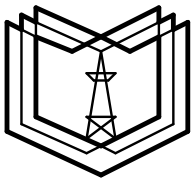


Министерство образования и науки Российской федерации

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

«Казанский государственный энергетический университет»

Л.В. АХМЕТВАЛЕЕВА, Л.Г. КУЛАГИНА

ОСНОВЫ ЦИФРОВОЙ ЭЛЕКТРОНИКИ

Учебно-методическое пособие

Казань
2018

УДК 621.38
ББК 32.859
А95

Рецензент – профессор, доктор физико-математических наук В. А. Уланов

Ахметвалеева Л. В., Кулагина Л. Г.

А95 Основы цифровой электроники: учебно-методическое пособие / Л. Г. Кулагина, Л. В. Ахметвалеева. – Казань: Казан. гос. энерг. ун-т, 2018. – 100 с.

Изложены принципы функционирования основных узлов цифровой электроники. Предложен комплекс практических заданий для закрепления студентами теоретического материала и навыков синтеза электронных схем на основе цифровых устройств комбинационного типа.

Предназначено для студентов всех форм обучения по образовательным программам «Промышленная электроника» и «Светотехника и источники света» направления подготовки 11.03.04 «Электроника и наноэлектроника»; по образовательной программе «Автоматизация технологических процессов и производств»; направления подготовки 15.03.04 «Автоматизация технологических процессов и производств»; по образовательным программам «Информационно-измерительная техника и технологии» и «Приборы и методы контроля качества и диагностика» направления подготовки 12.03.01 «Приборостроение»; по образовательной программе «Мехатроника» направления подготовки 15.03.06 Мехатроника и робототехника.

УДК 621.38
ББК 32.859
А95

ВВЕДЕНИЕ

Квалифицированный специалист электронной техники должен быть хорошо знаком с элементной базой и методами логического проектирования. Только при решении задач анализа и синтеза цифровых устройств можно достичь глубокого понимания проблем цифровой техники, умения творчески применять теоретические знания.

Целью данного учебно-методического пособия является изучение принципов построения, логической структуры и функциональных особенностей комбинационных цифровых устройств, получение практических навыков анализа и синтеза дискретных устройств различного назначения.

В учебно-методическом пособии рассмотрены принципы построения и функционирования интегральных логических элементов, шифраторов, дешифраторов, преобразователей кодов, мультиплексоров, демультиплексоров сумматоров, методы анализа и синтеза цифровых устройств комбинационного типа.

В пособии приводятся теоретические сведения, математические и логические основы, используемые при решении задач синтеза и анализа комбинационных цифровых схем; представлен большой объем заданий по анализу, проектированию и применению цифровых устройств различной сложности и назначения.

Содержание пособия основано на материале курса лекций по дисциплинам «Электроника и микропроцессорная техника», «Математические основы цифровой техники», «Анализ и синтез цифровых схем в изделиях микроэлектроники» и «Информационная электроника».

Учебно-методическое пособие предназначено для развития навыков решения задач по логическому проектированию, анализу и построению цифровых схем электронных устройств, рекомендуется использовать его как для аудиторной, так и самостоятельной работы; направлено на формирование следующих компетенций:

– способность учитывать современные тенденции развития электроники, измерительной и вычислительной техники, информационных технологий в своей профессиональной деятельности;

– способность осуществлять поиск, хранение, обработку и анализ информации из различных источников и баз данных, представлять ее в требуемом формате с использованием информационных, компьютерных и сетевых технологий;

– готовность выполнять расчет и проектирование электронных приборов, схем и устройств различного функционального назначения в соответствии с техническим заданием с использованием средств автоматизации проектирования.

В результате освоения дисциплин обучающийся должен:

– знать основы цифровой электроники, методы анализа и синтеза цифровых схем, элементную базу цифровых устройств;

– уметь проводить анализ, расчет и проектирование цифровых устройств на основе специализированных интегральных микросхем;

– владеть методами и навыками расчета и решения задач по проектированию цифровых устройств на базе интегральных микросхем.

1. ЦИФРОВОЕ КОДИРОВАНИЕ ИНФОРМАЦИИ. ПРЕДСТАВЛЕНИЕ ЧИСЛОВОЙ ИНФОРМАЦИИ С ПОМОЩЬЮ СИСТЕМ СЧИСЛЕНИЯ

Обозначение различной цифровой информации соответствующими символами называется кодированием, а состав символов для данной информации – алфавитом этого кода.

Система счисления – это совокупность правил для обозначения и наименования чисел.

В позиционных системах счисления значимость конкретной цифры определяется ее местоположением в записи числа.

Основание позиционной системы счисления – это количество различных знаков или символов, используемых для изображения цифр в данной системе.

Запись чисел в каждой из систем счисления с основанием q означает сокращенную запись выражения

$$a_{n-1}q^{n-1} + a_{n-2}q^{n-2} + \ldots + a_1q^1 + a_0q^0 + a_{-1}q^{-1} + \ldots + a_{-m}q^{-m}, \qquad (1.1)$$

где n и m – число целых и дробных разрядов, соответственно; $a_i, i = \overline{n-1, m}$ – цифры системы счисления. Номер позиции цифры a_i называют разрядом числа. Разряды с положительными степенями q образуют целую часть числа, а с отрицательными – дробную. Цифры a_n и a_{-m} являются старшим и младшим разрядами числа соответственно.

В десятичной системе счисления любое число может быть представлено через степени числа 10 (основание системы).

Для записи чисел в двоичной системе счисления используются только две цифры – 0 и 1, в восьмеричной системе – восемь цифр от 0 до 7 включительно. Для записи чисел в шестнадцатеричной системе необходимо уже шестнадцать символов, используемых как цифры. В качестве первых десяти цифр используются те же, что и в десятичной системе. Для обозначения остальных шести (в десятичной они соответствуют числам от 10 до 15) – буквы латинского алфавита с A по F включительно. Четверичная система счисления – система счисления с целочисленным основанием равным 4, в которой цифры от 0 до 3 включительно, а базу системы составляют степени числа 4.

Для перевода чисел из любой системы счисления в десятичную можно использовать запись этого числа в виде полинома (1.1). Выполнив сложение, получаем искомое десятичное число.

Пример 1.1. Перевести восьмеричное число 745 в десятичную систему счисления.

Решение: $745_8 = 7\cdot8^2 + 4\cdot8^1 + 5\cdot8^0 = 520$.

Пример 1.2. Перевести число $F17B_{16}$ из шестнадцатеричной системы в десятичную.

Решение: $F17B_{16} = 15\cdot16^3 + 1\cdot16^2 + 7\cdot16^1 + 11\cdot16^0 = 618119_{10}$.

Переход от системы счисления с меньшим основанием к системе с большим основанием осуществляется при помощи выражения (1.1), которое справедливо как для целой, так и дробной частей числа.

Пример 1.3. Перевести число $110110{,}101_2$ из двоичной системы счисления в десятичную.

Решение:

$$110110{,}101_2 = 1\cdot2^5 + 1\cdot2^4 + 1\cdot2^2 + 1\cdot2^1 + 0\cdot2^0 + 2^{-1} + 0^{-2} + 2^{-3} = 54{,}625_{10}.$$

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

1. Целая часть исходного числа делится на основание новой системы счисления.

2. Дробная часть исходного числа умножается на основание новой системы счисления.

Пример 1.4. Представленное в десятичной системе счисления число $X_1 = 327{,}125$ перевести в двоичную систему счисления.

Решение. 1. Преобразуем целую часть:

$$\begin{array}{rl}
3\ 2\ 7 & \big|\,2 \\
\underline{3\ 2\ 6}\ 1\ 6\ 3 & \big|\,2 \\
a_0 = 1\ \underline{1\ 6}\ 2\ 8\ 1 & \big|\,2 \\
a_1 = 1\ \underline{8\ 0}\ 4\ 0 & \big|\,2 \\
a_2 = 1\ \underline{4\ 0}\ 2\ 0 & \big|\,2 \\
a_3 = 0\ \underline{2\ 0}\ 1\ 0 & \big|\,2 \\
a_4 = 0\ \underline{1\ 0}\ 5 & \big|\,2 \\
a_5 = 0\ \underline{4}\ 2 & \big|\,2 \\
a_6 = 1\ \underline{2}\ 1 & \big|\,2 \\
a_7 = 0\ \underline{0}\ 0 & \\
a_8 = 1 &
\end{array}$$

Целая часть двоичного числа X_2 записывается в виде остатков деления $a_i \left(i=\overline{0,8}\right)$, начиная с последнего. Таким образом $327_{10} = 101000111_2$.

2. Преобразуем дробную часть:

$$0{,}125 \cdot 2 = 0{,}25 + 0 \ (a_{-2} = 0)$$

$$0{,}25 \cdot 2 = 0{,}5 + 0 \ (a_{-1} = 0)$$

$$0{,}5 \cdot 2 = 1 + 0 \ (a_{-3} = 1)$$

В произведениях выделяется целая часть, которая принимается в качестве значения первого после запятой разряда числа в двоичной системе счисления, т.е. $0{,}125_{10} = 0{,}001_2$.

Ответ: $327{,}125_{10} = 101000111{,}001_2$.

Взаимосвязь двоичной, восьмеричной и шестнадцатеричной систем счисления приведена в табл. 1.

Таблица 1.1

Таблица соответствия систем счисления

Десятичная	Двоичная	Восьмеричная	Шестнадцатеричная
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	*A*
11	1011	13	*B*
12	1100	14	*C*
13	1101	15	*D*
14	1110	16	*E*
15	1111	17	*F*
16	10000	20	10
17	10001	21	11
…	…	…	…
26	11010	32	1*A*

Так как основания восьмеричной и шестнадцатеричной систем счисления являются степенями двойки, то перевод чисел из этих систем счисления в двоичную и наоборот прост и основан на методах триад и тетрад. Число делится на триады (тетрады) вправо и влево от десятичной точки. Если крайние триады (тетрады) оказались неполными, они дополняются нулями.

Пример 1.5. Перевести восьмеричное число 4011_8 в двоичную систему счисления, используя табл. 1.1.

Р е ш е н и е: $4011_8 = 100.000.001.001_2$ (точки отделяют триады).

Пример 1.6. Перевести число $11101001000,11010010_2$ в шестнадцатеричную систему счисления, используя табл. 1.1.

Р е ш е н и е: $0111.0100.1000,1101.0010_2 = 748,D2_{16}$.

Двоично-десятичные коды

Неудобство использования двоичного кода – в громоздкости записи чисел. Для устранения этого неудобства применяется двоично-десятичный код (ДДК) или смешанная система счисления (ССС). В этой системе цифры десятичной системы счисления представлены в виде двоичного кода. При двоично-десятичном кодировании каждая десятичная цифра заменяется тетрадой (четверкой) двоичных цифр, а сами тетрады записываются последовательно в соответствии с порядком следования десятичных цифр. При обратном преобразовании двоично-десятичного кода в десятичный исходный код разбивается на тетрады вправо и влево от запятой, которые затем заменяются десятичными цифрами. При помощи четырех бит можно закодировать шестнадцать цифр. Лишние комбинации в двоично-десятичном коде являются избыточными.

Например, число 311_{10} будет записано в двоичной системе счисления в двоичном коде как $1\,0011\,0111_2$, а в десятичной системе счисления в двоично-десятичном коде как $0011\ 0001\ 0001_{10}$.

Чаще всего используются взвешенные двоично-десятичные коды, позволяющие достаточно просто переводить десятичные цифры $(X)_{10}$ в двоичные $(X)_2$ по формуле

$$(X)_{10} = a_3x_3 + a_2x_2 + a_1x_1 + a_0x_0, \qquad (1.2)$$

где символы $a_3 \ldots a_0$ являются постоянными весовыми коэффициентами соответствующего кода, а символы $x_3 \ldots x_0$ – двоичные цифры 1 или 0. Кодовые комбинации, соответствующие тем или иным символам в различных кодах, легко определяются из выражения (1.2).

Так, для кода «8421» веса разрядов следующие: $a_3 = 8$, $a_2 = 4$, $a_1 = 2$, $a_0 = 1$. Цифру 5 в двоично-десятичном коде «8421» можно получить, подставив соответствующие веса в двоичное представление:

$$0101: 0 \cdot 8 + 1 \cdot 4 + 0 \cdot 2 + 1 \cdot 1 = 5.$$

Аналогично формируются значения для цифр в других двоично-десятичных кодах с весами, по названию которых они и идентифицируются, таких как 5121, 6423 и (8421)+3, представленных в табл. 1.2.

Таблица 1.2

Двоично-десятичные коды

Десятичные цифры	Коды					
	8421	5121	Невзвешенный	6423	С избытком 3	5321
0	0000	0000	0000	0000	0011	0000
1	0001	0001	0111	0101	0100	0011
2	0010	0010	0110	0010	0101	0010
3	0011	0011	0101	0111	0110	0100
4	0100	0111	0100	0100	0111	1001
5	0101	1000	1011	1011	1000	0110
6	0110	1100	1010	1000	1001	1011
7	0111	1101	1001	1101	1010	1101
8	1000	1110	1000	1010	1011	1100
9	1001	1111	1111	1111	1100	1111

Например, число 576 различными двоично-десятичными кодами будет записано следующим образом:

в коде 8421 $576 \rightarrow 010101110110$;
в коде 2421 $576 \rightarrow 101111011100$;
в коде 4221 $576 \rightarrow 100111011010$;
в коде 5121 $576 \rightarrow 100010101001$.

Особенностью взвешенных двоично-десятичных кодов (ДДК), имеющих только положительные веса, кроме кода группы «8421», является отсутствие однозначного представления десятичных цифр. Это означает, что некоторые десятичные цифры могут быть записаны несколькими комбинациями двоичных цифр. Например, в ДДК «4421» цифру 4 можно представить как 1000 и как 0100, цифру 5 – как 1001 и как 0101, цифру 6 – как 1010 и как 0110, цифру 7 – как 1011 и как 0111.

Коды с весовыми коэффициентами «2421» называются самодополняющимися, так как инвертированный код, полученный заменой 0 на 1 и 1 на 0 в каждом разряде, всегда дополняет основной до числа 9 (1111). Например, если инвертировать комбинацию 0100 (цифра 4 в коде «2421»), то получится комбинация 1011, соответствующая цифре 5. При этом сложение прямой и инвертированной комбинации 0100 и 1011 дает в сумме комбинацию 1111, что соответствует цифре 9.

Двоичная арифметика

Правила выполнения арифметических действий над двоичными числами определяются арифметическими действиями над одноразрядными двоичными числами и представлены в табл. 1.3.

Таблица 1.3

Арифметические действия над двоичными числами

Сложение	Вычитание	Умножение
0 + 0 = 0	0 – 0 = 0	0 · 0 = 0
0 + 1 = 1	1 – 0 = 0	1 · 0 = 0
1 + 0 = 1	1 – 1 = 0	0 · 1 = 0
1 + 1 = 10 ↑↵ перенос в старший разряд	10 – 1 = 1 ↳↑ заём из старшего разряда	1 · 1 = 1

Правила выполнения арифметических действий во всех позиционных системах счисления аналогичны.

При сложении двух единиц в двоичном коде происходит переполнение разряда и производится перенос в старший разряд. Переполнение разряда наступает тогда, когда значение числа в нем становится равным или большим основания. Для двоичной системы счисления это число равно двум. То есть при сложении необходимо помнить, что 1 + 1 дают нуль в младшем разряде и единицу переноса в старшем разряде.

При выполнении операции вычитания всегда из большего числа вычитается меньшее и перед результатом ставится соответствующий знак (плюс или минус). Между тем, вычитание из меньшего числа (0) большего (1) сопровождается заемом из старшего разряда.

Пример 1.7. Сложить два числа в десятичном и двоичном представлении (формат – 1 байт).

Р е ш е н и е .

Перенос (единицы)	1 1	1 1 1 1 1 1 1
Слагаемое 1	0 9 9 $_{(10)}$	0 1 1 0 0 0 1 1 $_{(2)}$
Слагаемое 2	0 9 5 $_{(10)}$	0 1 0 1 1 1 1 1 $_{(2)}$
Сумма	1 9 4 $_{(10)}$	1 1 0 0 0 0 1 0 $_{(2)}$

Пример 1.8. Найти разность двоичных чисел 1111 и 11.

Р е ш е н и е . Так как число 1111 больше чем 11, то выполним следующее действие 1111 – 11 = 1100. Здесь разность будет положительным числом, так как уменьшаемое больше вычитаемого.

Рассмотрим случай, когда вычитаемое больше уменьшаемого.

Пример 1.9. Найти разность двоичных чисел 1010 и 11111.

Решение. Так как 1010 меньше чем 11111, то разность можно записать в виде:

$$1010-11111=-(11111-1010)=-10101$$

и знак разности изменен на минус.

Как и десятичные числа, двоичные можно вычитать столбиком. При этом следует помнить, что если при вычитании столбиком цифра какого-либо разряда уменьшаемого меньше цифры того же разряда вычитаемого, то производится заем единицы из следующего разряда. Так как каждая единица следующего разряда равна основанию системы счисления, то есть двум (в двоичной системе счисления), то в предыдущий разряд при займе приходит две единицы.

Пример 1.10. Выполнить вычитание двоичных чисел 1011 и 111 столбиком.

Решение:

$$\begin{array}{r} 1\ 0\ 1\ 1 \\ -\quad 1\ 1\ 1 \\ \hline 1\ 0\ 0 \end{array}$$

Так как в третьем разряде уменьшаемого 0, а вычитаемого 1, то происходит заем единицы из четвертого разряда уменьшаемого. Таким образом, в третьем разряде получаем «2 – 1» и в результате «1».

Пример 1.11. Произвести вычитание чисел $109_{(10)}$ и $49_{(10)}$ в десятичном и двоичном представлении.

Решение.

Заем (единица)	1	0 1 1 0 0 0 0 0
Уменьшаемое	$1\ 0\ 9_{(10)}$	$0\ 1\ 1\ 0\ 1\ 1\ 0\ 1_{(2)}$
Вычитаемое	$0\ 4\ 9_{(10)}$	$0\ 0\ 1\ 1\ 0\ 0\ 0\ 1_{(2)}$
Разность	$0\ 6\ 0_{(10)}$	$0\ 0\ 1\ 1\ 1\ 1\ 0\ 0_{(2)}$

Прямой, обратный и дополнительный коды

Для выполнения арифметических операций используют специальные коды представления чисел, которые позволяют свести операцию вычитания чисел к арифметическому сложению этих кодов. Различают прямой, обратный и дополнительный коды. Прямой код используется для представления отрицательных чисел в памяти компьютера, а также при выполнении операций

умножения и деления. Обратный и дополнительный коды применяются для выполнения операции вычитания, которую заменяют операцией сложения чисел с разными знаками: $a - b = a + (-b)$.

При записи кода знаковый разряд числа отделяют запятой от других разрядов. Если формат числа не указан, считают, что число 8-разрядное (байт).

Под прямым кодом двоичного числа понимают запись самого числа. Значение знакового разряда для положительных чисел определяют равным нулю (0), для отрицательных чисел – единице (1). Например, если для записи кода используется байт, то крайний левый разряд в прямом коде отведен под знак числа, остальные разряды – под само число:

Число	Прямой код
+1101	0,0001101
– 1101	1,0001101

Число располагается в разрядной сетке так, чтобы цифра младшего разряда числа занимала крайнюю правую ячейку:

знаковый разряд $\rightarrow$ 0, 0 0 0 1 1 0 1.

Обратный код целого положительного числа совпадает с его прямым кодом. Для отрицательного числа обратный код строится заменой каждого незнакового байта его представления в прямом коде на противоположный (заменой 1 на 0 и наоборот), знаковый разряд при этом не изменяется:

Число	Коды		Замечание
	Прямой	Обратный	
+11011	0,0011011	0,0011011	Число положительное, обратный и прямой коды совпадают
–11011	1,0011011	1,1100100	Число отрицательное, каждый байт, кроме знакового, изменен на противоположный

Дополнительный код положительного числа совпадает с его прямым кодом. Для отрицательного числа дополнительный код образуется путем получения обратного кода и добавлением к младшему разряду единицы:

Число	Коды		
	Прямой	Обратный	Дополнительный
+ 1110	0,0001110	0,0001110	0,0001110
–1110	1,0001110	1,1110001	1,1110010

При сложении чисел в знаковом разряде могут появиться две цифры, вторую единицу от запятой называют единицей переноса. В дополнительном коде возникающая при сложении чисел единица переноса в знаковом разряде отбрасывается, а в обратном коде – прибавляется к младшему разряду суммы кодов.

Если результат арифметических действий является кодом отрицательного числа, необходимо преобразовать его в прямой код. При этом обратный код преобразуется в прямой заменой цифр во всех разрядах на противоположные, кроме знакового. Дополнительный код преобразуется в прямой так же, как и обратный, с последующим прибавлением единицы к младшему разряду.

<u>Пример 1.12.</u> Сложить числа $X = 1111$ и $Y = -101$ в обратном и дополнительном кодах.

Р е ш е н и е. 1. Сложим числа, используя правила двоичной арифметики:

$$\begin{array}{rr} X & 1\,1\,1\,1 \\ Y & -\;\;1\,0\,1 \\ \hline X+Y= & 1\,0\,1\,0 \end{array}$$

2. Выполним сложение, используя коды:

Обратный код	Дополнительный код
$X_{\text{обр}} = 0,\ 0\ 0\ 0\ 1\ 1\ 1\ 1$	$X_{\text{доп}} = 0,0\,0\,0\,1\,1\,1\,1$
$Y_{\text{обр}} = 1,\ 1\ 1\ 1\ 1\ 0\ 1\ 0$	$Y_{\text{доп}} = 1,1\,1\,1\,1\,0\,1\,1$
$\mathbf{1}\,0,\ 0\ 0\ 0\ 1\ 0\ 0\ 1$	$\mathbf{1}\,0,0\,0\,0\,1\,0\,1\,0$
$+1$	отбрасывается
$(X+Y)_{\text{обр}} = 0,0001010$	$(X+Y)_{\text{доп}} = 0,0001010$

Так как результат сложения является кодом положительного числа, знаку плюс соответствует 0 в знаковом разряде, получаем

$$(X+Y)_{\text{обр}} = (X+Y)_{\text{доп}} = (X+Y)_{\text{пр}}.$$

<u>Пример 1.13.</u> Сложить числа $X = -101$ и $Y = -111$ в обратном и дополнительном кодах.

Р е ш е н и е. 1. Сложим числа, используя правила двоичной арифметики:

$$\begin{array}{rr} X & -1\,0\,1 \\ Y & -1\,1\,1 \\ \hline X+Y= & -11\,0\,0 \end{array}$$

2. Выполним сложение, используя коды:

Обратный код	Дополнительный код
$X_{\text{обр}} = 1,\ 1\ 1\ 1\ 1\ 0\ 1\ 0$	$X_{\text{доп}} = 1,1\ 1\ 1\ 1\ 0\ 1\ 1$
$Y_{\text{обр}} = 1,\ 1\ 1\ 1\ 1\ 0\ 0\ 0$	$Y_{\text{доп}} = 1,1\ 1\ 1\ 1\ 0\ 0\ 1$
$\mathbf{1}\ 1,\ 1\ 1\ 1\ 0\ 0\ 1\ 0$	$\mathbf{1}\ 1,1\ 1\ 1\ 0\ 1\ 0\ 0$
$+1$	отбрасывается
$(X+Y)_{\text{обр}} = 1{,}1110010$	$(X+Y)_{\text{доп}} = 1{,}1110100$

Так как сумма является кодом отрицательного числа, знаку минус соответствует 1 в знаковом разряде, то необходимо перевести результаты в прямой код:

$$(X+Y)_{\text{обр}} = 1{,}1110011 \Rightarrow (X+Y)_{\text{пр}} = 1{,}0001100;$$

$$(X+Y)_{\text{доп}} = 1{,}1110100 \Rightarrow (X+Y)_{\text{пр}} = 1{,}0001011 + 0{,}0000001 = 1{,}0001100.$$

Получили $X + Y = -1100$, результат совпадает с суммой, полученной по правилам двоичной арифметики.

Код Грея

Характерной особенностью кода Грея является изменение только одной позиции при переходе от одного кодовой комбинации к другой. Этот код иногда называют рефлексным (отраженным), его применяют для преобразования линейных и угловых перемещений в кодовые комбинации. Если при таком преобразовании используется обычный двоичный код, то некоторые расположенные рядом кодовые комбинации различаются в нескольких разрядах. Например, комбинации 0111 (цифра 7) и 1000 (цифра 8) различаются во всех разрядах.

Вес разрядов кода Грея определяется выражением:

$$q_i = 2^i - 1, \quad i = \overline{1,n}.$$

Таким образом, веса разрядов, начиная с младшего, будут следующими: 1, 3, 7, 15, 31… и т.д. Для того чтобы прочесть число в коде Грея, под каждым разрядом записывают его десятичный эквивалент, старший значащий разряд берется со знаком плюс, а перед остальными значащими разрядами знаки чередуются. Например, перевод комбинации из кода Грея 101111 и 010011 в десятичный код производится следующим образом:

$$\begin{array}{cccccc} 1 & 0 & 1 & 1 & 1 & 1 \\ 63 & 31 & 15 & 7 & 3 & 1 \end{array} = 63 - 15 + 7 - 3 + 1 = 53;$$

$$\begin{array}{cccccc} 0 & 1 & 0 & 0 & 1 & 1 \\ 63 & 31 & 15 & 7 & 3 & 1 \end{array} = 31 - 3 + 1 = 29.$$

Код Грея относится к неарифметическим кодам. Поэтому перед обработкой информации производят преобразование в двоичный код.

Существует несколько алгоритмов перевода кода Грея в двоичный код и обратно. В общем виде число в двоичном коде можно записать как $a_n a_{n-1} \ldots a_i \ldots a_1$, а в коде Грея – $b_n b_{n-1} \ldots b_i \ldots b_1$. Преобразование из кода Грея в двоичный код можно осуществить по следующему правилу:

1) цифра старшего разряда записывается без изменений, т.е. $a_n = b_n$;

2) значение каждого последующего разряда двоичного числа находят путем сложения по модулю 2 этого же разряда в коде Грея с предыдущими, т.е. $a_{n-1} = b_n \oplus b_{n-1}$. В общем случае можно записать:

$$a_i = \sum_{i=n}^{i} b_i \left(\mathrm{mod}\, 2\right).$$

Комбинации кода Грея, соответствующие десятичным числам от 0 до 15, приведены в табл. 1.4.

Таблица 1.4

Код Грея

N_{10}	Код Грея	N_{10}	Код Грея	N_{10}	Код Грея	N_{10}	Код Грея
0	0000	4	0110	8	1100	12	1010
1	0001	5	0111	9	1101	13	1011
2	0011	6	0101	10	1111	14	1001
3	0010	7	0100	11	1110	15	1000

Формируются комбинации кода Грея из обычного двоичного кода путем суммирования по модулю 2 исходной кодовой комбинации с такой же комбинацией, сдвинутой вправо на 1 разряд. Младший разряд в сдвинутой комбинации при этом отбрасывается.

Например: 1010 (код Грея)

1100 (двоичный код).

Правило перехода от кода Грея к натуральному двоичному коду: если слева от данной цифры находится четное число единиц, то цифра сохраняется, в противном случае цифра меняется.

Пример 1.14. Перевести число 11010110 в коде Грея в двоичный код.

Решение:

11010110 – код Грея

10011011 – двоичный код

Проверка:

$$\begin{array}{r} 10011011 \\ \oplus\ \ 1001101 \\ \hline 11010110 \end{array}$$ – код Грея.

Задания для самостоятельной работы

1. Переведите в десятичную систему счисления: $1001{,}1_2$; $7764{,}1_8$; $67{,}5_8$; $3AF_{16}$; 110110_2; $234{,}6_8$.

2. Переведите в восьмеричную систему счисления: 10101011111101_2; 1011.0101_2; 241_{10}; $123{,}5625_{10}$; 11110000010110_2; $0{,}5625_{10}$; $111100001{,}0111_2$; FEA_{16}; 10101011111101_2.

3. Переведите в шестнадцатеричную систему счисления: 571_8; 7467_{10}; 6635_8; 10101011111101_2; $11101{,}01111_2$; 3627_{10}; 11100010_2; 11110000010110_2; $1101\ 1001\ 1101_2$.

4. Переведите в двоичную систему счисления: $0{,}5625_{10}$; $26{,}25_{10}$; $EE8_{16}$; 6521_8; 1987_{10}; 245_{10}; $173{,}5625_{10}$; 0,1275; $373{,}25_8$; $3DF{,}CA_{16}$; $67{,}125_{10}$.

5. Записать десятичное число в двоично-десятичной системе счисления: 572,38; 8072; 3691; 167; 334.

