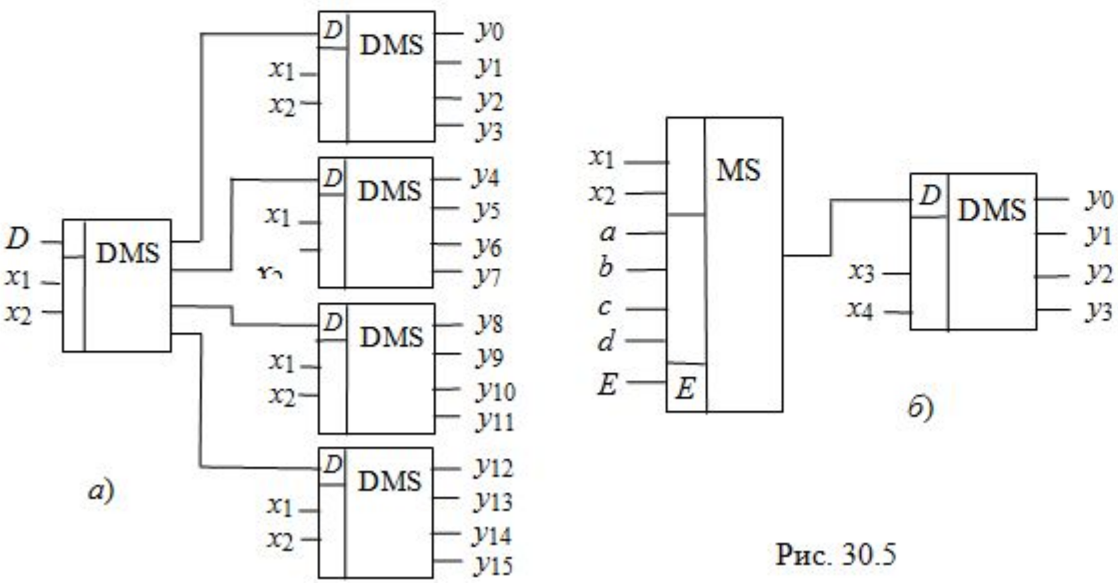


Рис. 30.5

Цифровым компаратором наз. комбинационное логическое устройство, предназначенное для сравнения чисел, представленных в виде двоичных кодов.

Число входов компаратора определяется разрядностью сравниваемых кодов. На выходе компаратора обычно формируется три сигнала:

$F_{=}$ — равенство кодов,

$F_{>}$ — если числовой эквивалент первого кода больше второго,

$F_{<}$ — если числовой эквивалент первого кода меньше второго.

Таблица истинности компаратора одноразрядных кодов

x_1	x_0	$F_{=}$	$F_{>}$	F
0	0	1	0	0
0	1	0	0	1
1	0	0	1	0
1	1	1	0	0

Система ФАЛ, соответствующая приведенной таблице истинности, имеет вид

$$F_{=} = \bar{x}_1\bar{x}_0 + x_1x_0 = \overline{x_1 \oplus x_0} = \bar{F}_{>}\bar{F}_{<},$$

$$F_{<} = \bar{x}_1x_0 = \bar{F}_{=}\bar{F}_{>}, \qquad (16.9)$$

$$F_{>} = x_1\bar{x}_0 = \bar{F}_{=}\bar{F}_{<}.$$

$$F_{=} = F_{1=}F_{0=},$$
$$F_{>} = F_{1>} + F_{1=}F_{0>},$$
$$F_{<} = \overline{F_{=} + F_{>}}.$$

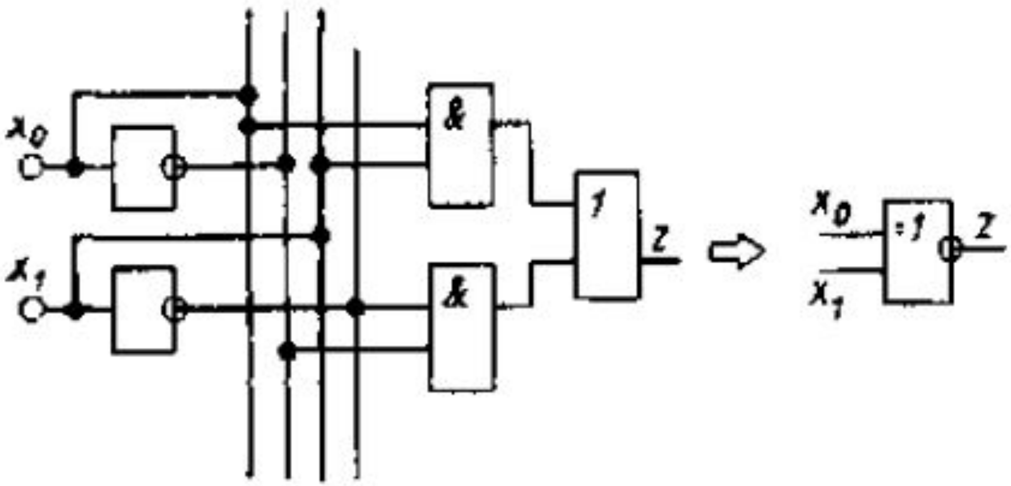


Рис. 16.16. Схемная реализация операции Исключающее ИЛИ—НЕ

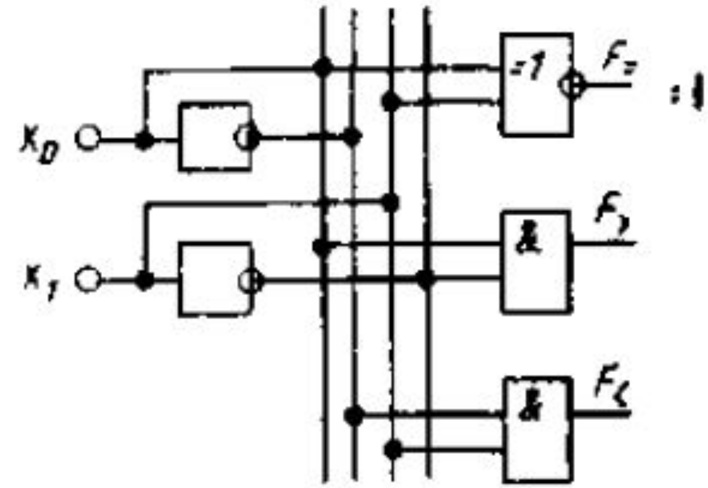


Рис. 16.17 Логическая схема компаратора