6. Запишите двоично-десятичное число в десятичной системе счисления: $10010{,}010101_{2\text{-}10}$; 0011 1001 0010 0111; 0100 1000 0101 0110; 0001 0010 0101 1000; 0000 0101 1000 1001.

7. Представьте числа 576, 673, 817 различными двоично-десятичными кодами: 8421, 2421, 4221 и 5121.

8. Выполните сложение чисел A и B в двоичной системе счисления: $A = 1100111{,}011_2$ и $B = 10011{,}111_2$; $A = 53_{10}$ и $B = 14_{10}$; $A = A52B_{16}$ и $B = 340F_{16}$**;** $A = 1011_2$ и $B = -11001_2$**.**

9. Запишите в коде 5211 числа: 7, 9, 15.

10. Запишите следующие числа в прямом, обратном и дополнительном кодах: 1101011; –101011; –101101; –1100111; $+111001_2$; -110011011_2.

11. В прямом и обратном кодах вычислите выражения: $-3_{10} - 2_{10}$; $7_{10} - 3_{10}$.

12. Найдите прямой, обратный и дополнительный коды в однобайтовом представлении для чисел: -56_{10}; $+28_{10}$ и -28_{10}; -5_{10}; -44_{10}.

13. В дополнительном коде вычислите выражения: 58 – 23; 26 – 34; 33 – 11.

14. Представьте в обратном коде дробные двоичные числа: +0,10012; – 0,01101.

15. Задан дополнительный код числа в однобайтовом представлении: 1,1011100. Найдите число в десятичной системе счисления.

16. Представьте в дополнительном коде дробные двоичные числа: +0,10012; –0,011012.

17. Переведите числа X и Y в прямой, обратный и дополнительный коды. Выполните сложение в обратном и дополнительном кодах. Результат переведите в прямой код. Проверьте полученный результат, используя правила двоичной арифметики.

а) $X = –11010$, $Y = 100111$;

б) $X = –11101$, $Y = –10011$;

в) $X = 111010$, $Y = –101111$;

г) $X = –101110$, $Y = –11101$;

д) $X = 1101011$, $Y = –1001110$;

е) $X = –11011$, $Y = –10111$.

18. Преобразуйте двоичные числа в код Грея: 110011; 111011; 1011001; 1101; 1001; 1111.

19. Преобразуйте числа из кода Грея в натуральный двоичный код: 10111; 100011; 1000110; 100011, 111011. Провести проверку.

2. ФОРМЫ ПРЕДСТАВЛЕНИЯ ЛОГИЧЕСКИХ ФУНКЦИЙ

Для описания алгоритмов работы цифровых устройств необходим соответствующий математический аппарат – алгебра логики. С помощью алгебры логики записывают, упрощают, вычисляют и преобразовывают логические выражения.

Логическая функция (ЛФ) – это функция логических переменных, которая может принимать только два значения: 0 или 1. Сама логическая переменная (аргумент логической функции) тоже может принимать только два значения: 0 или 1.

Значение логической функции *n* переменных определяется или задается для каждого набора (сочетания) двоичных переменных. Количество возможных различных наборов, которые могут быть составлены из *n* аргументов, очевидно, равно 2^n. При этом, поскольку сама функция на каждом наборе может принимать значение 0 или 1, общее число возможных функций от *n* переменных равно 2^{2^n}.

Таким образом, количество состояний (значений), которые могут принимать как аргументы, так и функции, равно двум. Для этих состояний в алгебре логики определяются отношение эквивалентности, обозначаемое символом равенства и три основные операции:

а) ИЛИ – логическое сложение (дизъюнкция), обозначается символами «+» или «$\vee$»;

б) И – логическое умножение (конъюнкция), обозначается символами «$\cdot$», «$\wedge$» или «&»;

в) НЕ – логическое отрицание (инверсия), обозначается чертой над переменной $\overline{x}$.

Остальные операции алгебры логики выражаются через три вышеперечисленные. Они были введены для сокращения и упрощения записи логических выражений.

Постулативно полагается, что при выполнении перечисленных операций отношения эквивалентности имеют вид:

$$\begin{gathered} 0+0=0, \quad 0+1=1, \quad 1+0=1, \quad 1+1=1; \\ 0\cdot 0=0, \quad 0\cdot 1=0, \quad 1\cdot 0=0, \quad 1\cdot 1=1; \quad \overline{0}=1, \quad \overline{1}=0. \end{gathered} \tag{2.1}$$

На основании постулатов (2.1) можно вывести соотношения (законы) алгебры логики. Логические выражения можно преобразовать из одного вида в другой, применяя законы и тождества алгебры логики (табл. 2.1).

Таблица 2.1

Законы и тождества алгебры логики

Законы алгебры логики		Тождества (аксимомы) алгебры логики
Закон коммутативности (переместительный)	$A \vee B = B \vee A$ $A \wedge B = B \wedge A$	$1 \vee A = 1$ $0 \vee A = A$ $A \vee A = A$ $A \vee \overline{A} = 1$ $1 \wedge A = A$ $0 \wedge A = 0$ $A \wedge A = A$ $A \wedge \overline{A} = 0$ $\overline{\overline{A}} = A$
Закон ассоциативности (сочетательный)	$A \vee B \vee C = A \vee (B \vee C)$ $A \wedge B \wedge C = A \wedge (B \wedge C)$	
Закон дистрибутивности (распределительный)	$A \wedge (B \vee C) = A \wedge B \vee A \wedge C$ $A \vee B \wedge C = (A \vee B) \wedge (A \vee C)$	
Закон дуальности (теоремы де Моргана)	$\overline{A \vee B} = \overline{A} \wedge \overline{B}$ $\overline{A \wedge B} = \overline{A} \vee \overline{B}$	
Закон поглощения	$A \vee A \wedge B = A$ $A \wedge (A \vee B) = A$	

Дополнительно рассмотрим ряд соотношений, не приведенных в табл. 2.1.

Законы склеивания:

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

$$\left(A + B\right) \cdot \left(A + \overline{B}\right) = A.$$

Законы обобщенного склеивания:

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

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

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

$$A \cdot \left(\overline{A} + B\right) = A \cdot B.$$

Порядок выполнения логических операций в сложном логическом выражении следующий: инверсия; конъюнкция; дизъюнкция; исключающее ИЛИ; импликация; эквивалентность.

Для изменения указанного порядка выполнения операций используются скобки.

Полный набор логических функций, зависящих от двух переменных, представлен в табл. 2.2.

Таблица 2.2

Полный набор логических функций

A	0	0	1	1	Алгебраическая форма записи	Название функции
B	0	1	0	1		
F_0	0	0	0	0	$F_0 = 0$	Постоянный ноль
F_1	0	0	0	1	$F_1 = A \cdot B = A \wedge B$	Конъюнкция (И)
F_2	0	0	1	0	$F_2 = A \cdot \overline{B}$	Запрет
F_3	0	0	1	1	$F_3 = A$	Тождественность A
F_4	0	1	0	0	$F_4 = \overline{A} \cdot B$	Запрет
F_5	0	1	0	1	$F_5 = B$	Тождественность B
F_6	0	1	1	0	$F_6 = A \oplus B = A\overline{B} + \overline{A}B$	Неравнозначность (сложение по модулю 2)
F_7	0	1	1	1	$F_7 = A + B = A \vee B$	Дизъюнкция (ИЛИ)
F_8	1	0	0	0	$F_8 = A \downarrow B = \overline{A + B}$	Стрелка пирса (ИЛИ – НЕ)
F_9	1	0	0	1	$F_9 = A \equiv B = AB + \overline{A}\overline{B}$	Равнозначность (эквивалентность)
F_{10}	1	0	1	0	$F_{10} = \overline{B}$	Инверсия B
F_{11}	1	0	1	1	$F_{11} = A + \overline{B} = B \rightarrow A$	Импликация от B к A
F_{12}	1	1	0	0	$F_{12} = \overline{A}$	Инверсия A
F_{13}	1	1	0	1	$F_{13} = \overline{A} + B = A \rightarrow B$	Импликация от A к B
F_{14}	1	1	1	0	$F_{14} = A / B = \overline{A \wedge B}$	Штрих Шеффера (И – НЕ)
F_{15}	1	1	1	1	$F_{15} = 1$	Постоянная 1

Логические функции и их свойства

Сложение по модулю два

Логическая функция сложения по модулю два выражается через дизъюнкцию, конъюнкцию и отрицание:

$$x_1 \oplus x_2 = \overline{x_1 \equiv x_2} = \overline{(\overline{x}_1 \vee x_2) \cdot (x_1 \vee \overline{x}_2)} = \overline{(\overline{x}_1 \vee x_2)} \vee \overline{(x_1 \vee \overline{x}_2)} =$$
$$= x_1 \cdot \overline{x}_2 \vee \overline{x}_1 \cdot x_2 = (x_1 \vee x_2) \cdot (\overline{x}_1 \vee \overline{x}_2) \qquad (2.1)$$

Функция обладает следующими свойствами:

1) коммутативности (переместительный закон):

$$x_1 \oplus x_2 = x_2 \oplus x_1;$$

2) ассоциативности (сочетательный закон):

$$x_1 \oplus (x_2 \oplus x_3) = (x_1 \oplus x_2) \oplus x_3;$$

3) дистрибутивности (распределительный закон):

$$x_1 \cdot (x_2 \oplus x_3) = x_1 \cdot x_2 \oplus x_1 \cdot x_3.$$

Для нее справедливы аксиомы:

$$x \oplus x = 0; \qquad x \oplus 1 = \bar{x};$$
$$x \oplus 0 = x; \qquad x \oplus \bar{x} = 1.$$

На основании аксиом, тождеств и свойств можно вывести правила перевода функций И, ИЛИ, НЕ через функцию сложения по модулю два и наоборот.

$$\bar{x} = x \oplus 1;$$

$$x_1 \vee x_2 = x_1 \oplus x_2 \oplus x_1 \cdot x_2;$$

$$x_1 \cdot x_2 = (x_1 \oplus x_2) \oplus (x_1 \vee x_2).$$

Функция импликации

Чаще всего функция импликации выражается через дизъюнкцию и отрицание:

$$x_1 \to x_2 = \bar{x}_1 \vee x_2. \tag{2.2}$$

Для функции импликации справедливы следующие законы:

$$x \to x = 1; \qquad x \to 1 = 1; \qquad 0 \to x = 1;$$
$$x \to \bar{x} = \bar{x}; \qquad x \to 0 = \bar{x}; \qquad 1 \to \bar{x} = x.$$

Из этих законов следует, что функция импликации обладает только свойством коммутативности (переместительный закон) в измененном виде:

$$x_1 \to x_2 = \bar{x}_2 \to \bar{x}_1.$$

Свойство ассоциативности (сочетательный закон) для этой функции не справедливо, т.е. $x_1 \to (x_2 \to x_3) \neq (x_1 \to x_2) \to x_3$. Действительно:

$$x_1 \to (x_2 \to x_3) = \bar{x}_1 \vee (x_2 \to x_3) = \bar{x}_1 \vee \bar{x}_2 \vee x_3 = \overline{x_1 \cdot x_2} \vee x_3;$$
$$(x_1 \to x_2) \to x_3 = \overline{(x_1 \to x_2)} \vee x_3 = \overline{(\bar{x}_1 \vee x_2)} \vee x_3 = x_1 \cdot \bar{x}_2 \vee x_3.$$

Функции НЕ, ИЛИ, И выражаются через импликацию следующим образом:

$$\overline{x} = x \to 0;$$

$$x_1 \vee x_2 = \overline{x}_1 \to x_2 = (x_1 \to 0) \to x_2;$$

$$x_1 \cdot x_2 = \overline{x_1 \to \overline{x}_2} = \overline{x_1 \to (x_2 \to 0)} = [x_1 \to (x_2 \to 0)].$$

Функция Шеффера

Функция Шеффера выражается через конъюнкцию и отрицание:

$$x_1 | x_2 = \overline{x_1 \cdot x_2}. \qquad (2.3)$$

Для функции Шеффера (2.3) справедливы следующие законы:

$$x|x = \overline{x}; \qquad x|\overline{x} = 1; \qquad x|0 = 1;$$

$$x|1 = \overline{x}; \qquad \overline{x}|0 = 1; \qquad \overline{x}|1 = x.$$

Для двух переменных функции Шеффера справедливо только свойство коммутативности

$$x_1 | x_2 = x_2 | x_1.$$

Однако для трех и более переменных функции Шеффера имеет место

$$x_1 | (x_2 | x_3) \neq (x_1 | x_2) | x_3,$$

т.е. свойство ассоциативности для функции Шеффера несправедливо. Действительно, рассмотрим функции трех переменных: $(x_1|x_3)|x_2$ и $(x_1|x_2)|x_3$. Так как

$$(x_1|x_3)|x_2 = \overline{\overline{x_1 \cdot x_3} \cdot x_2} = x_1 \cdot x_3 \vee \overline{x}_2,$$

а

$$(x_1|x_2)|x_3 = \overline{\overline{x_1 \cdot x_2} \cdot x_3} = x_1 \cdot x_2 \vee x_3,$$

получаются совершенно разные функции. Следовательно, свойство ассоциативности несправедливо как для трех, так и для *n* переменных.

Свойство дистрибутивности также несправедливо для функции Шеффера, т.е. $x_1 | (x_2 | x_3) \neq x_1 \cdot x_2 | x_1 \cdot x_3$. Действительно, $x_1(x_2|x_3) = x_1 \cdot \overline{x_2 \cdot x_3} =$ $= x_1(\overline{x}_2 \vee \overline{x}_3) = x_1 \cdot \overline{x}_2 \vee x_1 \cdot \overline{x}_3 = \overline{\overline{x_1 \cdot \overline{x}_2} \cdot \overline{x_1 \cdot \overline{x}_3}} = \overline{x_1 \cdot \overline{x}_2} | \overline{x_1 \cdot \overline{x}_3}$, что не соответствует распределительному закону (свойству дистрибутивности).

Формулы преобразования, полученные из основных свойств алгебры логики, для функции Шеффера имеют вид:

$$x_1 \cdot x_2 = \overline{x_1 | x_2} = (x_1 | x_2) | (x_1 | x_2);$$

$$\bar{x} = x | x;$$

$$x_1 \vee x_2 = \overline{\bar{x}_1 \cdot \bar{x}_2} = \bar{x}_1 | \bar{x}_2 = (x_1 | x_1) | (x_2 | x_2).$$

Функция Пирса (Вебба)

Логическая функция Пирса (Вебба) выражается через дизъюнкцию и отрицание:

$$x_1 \downarrow x_2 = \overline{x_1 \cdot x_2} = \bar{x}_1 \vee \bar{x}_2. \qquad (2.4)$$

Для функции Пирса справедливы законы:

$$x \downarrow x = \bar{x}; \qquad x \downarrow \bar{x} = 0;$$
$$x \downarrow 0 = \bar{x}; \qquad x \downarrow 1 = 0.$$

Для функции Пирса справедливо только свойство коммутативности:

$$x_1 \downarrow x_2 = x_2 \downarrow x_1.$$

Функции И, ИЛИ, НЕ выражаются через функцию Пирса следующим образом (формулы преобразования):

$$x_1 \cdot x_2 = (x_1 \downarrow x_1) \downarrow (x_2 \downarrow x_2);$$

$$x_1 \vee x_2 = \overline{x_1 \downarrow x_2} = (x_1 \downarrow x_2) \downarrow (x_1 \downarrow x_2);$$

$$\bar{x} = x \downarrow x.$$

Функции Шеффера и Пирса связаны между собой соотношениями, аналогичными формулам Де-Моргана для конъюнкции и дизъюнкции:

$$x_1 / x_2 = \overline{\bar{x}_1 \downarrow \bar{x}_2}; \qquad (2.5)$$

$$x_1 \downarrow x_2 = \overline{\bar{x}_1 / \bar{x}_2}. \qquad (2.6)$$

Действительно, например, для (2.5) справедливо:

$$x_1 / x_2 = \overline{x_1 \cdot x_2} = \bar{x}_1 \vee \bar{x}_2 = \overline{\overline{\bar{x}_1 \vee \bar{x}_2}} = \overline{\bar{x}_1 \downarrow \bar{x}_2}.$$

Аналогично доказывается справедливость равенства (2.6):

$$x_1 \downarrow x_2 = \overline{x_1 \vee x_2} = \bar{x}_1 \cdot \bar{x}_2 = \overline{\overline{\bar{x}_1 \cdot \bar{x}_2}} = \overline{\bar{x}_1 / \bar{x}_2}.$$

Построение таблиц истинности для логических выражений

ЛФ можно выразить через три основные функции: конъюнкцию, дизъюнкцию и отрицание.

Простой конъюнкцией называется конъюнкция одной или нескольких переменных, при этом каждая переменная встречается не более одного раза (либо сама, либо ее отрицание). Число переменных, составляющих элементарные конъюнкции или элементарные дизъюнкции, называют ее рангом. Например, $x \cdot y \cdot \bar{z}$ является простой конъюнкцией.

Дизъюнктивной нормальной формой (ДНФ) называется дизъюнкция простых конъюнкций. Например, выражение $x \cdot y \vee \bar{y} \cdot z$ является ДНФ.

Совершенной дизъюнктивной нормальной формой (СДНФ) называется такая дизъюнктивная нормальная форма, у которой в каждую конъюнкцию входят все переменные данного списка (либо сами, либо их отрицания), причем в одном и том же порядке. Например, выражение $x \vee y \cdot \bar{z}$ является ДНФ, но не СДНФ. Выражение $x \cdot y \cdot z \vee x \cdot \bar{y} \cdot \bar{z} \vee \bar{x} \cdot y \cdot z$ является СДНФ.

Простой дизъюнкцией называется дизъюнкция одной или нескольких переменных, при этом каждая переменная входит не более одного раза (либо сама, либо ее отрицание). Например, выражение $x \vee \bar{y} \vee z$ – простая дизъюнкция.

Конъюнктивной нормальной формой (КНФ) называется конъюнкция простых дизъюнкций. Например, выражение $\left(x \vee y \vee \bar{z}\right) \cdot \left(x \vee z\right) \cdot \left(y \vee \bar{z}\right)$ – КНФ.

Совершенной конъюнктивной нормальной формой (СКНФ) называется такая КНФ, у которой в каждую простую дизъюнкцию входят все переменные данного списка (либо сами, либо их отрицания), причем в одинаковом порядке. Например, выражение $\left(x \vee y \vee \bar{z}\right) \cdot \left(\bar{x} \vee \bar{y} \vee z\right) \cdot \left(\bar{z} \vee y \vee \bar{z}\right)$ является СКНФ.

Существует несколько форм представления ЛФ, используемых для проектирования комбинационных схем: словесная, табличная, аналитическая, геометрическая и кубическая.

Любая ЛФ может быть задана с помощью таблицы истинности (ТИ). В левой части ТИ записывается набор аргументов (логических переменных), она содержит полный перечень возможных комбинаций логических переменных. В правой части ТИ записываются соответствующие значения логической функции. При построении таблицы истинности необходимо учитывать порядок выполнения логических операций.

В качестве примера рассмотрим ТИ функции *Y* для трех аргументов x_1, x_2, x_3 (табл. 2.3). Функция *Y* принимает значение 1 или 0 на каждом наборе аргументов. Если значение функции не определено, то в соответствующей позиции ТИ ставится прочерк.

Таблица 2.3

Таблица истинности логической функции $Y(x_1, x_2, x_3)$

N	x_1	x_2	x_3	$Y(x_1, x_2, x_3)$
0	0	0	0	1
1	0	0	1	1
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

Иногда используют списочную (цифровую) форму представления ТИ, в которой приводится список единичных и нулевых наборов. Так, рассматриваемая в примере функция в списочной форме может быть представлена в виде:

$$Y = F(x_1, x_2, x_3) = \bigvee_1 (0,\ 1,\ 3,\ 6,\ 7);$$

$$Y = \bigwedge_0 (2,\ 4,\ 5).$$

Здесь в скобках приведены десятичные эквиваленты двоичных кодов наборов.

Алгоритм построения таблицы истинности:

1) подсчитать количество переменных n в логическом выражении;

2) определить число строк в таблице по формуле $m=2^n$;

3) подсчитать количество логических операций в выражении;

4) установить последовательность выполнения логических операций с учетом скобок и приоритетов;

5) определить количество столбцов как сумму количества переменных и логических операций;

6) выписать наборы входных переменных;

7) провести заполнение ТИ по столбцам, выполняя логические операции в соответствии с установленной в п. 4 последовательностью.

Пример 2.1. Составить таблицу истинности для выражения

$$F(x, y) = \bar{x} \wedge y \vee \overline{(x \wedge y)} \vee x.$$

Р е ш е н и е . Составим ТИ для заданной функции, которая содержит две переменные x и y. В первых двух столбцах ТИ запишем четыре возможных пары значений этих переменных, в последующих пяти – значения промежуточных функций, а в последнем – значения заданной функции (табл. 2.4).

Таблица 2.4

Таблица истинности логического выражения

Переменные		Промежуточные функции					$F(x, y)$
x	y	$\overline{x}$	$\overline{x} \wedge y$	$x \wedge y$	$\overline{x \wedge y}$	$\overline{x} \wedge y \vee \overline{(x \wedge y)}$	$\overline{x} \wedge y \vee \overline{(x \wedge y)} \vee x$
0	0	1	0	0	1	1	1
0	1	1	1	1	0	1	1
1	0	0	0	1	0	0	1
1	1	0	0	1	0	0	1

От табличной формы легко перейти к аналитической записи. Каждое из произведений аргументов, для которых значение равно 1, называют минтермом. Минтерм – это функция, образованная конъюнкцией некоторого числа переменных или их отрицаний. Минтерм принимает значение 1 при единственном из всех возможных наборов аргументов, и значение 0 при всех остальных. Например: $Mn_{11} = x_1 \cdot \overline{x}_2 \cdot x_3 \cdot x_4$.

Макстерм – это функция, образованная дизъюнкцией некоторого числа переменных или их отрицаний. Макстерм принимает значение 0 в одном из возможных наборов, и 1 при всех других. Пример: $Mx_6 = x_1 + x_2 + \overline{x}_3$.

Переход от табличной формы задания логической функции к алгебраической

Если таблица состояний задана или получена экспериментально, то можно найти такое логическое алгебраическое выражение, которое соответствует этой таблице. Рассмотрим один из возможных способов – с применением минтермов.

Любая логическая функция может быть выражена в виде СДНФ или СКНФ. В качестве примера рассмотрим функцию $f(x_1, x_2, x_3)$, представленную в табличной форме (табл. 2.5).

Таблица 2.5

Минтермы функции $f(x_1, x_2, x_3)$

N	x_1	x_2	x_3	$f(x_1, x_2, x_3)$	Mn_0	Mn_1	Mn_4	Mn_5	Mn_7
0	0	0	0	1	1	0	0	0	0
1	0	0	1	1	0	1	0	0	0
2	0	1	0	0	0	0	0	0	0
3	0	1	1	0	0	0	0	0	0
4	1	0	0	1	0	0	1	0	0
5	1	0	1	1	0	0	0	1	0
6	1	1	0	0	0	0	0	0	0
7	1	1	1	1	0	0	0	0	1

Функции Mn_0, Mn_1, Mn_4, Mn_5, Mn_7 – это минтермы. Каждая из этих функций является произведением трех переменных или их инверсий и принимает значение 1 только в одной ситуации. Видно, что для того, чтобы получить 1 в значении функции *f*, нужен один минтерм. Следовательно, количество минтермов, составляющих СДНФ этой функции, равно количеству единиц в значении функции:

$$f = Mn_0 + Mn_1 + Mn_4 + Mn_5 + Mn_7.$$

Таким образом, СДНФ имеет вид

$$f(x_1, x_2, x_3) = \overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3} + \overline{x_1} \cdot \overline{x_2} \cdot x_3 + x_1 \cdot \overline{x_2} \cdot \overline{x_3} + x_1 \cdot \overline{x_2} \cdot x_3 + x_1 \cdot x_2 \cdot x_3.$$

Аналогично можно построить СКНФ. Функции Mx_2, Mx_3, Mx_6 – это макстермы. Количество сомножителей равно количеству нулей в значениях функции:

$$f(x_1, x_2, x_3) = \left(x_1 + \overline{x_2} + x_3\right) \cdot \left(x_1 + \overline{x_2} + \overline{x_3}\right) \cdot \left(\overline{x_1} + \overline{x_2} + x_3\right).$$

Таким образом, в алгебраической форме возможно записать любую логическую функцию, которая задана в виде таблицы.

Алгоритм построения СДНФ по таблице истинности:

1. Необходимо выбрать все строки таблицы, в которых функция принимает значение 1.

2. Каждой такой строке поставить в соответствие конъюнкцию всех аргументов или их инверсий (минтерм). При этом аргумент, принимающий значение 0, входит в минтерм с инверсией, а значение 1 – без инверсии.

3. Образовать дизъюнкцию всех полученных минтермов. Количество минтермов должно совпадать с количеством единиц логической функции.

Алгоритм построения СКНФ по таблице истинности:

1. Необходимо выбрать все строки таблицы, в которых функция принимает значение 0.

2. Каждой такой строке поставить в соответствие дизъюнкцию всех аргументов или их инверсий (макстерм). При этом аргумент, принимающий значение 1, входит в макстерм с инверсией, а значение 0 – без инверсии.

3. Образовать конъюнкцию всех полученных макстермов. Количество макстермов должно совпадать с количеством нулей логической функции.

Если условиться отдавать предпочтение форме, которая содержит меньше букв, то СДНФ предпочтительней, если среди значений функции таблицы истинности меньше единиц, СКНФ – если меньше нулей.

Пример 2.2. Дана таблица истинности логической функции от трех переменных. Построить логическую формулу, реализующую эту функцию.

A	B	C	$F(A, B, C)$
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Р е ш е н и е . Выберем те строки в данной таблице истинности, в которых значения функции равны 0.

$$F\left(A,B,C\right)=\left(\overline{A}+B+C\right)\cdot\left(\overline{A}+B+\overline{C}\right).$$

Проверим выведенную функцию, составив таблицу истинности.

A	B	C	$\overline{A}+B+C$	$\overline{A}+B+\overline{C}$	$F\left(A,B,C\right)$
0	0	0	1	1	1
0	0	1	1	1	1
0	1	0	1	1	1
0	1	1	1	1	1
1	0	0	0	1	0
1	0	1	1	0	0
1	1	0	1	1	1
1	1	1	1	1	1

Синтез цифровых схем по заданной функции

По заданной функции можно построить схему, ее реализующую. Так как в соответствие заданной функции можно поставить множество схем, то задача синтеза решается неоднозначно.

Набор функций, через которые можно выразить любые другие функции, называется полным набором. Любая ЛФ может быть выражена через три основные функции – И, ИЛИ, НЕ. Таким образом, конъюнкция, дизъюнкция и отрицание – полный набор.

Логический элемент (ЛЭ) – часть электронной логической схемы, которая выполняет элементарную логическую операцию.

Каждый логический элемент имеет свое условное обозначение, имеет один или несколько входов, на которые подаются сигналы «высокого» напряжения (лог. 1) и «низкого» напряжения (лог. 0), и только один выход.

Логическая схема (ЛС) – это электронное устройство, реализующее логическую функцию. Для построения ЛС необходимо ЛЭ располагать в порядке, указанном в булевом выражении.

Выбор способа построения логической функции зависит от количества нулей и единиц: если в ней значительно меньше единиц, чем нулей, то лучше строить ДНФ.

В цифровой технике существуют специальные ЛС, реализующие базовые логические операции. По виду реализуемой ЛФ основные логические элементы условно разделяют на элементы одноступенчатой логики, реализующие функции И (конъюнкция), ИЛИ (дизъюнкция), НЕ (отрицание), И-НЕ, ИЛИ-НЕ, и на элементы двухступенчатой логики, реализующие функции И-ИЛИ, ИЛИ-И, И-ИЛИ-НЕ, ИЛИ-И-НЕ, И-ИЛИ и т.п.

Количество входов ЛС соответствует числу входных переменных, воспроизводимой им логической функции.

Схема, значение выходного сигнала которой определяется только комбинацией входных сигналов, называется комбинационной.

На рис. 2.1 приведены примеры условных обозначений наиболее часто используемых логических элементов.

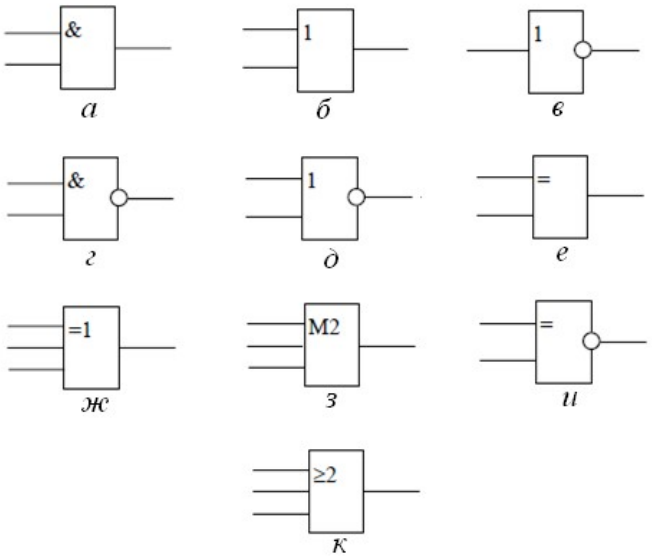


Рис. 2.1. Обозначение логических элементов на функциональных схемах: *а* – И; *б* – ИЛИ; *в* – НЕ; *г* – И-НЕ; *д* – ИЛИ-НЕ; *е* – эквивалентность; *ж* – исключающее ИЛИ; *з* – сумма по модулю 2; *и* – неэквивалентность; *к* – трехвходовой мажоритарный элемент

Алгоритм построения логических схем следующий:

1. Необходимо определить число логических переменных.

2. Определить количество логических операций и их порядок.

3. Изобразить для каждой логической операции соответствующий ей логический элемент.

4. Соединить логические элементы в порядке выполнения логических операций.

5. По возможности минимизировать полученную формулу.

6. Если заданы базисные элементы, то с помощью законов Моргана привести к заданному базису.

Совокупность элементарных функций, с помощью которых можно записать любую, сколь угодно сложную функцию, называют базисом. Функционально полными в алгебре логики являются три базиса: И-ИЛИ-НЕ – базис конъюнкции, дизъюнкции, инверсии (рис. 2.1, *а*, *б*, *в*); И-НЕ – базис Шеффера (рис. 2.1, *г*); ИЛИ-НЕ – базис Пирса или функция Вебба (рис. 2.1, *д*).

Пример 2.3. Построить логическую схему для функции

$$F(A,B,C)=\left(\overline{\left(A+\overline{B}\right)}\vee C\right)\wedge\left(B\vee\overline{C}\right).$$

Решение. В соответствии с приоритетами выполнения логических операций логическую схему строим в следующем порядке:

1) инвертируем значения параметров *B* и *C*;

2) при помощи одного элемента 2-ИЛИ складываем *A* и $\overline{B}$ (цифра 2 обозначает количество входов, но их может быть более двух) и инвертируем получившийся на выходе сигнал;

3) выход первого элемента 2-ИЛИ после инверсии соединяем со входом второго элемента 2-ИЛИ. На другой вход этого элемента подаем сигнал *C*;

4) отдельно при помощи третьего элемента 2-ИЛИ складываем *B* и $\overline{C}$;

5) соединяем выходы второго и третьего элементов 2-ИЛИ со входами элемента 2-И.

Логическая схема будет иметь вид (рис. 2.2):

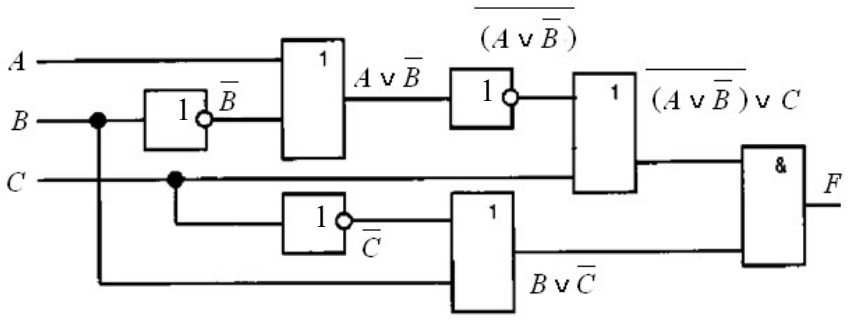


Рис. 2.2. Логическая схема для функции *F*

На вход логической схемы подаются все сигналы (*A*, *B*, *C*), участвующие в логической функции в не инвертированном виде. Количество входов логической схемы равно количеству переменных, участвующих в логической функции, а выход всегда один.

Таким образом, любую логическую функцию можно реализовать непосредственно по выражениям, представленным в виде СДНФ или СКНФ. Однако, полученная таким образом схема, как правило, не оптимальна Под оптимальной структурой принято понимать такое построение логического устройства, при котором число входящих в его состав элементов минимально.

Анализ цифровых схем. Переход от логической схемы к логической функции

Иногда по заданной схеме требуется определить функцию, реализующуюся данной схемой. При решении задачи анализа следует придерживаться следующей последовательности действий:

1. Заданная схема разбивается по ярусам.

2. Начиная с последнего, выходы каждого элемента обозначаются проиндексированными функциями в зависимости от яруса, к которому относится элемент.

3. Записываются выходные функции каждого элемента в виде формул в соответствии с введенными обозначениями.

4. Производится подстановка одних выходных функций через другие, используя входные переменные.

5. Записывается получившаяся булева функция через входные переменные.

Пример 2.4. По заданной логической схеме (рис. 2.4) составить булеву функцию.

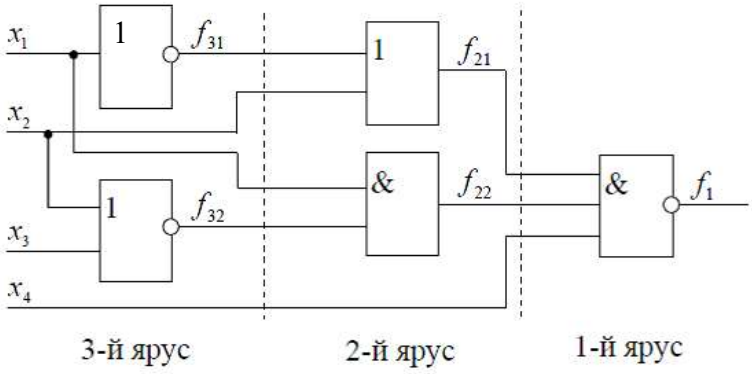


Рис. 2.4. Пример логической схемы устройства

Согласно приведённой выше последовательности действий, произведём разбиение схемы на ярусы. Пронумеровав получившиеся ярусы, введём обозначения для каждой выходной функции (см. рис. 2.4). Запишем все функции, начиная с 1-го яруса:

$$f_1 = \overline{f_{21} \cdot f_{22} \cdot x_4};$$

$$f_{21} = f_{31} \vee x_2, \;\; f_{22} = f_{32} \cdot x_1;$$

$$f_{31} = \overline{x_1}; \;\; f_{32} = \overline{x_2 \vee x_3}.$$

Теперь запишем все функции, подставляя входные переменные:

$$f_{21} = \overline{x_1} \vee x_2, \quad f_{22} = x_1 \cdot (\overline{x_2 \vee x_3}).$$

В итоге, получим выходную функцию:

$$f = f_1 = \overline{x_1 \cdot (\overline{x_1} \vee x_2) \cdot (\overline{x_2 \vee x_3}) \cdot x_4}.$$

Задания для самостоятельной работы

1. Составьте СДНФ и СКНФ. Приведите схемы для функций: импликации, сложения по модулю 2, мажоритарной (функции, дающей сигнал лог. 1 на выходе, если на входе более половины сигналов лог. 1).

2. Постройте таблицы истинности и ЛС для функций:

а) $f(x,y) = (x \vee y) \wedge \overline{x}$;

б) $F = \overline{A} \cdot (B + C)$;

в) $((A \to B) \wedge A) \leftrightarrow \overline{B}$;

г) $f(x_1, x_2, x_3) = (\overline{x_1} \vee x_2) \to (x_1 \cdot x_3)$;

д) $F = (A + B) \cdot \overline{C}$;

е) $f(x_1, x_2, x_3) = (x_1 \to x_3) \wedge (x_2 / x_1) \wedge \overline{(x_1 \vee x_2)}$.

3. Запишите логическую функцию трех переменных, если функция на наборах $F(0,0,1) = F(0,1,0) = F(1,0,1) = 1$, а на остальных – равно 0.

4. Постройте логические схемы устройств, реализующих логические функции:

а) $f = \overline{x}_1 \cdot x_2 \cdot x_3 \vee x_1 \cdot \overline{x}_2 \cdot x_3 \vee x_1 \cdot x_2 \cdot \overline{x}_3 \vee x_1 \cdot x_2 \cdot x_3$;

б) $f(a,b,c) = a \vee (\overline{c} \wedge b)$;

в) $F(A,B) = \overline{A} \cdot B \vee A \cdot \overline{B}$;

г) $F = \overline{A \cdot B} \cdot (A + B)$;

д) $F(A,B,C) = \overline{B} \cdot A + B \cdot \overline{A} + C \cdot \overline{B}$;

е) $F = \overline{A \cdot B} \cdot (A + B)$;

ж) $F = \overline{\overline{A \vee B}(\overline{A} \to B)}$;

з) $F = \overline{(\overline{A \cdot B} \to A)(\overline{A} \cdot B \to B)}$;

е) $F = \overline{(A \to B) \leftrightarrow (B \to A)}$;

и) $F = (A \oplus B)(\overline{A} \to B)(A \cdot B \vee \overline{A} \cdot \overline{B})$;

к) $F = (A\overline{B} \vee \overline{A} \vee B)\overline{A \oplus B}$;

л) $f(x_1, x_2, x_3, x_4) = x_1 \cdot (x_2 + \overline{x}_3) + \overline{x}_1 \cdot x_3 + \overline{x}_1 \cdot \overline{x}_2 \cdot x_4$;

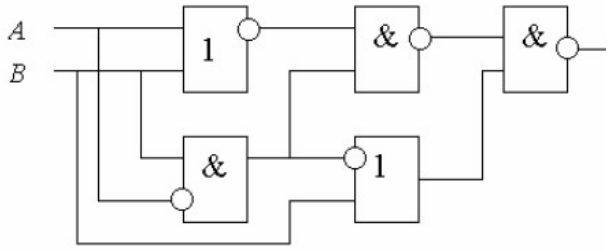
м) $y = (a + b + c) \cdot (a + b + \overline{c}) \cdot (\overline{a} + b + c) \cdot (\overline{a} + \overline{b} + c)$.

5. По таблице истинности постройте ЛС для логических функций:

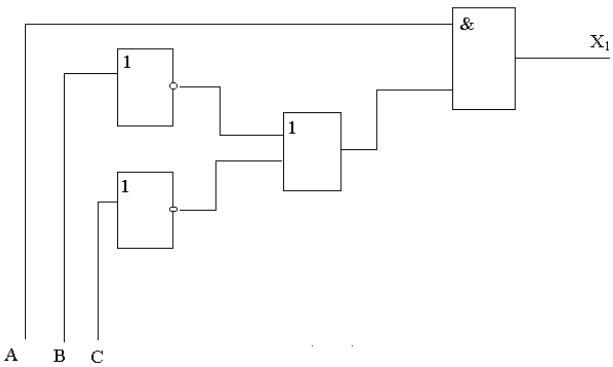
A	B	C	$F_1(A, B, C)$	$F_2(A, B, C)$	$F_3(A, B, C)$
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	0	0
0	1	1	1	1	0
1	0	0	0	0	0
1	0	1	1	1	0
1	1	0	1	0	1
1	1	1	0	0	0

6. Постройте и проанализируйте логические функции по заданным схемам:

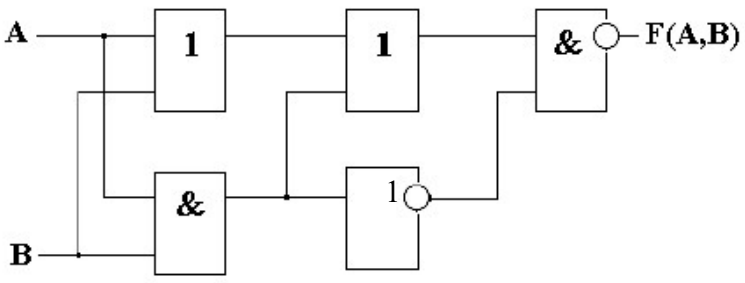
а)



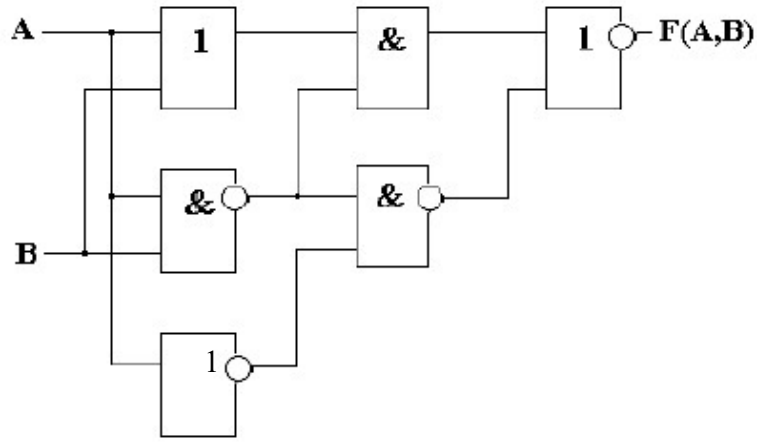
б)



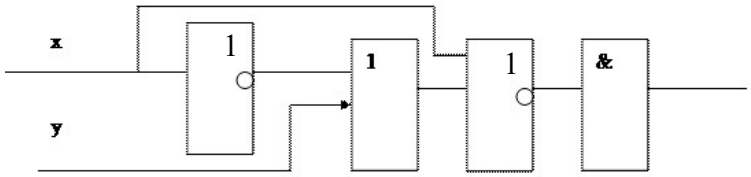
в)



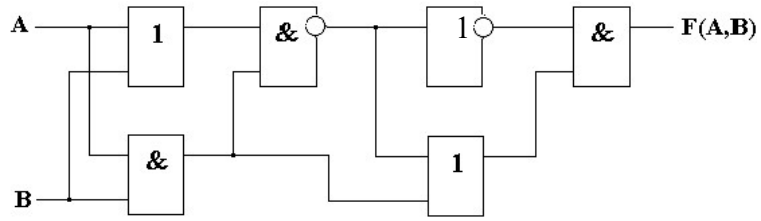
г)



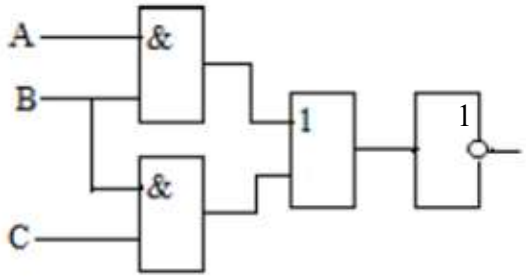
д)



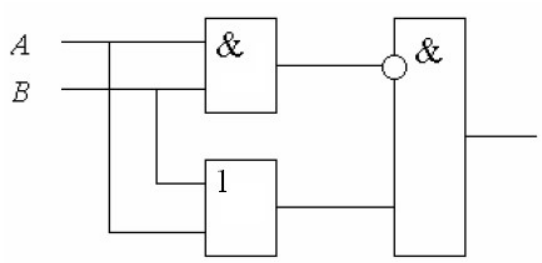
е)



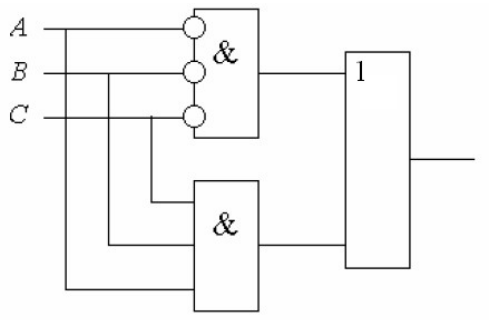
ж)



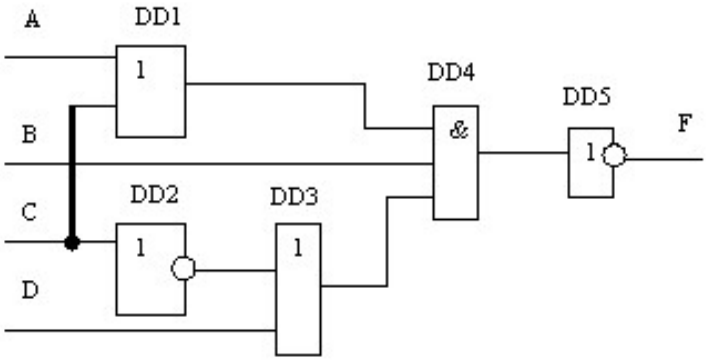
з)



и)



к)



7. Постройте таблицу истинности и функциональную схему для функций $F(A,B,C)=\overline{C+(A\oplus B)}$ и $F = (x \sim z) \mid ((x \cdot y) \sim (y \cdot z))$.

8. Постройте функциональную схему по таблице истинности:

A	B	C	$F_1(A, B, C)$	$F_2(A, B, C)$
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	0	1

9. Составьте таблицу истинности, СКНФ и функциональную схему для выражений: $F=(B\oplus C)\to A$, $F=\mathrm{C}\to\overline{A+B}$ и $Y = A \cdot B \cdot \overline{C} + C \cdot \left(\overline{A} + B\right)$.

10. По таблице истинности составьте СДНФ и СКНФ, нарисуйте функциональную схему.

x_1	x_2	x_3	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

11. Вычислите значение выражения $F=\overline{\left(A\wedge\left(B\vee C\right)\wedge D\right)}$ с помощью логической схемы, если $A = 0$; $B = 1$; $C = 1$; $D = 0$.

12. Постройте схему на логических элементах НЕ, ИЛИ, И, заданную функцией $F=\overline{A}BC+A\overline{B}C+AB\overline{C}$.

3. МИНИМИЗАЦИЯ ЛОГИЧЕСКИХ ФУНКЦИЙ

СДНФ не всегда является самым простым выражением функции. Реализации логической функции должна предшествовать операция минимизации функции. Минимизацией называют процедуру упрощения аналитического выражения ЛФ, для того, чтобы это выражение содержало минимальное количество членов с минимальным числом переменных.

Способы минимизации:

– алгебраический;

– с помощью диаграмм Вейча (карт Карно).

Метод минимизации с помощью диаграмм Вейча (карт Карно) используется для функций с малым числом переменных. Диаграмма Вейча представляет собой развертку *n*-мерного куба на плоскости. При этом вершины куба представляются клетками карты, каждой из которых поставлена в соответствие конститутиента единицы или нуля. В клетку карты, соответствующую конститутиенте единицы, заносится 1. Таким образом, для минимизации функции она должна быть представлена в форме СДНФ. Минимизация булевой функции с использованием карт в дизъюнктивной (конъюнктивной) форме заключается в объединении единичных (нулевых) клеток в контуры, каждому такому контуру соответствует простая импликанта.

Для булевой функции двух переменных диаграмма Вейча имеет вид, показанный на рис. 3.1, *а*. Каждая клетка диаграммы соответствует набору переменных булевой функции в ее таблице истинности. В клетке диаграммы Вейча ставится единица, если булева функция принимает единичное значение на соответствующем наборе. Нулевые значения булевой функции в диаграмме Вейча не ставятся. Диаграмма Вейча для ЛФ трех переменных показана на рис. 3.1, *б*. На рис. 3.1, *в* представлена диаграмма для функции четырех переменных.

Клетки на противоположных концах карты тоже являются соседними. Если представить карту свернутой по вертикали в цилиндр, то крайние клетки окажутся рядом, их тоже можно склеить (рис. 3.2).

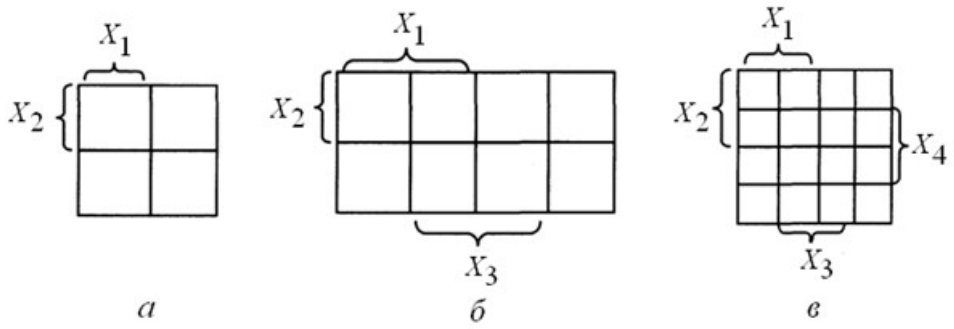


Рис. 3.1. Диаграммы Вейча для функции двух (*а*), трех (*б*) и четырех (*в*) переменных

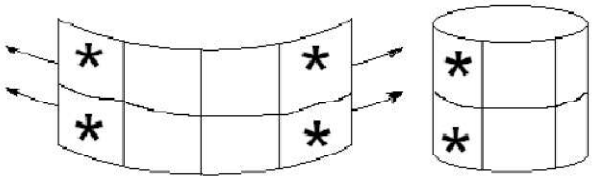


Рис. 3.2. Склеивание соседних клеток диаграммы Вейча

Соседние (по строке или столбцу) клетки отличаются значением только одной переменной. Диаграмму Вейча следует считать сложной фигурой, у которой крайние боковые стороны могут быть соединены в цилиндр, и крайние горизонтальные стороны также могут быть замкнуты в цилиндр. При этом можно полагать, что вся карта размещена на торе.

Правила минимизации с использованием карт Карно (диаграмм Вейча):

1. В карте Карно группы единиц (для получения ДНФ) и группы нулей (для получения КНФ) необходимо обвести четырехугольными контурами. Внутри контура должны находиться только одноименные значения функции. Этот процесс соответствует операции склеивания или нахождения импликант данной функции. Совокупность прямоугольников, покрывающих все единицы, называется покрытием. Заметим, что одна и та же ячейка может покрываться несколько раз.

2. Количество клеток внутри контура должно быть целой степенью двойки (1, 2, 4, 8, 16...).

3. При проведении контуров крайние строки карты (верхние и нижние, левые и правые), а также угловые клетки, считаются соседними (для карт до 4-х переменных).

4. Каждый контур должен включать максимально возможное количество клеток. В этом случае он будет соответствовать простой импликанте.

5. Все единицы (нули) в карте (даже одиночные) должны быть охвачены контурами. Любая единица (нуль) может входить в контуры произвольное количество раз.

6. Множество контуров, покрывающих все единицы (нули) функции, образуют тупиковую ДНФ (КНФ). Целью минимизации является нахождение минимальной из множества тупиковых форм.

7. В элементарной конъюнкции (дизъюнкции), которая соответствует одному контуру, остаются только те переменные, значение которых не изменяется внутри обведенного контура. Переменные булевой функции входят в элементарную коньюнкцию (для значений функции 1) без инверсии, если их значение на соответствующих координатах равно 1 и с инверсией – если 0. Для значений булевой функции, равных 0, записываются элементарные дизьюнкции, куда переменные входят без инверсии, если их значение на соответствующих координатах равно 0 и с инверсией – если 1.

С помощью диаграмм Вейча можно находить простые импликанты. Две ячейки, содержащие единицы, склеиваются, если они являются соседними, то есть расположены рядом (но не на диагонали). Переменная, значения которой для этих клеток различны (0 и 1), в соответствии с законом склеивания исчезает.

Рассмотрим рис. 3.3, *а*. В левом верхнем углу карты объединены две ячейки, соответствующие конъюнкциям $x_1x_2x_3\overline{x}_4$ и $x_1x_2\overline{x}_3\overline{x}_4$. В результате склеивания этих конъюнкций по переменной x_3 получается конъюнкция $x_1x_2\overline{x}_4$. В левом нижнем углу этой же карты склеивание ячеек определяет конъюнкцию $x_1\overline{x}_2\overline{x}_3$, ячейки, помеченные единицами в правом верхнем и правом нижнем углах, определяют конъюнкцию $\overline{x}_1\overline{x}_3\overline{x}_4$.

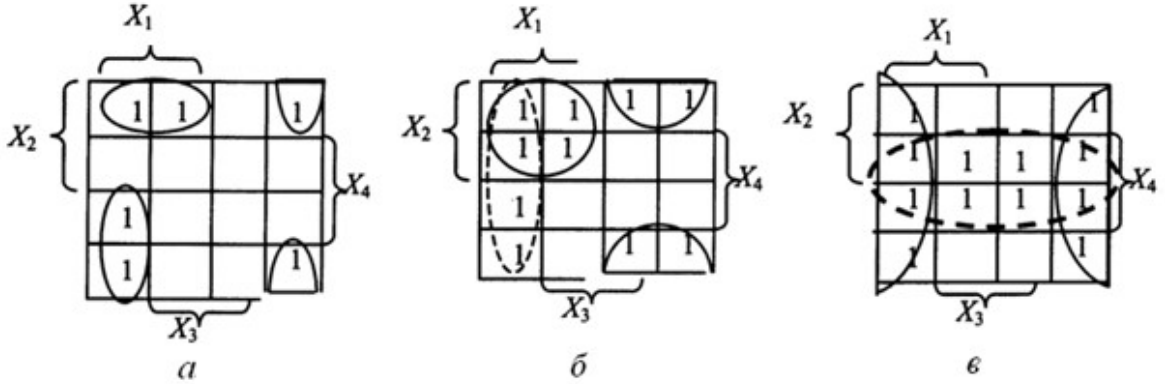


Рис. 3.2. Варианты склеивания двух (*а*), четырех (*б*) и восьми (*в*) соседних ячеек

Четыре ячейки, содержащие единицы, склеиваются, если они расположены в одной строке или в столбце, или в квадрате. Четырехклеточное объединение позволяет исключить две переменные. На диаграмме Вейча (рис. 3.2, *б*) представлены конъюнкции x_1x_2, $x_1\overline{x}_3$ и $\overline{x}_1\overline{x}_4$ – ячейки, образующие квадраты.

Восемь ячеек, содержащих единицы, склеиваются, если все они лежат в зоне, относящейся к какой-либо переменной или ее инверсии (см. рис. 3.2). Объединение восьми клеток позволяет исключить три переменные. На рис. 3.2, *в* представлены конъюнкции $\overline{x}_3$ (пунктир) и $\overline{x}_4$.

Пример 3.1. Пусть СДНФ функции $f(x_1,x_2,x_3,x_4)$ имеет вид

$$f(x_1,x_2,x_3,x_4)=x_1\cdot x_2\cdot\overline{x}_3\cdot\overline{x}_4+x_1\cdot x_2\cdot x_3\cdot\overline{x}_4+x_1\cdot x_2\cdot x_3\cdot x_4+x_1\cdot\overline{x}_2\cdot x_3\cdot\overline{x}_4+$$
$$+x_1\cdot\overline{x}_2\cdot x_3\cdot x_4+\overline{x}_1\cdot\overline{x}_1\cdot x_3\cdot x_4+\overline{x}_1\cdot\overline{x}_2\cdot x_3\cdot\overline{x}_4+\overline{x}_1\cdot\overline{x}_2\cdot\overline{x}_3\cdot\overline{x}_4. \qquad (3.1)$$

Минимизировать функцию (3.1) и составить функциональную схему устройства.

Р е ш е н и е. Функциональная схема устройства, реализующего функцию (3.1), показана на рис. 3.4.

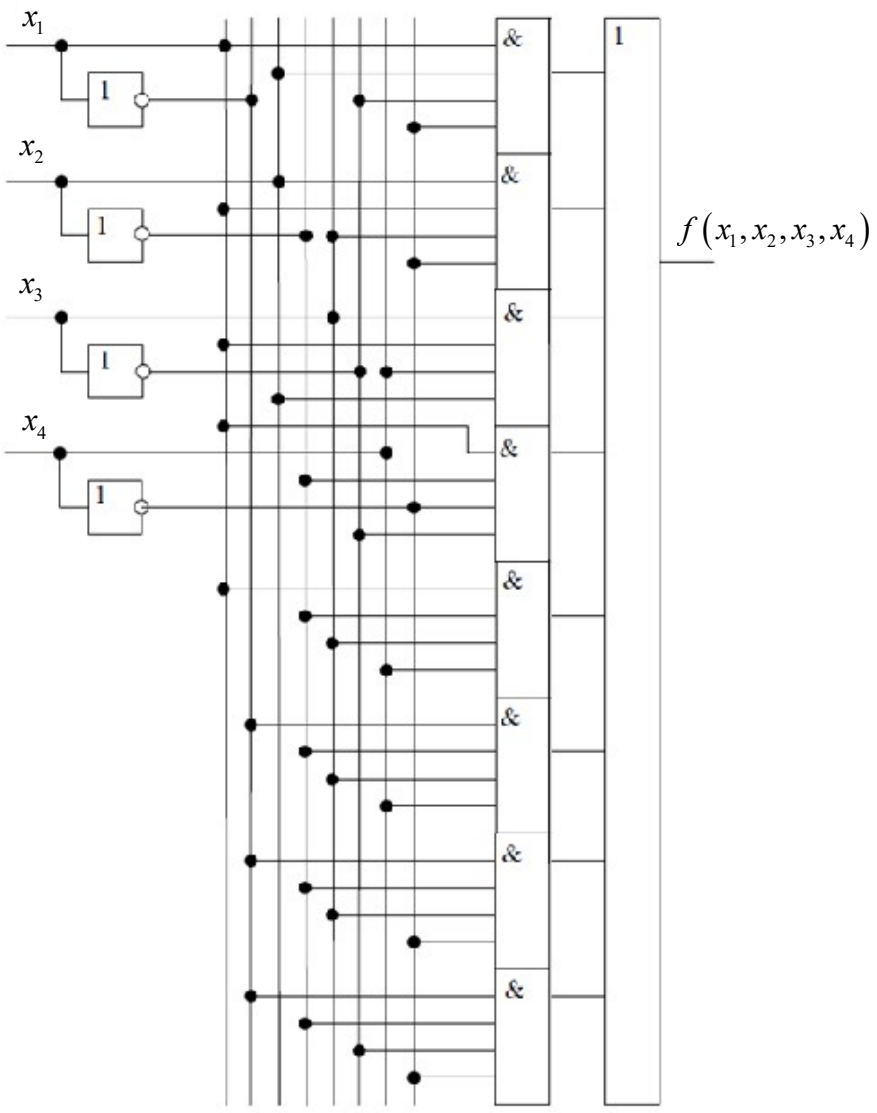


Рис. 3.4. Функциональная схема устройства

Минимизация функции с помощью диаграмм Вейча показана на рис. 3.5.

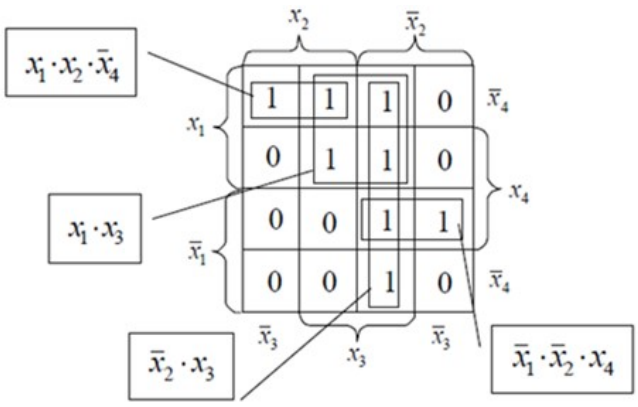


Рис. 3.5. Диаграмма Вейча

Следовательно, МДНФ функции имеет вид:

$$f\left(x_1, x_2, x_3, x_4\right) = x_1 \cdot x_2 \cdot \bar{x}_4 + x_1 \cdot x_3 + \bar{x}_2 \cdot x_3 + \bar{x}_1 \cdot \bar{x}_2 \cdot x_4. \qquad (3.2)$$

Функциональная схема устройства, реализующего функцию (3.2), показана на рис. 3.6.

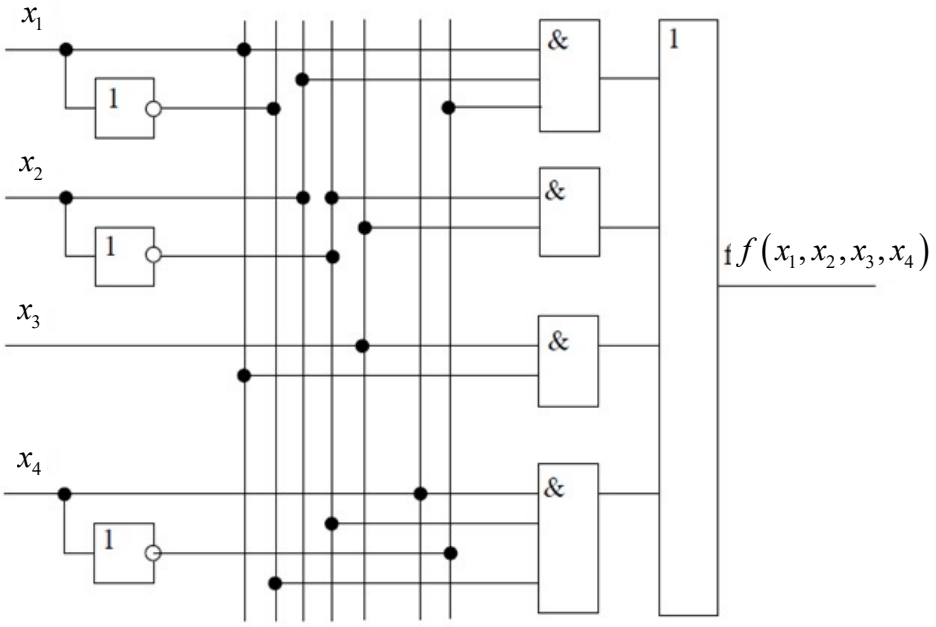


Рис. 3.6. Схема устройства, реализующего МДНФ вида (3.2)

Сопоставление схем, представленных на рис. 3.4 и 3.6, доказывает эффективность минимизации.

Пример 3.2. Найти МКНФ функции, заданной таблицей, с помощью карты Карно.

x	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
y	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
z	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
t	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
f	0	1	0	0	1	0	1	1	0	0	1	1	0	0	1	1

Р е ш е н и е. Минимизация функции с помощью диаграммы Вейча показана на рис. 3.7.

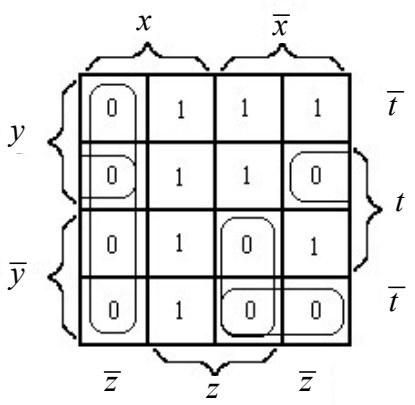


Рис. 3.7. Диаграмма Вейча

Следовательно, МКНФ функции имеет вид:

$$f(x,y,z,t)=\left(\overline{x}+z\right)\left(\overline{y}+z+\overline{t}\right)\left(x+y+\overline{x}\right)\left(x+y+t\right).$$

Диаграммы Вейча отличаются от карт Карно расположением столбцов и строк. В карту Карно булевы переменные передаются из таблицы истинности и упорядочиваются с помощью кода Грея, в котором каждое следующее число отличается от предыдущего только одним разрядом (используется циклический порядок следования символов, а именно 00, 01, 11, 10). В диаграмме Вейча символы располагаются в порядке возрастания двоичных чисел, а именно 00, 01, 10, 11. В остальном эти методы идентичны. Карты Карно более удобны в обращении и не требуют столь большой затраты времени.

Чтобы представить функцию на карте Карно, достаточно в клетки карты, где функция имеет значение 1, поместить единицы (рис. 3.8). Клетки, в которых записаны единицы, называют конституентами единицы функции или просто конституентами. Две соседние конституенты склеиваются и образуют простую импликанту.

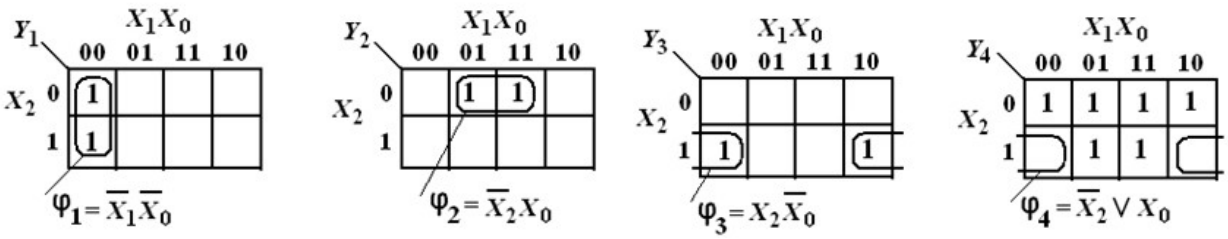


Рис. 3.8. Примеры минимизации для ЛФ трех переменных

На рис. 3.9 показаны примеры минимизации ЛФ четырех переменных.

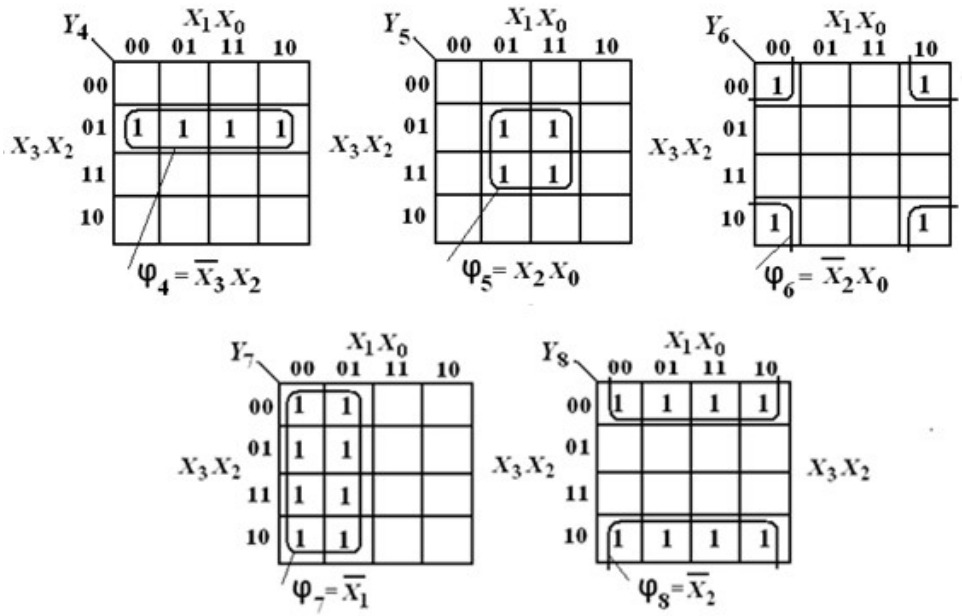


Рис. 3.9. Примеры минимизации для ЛФ четырех переменных

Пример 3.3. Минимизировать с помощью карт Карно функцию:

$$y = x_2x_1x_0 + x_2x_1\overline{x}_0 + x_2\overline{x}_1\overline{x}_0 + \overline{x}_2\overline{x}_1\overline{x}_0 + \overline{x}_2x_1\overline{x}_0. \qquad (3.3)$$

Р е ш е н и е. Из (3.3) видно, что для минимизации функции используем карту Карно для трех переменных (рис. 3.10). Каждое слагаемое в этой логической функции представляет собой минтерм – конъюнкцию входных переменных, обращающую функцию в единицу. Поэтому на рис. 3.10 пять единиц.

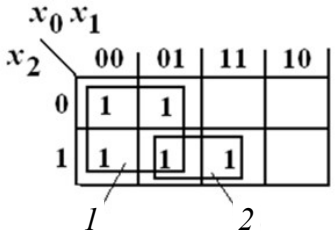


Рис. 3.10. Карта Карно

Из рис. 3.10 видно, что группе *1* соответствует повторяющая переменная $\overline{x}_0$, а *2* – переменные x_1 и x_2. Поэтому минимизированное алгебраическое выражение функции имеет вид $y = \overline{x}_0 + x_2x_1$.

Минимизация частично заданных логических функций

Не полностью определенной логической функцией называется такая функция алгебры логики (ФАЛ), значение которой определены не на всех наборах аргументов. При представлении такой ФАЛ в виде таблицы истинности наборы, на которых значение функции не определено, отмечаются особым символом, отличным от нуля и единицы, например, «~», «*»,«×».

Основная задача минимизации не полностью определенных функций заключается в отыскании оптимального варианта ее доопределения, позволяющего получить минимальную нормальную форму. Если значения ФАЛ не определены на n наборах, то ее можно доопределить 2^n способами. Поэтому минимизация не полностью определенной логической функции состоит в выборе одной из 2^n полностью определенных функций, которая позволяет получить форму с минимальным количеством букв.

В табл. 3.1 приведен пример функции трех переменных: функция принимает значение единица на наборах 0, 5 и 7, значение нуль – на 2 и 6. На наборах 1, 3 и 4 значение функции не определено.

Таблица. 3.1

Таблица истинности не полностью определенной логической функции

Номер набора	a	b	c	$f(a, b, c)$
0	0	0	0	1
1	0	0	1	×
2	0	1	0	0
3	0	1	1	×
4	1	0	0	×
5	1	0	1	1
6	1	1	0	0
7	1	1	1	1

Сокращенная запись этой функции имеет вид:

$$f(a,b,c)=\sum(0,5,7),\times(1,3,4) \text{ или } f(a,b,c)=\prod(2,6),\times(1,3,4).$$

При составлении карт Карно (диаграмм Вейча) для не полностью заданной функции в клетках, соответствующих координатам неиспользованных входных наборов, проставляются звездочки (прочерки). Клетки со звездочками (прочерками) при необходимости включаются в контуры как вместе с единицами (при получении формулы на базе ДНФ), так и вместе с нулями (при получении формулы на базе КНФ). При этом выражение функции после минимизации оказывается более простым.

Пример 3.4. Рассмотрим карту Карно для четырех переменных, представленную на рис. 3.11.

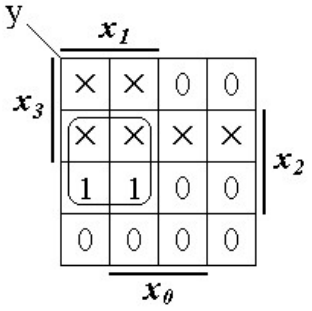


Рис. 3.11. Минимизации функции четырех переменных с помощью карты Карно

Доопределив безразличные значения y на наборах 14 и 15 единицами, получим следующее минимальное выражение: $y = x_2 x_1$.

После реализации этой функции она становится полностью определенной, т. е. на безразличных наборах, включенных в контур, будут реализовываться значения 1, а на невключенных в контур – значения 0.

Задания для самостоятельной работы

1. Минимизируйте функцию с помощью диаграммы Вейча, заданную таблицей:

x	0	0	0	0	1	1	1	1
y	0	0	1	1	0	0	1	1
z	0	1	0	1	0	1	0	1
f	1	1	1	0	0	1	0	1

2. Используя диаграммы Вейча получите минимальные КНФ и ДНФ:

а) $f(x,y,z,t) = \overline{x} \cdot y \cdot z \cdot \overline{t} + \overline{x} \cdot \overline{y} \cdot z \cdot \overline{t} + x \cdot \overline{z} \cdot t + \overline{x} \cdot \overline{z} \cdot t$;

б) $F = x \cdot y \cdot v \cdot z + \overline{x} \cdot y \cdot v \cdot z + x \cdot \overline{y} \cdot \overline{v} \cdot z + \overline{x} \cdot y \cdot v \cdot \overline{z} + x \cdot y \cdot \overline{v} \cdot \overline{z} + \overline{x} \cdot \overline{y} \cdot \overline{v} \cdot z$;

в) $f(x,y,z)_{\text{СДНФ}} = \sum(0,1,2,5,7)$;

г) $f(x,y,z)_{\text{СДНФ}} = x \cdot y \cdot z \vee x \cdot \overline{y} \cdot \overline{z} \vee \overline{x} \cdot \overline{y} \cdot \overline{z}$;

д) $f(a,b,c,d)_{\text{СДНФ}} = \sum(0,1,2,3,4,7,11,15)$;

е) $f(a,b,c,d)_{\text{СДНФ}} = \sum(0,2,3,7,9,10,11,14)$;

ж) $f(a,b,c,d)_{\text{СДНФ}} = \prod(2,3,5,6,7,10,11,13,14)$;

з) $f(x,y,z)_{\text{СДНФ}} = \prod(0,2,5,6,7)$;

и) $f(x_1,x_2,x_3,x_4)_{\text{СДНФ}} = \sum(0,2,3,4,6,8,10,11,14)$.

3. Составьте карту Карно и найдите минимальные ДНФ и КНФ функции четырёх переменных $f(x_1,x_2,x_3,x_4)$, заданной таблицей истинности:

Номер набора	Наборы аргументов				Значение функции	Номер набора	Наборы аргументов				Значение функции
	x_4	x_3	x_2	x_1			x_4	x_3	x_2	x_1	
0	0	0	0	0	1	8	1	0	0	0	1
1	0	0	0	1	1	9	1	0	0	1	1
2	0	0	1	0	1	10	1	0	1	0	0
3	0	0	1	1	1	11	1	0	1	1	0
4	0	1	0	0	0	12	1	1	0	0	1
5	0	1	0	1	0	13	1	1	0	1	1
6	0	1	1	0	0	14	1	1	1	0	0
7	0	1	1	1	1	15	1	1	1	1	0

4. С помощью карт Карно найдите МДНФ и МКНФ следующих функций:

а) $f(x_1,x_2,x_3,x_4)=x_1x_2\overline{x}_3\overline{x}_4+x_1\overline{x}_2\overline{x}_3\overline{x}_4+\overline{x}_1x_2x_3\overline{x}_4+\overline{x}_1x_2\overline{x}_3\overline{x}_4+\overline{x}_1x_2x_3x_4+$
$+\overline{x}_1\overline{x}_2x_3x_4+\overline{x}_1\overline{x}_2\overline{x}_3x_4+\overline{x}_1\overline{x}_2x_3\overline{x}_4+\overline{x}_1\overline{x}_2\overline{x}_3\overline{x}_4;$

б) $f(x_1,x_2,x_3,x_4)=x_1x_2\overline{x}_3\overline{x}_4+x_1x_2x_3\overline{x}_4+\overline{x}_1x_2x_3\overline{x}_4+\overline{x}_1x_2\overline{x}_3\overline{x}_4+\overline{x}_1x_2x_3x_4+$
$+x_1\overline{x}_2x_3x_4+\overline{x}_1\overline{x}_2x_3x_4+\overline{x}_1\overline{x}_2x_3\overline{x}_4;$

в) $f(x_1,x_2,x_3,x_4)=x_1x_2\overline{x}_4+x_1x_2x_3+x_1\overline{x}_2x_3\overline{x}_4+\overline{x}_1x_2x_4+\overline{x}_1\overline{x}_2\overline{x}_3x_4;$

г) $f(x_1,x_2,x_3,x_4)=x_1\overline{x}_2x_4+x_1x_2x_3+x_1\overline{x}_2x_3\overline{x}_4+\overline{x}_1x_2x_4+\overline{x}_1\overline{x}_2\overline{x}_3x_4;$

д) $f(x_1,x_2,x_3,x_4)=x_1x_2\overline{x}_4+x_1x_2x_3x_4+\overline{x}_1\overline{x}_3\overline{x}_4+\overline{x}_2x_3\overline{x}_4+\overline{x}_2\overline{x}_3\overline{x}_4;;$

е) $f(x_1,x_2,x_3,x_4)=x_1x_2x_4+\overline{x}_1x_3x_4+\overline{x}_1\overline{x}_2x_4+x_1x_2x_3\overline{x}_4+x_1\overline{x}_2\overline{x}_3x_4;$

ж) $f(x_1,x_2,x_3,x_4)=x_1x_2\overline{x}_3\overline{x}_4+x_2x_3x_4+\overline{x}_1x_2x_3+\overline{x}_1x_2\overline{x}_4+\overline{x}_1\overline{x}_2\overline{x}_4+x_1\overline{x}_2\overline{x}_3\overline{x}_4;$

з) $f(x_1,x_2,x_3,x_4)=x_1x_2\overline{x}_3+x_1x_2x_4+x_1\overline{x}_2x_4+\overline{x}_1\overline{x}_3x_4+\overline{x}_1\overline{x}_2\overline{x}_3;$

и) $f(x_1,x_2,x_3,x_4)=x_1\overline{x}_3\overline{x}_4+x_2\overline{x}_3\overline{x}_4+\overline{x}_1x_2x_3+x_1\overline{x}_2x_3\overline{x}_4+\overline{x}_1x_2x_3\overline{x}_4+\overline{x}_1\overline{x}_2\overline{x}_3\overline{x}_4;$

к) $y=\overline{x}_1\overline{x}_2x_3+\overline{x}_1x_2x_3+\overline{x}_1x_2\overline{x}_3+x_1\overline{x}_2x_3+x_1x_2x_3.$

5. Найдите с помощью карт Карно МДНФ и МКНФ следующих функций:

x	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
y	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
z	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
t	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
F_1	0	1	0	0	0	1	0	1	1	0	1	0	1	0	1	1
F_2	0	0	0	1	0	0	0	1	1	1	0	1	1	0	1	1
F_3	1	0	0	0	1	0	1	1	1	0	0	0	1	1	0	1

6. С помощью диаграмм Вейч найдите МКНФ следующих функций:

а) $f(x_1,x_2,x_3,x_4)=(x_1+\overline{x}_2+x_4)(\overline{x}_1+x_3+\overline{x}_4)(x_2+\overline{x}_3+\overline{x}_4)(x_1+x_2+x_4);$

б) $f(x_1,x_2,x_3,x_4)=(x_1+\overline{x}_2+x_4)(\overline{x}_2+x_3+\overline{x}_4)(x_1+x_2+x_3)(x_2+\overline{x}_3+x_4);$

в) $f(x_1,x_2,x_3,x_4)=(\overline{x}_1+\overline{x}_3+x_4)(\overline{x}_1+x_2+\overline{x}_4)(x_1+x_2+\overline{x}_3)(x_1+x_3+\overline{x}_4).$

7. Найдите МДНФ следующих функций с помощью карты Карно:

x	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
y	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
z	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
t	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
F_1	0	–	1	1	1	0	0	0	0	0	–	–	–	0	0	1
F_2	0	–	–	0	0	0	–	1	0	1	–	0	0	0	1	1
F_3	1	0	1	1	–	0	1	–	–	0	0	0	1	0	0	–
F_4	0	–	1	1	1	0	0	0	0	0	–	–	–	0	0	1
F_5	0	–	–	0	0	0	–	1	0	1	–	0	0	0	1	1
F_6	1	0	1	1	–	0	1	–	–	0	0	0	1	0	0	–

8. Минимизируйте с помощью диаграммы Вейча логические выражения:

а) $y = x_1\overline{x}_2\overline{x}_3x_4 + x_1x_2\overline{x}_3\overline{x}_4 + \overline{x}_1x_2\overline{x}_3x_4 + \overline{x}_1x_2x_3\overline{x}_4 + \overline{x}_1x_2\overline{x}_3\overline{x}_4 + x_1x_2x_3\overline{x}_4 +$
$+x_1\overline{x}_2x_3\overline{x}_4 + x_1x_2x_3\overline{x}_4 + x_1x_2\overline{x}_3x_4 + \overline{x}_1\overline{x}_2x_3\overline{x}_4;$

б) $y = \overline{x}_1\overline{x}_2x_3x_4 + x_1x_2\overline{x}_3\overline{x}_4 + \overline{x}_1x_2\overline{x}_3x_4 + \overline{x}_1x_2\overline{x}_3\overline{x}_4 + \overline{x}_1x_2x_3\overline{x}_4 + x_1x_2\overline{x}_3x_4 +$
$+x_1\overline{x}_2x_3\overline{x}_4 + x_1x_2x_3\overline{x}_4 + x_1\overline{x}_2x_3x_4 + \overline{x}_1\overline{x}_2x_3\overline{x}_4.$

4. СИНТЕЗ ЛОГИЧЕСКИХ УСТРОЙСТВ В ЗАДАННОМ БАЗИСЕ

Набор ЛЭ обладает функциональной полнотой, если с помощью этого набора можно реализовать схему с любым законом функционирования.

Таким образом, любая комбинационная схема может быть построена с применением логических элементов И, ИЛИ, НЕ, совокупность которых является функционально полной системой (базисом), а также из элементов И-НЕ или ИЛИ-НЕ, или И-ИЛИ-НЕ, каждый из которых представляет функционально полною систему.

Элементы, обеспечивающие выполнение любой из трех основных операций (И, НЕ, ИЛИ), называют универсальными. Если задан базис И-НЕ, то путем двойного инвертирования исходного выражения или его части и применения теорем де Моргана логическая функция приводится к виду, содержащему только операции логического умножения и инвертирования. Если же задан базис ИЛИ-НЕ, исходную логическую функцию теми же приемами приводят к виду, содержащему только операции логического сложения и инверсии. Далее логическое выражение записывается через условные обозначения выбранных операций.

Для преобразования ЛФ в базис И-НЕ следует заменить все инверсии, конъюнкции и дизъюнкции, входящие в МДНФ, на штрих Шеффера. Эта замена основана на соотношениях:

$$\overline{a} = a|a = a|1; \quad ab = \overline{a|b} = (a|b)|(a|b) = (a|b)|1;$$

$$a \vee b = \overline{\overline{a}\,\overline{b}} = (a|b)|(b|b) = (a|1)|(b|1).$$

Для того чтобы доказать функциональную полноту элемента Шеффера, покажем возможность построения на его основе логических цепей, реализующих простейшие функции НЕ, И, ИЛИ (рис. 4.1).

Действительно, в базисе И-НЕ имеем:

а) $\overline{x} = \overline{x \cdot x} = x|x$, т.е. инвертор реализуется элементом И-НЕ с запараллеленными входами (рис. 4.1, *а*);

б) $x_1 \cdot x_2 = \overline{\overline{x_1 \cdot x_2}} = \left(x_1 | x_2\right) | \left(x_1 | x_2\right)$, т.е. конъюнктор реализуется двумя элементами Шеффера (рис. 4.3, *б*);

в) $x_1 \vee x_2 = \overline{\overline{x_1} \cdot \overline{x_2}} = \left(x_1 | x_1\right) | \left(x_2 | x_2\right)$, т.е. дизъюнктор реализуется тремя элементами И-НЕ (рис. 4.1, *в*).

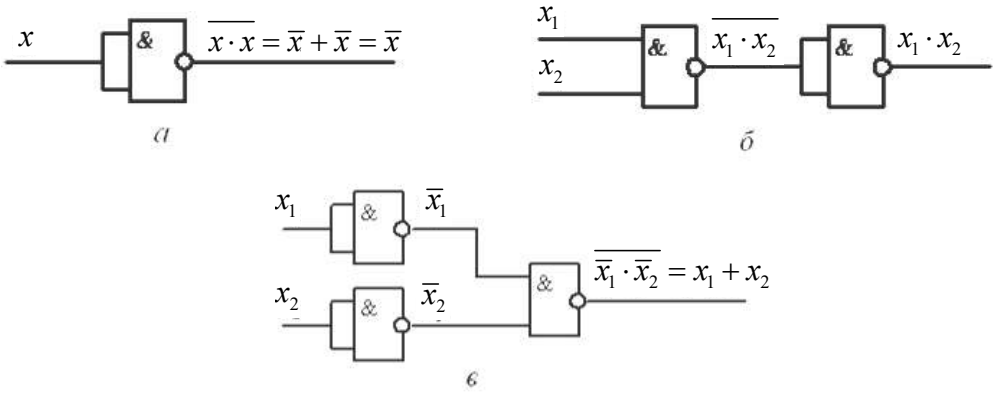


Рис. 4.1. Реализация простейших функций НЕ (*а*), И (*б*), ИЛИ (*в*) на основе штриха Шеффера

Покажем возможность построения логических функций НЕ, И и ИЛИ на основе элемента Пирса (рис. 4.2).

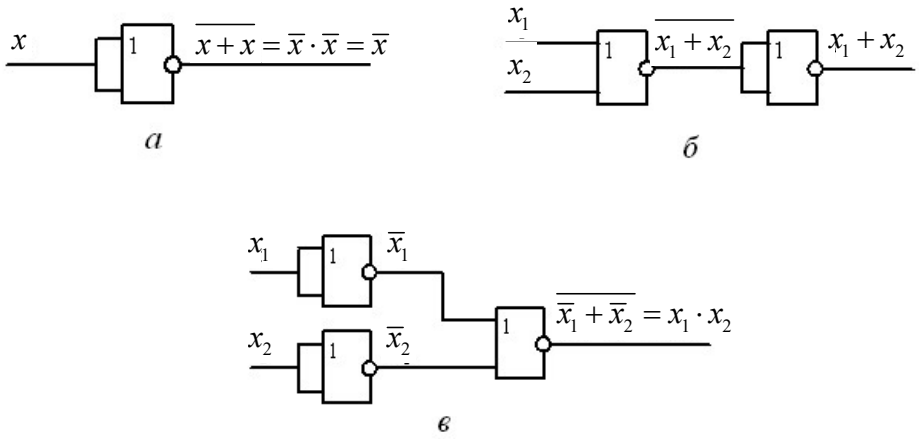


Рис. 4.2. Реализация функций НЕ (*а*), И (*б*), ИЛИ (*в*) на элементах ИЛИ-НЕ (стрелка Пирса)

Действительно, в базисе ИЛИ-НЕ:

а) $\overline{x} = \overline{x \vee x} = x \downarrow x$, т. е. инвертор реализуется элементом ИЛИ-НЕ с запараллеленными входами (см. рис. 4.2, *а*);

б) $x_1 \vee x_2 = \overline{\overline{x_1 \vee x_2}} = \left(x_1 \downarrow x_2\right) \downarrow \left(x_1 \downarrow x_2\right)$, т. е. дизъюнктор реализуется двумя элементами ИЛИ-НЕ (см. рис. 4.2, *б*);

в) $x_1 \cdot x_2 = \overline{\overline{x_1} \vee \overline{x_2}} = \left(x_1 \downarrow x_1\right) \downarrow \left(x_2 \downarrow x_2\right)$, т. е. конъюнктор реализуется тремя элементами ИЛИ-НЕ (см. рис. 4.2, *в*).

Таким образом, все основные операции (НЕ, И, ИЛИ) могут быть реализованы на элементах любого из рассмотренных выше базисов.

Преобразование дизъюнктивной формы (И, НЕ) также как и преобразование конъюнктивной формы (ИЛИ, НЕ) в базис приводит к схеме с дополнительным выходным инвертором.

Запись в базис И-НЕ (ИЛИ-НЕ) производится в два этапа:

1-й этап – логическая формула, минимизированная в основном базисе, представляется в форме ДНФ (КНФ);

2-й этап – над правой частью полученной формулы ставится два знака инверсии и с помощью формул де Моргана осуществляется переход в базис И-НЕ (ИЛИ-НЕ).

Запись в базисе И-ИЛИ-НЕ производится также в два этапа:

1-й этап – логическая формула для инверсного значения функции $\overline{f}$ минимизируется в основном базисе и представляется в форме ДНФ;

2-й этап – над обеими частями формулы ставится один знак инверсии, и с помощью формул де Моргана производится переход в базис И-ИЛИ-НЕ.

Пример 4.1. Функцию $f = x_1x_4 + \overline{x}_1x_3\overline{x}_4 + x_1\overline{x}_2\overline{x}_3$ перевести в базисы И-НЕ и ИЛИ-НЕ.

Р е ш е н и е . Исходная ДНФ в базисе И-НЕ имеет вид:

$$f = x_1x_4 + \overline{x}_1x_3\overline{x}_4 + x_1\overline{x}_2\overline{x}_3 = \overline{\overline{x_1x_4 + \overline{x}_1x_3\overline{x}_4 + x_1\overline{x}_2\overline{x}_3}} =$$

$$= \overline{\overline{(x_2x_4)}\,\overline{(\overline{x}_1x_3\overline{x}_4)}\,\overline{(x_1\overline{x}_2\overline{x}_3)}} = (x_2/x_4)\,(\overline{x}_1|x_3|\overline{x}_4)\,(x_1|\overline{x}_2|\overline{x}_3).$$

Аналогично, КНФ в базисе ИЛИ-НЕ имеет вид:

$$f = \overline{\overline{(x_1 + x_4)(\overline{x}_1 + x_3 + \overline{x}_4)(x_1 + \overline{x}_2 + \overline{x}_3)}} = \overline{\overline{(x_1 + x_4)}} + \overline{\overline{(\overline{x}_1 + x_3 + \overline{x}_4)}} + \overline{\overline{(x_1 + \overline{x}_2 + \overline{x}_3)}} =$$

$$= (x_1 \downarrow x_4) \downarrow (\overline{x}_1 \downarrow x_3 \downarrow \overline{x}_4) \downarrow (x_1 \downarrow \overline{x}_2 \downarrow \overline{x}_3).$$

Пример 4.2. Построить логическую схему в базисе И-НЕ для функции, заданной таблицей истинности:

x_1	x_2	x_3	$f(x_1, x_2. x_3)$
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

Р е ш е н и е . Минимизируем заданную функцию. Заполняем диаграмму и обводим прямоугольными контурами соседние клетки с единицами, как показано на рис. 4.3.

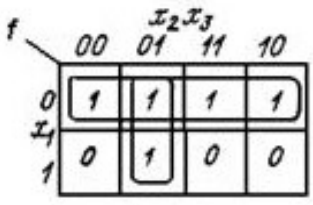


Рис. 4.3. Диаграмма Вейча минимизируемой функции

Используя контуры, показанные на карте Карно, получаем следующее логическое выражение $f(x_1, x_2, x_3) = \overline{x}_1 \vee \overline{x}_2 x_3$. Преобразуем полученное логическое выражение к базису И-НЕ:

$$f(x_1, x_2, x_3) = \overline{\overline{\overline{x}_1 \vee \overline{x}_2 x_3}} = \overline{x_1 \overline{x}_2 x_3}. \qquad (4.1)$$

Логическая схема, реализующая функцию (4.1) на элементах И-НЕ, представлена на рис. 4.4.

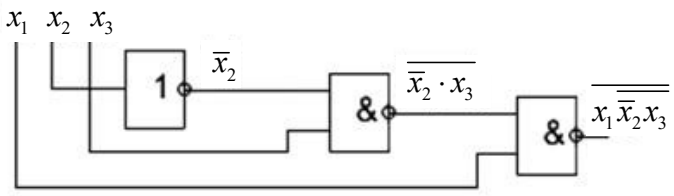


Рис. 4.4. Логическая схема на элементах И-НЕ

Пример 4.3. В базисе ИЛИ-НЕ построить схему, реализующую функцию $f = (\overline{a} + \overline{c}) \cdot (a + \overline{b})$.

Р е ш е н и е. МКНФ функции $f = (\overline{a} + \overline{c}) \cdot (a + \overline{b})$ приводится к базису {ИЛИ, НЕ} следующим образом:

$$f = \overline{\overline{(\overline{a} + \overline{c}) \cdot (a + \overline{b})}} = \overline{\overline{(\overline{a} + \overline{c}) + \overline{(a + \overline{b})}}} = (\overline{a} \downarrow \overline{c}) \downarrow (a \downarrow \overline{b}). \qquad (4.2)$$

При инвертировании функции используется правило де Моргана. Схема, реализующая выражение (4.2), приведена на рис. 4.5.

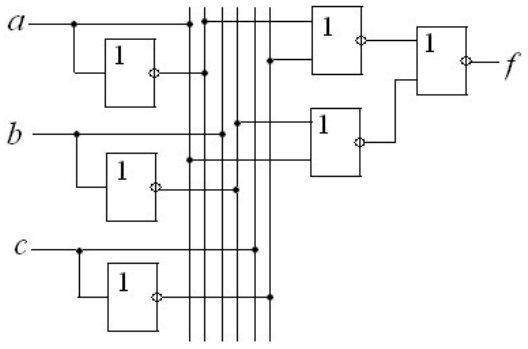


Рис. 4.5. Схема на элементах ИЛИ-НЕ

В состав стандартных серий кроме простых ЛЭ входят и более сложные. Они представляют собой комбинацию из простых ЛЭ и выпускаются в составе серий интегральных микросхем (ИМС).

Для логических элементов (Л) различных видов используются следующие обозначения: ЛН – инвертор; ЛА – логическое умножение с инверсией (И-НЕ); ЛЕ – логическое сложение с инверсией (ИЛИ-НЕ); ЛР – совмещенные логические элементы, например И-ИЛИ-НЕ; ЛБ – элементы И-НЕ/ ИЛИ-НЕ; ЛИ – элемент И; ЛЛ – элемент ИЛИ; ЛС – элемент И-ИЛИ; ЛМ – ИЛИ-НЕ/ИЛИ; ЛД – расширители; ЛП – прочие.

Как правило, за редким исключением, у микросхем различных серий, имеющих одинаковые названия, совпадают и функциональное назначение, и логика работы, и расположение выводов. Но другие параметры, такие как быстродействие, потребляемая мощность, входные и выходные токи и т. д. будут отличными.

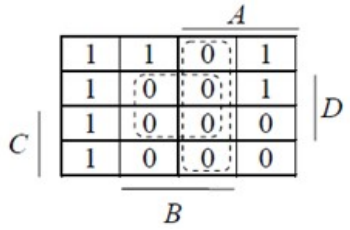
В ряду ИМС, сделанных по технологиям ТТЛ и ТТЛШ, это серии 130, 131, 133, 134, 136, 155, 158, 530, 531, 533, 555, 1530, КР1530, 1531, КР1531, 1533 и КР1533. Среди ИМС, сделанных по технологии КМОП, аналогично друг другу работают микросхемы серий 164, 176, 561, 564, 1561. Микросхемы КМОП серий 1564 и 1554 функционируют аналогично одноименным микросхемам ТТЛ-ТТЛШ.

Условно-графические (УГО) и буквенно-цифровые обозначения логических элементов приведены в приложении.

Пример 4.9. Реализовать устройство с четырьмя входами, логическая функция которого задана таблицей истинности:

n	*A*	*B*	*C*	*D*	*F*
0	0	0	0	0	1
1	0	0	0	1	1
2	0	0	1	0	1
3	0	0	1	1	1
4	0	1	0	0	1
5	0	1	0	1	0
6	0	1	1	0	0
7	0	1	1	1	0
8	1	0	0	0	1
9	1	0	0	1	1
10	1	0	1	0	0
11	1	0	1	1	0
12	1	1	0	0	0
13	1	1	0	1	0
14	1	1	1	0	0
15	1	1	1	1	0

Р е ш е н и е . Представим логическую функцию в виде соответствующей ей карты Карно (рис. 4.7, *а*). На рис. 4.7, *б* представлена таблица соответствия ее клеток наборам таблицы истинности.



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

а *б*

Рис. 4.7. Карта Карно (диаграмма Вейча) исследуемой функции

Организовав блоки по нулям (блоки *AB* и *BD* выделены на карте Карно пунктирной линией), записываем минимизированное выражение для логической функции по карте Карно: $\overline{F} = AB + BC + AC + BD$, которое легко реализовать на микросхеме К555ЛР3 (рис. 4.8).

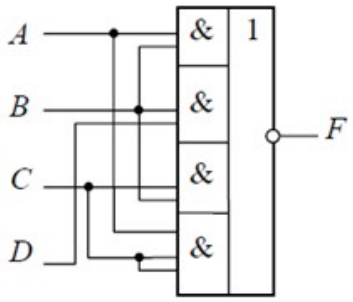


Рис. 4.8. Реализация устройства на микросхеме К555ЛР3

Если блоки организовать по единицам, то их число уменьшится до трех, но потребуются дополнительные инверторы: $F = \overline{A} \cdot \overline{B} + \overline{B} \cdot \overline{C} + \overline{A} \cdot \overline{C} \cdot \overline{D}$.

Пример 4.10. Реализовать в разных базисах логические схемы функции, заданной таблицей истинности:

n	x_1	x_2	x_3	x_4	*F*	*n*	x_1	x_2	x_3	x_4	*F*
0	0	0	0	0	1	8	1	0	0	0	*
1	0	0	0	1	1	9	1	0	0	1	1
2	0	0	1	0	*	10	1	0	1	0	1
3	0	0	1	1	0	11	1	0	1	1	0
4	0	1	0	0	1	12	1	1	0	0	0
5	0	1	0	1	0	13	1	1	0	1	0
6	0	1	1	0	0	14	1	1	1	0	0
7	0	1	1	1	1	15	1	1	1	1	0

Р е ш е н и е . Проведем минимизацию частично определенной функции *F* с помощью диаграммы Вейча. На рис. 4.9 приведен выбор минимизирующих контуров при минимизации логической функции *F* по единичным и нулевым значениям.

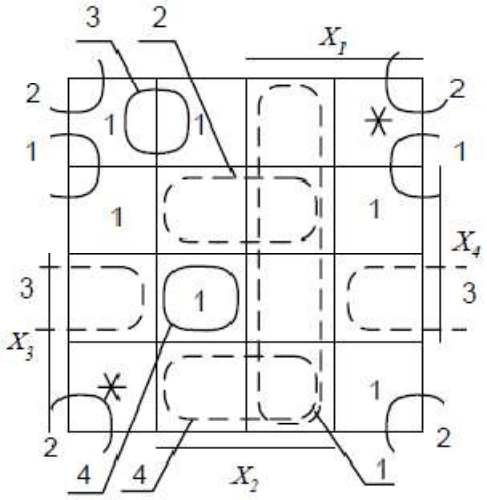


Рис. 4.9. Минимизирующие контуры на карте Карно по единичным (———)
и по нулевым (– – – – –) значениям функции

Для логической функции F (см. рис.4.9), минимизированной по единичным значениям, МДНФ представляется алгебраическим выражением:

$$F^1 = \overline{x}_2 \cdot \overline{x}_3 + \overline{x}_2 \cdot \overline{x}_4 + \overline{x}_1 \cdot \overline{x}_3 \cdot \overline{x}_4 + \overline{x}_1 \cdot x_2 \cdot x_3 \cdot x_4. \qquad (4.3)$$

Минимизация логической функции по приведенному алгоритму может быть выполнена и по нулевым значениям, при этом получают инверсное значение исходной функции также в МДНФ (см. рис. 4.9):

$$\overline{F^0} = x_1 \cdot x_2 + x_2 \cdot \overline{x}_3 \cdot x_4 + \overline{x}_2 \cdot x_3 \cdot x_4 + x_2 \cdot x_3 \cdot \overline{x}_4. \qquad (4.4)$$

Проведем схемную реализацию каждой МДНФ в заданных элементных базисах И-НЕ, И-ИЛИ-НЕ. Перед этим необходимо выбрать серию интегральных микросхем, удовлетворяющих требованиям быстродействия, потребляемой мощности и имеющих наиболее широкий функциональный набор логических элементов. Например, серию К155(КР155). При одновременном использовании логических элементов разных серий необходимо обратить внимание на совместимость их основных электрических и динамических параметров.

При схемной реализации в элементном базисе И-НЕ необходимо предварительное преобразование алгебраического выражения с помощью законов инверсии (теорем де Моргана) к такому виду, в котором используется только конъюнкция и инверсия, при этом каждое элементарное произведение логических переменных (импликанта) рассматривается как некоторая эквивалентная логическая переменная. Для получения инверсных значений логических переменных и их функций целесообразно использовать специализированную цифровую микросхему в виде блока инверторов К155ЛН1. К примеру,

логическая функция F, минимизированная по ее единичным значениям (4.4), для схемной реализации в элементном базисе И-НЕ приводится к виду

$$F^1 = \overline{\overline{F^1}} = \overline{\overline{\overline{x}_2 \cdot \overline{x}_3} \cdot \overline{\overline{x}_2 \cdot \overline{x}_4} \cdot \overline{\overline{x}_1 \cdot \overline{x}_3 \cdot \overline{x}_4} \cdot \overline{\overline{x}_1 \cdot x_2 \cdot x_3 \cdot x_4}}, \qquad (4.5)$$

а структурная схема в соответствии с выражением (4.5) приведена на рис. 4.10.

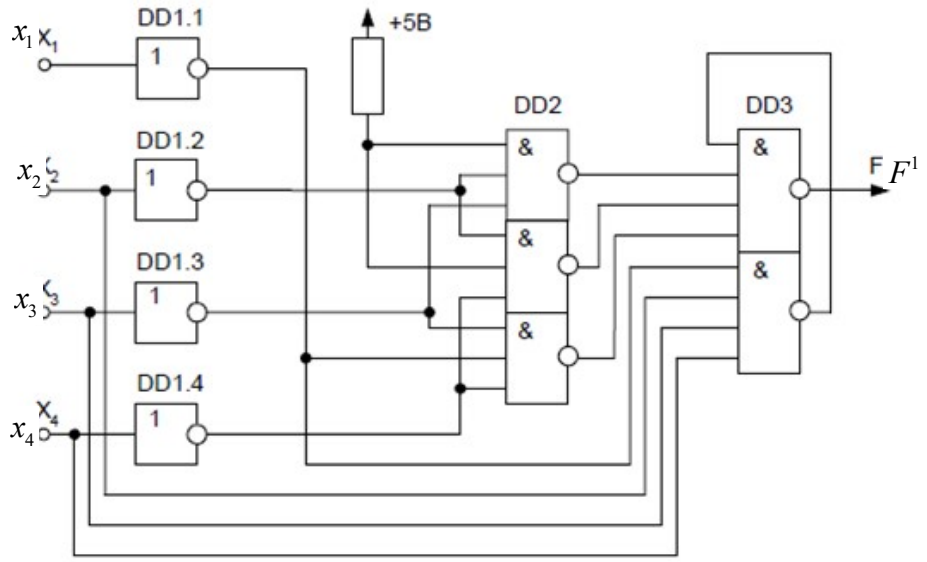


Рис. 4.10. Структурная схема реализации логической функции F^1 в базисе И-НЕ: DD1 – К155ЛН, DD2 – К155ЛА4, DD3 – К155ЛА1

При схемной реализации в базисе И-НЕ свободные информационные входы логических элементов с целью повышения помехоустойчивости цифрового устройства подключаются к источнику питания через резистор сопротивлением 1 кОм, при этом к одному резистору может быть подключено до двадцати информационных входов цифровых микросхем

При схемной реализации логических функций F^1 и F^2 в базисе И-НЕ целесообразно оптимизировать обобщенную структуру путем использования выходных сигналов логических элементов, реализующих одинаковые импликанты обеих функций.

При схемной реализации минимизированных логических функций, представленных в МДНФ, в элементном базисе И-ИЛИ-НЕ необходимо хотя бы один информационный вход неиспользуемого элемента И подключить к общей шине, в противном случае он будет постоянно находиться в единичном состоянии, а в целом элемент И-ИЛИ-НЕ – в нулевом состоянии независимо от комбинации информационных сигналов на других входах.

Необходимо обратить внимание на то, что цифровые микросхемы И-ИЛИ-НЕ часто имеют дополнительный вход расширения по ИЛИ (входы *A*, *B* микросхем К155ЛР1, К155ЛР3, К155Л4 и др.) для подключения

выходов микросхем расширителей по ИЛИ К155ЛД1, К155ЛД3, что существенно увеличивает функциональные возможности исходных цифровых микросхем. При этом неиспользуемые входы расширения по ИЛИ микросхем И-ИЛИ-НЕ остаются свободными в отличие от входов элементов И. Структурная схема реализации логической функции F^1 в соответствии с алгебраическим выражением (4.4) приведена на рис. 4.11.

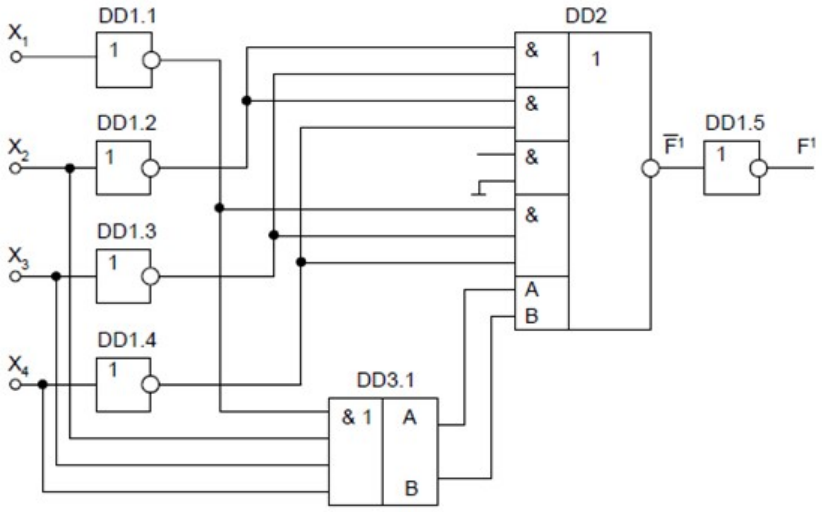


Рис. 4.11. Структурная схема реализации логической функции F^1 в базисе И-ИЛИ-НЕ: DD1 – К155ЛН1, DD2 – К155ЛР3, DD3 – К155ЛД1

Задания для самостоятельной работы

1. Приведите к базисам И-НЕ и ИЛИ-НЕ логические функции:

а) $y = x_1x_2 \vee x_1x_3 \vee x_2x_3$;

б) $y = (x_1 \vee x_2)\cdot(x_1 \vee x_3)\cdot(x_2 \vee x_3)$;

в) $y = x_3x_0 + \left(\overline{\overline{x}_3x_2\overline{x}_0}\right)(\overline{x}_2 + x_1)$;

г) $f = x_1x_4 + \left(\overline{\overline{x}_1 + x_3 + \overline{x}_4}\right)(x_2 + \overline{x}_3)$.

2. Синтезируйте в базисах И, ИЛИ, НЕ и в И-НЕ, ИЛИ-НЕ устройства, выходной сигнал которых равен 1 только в том случае, если на двух их входах x_1 и x_2 действуют:

а) различные сигналы (узел неравнозначности, сумматор по модулю два);

б) одинаковые сигналы (узел равнозначности).

3. Синтезируйте в базисах И, ИЛИ, НЕ, И-НЕ и ИЛИ-НЕ логические схемы, реализующие логические выражения:

а) $F = \overline{A}D + A\overline{B}\overline{C}$;

б) $F = a\overline{b} \vee c$;

в) $f = x(y \vee z)$;

г) $F = (\overline{B} + \overline{C})(\overline{C} + \overline{D})$;

д) $Y = \overline{A}B\overline{C} \vee \overline{A}BC \vee A\overline{B}C \vee ABC$

е) $y = \overline{(x_1x_2 + x_1x_3 + x_2x_3)}(x_1 + x_2 + x_3) + x_1x_2x_3$.

4. Синтезируйте мажоритарный элемент на три и четыре входа в базисах ИЛИ-НЕ и И, ИЛИ, НЕ. У такого элемента значение выходного сигнала совпадает с значением большинства входных.

5. Синтезировать логические схемы:

а) в базисе И, ИЛИ, НЕ для функции, заданной таблицей истинности по СДНФ и СКНФ:

x_1	0	0	0	0	1	1	1	1
x_2	0	0	1	1	0	0	1	1
x_3	0	1	0	1	0	1	0	1
$F(x_1, x_2, x_3)$	0	0	1	1	0	1	0	1

б) для минимальной ЛФ.

6. Реализуйте булеву функцию $F = (A \equiv B)\overline{C}$ на логических элементах ИЛИ-НЕ серии К555.

7. Приведите ФАЛ $y = x_3x_0 + \overline{x}_3\overline{x}_2x_1 + x_2x_1\overline{x}_0$ к базису 2И-НЕ, синтезируйте схемы исходной и приведенной функций.

8. Постройте логическое устройство в базисах И-НЕ и ИЛИ-НЕ, реализующее функцию, приведенную в таблице истинности:

x_1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
x_2	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
x_3	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
x_4	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
$F(x_1, x_2, x_3, x_4)$	0	0	1	1	0	0	0	1	1	1	0	0	1	1	0	1

9. Синтезируйте логические схемы в базисе И-НЕ, соответствующие выражениям:

а) $f = a \cdot b + \overline{a} \cdot c$;

б) $f(x_1, x_2, x_3) = x_3x_1 \vee \overline{x}_2x_1 \vee \overline{x}_3x_2\overline{x}_1$.

10. Запишите в базисе ИЛИ-НЕ минимальную КНФ функции

$$f(x_1, x_2, x_3) = (\overline{x}_3 \vee x_2 \vee x_1)(x_3 \vee x_2 \vee \overline{x}_1)$$

и постройте схему.

11. Запишите в базисе И-ИЛИ-НЕ инверсное значение функции

$$f(x_1, x_2, x_3) = x_3\overline{x}_1 \vee \overline{x}_2\overline{x}_1 \vee \overline{x}_3x_2x_1$$

и приведите схему.

12. В базисах И-НЕ, И-ИЛИ-НЕ реализуйте схему ЛФ, заданной таблицей истинности:

n	x_1	x_2	x_3	x_4	F	n	x_1	x_2	x_3	x_4	F
0	0	0	0	0	1	8	1	0	0	0	*
1	0	0	0	1	1	9	1	0	0	1	1
2	0	0	1	0	*	10	1	0	1	0	1
3	0	0	1	1	0	11	1	0	1	1	0
4	0	1	0	0	1	12	1	1	0	0	0
5	0	1	0	1	0	13	1	1	0	1	0
6	0	1	1	0	0	14	1	1	1	0	0
7	0	1	1	1	1	15	1	1	1	1	0

13. Получите МДНФ функции:

$$f(x_1,x_2,x_3,x_4) = x_1 \cdot x_2 \cdot \overline{x}_3 \cdot \overline{x}_4 + x_1 \cdot x_2 \cdot x_3 \cdot \overline{x}_4 + x_1 \cdot x_2 \cdot x_3 \cdot x_4 + x_1 \cdot \overline{x}_2 \cdot x_3 \cdot \overline{x}_4 + \\ + x_1 \cdot \overline{x}_2 \cdot x_3 \cdot x_4 + \overline{x}_1 \cdot \overline{x}_2 \cdot x_3 \cdot x_4 + \overline{x}_1 \cdot \overline{x}_2 \cdot x_3 \cdot \overline{x}_4 + \overline{x}_1 \cdot \overline{x}_2 \cdot \overline{x}_3 \cdot \overline{x}_4.$$

Постройте схему на логических элементах И-НЕ. Реализуйте ее, используя только двухвходовые элементы И-НЕ.

14. Синтезируйте с использованием двухвходовых элементов И-НЕ логическую функцию, МДНФ которой представляется выражением:

$$f(x_1,x_2,x_3,x_4) = x_1 \cdot x_2 + x_3 \cdot x_4 + \overline{x}_1\overline{x}_2.$$

15. Проведите минимизацию и синтез в базисе И-НЕ функции

$$F = AB\overline{C}D + AB\overline{C}\,\overline{D} + \overline{A}\,\overline{B}CD + A\overline{B}CD + ABCD.$$

16. Проведите минимизацию и синтез в базисе ИЛИ-НЕ функции

$$F = ABC\overline{D} + AB\overline{C}\,\overline{D} + AB\overline{C}D + \overline{A}\,\overline{B}\,\overline{C}D + \overline{A}\,\overline{B}CD.$$

17. Спроектируйте схему устройства на микросхемах серии К155, используя диаграмму Вейча:

		A		
	X	1	0	1
C	1	0	X	0
		B		

18. Реализуйте на логических элементах ИЛИ-НЕ серии К555 булеву функцию $F = \left[\left(A \equiv B\right) + C \cdot \left(B \oplus C\right)\right] \cdot D$.

19. Проведите оптимизацию методом карт Карно и построить в базисах И-НЕ, ИЛИ-НЕ, смешанном оптимальную схему функций:

а) $X(a,b,c,d) = \sum 4,6,12,14,11,13.$ Неопределенные условия – 10, 15;

б) $X(a,b,c,d) = \prod 0,1,4,5,7,13,15.$ Неопределенные условия – 9, 11, 14;

в) $X(a,b,c,d) = \sum 1,2,8,9,10,11,12,13$. Неопределенные условия – 14, 15, 3;

г) $X(a,b,c,d) = \prod 2,3,6,7,9,0,1$. Неопределенные условия – 8,12,13;

д) $X(a,b,c,d) = \prod 1,2,5,6,7,9,12,13$. Неопределенные условия – 14, 15, 3, 11;

е) $X(a,b,c,d) = \sum 0,4,8,12,6,2,10,3,7,9,13$. Неопределенные условия – 14;

ж) $X(a,b,c,d) = \sum 4,5,12,13,2,3,6,7,8,10,11$. Неопределенное условие – 0;

з) $X(a,b,c,d) = \sum 2,3,6,7,8,9,11,12,14$;

и) $X(a,b,c,d) = \prod 2,3,6,7,8,9,11,12,14$.

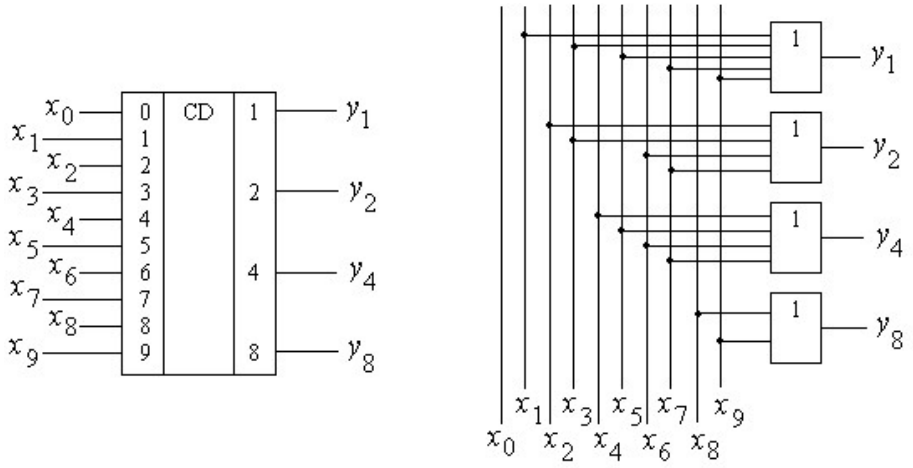
5. ПРЕОБРАЗОВАТЕЛИ КОДОВ. ШИФРАТОРЫ И ДЕШИФРАТОРЫ

Преобразователи кодов – комбинационные устройства, которые преобразуют код одной системы счисления в код другой системы. Шифраторы и дешифраторы представляют собой простейшие преобразователи кодов.

Шифратор (*coder*) – комбинационное устройство, преобразующее десятичные числа (позиционный код) в двоичную систему счисления, причем каждому входу может быть поставлено в соответствие десятичное число, а набор выходных логических сигналов соответствует определенному двоичному коду.

Шифратор содержит m входов и n выходов. Если в шифраторе используются все возможные комбинации сигналов на входе, то он называется полным шифратором. Число входов и выходов в полном шифраторе связано соотношением $n = 2^m$, если часть выходных наборов не используется и $n < 2^m$, то шифратор неполный.

На рис. 5.1 представлен шифратор, осуществляющий преобразование десятичных чисел в двоичный код 8421. Работа шифратора описывается таблицей истинности (табл. 5.1), где каждому десятичному числу ставится в соответствие двоичный эквивалент.



а *б*

Рис. 5.1. Шифратор с прямыми входами и выходами:
а – УГО; *б* – функциональная схема

Таблица 5.1

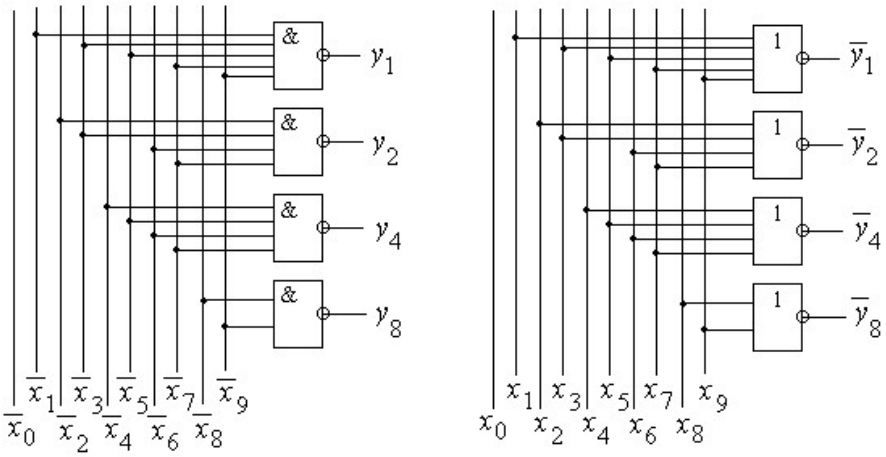
Таблица истинности шифратора

Входы (десятичное число)	Выходы (код 8421)			
	y_4	y_3	y_2	y_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

На основании табл. 5.1 получим следующие логические выражения для ФАЛ:

$$y_1 = x_1 + x_3 + x_5 + x_7; \quad y_2 = x_2 + x_3 + x_6 + x_7;$$
$$y_3 = x_4 + x_5 + x_6 + x_7; \quad y_4 = x_8 + x_9.$$

Возможна реализация шифратора в базисах И-НЕ (рис. 5.2, *а*) и ИЛИ-НЕ (рис. 5.2, *б*).



а *б*

Рис. 5.2. Схема шифратора в базисах: *а* – И-НЕ; *б* – ИЛИ-НЕ

Выполненный на элементах И-НЕ шифратор имеет инверсные входы. Следовательно, система логических выражений для этого шифратора имеет вид:

$$y_1 = \overline{\overline{x_1 \vee x_3 \vee x_5 \vee x_7 \vee x_9}} = \overline{\overline{x}_1\overline{x}_3\overline{x}_5\overline{x}_7\overline{x}_9} = \overline{x}_1 \,|\, \overline{x}_3 \,|\, \overline{x}_5 \,|\, \overline{x}_7 \,|\, \overline{x}_9;$$

$$y_2 = \overline{x}_2 \,|\, \overline{x}_3 \,|\, \overline{x}_6 \,|\, \overline{x}_7; \quad y_4 = \overline{x}_4 \,|\, \overline{x}_5 \,|\, \overline{x}_6 \,|\, \overline{x}_7; \quad y_8 = \overline{x}_8 \,|\, \overline{x}_9.$$

Система логических выражений для шифратора, выполненного на элементах ИЛИ-НЕ с инверсными выходами, имеет вид:

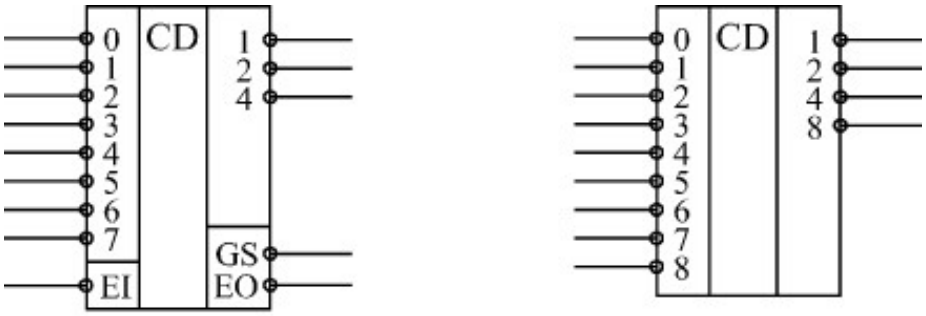
$$\overline{y}_1 = \overline{\overline{x_1 \vee x_3 \vee x_5 \vee x_7 \vee x_9}} = x_1 \downarrow x_3 \downarrow x_5 \downarrow x_7;$$

$$\overline{y}_2 = x_2 \downarrow x_3 \downarrow x_6 \downarrow x_7;\ \overline{y}_4 = x_4 \downarrow x_5 \downarrow x_6 \downarrow x_7;$$

$$\overline{y}_8 = x_8 \downarrow x_9.$$

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

В отечественных сериях шифраторы имеют в названии буквы ИВ. На рис. 5.3 представлены две микросхемы приоритетных шифраторов ИВ1 и ИВ3. Микросхема ИВ1 (рис. 5.3, *а*) имеет 8 входов и 3 выхода (шифратор 8-3), а ИВ3 – 9 входов и 4 выхода (шифратор 9-4). Все входы и выходы шифраторов – инверсные (активные входные сигналы – нулевые, а на выходе формируется инверсный код . Микросхема ИВ1, помимо 8 информационных входов и 3 разрядов выходного кода (1, 2, 4), имеет инверсный вход разреш ния – *EI*, выход признака прихода любого входного сигнала – *GS*, а также выход переноса – *EO*, позволяющий объединять несколько шифраторов для увеличения разрядности.



а *б*

Рис. 5.3. Микросхемы шифраторов: *а* – ИВ1, *б* – ИВ3

Для синтеза шифратора строится таблица истинности, на основании которой получают аналитические зависимости выходов от входов. Затем, если необходимо, выражения преобразовывают к требуемому базису и строят схему.

Пример 5.1. Построить шифратор для перевода десятичных чисел от 3 до 9 в двоичный код 8421 в базисе ИЛИ-НЕ.

Р е ш е н и е . Строим таблицу истинности шифратора (табл. 5.2).

Таблица 5.2

Таблица истинности шифратора

Входы	Выходы (код 8421)			
	y_4	y_3	y_2	y_1
x_3	0	0	1	1
x_4	0	1	0	0
x_5	0	1	0	1
x_6	0	1	1	0
x_7	0	1	1	1
x_8	1	0	0	0
x_9	1	0	0	1

Записываем выражения ДНФ каждого из выходов путем сложения тех аргументов, у которых выход будет находиться в состоянии лог. «1»:

$$y_4 = x_8 \vee x_0; \quad y_3 = x_4 \vee x_5 \vee x_5 \vee x_7;$$

$$y_2 = x_3 \vee x_6 \vee x_7; \quad y_1 = x_3 \vee x_5 \vee x_7 \vee x_9.$$

Затем по закону двойного отрицания преобразовываем выражения к базису ИЛИ-НЕ и строим схему (рис. 5.4):

$$y_4 = \overline{\overline{x_8 \vee x_9}} = \overline{x_8 \downarrow x_9}; \quad y_3 = \overline{\overline{x_4 \vee x_5 \vee x_6 \vee x_7}} = \overline{x_4 \downarrow x_5 \downarrow x_6 \downarrow x_7};$$

$$y_2 = \overline{\overline{x_3 \vee x_6 \vee x_7}} = \overline{x_3 \downarrow x_6 \downarrow x_7}; \quad y_1 = \overline{\overline{x_3 \vee x_5 \vee x_7 \vee x_9}} = \overline{x_3 \downarrow x_5 \downarrow x_7 \downarrow x_9}$$

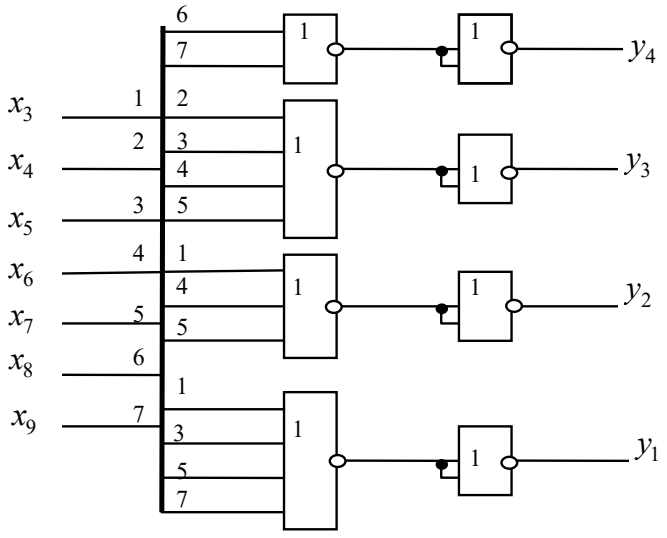


Рис. 5.4. Схема шифратора в базисе ИЛИ-НЕ

Дешифратор (*decoder*) – это комбинационное цифровое устройство, преобразующее двоичный код в код «1 из *n*» (унитарный десятичный код). Дешифратор позволяет распознать *m*-разрядное входное двоичное число путём получения логической единицы на одном из выходов.

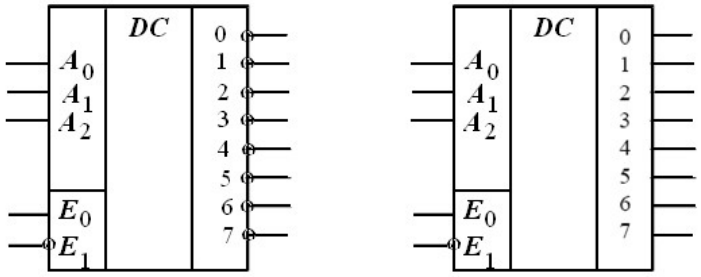
Если у дешифратора m-входов, то он должен иметь $n = 2^m$ выходов, соответствующих числу разных комбинаций в m-разрядном двоичном коде. Если у дешифратора используются все входные комбинации, то его называют полным. Если часть входных наборов не используется, то дешифратор называют неполным, $n < 2^m$.

Дешифратор выполняет преобразование всех комбинаций значений n булевых аргументов в один из 2^n выходных сигналов, который будет соответствовать минтерму m_i (или макстерму M_0 для дешифратора с инверсными выходами). Поскольку выходной сигнал появляется только на одном выходе дешифратора, то говорят, что дешифратор преобразует двоичный код в «позиционный», то есть значение выходной функции дешифратора зависит от того, в какой позиции располагается выходной сигнал.

Принципы построения схем дешифраторов остаются одинаковыми для любой разрядности двоичного кода.

Для обозначения микросхем дешифраторов применяется обозначение ИД. В интегральном исполнении выпускаются дешифраторы как с прямыми выходами, так и с инверсными, а также дополнительным входом (входами) стробирования, разрешающим процесс дешифрации.

Современные логические дешифраторы имеют три адресных входа (A_0, A_1, A_2), два разрешающих (E_0, E_1) и восемь информационные выходов (с 0 по 7). На рис. 5.5 представлены условно-графические обозначения (УГО) дешифратора, различающиеся уровнями активного сигнала на выходах.



а *б*

Рис. 5.5. УГО дешифратора с уровнями активного сигнала на выходе лог. «0» (*а*) и лог. «1» (*б*)

Десятичный номер активизированного выхода при этом соответствует двоичному эквиваленту входного кода (табл. 5.3, см. рис. 5.5, *а*). На остальных выходах дешифратора при этом устанавливается уровень лог. «1» (см. рис. 5.5, *а*) или уровень лог. «0» (см. рис. 5.5, *б*). Таким образом, входной двоичный код адресует соответствующий выход, поэтому эти входы дешифратора и называют адресными.

Таблица 5.3

Обобщенная таблица истинности дешифратора

Входы					Выходы							
Разрешения		Адресные			Информационные							
E_1	E_0	A_2	A_1	A_0	0	1	2	3	4	5	6	7
0	1	0	0	0	0	1	1	1	1	1	1	1
0	1	0	0	1	1	0	1	1	1	1	1	1
0	1	0	1	0	1	1	0	1	1	1	1	1
0	1	0	1	1	1	1	1	0	1	1	1	1
0	1	1	0	0	1	1	1	1	0	1	1	1
0	1	1	0	1	1	1	1	1	1	0	1	1
0	1	1	1	0	1	1	1	1	1	1	0	1
0	1	1	1	1	1	1	1	1	1	1	1	0
Прочие комбинации		*	*	*	1	1	1	1	1	1	1	1

Большинство дешифраторов имеют один или несколько входов стробирования (разрешения) – E_0 и E_1. При разрешающей комбинации $E_0 = 1$ и $E_1 = 0$ (см. рис. 5.5) функционирование дешифратора разрешено. При прочих комбинациях E_0 и E_1 независимо от состояния адресных входов на всех выходах дешифратора формируются сигналы лог. «1» (см. рис. 5.5, *а*) или лог. «0» (см. рис. 5.5, *б*).

Пример 5.2. Построить линейный дешифратор для перевода двоичных чисел от 3 до 8 кода 8421 в десятичный код в базисе И-НЕ.

Р е ш е н и е . Составим таблицу истинности дешифратора (табл. 5.4).

Таблица 5.4

Таблица истинности дешифратора

Входы				Выходы
X_4	X_3	X_2	X_1	
0	0	1	1	Y_3
0	1	0	0	Y_4
0	1	0	1	Y_5
0	1	1	0	Y_6
0	1	1	1	Y_7
1	0	0	0	Y_8

В соответствии с табл. 5.4 записываем КНФ для каждого выхода:

$$Y_3 = \overline{X_4}\,\overline{X_3}X_2X_1; \quad Y_4 = \overline{X_4}X_3\overline{X_2}\,\overline{X_1};$$

$$Y_5 = \overline{X_4}X_3\overline{X_2}X_1; \quad Y_6 = \overline{X_4}X_3X_2\overline{X_1};$$

$$Y_7 = \overline{X_4}X_3X_2X_1; \quad Y_8 = X_4\overline{X_3}\,\overline{X_2}\,\overline{X_1}.$$

Преобразуем выражения к базису И-НЕ и строим схему (рис. 5.6):

$$Y_3 = \overline{\overline{X_4}\,\overline{X_3}X_2X_1} = \overline{\overline{X_4}\,|\,\overline{X_3}\,|\,X_2\,|\,X_1};\quad Y_4 = \overline{\overline{X_4}X_3\overline{X_2}\,\overline{X_1}} = \overline{\overline{X_4}\,|\,X_3\,|\,\overline{X_2}\,|\,\overline{X_1}};$$

$$Y_5 = \overline{\overline{X_4}X_3\overline{X_2}X_1} = \overline{\overline{X_4}\,|\,X_3\,|\,\overline{X_2}\,|\,X_1};\quad Y_6 = \overline{\overline{X_4}X_3X_2\overline{X_1}} = \overline{\overline{X_4}\,|\,X_3\,|\,X_2\,|\,\overline{X_1}};$$

$$Y_7 = \overline{\overline{\overline{X_4}X_3X_2X_1}} = \overline{\overline{X_4}\,|\,X_3\,|\,X_2\,|\,X_1};\quad Y_8 = \overline{X_4\overline{X_3}\,\overline{X_2}\,\overline{X_1}} = \overline{X_4\,|\,\overline{X_3}\,|\,\overline{X_2}\,|\,\overline{X_1}}.$$

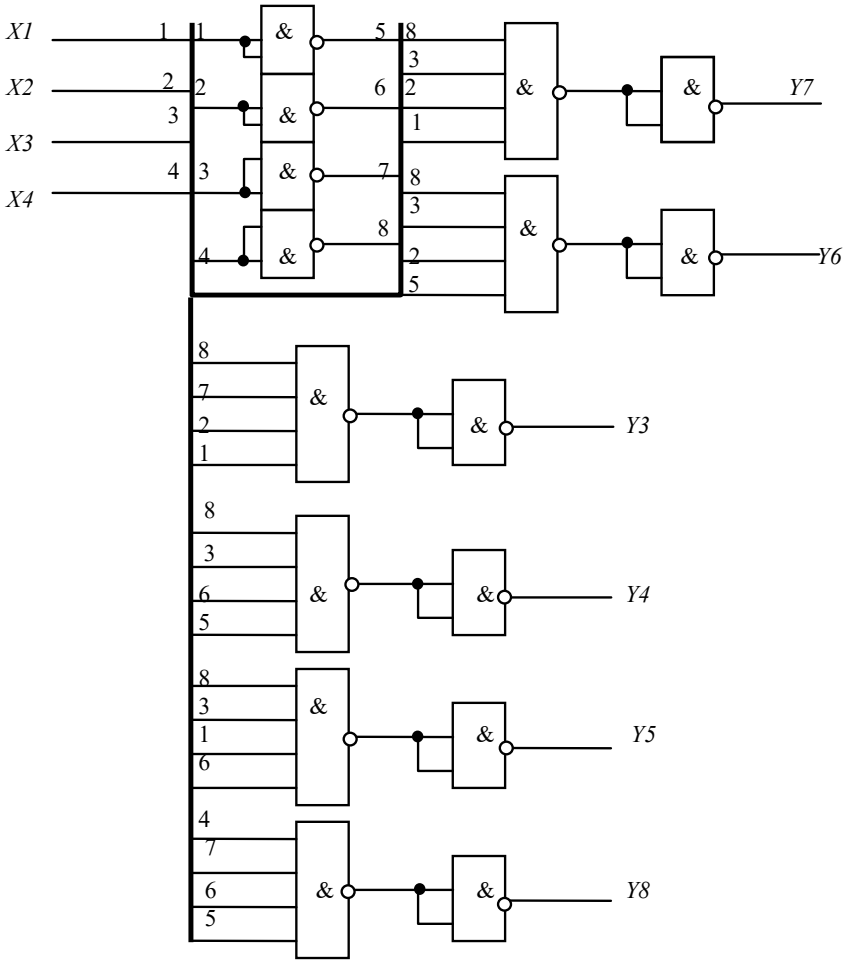


Рис. 5.6. Схема дешифратора в базисе И-НЕ

Преобразователи кодов

Под преобразованием кодов понимается преобразование *n*-разрядных двоичных чисел, представляющих информацию в одном заданном коде, в *m*-разрядные, представляющие эту информацию в другом коде. Наиболее распространены следующие два подхода к построению преобразователей кодов:

1) синтез *m* независимых одновыходных функций по заданной таблице истинности – таблице соответствия кодов;

2) построение преобразователя кодов по методу «дешифратор-шифратор».

Алгоритм синтеза ПК с использованием свойства независимости входов и выходов следующий:

1. Определяется количество двоичных чисел, которые необходимо перевести из одного кода в другой.

2. Составляется таблица истинности, в которой описываются входные и выходные коды двоичного числа.

3. Выполняется синтез по методике построения устройств с несколькими выходами и не полностью определяемыми функциями (так как выходы независимы друг от друга, то для каждого из них составляется логическая функция и затем строится схема на общих входах). Обязательным этапом в данном случае является минимизация функций выходов одним из известных методов.

Пример 5.3. Синтезировать преобразователь 2-х разрядного двоичного кода в 3-х разрядный по таблице истинности (табл. 5.5).

Таблица 5.5

Таблица истинности

Входы		Выходы		
a_1	a_0	b_2	b_1	b_0
0	0	1	0	0
0	1	0	0	1
1	0	0	1	1
1	1	0	0	0

Решение. Считая b_0, b_1 и b_2 независимыми одновыходными функциями, запишем для каждой из них булевые выражения: $b_0 = \overline{a}_1 a_0 + a_1 \overline{a}_0$; $b_1 = a_1 \overline{a}_0$; $b_2 = \overline{a}_1 \overline{a}_0$.

Используя приведенные булевы выражения с учетом того, что инверсии некоторых переменных и произведение $a_1 \overline{a}_0$ встречаются ни в одной функции, составляем схему преобразователя с применением, где возможно, одних и тех же логических элементов (рис. 5.7).

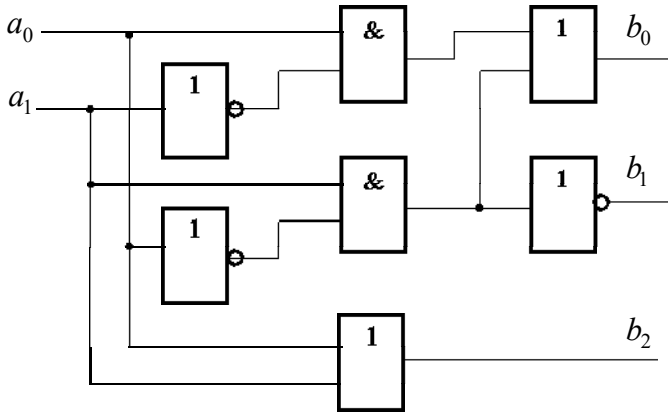


Рис. 5.7. Схема преобразователя 2-х разрядного двоичного кода в 3-х разрядный

Построение преобразователя кодов по принципу «дешифратор – шифратор» заключается в следующем:

1. Строится схема преобразователя двоичного кода в унарный, т. е. такой код, когда логическая «1» может быть только на одном из N выходов преобразователя, номер которого совпадает с числом, представленным входным двоичным кодом. Такой преобразователь называется дешифратором. Число выходов дешифратора равно $N = 2^n$, где n – число разрядов входного кода дешифратора.

2. Строится схема преобразователя, осуществляющего обратную операцию, т. е. преобразование унарного кода в двоичный – шифратор. Число входов такого преобразователя, равно $M = 2^m$, где m – число разрядов выходного кода шифратора.

Применительно к случаю, рассмотренному в примере 5.3 по таблице истинности (см. табл. 5.5) составляется таблица соответствия десятичных цифр (табл. 5.6).

Таблица 5.6

Таблица соответствия десятичных цифр

A	0	1	2	3
B	4	1	3	0

Схема преобразователя кода образуется соединением выходов дешифратора и входов шифратора в соответствии с табл. 5.6 (рис. 5.8).

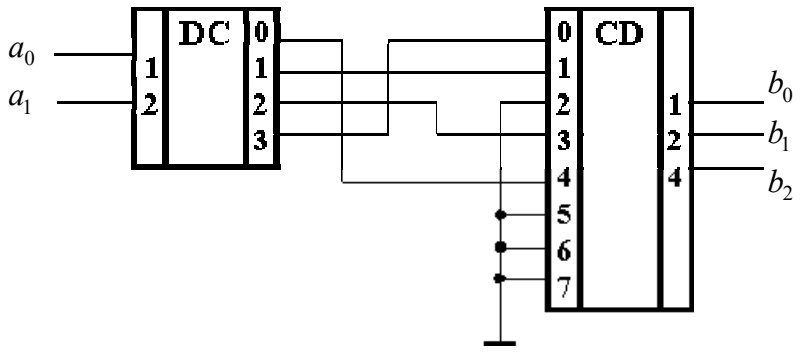


Рис. 5.8. Схема преобразователя кода по принципу «дешифратор – шифратор»

Пример 5.4. Синтезировать преобразователь кода прямого замещения в двоично-десятичный код 2421.

Р е ш е н и е. Код прямого замещения представляет собой обычное представление одноразрядного десятичного числа в двоичной системе счисления:

$$x_1 \cdot 2^3 + x_2 \cdot 2^2 + x_3 \cdot 2^1 + x_4 \cdot 2^0 = 8x_1 + 4x_2 + 2x_1 + x_4 .$$

Двоично-десятичный код 2421 соответствует представлению числа в виде: $y_1 \cdot 2^1 + y_2 \cdot 2^2 + y_3 \cdot 2^1 + y_4 \cdot 2^0 = 2y_1 + 4y_2 + 2y_1 + y_4$.

Таким образом, преобразователь кодов представляет собой схему с четырьмя входами и четырьмя выходами.

Составим таблицу истинности для логической функции преобразователя кодов (табл. 5.7).

Таблица 5.7

Таблица истинности преобразователя кодов

Десятичное число	Код прямого замещения				Выходы (код 2421)			
	x_1	x_2	x_3	x_4	y_1	y_2	y_3	y_4
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	0
3	0	0	1	1	0	0	1	1
4	0	1	0	0	0	1	0	0
5	0	1	0	1	1	0	1	1
6	0	1	1	0	1	1	0	0
7	0	1	1	1	1	1	0	1
8	1	0	0	0	1	1	1	0
9	1	0	0	1	1	1	1	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	×	×	×	×

Получим логическую функцию преобразователя кодов в виде СДНФ путем записи «по единицам», представленную системой уравнений:

$$y_1 = \overline{x}_1 x_2 \overline{x}_3 x_4 + \overline{x}_1 x_2 x_3 \overline{x}_4 + \overline{x}_1 x_2 x_3 x_4 + x_1 \overline{x}_2 \overline{x}_3 \overline{x}_4 + x_1 \overline{x}_2 \overline{x}_3 x_4;$$

$$y_2 = \overline{x}_1 x_2 \overline{x}_3 \overline{x}_4 + \overline{x}_1 x_2 x_3 \overline{x}_4 + \overline{x}_1 x_2 x_3 x_4 + x_1 \overline{x}_2 \overline{x}_3 \overline{x}_4 + x_1 \overline{x}_2 \overline{x}_3 x_4;$$

$$y_3 = \overline{x}_1 \overline{x}_2 x_3 \overline{x}_4 + \overline{x}_1 \overline{x}_2 x_3 x_4 + \overline{x}_1 x_2 \overline{x}_3 x_4 + x_1 \overline{x}_2 \overline{x}_3 \overline{x}_4 + x_1 \overline{x}_2 \overline{x}_3 x_4;$$

$$y_4 = \overline{x}_1 \overline{x}_2 \overline{x}_3 x_4 + \overline{x}_1 \overline{x}_2 x_3 x_4 + \overline{x}_1 x_2 \overline{x}_3 x_4 + \overline{x}_1 x_2 x_3 x_4 + x_1 \overline{x}_2 \overline{x}_3 x_4.$$

Синтезируемая схема реализует четыре функции. Ее можно представить как простое объединение схем, реализующих каждую функцию отдельно. Но это не экономично. Целесообразно преобразовать совокупность этих функций к такому виду, чтобы реализующие их схемы содержали общие части, а схема с четырьмя выходами представляла собой единое целое.

Для выполнения этого условия, используя избыточные наборы входных переменных x_1, x_2, x_3, x_4, отмеченные на картах Карно крестиками, образуют минимальные покрытия для каждой из функций, которые включали бы возможно больше однотипных объединений клеток на картах.

Таким образом с помощью карт Карно получим логические выражения в виде МДНФ для y_1, y_2, y_3 и y_4 (рис. 5.9).

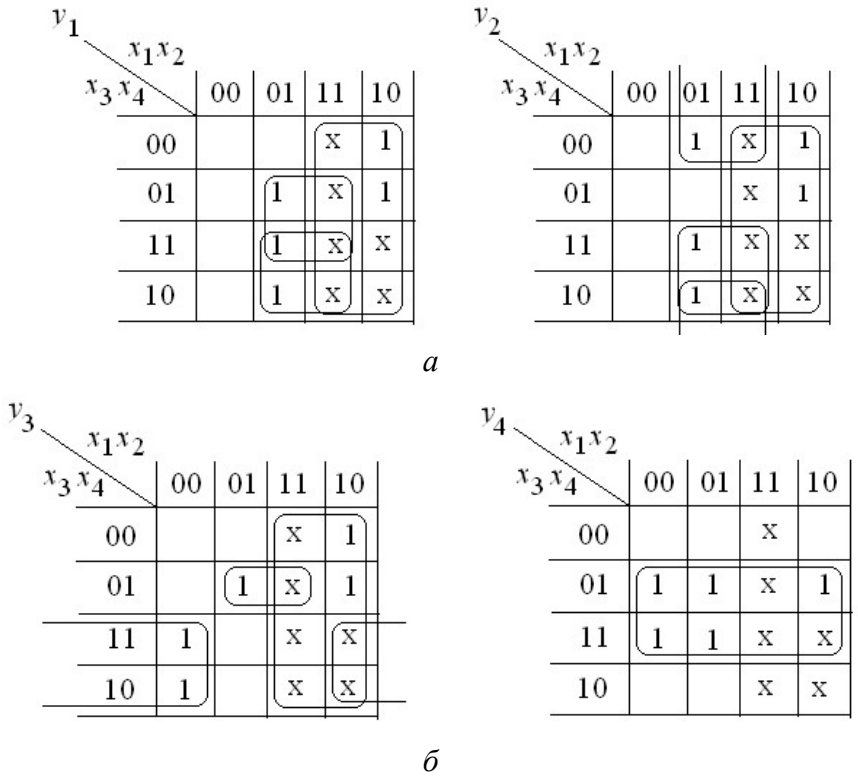


Рис. 5.9. Минимизация функций с помощью Карт Карно: *а* – y_1 и y_2; б – y_3 и y_4

В результате минимизации получим функции:

$$y_1 = x_1 + x_2x_3 + x_2x_4 = x_1 + x_2(x_3 + x_4);\ \ y_2 = x_1 + x_2x_3 + x_2\overline{x}_4 = x_1 + x_2(x_3 + \overline{x}_4);$$

$$y_3 = x_1 + \overline{x}_2x_3 + x_2\overline{x}_3x_4;\quad y_4 = x_4.$$

Составим функциональную схему устройства (рис. 5.10).

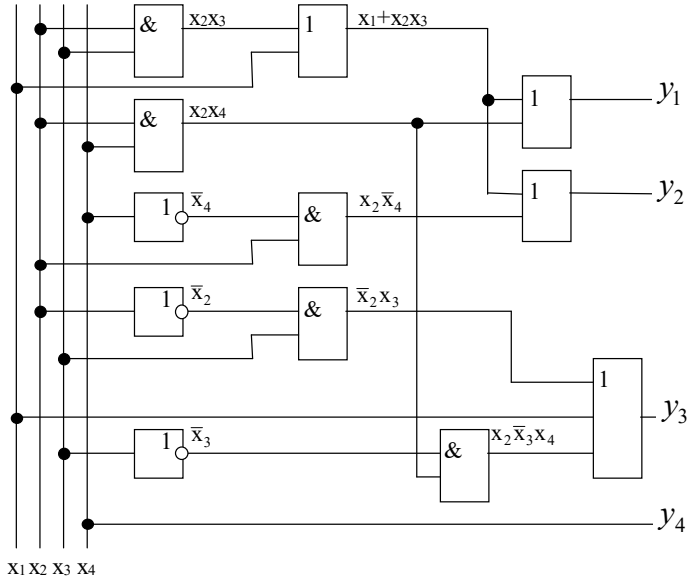


Рис. 5.10. Функциональная схема преобразователя кода прямого замещения в двоично-десятичный код 2421

Реализация произвольных функций алгебры логики с использованием дешифратора

Дешифратор может быть использован для реализации произвольных логических функций.

Поскольку каноническая сумма минтермов любой логической функции есть дизъюнкция конъюнкций аргументов, каждый минтерм которой принимает единичное значение при одном соответствующем наборе значений аргументов, то с помощью полного дешифратора можно реализовать любую логическую функцию. Поэтому дешифраторы часто называют универсальными преобразователями.

Для реализации логической функции на дешифраторе она должна быть задана в виде канонической суммы минтермов, при этом необходимо те выходы дешифратора, на которых появляется логическая единица, при подаче на входы дешифратора комбинаций, соответствующих каждому минтерму входящему в данную каноническую сумму минтермов, объединить через операцию «дизъюнкция».

Пример 5.5. Реализовать ЛФ, зависящую от трех переменных и представленную картой Карно (рис. 5.11):

y \ x_1				
x_2	0	1	0	1
	1	0	1	0
			x_0	

Рис. 5.11. Карта Карно для функции $y(x_0, x_1, x_2)$

Р е ш е н и е . Заданную функцию можно представить:

– в СДНФ:

$$y = 1 + 2 + 4 + 7 = \overline{x}_2\overline{x}_1x_0 + \overline{x}_2x_1\overline{x}_0 + x_2\overline{x}_1\overline{x}_0 + x_2x_1x_0; \qquad (5.1)$$

– в базисе И-НЕ:

$$y = \overline{\overline{1}\cdot\overline{2}\cdot\overline{4}\cdot\overline{7}} = \overline{\overline{x}_2\overline{x}_1x_0}\cdot\overline{\overline{x}_2x_1\overline{x}_0}\cdot\overline{x_2\overline{x}_1\overline{x}_0}\cdot\overline{x_2x_1x_0}; \qquad (5.2)$$

– в СКНФ:

$$y = 0\cdot 3\cdot 5\cdot 6 = (x_2 + x_1 + x_0)(x_2 + \overline{x}_1 + \overline{x}_0)(\overline{x}_2 + x_1 + \overline{x}_0)(\overline{x}_2 + \overline{x}_1 + x_0); \qquad (5.3)$$

– в базисе ИЛИ-НЕ:

$$y = \overline{\overline{0} + \overline{3} + \overline{5} + \overline{6}} = \overline{(\overline{x_2 + x_1 + x_0})(\overline{x_2 + \overline{x}_1 + \overline{x}_0})(\overline{\overline{x}_2 + x_1 + \overline{x}_0})(\overline{\overline{x}_2 + \overline{x}_1 + x_0})}. \qquad (5.4)$$

Поскольку в полном дешифраторе реализуются все конституенты, то для получения ЛФ достаточно добавить к нему один логический элемент. Итак, для реализации ЛФ по уравнению (5.1) требуется дешифратор с активной единицей выхода и четырехвходовый элемент ИЛИ (рис. 5.12, *а*); по (5.2) – дешифратор с активным нулем выхода и четырехвходовый элемент И-НЕ (рис. 5.12, *б*); по (5.3) – дешифратор с активным нулем выхода и четырехвходовый элемент И (рис. 5.12, *в*); по (5.4) – дешифратор с активной единицей выхода и четырехвходовый элемент ИЛИ-НЕ (рис. 5.12, *г*).

Из примера 5.5 следует, что для реализации произвольной ЛФ, зависящей от n переменных, требуются две ИС: дешифратор 1 из 2^n и логический элемент с числом входов не более 2^{n-1}. Отметим также, что если используется дешифратор с открытым коллектором (с активным нулем выхода), то схема на рис. 5.12, *в* может быть реализована без дополнительного элемента И, но с использованием монтажной операции И как показано на рис. 5.12, *д*. А если используется дешифратор, выполненный на элементах ЭСЛ-типа (эмиттерно-связанной логики) с открытым эмиттером (с активной единицей выхода), то схема на рис. 5.12, *а* может быть реализована без дополнительного элемента ИЛИ, но с использованием монтажной операции ИЛИ как показано на рис. 5.12, *е*.

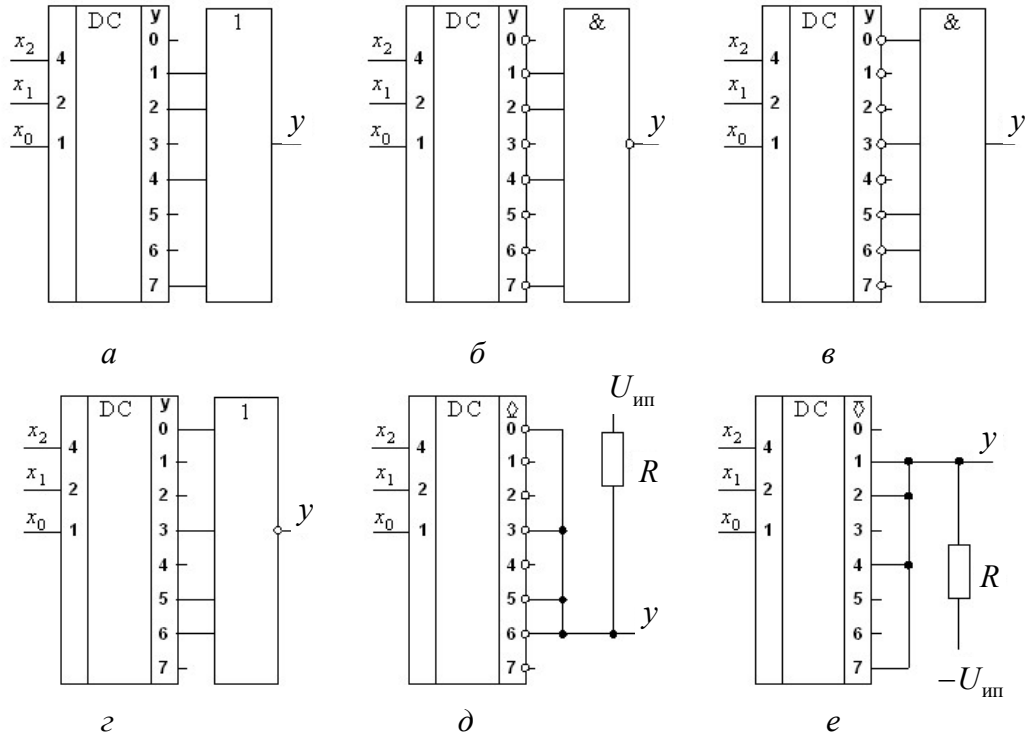


Рис. 5.12. Реализация ЛФ на дешифраторе: *а* – с элементом ИЛИ; *б* – с элементом И-НЕ; *в* – с элементом И; *г* – с элементом ИЛИ-НЕ; *д* – с использованием монтажной операции И; *е* – с использованием монтажной операции ИЛИ

Задания для самостоятельной работы

1.Синтезируйте схему шифратора 4→1 на элементах И-НЕ.

2. Синтезируйте дешифратор 4×10 с прямыми входами и инверсными выходами на элементах И-НЕ и инверторах.

3. Синтезируйте шифратор на пять входов, выход которого представляется в двоичном коде.

4. Синтезируйте в разных базисах: а) полный дешифратор 3×8 с прямыми выходами; полный дешифратор с инверсными выходами; б) дешифратор кода Джонсона.

5. Синтезируйте дешифратор для преобразования двоично-десятичного кода в код, предназначенный для управления десятичным индикатором (дешифратор 4×10).

6. Реализуйте трехразрядный дешифратор с функциями X_1 (СДНФ наборов 0, 4, 7) и X_2 (СКНФ наборов 2, 5, 7).

7. Постройте преобразователь кода в базисе И-НЕ для перевода кода 8421 чисел от 5 до 8 в код 7421 (см. табл. 1.2).

8. Синтезируйте преобразователи кодов в базисе элементов И-НЕ: а) код 8421 в 2421; б) код 2421 в 8421.

9. Спроектируйте дешифратор с тремя входами и шестью выходами $y_0 - y_5$.

10. Синтезируйте дешифраторы: а) полный на два входа на элементах И и инверторах, ИЛИ-НЕ, ИЛИ и инверторах, И-НЕ; б) на два входа со входом разрешения.

11. Постройте дешифратор четырехразрядного кода, если имеются дешифраторы двухразрядного кода.

12. Составьте схему преобразователя кода 8421 в 7421 на элементах: а) И, ИЛИ, НЕ; б) И-НЕ; ИЛИ- НЕ.

13. Постройте двухступенчатый дешифратор для $n = 4$.

14. Составьте схему дешифрации двоичного кода с помощью дешифраторов на три и два входа.

15. Синтезируйте полный шифратор на два выхода ($n = 2$).

16. Постройте в базисе ИЛИ-НЕ схему преобразователя кода для семисегментного индикатора.

17. Составьте схемы: а) линейного полного дешифратора на три входа со входом разрешения и с прямыми выходами; б) полного дешифратора на три входа со входом разрешения и с инверсными выходами.

18. Синтезируйте преобразователи: а) трехразрядного двоичного кода в код Грея; б) четырехразрядного двоичного кода в код Грея.

19. На основе трехразрядных дешифраторов создайте пятиразрядный дешифратор.

20. Синтезируйте преобразователь двоичного числа в код «2 из 5».

21. Реализуйте функцию $F = x_1x_2x_3 + x_1\bar{x}_2\bar{x}_3 + \bar{x}_1\bar{x}_2x_3$ на дешифраторе с прямыми и инверсными выходами с элементами И, И-НЕ, ИЛИ-НЕ.

22. Реализуйте ЛФ $F = x_2x_1\bar{x}_0 + x_2\bar{x}_1x_0 + \bar{x}_2x_1\bar{x}_0 + x_2\bar{x}_1\bar{x}_0 + \bar{x}_2\bar{x}_1\bar{x}_0$ на дешифраторе 3×8 со входом разрешения.

23. Постройте преобразователь двоично-десятичного кода в код с избытком 3.

24. Реализуйте на дешифраторе функции: а) F_1 + F_2: $F_1 = \bar{x}_2\bar{x}_3 + x_3x_2$ и $F_2 = \bar{x}_3x_2x_1 + x_2\bar{x}_1$; б) $F_1 = \bar{x}_2\bar{x}_3 + x_3x_1$ и $F_2 = \bar{x}_3\bar{x}_2x_1 + x_2\bar{x}_1$.

25. С применением дешифратора с четырьмя входами (m = 4) реализуйте функции $f_1 = ab\bar{c} \vee \bar{a}b\bar{c}\,\bar{d}$; $f_2 = a\bar{b}\bar{d} \vee \bar{b}\bar{c}\,\bar{d}$.

26. Реализуйте ЛФ, заданную таблицей истинности (табл. 5.8), на дешифраторе: а) с прямыми выходами с элементом И; б) с инверсными выходами с элементами И, И-НЕ.

Таблица 5.8

Таблица истинности

Минтермы		Входы			Выход
		x_3	x_2	x_1	y
m_0	$\bar{x}_3\bar{x}_2\bar{x}_1$	0	0	0	1
m_1	*	0	0	1	0
m_2	*	0	1	0	0
m_3	$\bar{x}_3x_2x_1$	0	1	1	1
m_4	*	1	0	0	0
m_5	$x_3\bar{x}_2x_1$	1	0	1	1
m_6	$x_3x_2\bar{x}_1$	1	1	0	1
m_7	*	1	1	1	0

27. Синтезируйте схему преобразователя двухразрядного двоичного кода в четырехразрядный по схеме дешифратора и шифратора.

28. Разработайте дешифратор, имеющий восемь входов и один выход на элементах И и ИЛИ-НЕ. Выход дешифратора должен выдавать сигнал, разрешающий прохождение управляющих команд при подаче на вход заданного двоичного числа, при всех других числах выходной сигнал дешифратора должен блокировать прохождение управляющих команд. Разрешающий код – число 105.

29. Разработайте дешифратор, имеющий восемь входов и один выход на элементах И и И-НЕ. Выход дешифратора должен выдавать сигнал, разрешающий прохождение управляющих команд при подаче на вход заданного двоичного числа, при всех других числах выходной сигнал дешифратора должен блокировать прохождение управляющих команд. Разрешающий код – число 150.

30. Реализуйте на дешифраторе ЛФ:

а) $X(a,b,c,d,e) = \sum 5,8,10,15,17,27;$

б) $X(a,b,c,d) = \prod 1,3,7,8,11,12;$

в) $X(a,b,c,d) = \sum 0,5,7,8,12,14;$

г) $X(a,b,c,d) = \prod 0,1,4,8,9,10,14;$

д) $X(a,b,c,d,e) = \prod 2,4,12,13,15,19,23,25;$

е) $X(a,b,c,d) = \sum 1,2,4,6,8,9,12,15.$

6. МУЛЬТИПЛЕКСОРЫ И ДЕМУЛЬТИПЛЕКСОРЫ

Мультиплексор – это комбинационное логическое устройство, предназначенное для управляемой передачи данных от нескольких источников информации в один выходной канал. Мультиплексоры обозначают MUX (*Multiplexor*) или MS (*Multiplexorselector*). Мультиплексор имеет m информационных входов (D_0, D_1, ...), n адресных входов (A_0, A_1, ...) и один выход. Адреса представляются в двоичном коде. Каждому адресу соответствует свой информационный вход, сигнал с которого при данном адресе проходит на выход. Информационные и адресные входы мультиплексора находятся в следующем соотношении: $m = 2^n$.

Кроме информационных и адресных входов, некоторые мультиплекаторы имеют разрешающий (стробирующий) вход E с активным лог. «0», т. е. при $E = 0$ функционирование мультиплексора разрешено.

Условное графическое обозначение мультиплексора, имеющего восемь информационных входов $D_0 - D_7$, три адресных A_0, A_1, A_2 входа, разрешающий E, прямой Y и инверсный $\overline{Y}$ выход, приведено рис. 6.1, *б*. На рис. 6.1, *а* представлена диаграмма Вейча, показывающая связь информационных и адресных входов для 8-канального мультиплексора.

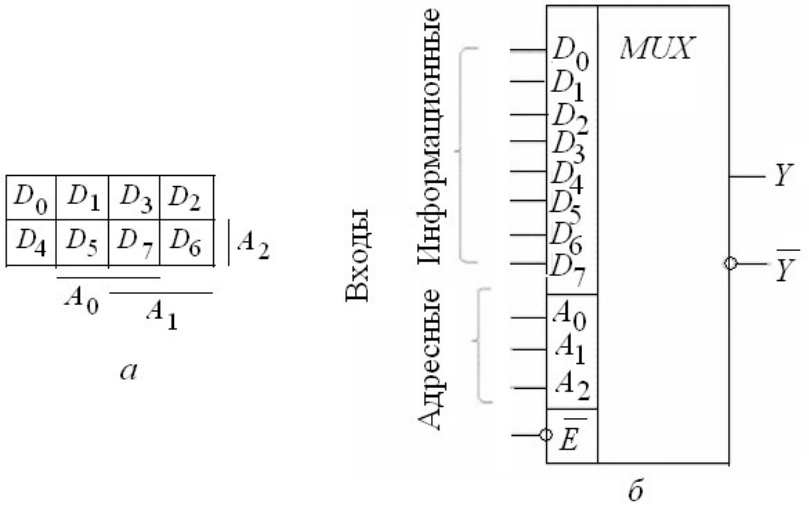


Рис. 6.1. Мультиплексор: *а* – диаграмма Вейча; *б* – УГО

Адресные входы A_0, A_1, A_2 обеспечивают выбор (селекцию) данных, т. е. мультиплексоры способны выбирать определенный канал. Поэтому их иногда называют селекторами.

Мультиплексор, представленный на рис. 6.1, реализует логическую функцию:

$$Y = \overline{E}\left(D_0 \overline{A_2}\,\overline{A_1}\,\overline{A_0} + D_1 \overline{A_2}\,\overline{A_1} A_0 + D_2 \overline{A_2} A_1 \overline{A_0} + D_3 \overline{A_2} A_1 A_0 + D_4 A_2 \overline{A_1}\,\overline{A_0} + \right.$$
$$\left. + D_5 A_2 \overline{A_1} A_0 + D_6 A_2 A_1 \overline{A_0} + D_7 A_2 A_1 A_0 \right) \qquad (6.1)$$

и функционирует согласно табл. 6.1.

Таблица 6.1

Таблица истинности восьмиканального мультиплексора

Входы				Выходы	
E	A_2	A_1	A_0	Y	$\overline{Y}$
0	0	0	0	D_0	$\overline{D_0}$
0	0	0	1	D_1	$\overline{D_1}$
0	0	1	0	D_2	$\overline{D_2}$
0	0	1	1	D_3	$\overline{D_3}$
0	1	0	0	D_4	$\overline{D_4}$
0	1	0	1	D_5	$\overline{D_5}$
0	1	1	0	D_6	$\overline{D_6}$
0	1	1	1	D_7	$\overline{D_7}$
1	×	×	×	0	1

Если на стробирующем входе лог. «0» ($E = 0$), то выбирают канал выбирают в соответствии с двоичным кодом, поданным на управляющие входы A_0, A_1, A_2, и соединяют его с выходами Y и $\overline{Y}$.

Логический сигнал, соответствующий выбранному каналу, проходит на выход Y в прямом виде $\left(D_i\right)$, а на выход $\overline{Y}$ – в инверсном $\left(\overline{D_i}\right)$. Если E= 1, то независимо от сигналов на адресных входах на прямом входе устанавливается сигнал лог. «0», а на инверсном лог. «1».

Мультиплексоры выполнены в виде отдельных ИМС, различаются числом информационных и адресных входов, отсутствием или наличием входа разрешения, а также видами выхода (стандартный и с тремя состояниями) и передачи информации (с инверсией и без). У некоторых мультиплексоров два выхода – инверсный и прямой.

На рис. 6.2 показана структурная схема мультиплексора К155КП7, построенного как совокупность двухвходовых конъюнкторов данных (их число равно числу информационных входов), выходы которых объединены схемой ИЛИ.

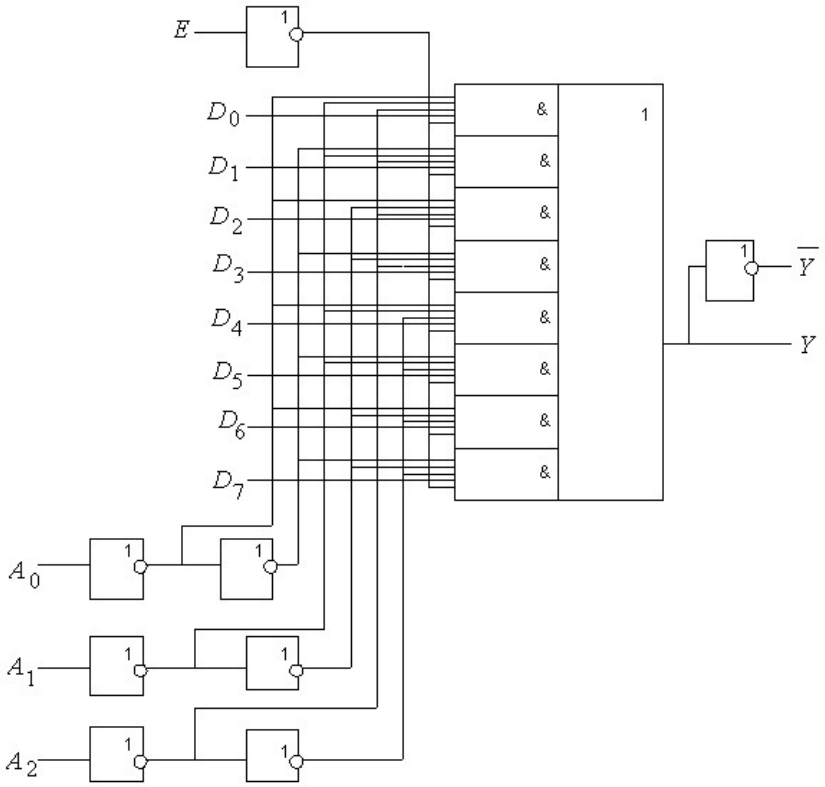


Рис. 6.2. Структурная схема мультиплексора К155КП7

Мультиплексор можно использовать в качестве универсального логического элемента для реализации любой логической функции от числа аргументов, равного числу его адресных входов.

Логическая функция, выполняемая мультиплексором $2^n \rightarrow 1$, по структуре полностью совпадает с канонической суммой минтермов n аргументов. Из этого следует, что любую логическую функцию n аргументов можно реализовать на мультиплексоре структуры $2^n \rightarrow 1$, подав на информационные входы D_i константы – логический нуль или логическую единицу.

На мультиплексорах, не имеющих вход стробирования E, можно реализовать любую логическую функцию $n+1$ входной переменной, а на мультиплексорах, имеющих вход стробирования – логическую функцию $n+2$ переменных, заменяя при этом несколько корпусов логических элементов малой степени интеграции. Здесь n – число адресных входов мультиплексора. Известно несколько методов синтеза таких схем на основе карты Карно, таблицы истинности или логического уравнения.

В зависимости от числа входных переменных булевой функции и параметров мультиплексора существует несколько способов её реализации.

Рассмотрим случай, когда число входных переменных логической функции равно или меньше числа адресных входов мультиплексора.

Наиболее просто логические функции реализуются на мультиплексорах, когда число логических переменных равно количеству их адресных входов.

Пример 6.1. Реализовать ФАЛ $y(x_2,x_1,x_0)$, представленную картой Карно (рис. 6.3, *а*), на мультиплексорах 8→1 и 4→1 (рис. 6.3, *б*, *в*).

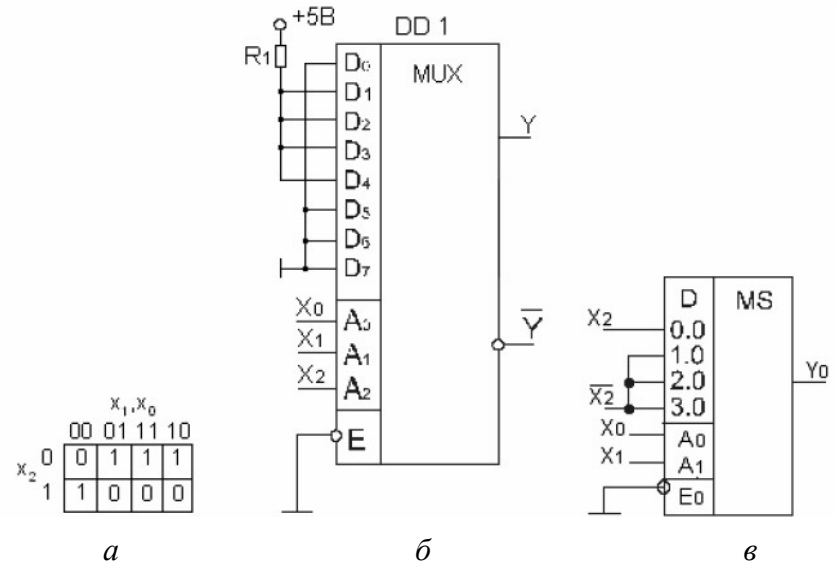


Рис. 6.3. Реализация логической функции: *а* – карта Карно; *б* – на мультиплексоре 8→1 К555КП7; *в* – на мультиплексоре 4→1

Р е ш е н и е. Анализ карты Карно показывает, что $y(0,0,1)=y(0,1,0)=$ $=y(0,1,1)=y(1,0,0)=1$, а $y(0,0,0)=y(1,0,1)=y(1,1,0)=y(1,1,1)=0$. Следовательно, для реализации этой функции на мультиплексоре 8→1 (см. рис. 6.3, *б*) достаточно на адресные входы A_0, A_1, A_2 подать сигналы x_2, x_1, x_0 соответственно, информационные входы D_1, D_2, D_3, D_4 соединить с лог. «1», а D_0, D_5, D_6, D_7 – с лог. «0».

Эту же функцию можно реализовать и на мультиплексоре 4→1. В этом случае воспользуемся непосредственным анализом карты Карно (см. рис. 6.3, *а*). В качестве адресных входов выберем переменные x_1 и x_0.

При последовательном анализе столбцов карты Карно, можно обнаружить, что значение столбца, для которого $x_1x_2=00(\overline{x}_1\overline{x}_0)$, совпадает с x_2, а значение остальных столбцов 01, 11 и 10 – с $\overline{x}_2$. Следовательно, для реализации исходной ФАЛ (см. рис. 6.3, *а*) на мультиплексоре 4→1 следует на адресные входы A_0 и A_1 подать соответственно логические переменные x_0 и x_1, на входы $D_{0.0} - x_2$, на $D_{1.0}$, $D_{2.0}$, $D_{3.0} - \overline{x}_2$ (см. рис. 6.3, *в*).

Теперь рассмотрим случай, когда число входных переменных логической функции больше числа адресных входов мультиплексора.

Пример 6.2. Реализуем логическую функцию, представленную картой Карно (см. рис. 6.3, *а*) на мультиплексоре 4→1, выбрав в качестве адресных переменных x_2, x_1.

Р е ш е н и е. В этом случае число логических переменных (x_2, x_1, x_0) на единицу больше числа адресных входов мультиплексора (A_0, A_1), поэтому синтез схемы выполняется иначе. Преобразуем карту Карно (см. рис. 6.3, *а*) в таблицу истинности (табл. 6.2), добавив к ней для удобства столбцы слева и справа: столбец *n* содержит номера информационных входов, а D_i – логические величины, поступающие на *i* вход. Разобьем таблицу на группы по две строки. В каждой группе логические переменные x_2 и x_1 неизменны, x_0 имеет два состояния (лог. «0» и лог. «1»), а выходной сигнал *y* может иметь одно из четырех состояний лог. «1», лог. «0», x_0, $\overline{x}_2$.

Таблица 6.2

Таблица истинности ЛФ

n	x_2	x_1	x_0	*y*	D_i
0	0 0	0 0	0 1	0 1	x_0
1	0 0	1 1	0 1	1 1	1
2	1 1	0 0	0 1	1 0	$\overline{x}_0$
3	1 1	1 1	0 1	0 0	0

Если подать логические переменные x_1 и x_2 соответственно на адресные входы мультиплексора A_0 и A_1, то конкретный набор этих переменных будет определять в двоичной системе счисления номер группы таблицы и информационного входа, который при этом будет соединяться с выходом *y*. Анализ сигналов x_0 и *y* позволяет заполнить столбец D_i. Подавая на соответствующие информационные входы D_i согласно таблице 6.2 постоянные логические сигналы «1» и «0» и переменные x_0 и $\overline{x}_0$, получим схему, реализующую заданную функцию (рис. 6.4).

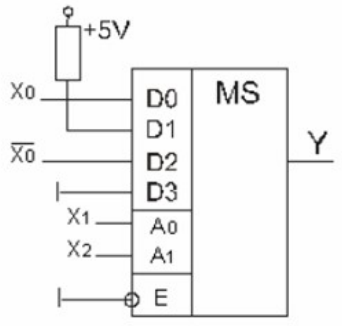


Рис. 6.4. Реализация логической функции на мультиплексоре 4→1

Пример 6.3. На мультиплексоре 8→1 реализовать логическую функцию четырёх аргументов

$$F = \bar{x}_1 x_2 \bar{x}_3 x_4 + x_1 \bar{x}_2 x_3 \bar{x}_4 + \bar{x}_1 \bar{x}_2 \bar{x}_3 \bar{x}_4 + x_1 \bar{x}_2 \bar{x}_3 x_4 + x_1 x_2 \bar{x}_3 \bar{x}_4.$$

Решение. Число входных переменных логической функции больше числа адресных входов мультиплексора. В этом случае одну или несколько из входных переменных следует «вынести» на информационные входы, используя разложение функции.

Сгруппируем минтермы и вынесем аргументы за скобку:

$$F = \bar{x}_1 \left(x_2 \bar{x}_3 x_4 + \bar{x}_2 \bar{x}_3 \bar{x}_4 \right) + x_1 \left(\bar{x}_2 x_3 \bar{x}_4 + \bar{x}_2 \bar{x}_3 x_4 + x_2 \bar{x}_3 \bar{x}_4 \right) =$$

$$= \bar{x}_1 \left(\underbrace{101}_{5} + \underbrace{000}_{0} \right) + x_1 \left(\underbrace{010}_{2} + \underbrace{001}_{1} + \underbrace{100}_{4} \right).$$

Из разложения видно, что на входы D_0 и D_5 следует подать инверсное значение аргумента x_1, на входы D_1, D_2, D_4 прямое значение этого аргумента, а входы D_3, D_6, D_7 соединить с общим проводом.

Схема, реализующая данную логическую функцию, представлена на рис. 6.5.

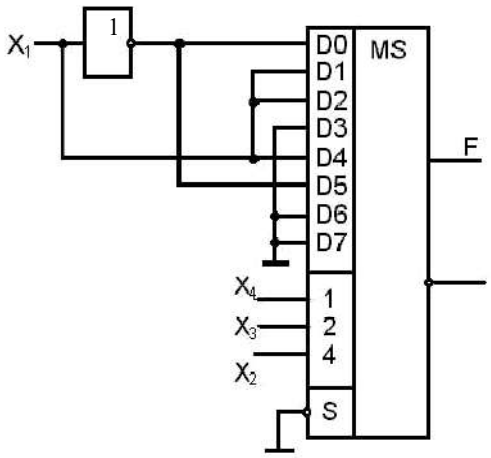


Рис. 6.5. Реализация логической функции на мультиплексоре 8→1

Рассмотрим еще один пример, в котором на информационные входы мультиплексора подается логическая переменная старшего разряда.

Пример 6.4. На мультиплексоре 8→1 реализовать функцию четырех переменных, представленную в СНДФ:

$$y = x_3 \bar{x}_2 x_1 x_0 + \bar{x}_3 x_2 \bar{x}_1 x_0 + x_3 x_2 x_1 \bar{x}_0 + \bar{x}_3 x_2 x_1 x_0 + x_3 x_2 x_1 x_0.$$

Р е ш е н и е . Переменные младших трех разрядов $x_2x_1x_0$ будем подавать на адресные входы мультиплексора, а x_3 – на информационные входы. По логическому выражению составим таблицу истинности (табл. 6.3), сгруппировав по два набора переменных так, что в каждой группе $x_2x_1x_0$ неизменны, x_3 имеет два состояния, а выходной сигнал y – одно из четырех значений 1, 0, x_3 или $\overline{x}_3$.

Таблица 6.3

Таблица истинности ЛФ

n	x_3	x_2	x_1	x_0	y	D_i
0	0	0	0	0	0	0
	1	0	0	0	0	
1	0	0	0	1	0	0
	1	0	0	1	0	
2	0	0	1	0	0	0
	1	0	1	0	0	
3	0	0	1	1	0	x_3
	1	0	1	1	1	
4	0	1	0	0	0	0
	1	1	0	0	0	
5	0	1	0	1	0	x_3
	1	1	0	1	1	
6	0	1	1	0	0	x_3
	1	1	1	0	1	
7	0	1	1	1	1	1
	1	1	1	1	1	

Схемная реализация такой функции показана на рис. 6.6.

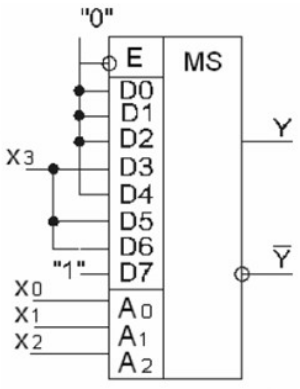


Рис. 6.6. Реализация ЛФ на мультиплексоре К555КП7

Приведенные примеры позволяют сформулировать следующий алгоритм реализации комбинационных функций на мультиплексорах:

1. Представить исходную ФАЛ в виде таблицы истинности, карты Карно или СНДФ.

2. Разделить входные логические переменные на адресные, информационные и стробирующие.

3. Определить значения информационных выходных переменных для всех наборов адресных логических переменных;

4. Изобразить логическую схему на мультиплексорах, реализующую требуемые функции.

Демультиплексор (рис. 6.7) – это устройство, предназначенное для коммутации входа данных D на один из выходов y_1, y_2, ..., y_n, адрес которого выбирается при помощи управляющих входов A_1, A_2, ..., A_k (причем $n = 2^k - 1$), при подаче сигнала синхронизации на вход C (если последний имеется). Демультиплексор имеет принцип действия, обратный мультиплексору.

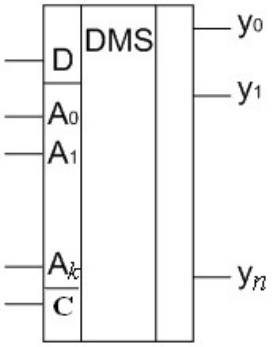


Рис. 6.7. Условное обозначение демультиплексора

Пример 6.5. Синтезировать синхронный демультиплексор на 4 выхода данных, функционирование которого задано таблицей истинности (табл. 6.4).

Таблица 6.4

Таблица истинности демультиплексора

Входы			Выходы			
A_1	A_2	C	y_0	y_1	y_2	y_3
0	0	1	D	0	0	0
0	1	1	0	D	0	0
1	0	1	0	0	D	0
1	1	1	0	0	0	D
×	×	0	0	0	0	0

Решение. Функции выходов составляются для каждого из выходов в отдельности на основе принципа независимости входов и выходов. Они имеют следующий вид: $y_0 = D\overline{A_1}\,\overline{A_2}C$; $y_1 = D\overline{A_1}A_2C$; $y_2 = DA_1\overline{A_2}C$; $y_3 = DA_1A_2C$.

Согласно формулам построим схему (рис. 6.8).

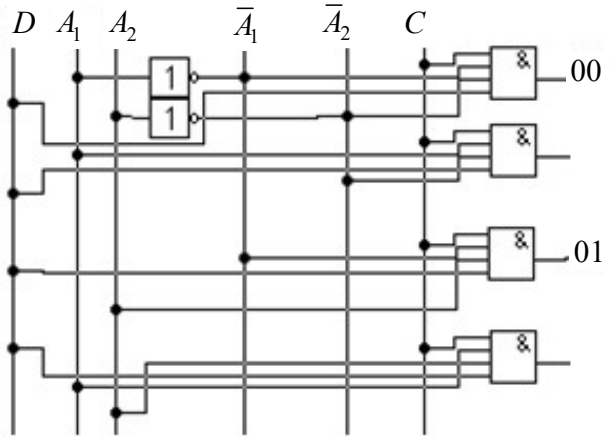


Рис. 6.8. Схема демультиплексора в базисе И-ИЛИ-НЕ

С использованием мультиплексоров и демультиплексоров можно строить схемы коммутации с *n* направлений в *m*. В схеме, приведенной на рис. 6.8, по заданным адресам любой из входов может быть подключен к любому из выходов.

Задания для самостоятельной работы

1. Реализуйте ЛФ, заданную таблицей истинности (табл. 6.5), с помощью мультиплексора.

Таблица 6.5

Таблица истинности ЛФ

N	*a*	*b*	*c*	*y*	*N*	*a*	*b*	*c*	*y*
0	0	0	0	0	4	1	0	0	0
1	0	0	1	0	5	1	0	1	1
2	0	1	0	0	6	1	1	0	1
3	0	1	1	1	7	1	1	1	0

2. Реализуйте ЛФ на мультиплексоре 4→1:

а) $y = \overline{x}_2\overline{x}_1x_0 + \overline{x}_2x_1\overline{x}_0 + \overline{x}_2x_1x_0 + x_2\overline{x}_1\overline{x}_0$;

б) функцию ИСКЛЮЧАЮЩЕЕ ИЛИ.

3. Реализуйте на базе мультиплексора 8→1:

а) $X(a,b,c) = \sum 0,1,5;$

б) схему мажоритарного элемента 2 из 3;

в) $y = \overline{x}_1x_2x_3 \vee x_1\overline{x}_2x_3 \vee x_1x_2\overline{x}_3 \vee x_1x_2x_3$;

г) $f = \overline{a}\,\overline{b}c \vee \overline{a}b\overline{c} \vee a\overline{b}\overline{c} \vee abc$;

д) $Y=\overline{A}\,\overline{B}\,\overline{C}\,\overline{D}+A\overline{B}\,\overline{C}D+\overline{A}B\overline{C}D+\overline{A}B\overline{C}\,\overline{D}+\overline{A}\,\overline{B}CD+$
$+A\overline{B}C\overline{D}+\overline{A}BCD+\overline{A}BC\overline{D}.$

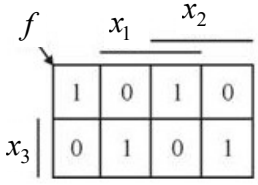
4. Реализуйте на базе MS $4\to1$ ЛФ: а) $y_1 = x_1 \vee x_2$; б) $y_1 = x_2 \wedge x_2$; в) $y_3 = \overline{x_1 \wedge x_2}$; г) $F = \overline{x}\,\overline{y}z \vee \overline{x}\,y\overline{z} \vee xyz \vee x\overline{y}\overline{z}$; д) $Y = \overline{x}_3\overline{x}_2\overline{x}_1\overline{x}_0 \vee \overline{x}_3\overline{x}_2x_1x_0 \vee \overline{x}_3x_2\overline{x}_1\overline{x}_0 \vee x_3x_2x_1\overline{x}_0 \vee \overline{x}_3x_2x_1x_0 \vee x_3x_2\overline{x}_1x_0 \vee x_3\overline{x}_2\overline{x}_1x_0 \vee x_3\overline{x}_2x_1\overline{x}_0$.

5. Постройте на мультиплексоре $8\to1$ функцию $F(x,y,z)$ заданную картами Карно и диаграммами Вейча:

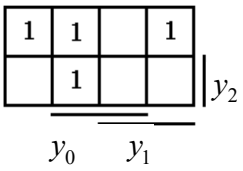
а)

xy		00	01	11	10
z	0	0	1	0	1
	1	1	0	1	0

б)



в)

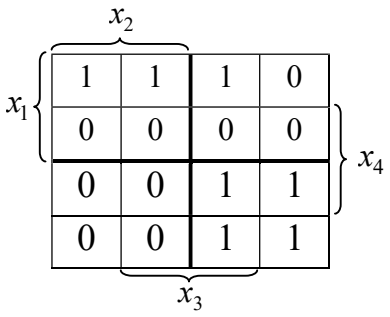


6. Постройте на мультиплексоре $4\to1$ функцию $F(A,B,C)$, заданную картами Карно и диаграммами Вейча

а)

AB		00	01	11	10
C	0	0	0	1	0
	1	0	1	1	1

б)



в)

	b		$\overline{b}$	
a_0	0	1	0	1
$\overline{a}$	1	0	1	0
	$\overline{c}$	c		$\overline{c}$

г)

	b		$\overline{b}$		
a_0	1	1	0	1	$\overline{d}$
	0	0	0	1	d
$\overline{a}$	1	1	0	0	
	1	1	0	0	$\overline{d}$
	$\overline{c}$	c		$\overline{c}$	

7. Реализуйте $f(x_1,x_2,x_3,x_4)$, заданную ТИ, на MS $4\to1$

x_1	x_2	x_3	x_4	y	Вход	Сигнал	x_1	x_2	x_3	x_4	y	Вход	Сигнал
0	0	0	0	0	D_0	x_4	1	0	0	0	0	D_0	x_4
0	0	0	1	1	D_0	x_4	1	0	0	1	1	D_0	x_4
0	0	1	0	0	D_0	x_4	1	0	1	0	0	D_0	x_4
0	0	1	1	1	D_0	x_4	1	0	1	1	1	D_0	x_4
0	1	0	0	1	D_1	$\overline{x}_3$	1	1	0	0	0	D_3	$\overline{x}_3x_4$
0	1	0	1	1	D_1	$\overline{x}_3$	1	1	0	1	1	D_3	$\overline{x}_3x_4$
0	1	1	0	0	D_1	$\overline{x}_3$	1	1	1	0	0	D_3	$\overline{x}_3x_4$
0	1	1	1	0	D_1	$\overline{x}_3$	1	1	1	1	0	D_3	$\overline{x}_3x_4$

8. Реализуйте функцию y в скобочной форме на двух MS 4→1:

$$y=\overline{x}_1\overline{x}_2\overline{x}_3\vee\overline{x}_1x_2\overline{x}_3\vee x_1\overline{x}_2\overline{x}_3\vee x_1x_2\overline{x}_3\vee\overline{x}_1\overline{x}_2x_3\vee\overline{x}_1x_2x_3\vee x_1\overline{x}_2x_3\vee x_1x_2x_3.$$

9. Реализуйте функцию вида $y=CB+AC$ в скобочной форме на двух MS2→1.

10. Реализуйте на MS 4→1 функции: а) $f=A\left(B+C\right)$; б) $f=x_1\overline{x}_2\vee\overline{x}_1x_2$; в) $f=U\,\overline{X}\,\overline{Y}+\overline{U}\,\overline{X}Y+\overline{U}X\overline{Y}+UXY$; г) $y=x_1x_2\overline{x}_3\vee x_1\overline{x}_2x_3\vee\overline{x}_1x_2x_3\vee\overline{x}_1\overline{x}_2\overline{x}_3$.

11. Постройте ЛФ: $y=x_3\overline{x}_2x_1\overline{x}_0\vee\overline{x}_3\overline{x}_2\overline{x}_1x_0\vee x_3\overline{x}_2x_1\overline{x}_0\vee x_3\overline{x}_2\overline{x}_1x_0\vee\overline{x}_3\overline{x}_2x_1\overline{x}_0\vee$ $\vee\,\overline{x}_3x_2\overline{x}_1\overline{x}_0$: а) на базе мультиплексора 8→1; б) на двух стробируемых мультиплексорах 4→1.

12. Постройте схему синхронного мультиплексора при наличии трех адресных входов в базисе И-ИЛИ-НЕ.

13. Постройте демультиплексор 1→4 на элементах: а) НЕ и И; б) ИЛИ-НЕ.

14. Реализуйте мультиплексор 16→1.

15. Реализуйте на мультиплексоре 2→1 ЛФ, заданную ТИ:

x_2	x_1	x_0	F	$F(x_0)$
0	0	0	1	1
0	0	1	1	
0	1	0	0	x_0
0	1	1	1	

16. Постройте на мультиплексоре 8→1 функции $Y_1=f_1\left(A,B,C\right)$ и $Y_2=f_2\left(A,B,C,D\right)$, заданные ТИ:

A	B	C	D	Y_1	Y_2	A	B	C	D	Y_1	Y_2
0	0	0	0	0	0	1	0	0	0	×	0
0	0	0	1	0	0	1	0	0	1	×	1
0	0	1	0	0	0	1	0	1	0	×	1
0	0	1	1	1	1	1	0	1	1	×	0
0	1	0	0	0	1	1	1	0	0	×	0
0	1	0	1	1	1	1	1	0	1	×	0
0	1	1	0	1	1	1	1	1	0	×	1
0	1	1	1	1	0	1	1	1	1	×	0

17. Постройте на мультиплексоре 8→1 функцию, заданную ТИ:

x_3	x_2	x_1	x_0	y	$y(x_0)$	x_3	x_2	x_1	x_0	y	$y(x_0)$
0	0	0	0	0	$y=x_0$	0	1	1	0	0	$y=x_0$
0	0	0	1	1		0	1	1	1	1	
0	0	1	0	1	$y=1$	1	0	0	0	1	$y=1$
0	0	1	1	1		1	0	0	1	1	
0	1	0	0	0	$y=0$	1	0	1	0	0	$y=x_0$
0	1	0	1	0							

18. Заданы МДНФ двух функций: $f = x_1x_3 \vee x_2\overline{x}_3\overline{x}_4 \vee \overline{x}_1\overline{x}_2\overline{x}_3 \vee \overline{x}_2x_3x_4$ и $f = x_1x_2x_3 \vee \overline{x}_2\overline{x}_3\overline{x}_4 \vee \overline{x}_1\overline{x}_2x_3x_4$. Постройте функциональную схему устройства на сдвоенном четырехканальном мультиплексоре.

19. Синтезируйте мультиплексор с восемью информационными входами и одним выходом на элементах И, ИЛИ, НЕ.

20. Синтезируйте демультиплексор на элементах И-НЕ.

21. Постройте схему на мультиплексорах с двумя управляющими и стробирующими входами для ФАЛ четырех переменных

$$f = x_1x_2x_3 \vee x_2\overline{x}_3x_4 \vee \overline{x}_1\overline{x}_2\overline{x}_4.$$

22. Реализуйте на базе мультиплексора $8 \rightarrow 1$ функции:

а) $X(a,b,c,d) = \sum 1,2,4,6,12,14,11,13;$

б) $X(a,b,c,d) = \prod 0,3,4,5,7,14,15;$

в) $X(a,b,c,d) = \sum 8,9,10,11,12,13,14,15;$

г) $X(a,b,c,d) = \prod 2,3,6,7,9,14;$

д) $X(a,b,c,d) = \prod 1,2,4,6,8,9,11,15;$

е) $X(a,b,c,d) = \sum 0,4,5,8,12,13;$

ж) $X(a,b,c,d) = \sum 4,5,8,10,11;$

з) $X(a,b,c,d) = \sum 3,6,8,9,10,12,14;$

и) $X(a,b,c,d) = \prod 1,2,4,6,8,9,11,,12,15;$

з) $X(a,b,c,d) = \prod 5,6,8,10,3,15.$

7. СУММАТОРЫ

Сумматор – комбинационная логическая схема, выполняющая арифметическое сложение кодов двух чисел.

Существует два вида сумматоров: с параллельным и последовательным действием. У сумматоров с параллельным действием сложение выполняется параллельно, сразу во всех разрядах суммируемых чисел. В сумматорах с последовательным действием имеется только одна одноразрядная суммирующая схема и результат образуется последовательным сложением отдельных разрядов.

На рис. 7.1 представлен простейший суммирующий элемент (четвертьсумматор) и схема его реализации на элементе Исключающее ИЛИ.

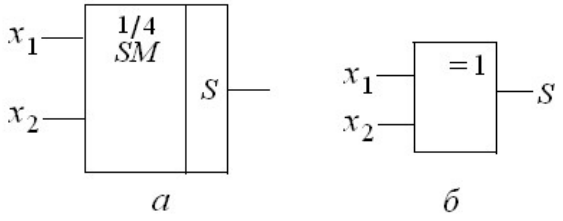


Рис. 7.1. Четвертьсумматор: *а* – УГО; *б* – схема реализации на элементе «Исключающее ИЛИ»

Схема (см. рис. 7.1, а) имеет входы x_1 и x_2 для двух слагаемых и один выход S для суммы. Ее работу отражает таблица истинности (табл. 7.1).

Таблица 7.1

Таблица истинности четвертьсумматора

x_1	x_2	S	m_i	M_i
0	0	0	$\overline{x}_1\overline{x}_2$	$x_1 + x_2$
0	1	1	$\overline{x}_1 x_2$	$x_1 + \overline{x}_2$
1	0	1	$x_1\overline{x}_2$	$\overline{x}_1 + x_2$
1	1	0	$x_1 x_2$	$\overline{x}_1 + \overline{x}_2$

Синтезируем функцию, заданную таблицей истинности, в базисе И, ИЛИ, НЕ. Из табл. 7.1 видно, что каждому набору переменных ставятся в соответствие минтермы m_i и макстермы M_i.

Таким образом, получаем следующее уравнение СДНФ функции:

$$S = \bar{x}_1 x_2 + x_1 \bar{x}_2 = x_1 \oplus x_2 . \qquad (7.1)$$

Схема реализации функции (7.1) в базисе И, ИЛИ, НЕ приведена на рис. 7.2.

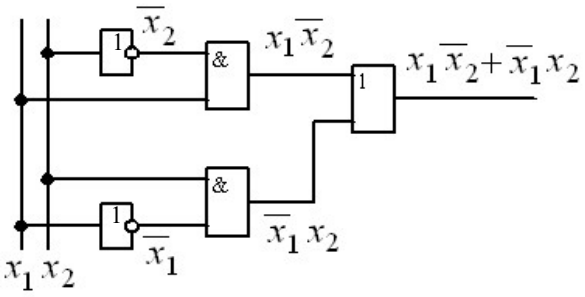


Рис. 7.2. Схема четвертьсумматора в базисе И, ИЛИ, НЕ

Данный элемент выпускается в виде ИС типа ЛП5 (серии 133, 155, 530, 531, 533, 555, 1531, 1533); ЛП2 (561, 564); ЛП4 (1561) и т.п.

Полусумматор (рис. 7.3) имеет два входа x_1 и x_2 для двух слагаемых и два выхода: S_i – сумма, P_{i+1} – перенос. Обозначением полусумматора служат буквы *HS* (*halfsum* – полусумма). Его работу отражает таблица истинности (табл. 7.2) и соответствующее ей уравнение.

Таблица 7.2

Таблица истинности полусумматора

Слагаемые		Результат суммирования	
x_1	x_2	S_i	Цифра переноса в старший разряд P_{i+1}
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Для получения логической функции одноразрядного суммирования в форме СДНФ производят запись «по единицам»:

$$S_i = \bar{x}_1 x_2 + x_1 \bar{x}_2 = x_1 \oplus x_2 \ , \ P_{i+1} = x_1 x_2 , \qquad (7.2)$$

т.е. схема полусумматора реализуется двумя логическими функциями, а устройство имеет два выхода: S_i и P_{i+1}.

Из уравнений (7.2) следует, что схему полусумматора можно представить как простое объединение схем (в базисе И, ИЛИ, НЕ), реализующих каждую функцию отдельно (рис. 7.3, *а*), или как схему полусумматора на логических элементах Исключающее ИЛИ и И (рис. 7.3, *б*).

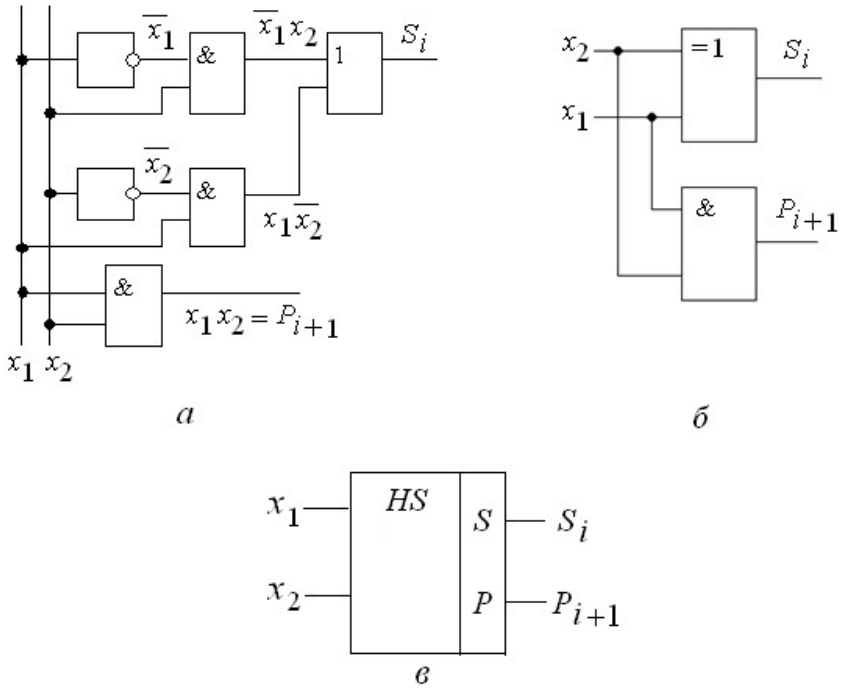


Рис. 7.3. Полусумматор: *а* – в базисе И, ИЛИ, НЕ;
б – с элементом «Исключающее ИЛИ»; *в* – УГО

Полный двоичный сумматор (рис. 7.4, *а*) имеет три входа: x_1, x_2 – это разряды соответственно первого и второго слагаемых и P_i – перенос из предыдущего (младшего) разряда и два выхода: S_i – сумма и P_{i+1} – перенос в следующий (старший) разряд.

Согласно правилам сложения для чисел, заданных в двоичной системе счисления, составим таблицу истинности (табл. 7.4) значений P_{i+1} и S_i. Например, для $x_1 = 1,\ \ x_2 = 0,\ \ P_i = 1$ имеем $1 + 0 + 1 = 10$, следовательно, $S_i = 0,\ \ P_{i+1} = 1$; для $x_1 = x_2 = P_i = 1$ будет $1 + 1 + 1 = 11$, т. е. $S_i = 1$ и $P_{i+1} = 1$. В табл. 7.4 x_1, x_2, P_i являются аргументами, а S_i и P_{i+1} – функциями.

Таблица 7.4

Таблица истинности полного одноразрядного сумматора

Слагаемые			Результат суммирования	
P_i	x_1	x_2	S_i	P_{i+1}
0	0	0	0	0
0	1	0	1	0
0	0	1	1	0
0	1	1	0	1
1	0	0	1	0
1	1	0	0	1
1	0	1	0	1
1	1	1	1	1

Для получения логической функции в алгебраической форме в виде СДНФ производят запись по «единицам»:

$$S_i = x_1\overline{x}_2\overline{P_i} + \overline{x}_1x_2\overline{P_i} + \overline{x}_1\overline{x}_1P_i + x_1x_2P_i; \tag{7.3}$$

$$P_{i+1} = x_1x_2\overline{P_i} + x_1\overline{x}_2P_i + \overline{x}_1x_2P_i + x_1x_2P_i. \tag{7.4}$$

Далее производят минимизацию логических функций. Выражение (7.3) не поддается минимизации изложенными ранее методами. Единственная возможность – это использовать вынесение за скобки:

$$S_i = \left(x_1\overline{x}_2 + \overline{x}_1x_2\right)\overline{P_i} + \left(\overline{x}_1\overline{x}_2 + x_1x_2\right)P_i.$$

Применив для выражения (7.4) операции склеивания и поглощения можно получить СДНФ:

по P_i: $x_1x_2\overline{P_i} + x_1x_2P_i = x_1x_2\left(\overline{P_i} + P_i\right) = x_1x_2$;

по x_2: $x_1\overline{x}_2P_i + x_1x_2P_i = x_1P_i(\overline{x}_2 + x_2) = x_1P_i$;

по x_1: $\overline{x}_1x_2P_i + x_1x_2P_i = x_2P_i(\overline{x}_1 + x_1) = x_2P_i$.

СДНФ логической функции:

$$P_{i+1} = x_1x_2 + x_1P_i + x_2P_i = x_1x_2 + (x_1 + x_2)P_i.$$

Таким образом, полный сумматор оказывается устройством с двумя выходами и реализуется двумя логическими функциями S_i и P_{i+1} с тремя аргументами x_1, x_2, P_i.

Схему, реализующую несколько функций, можно представить как простое объединение схем, реализующих каждую из них отдельно. Функциональная схема в логическом базисе И, ИЛИ, НЕ представлена на рис. 7.4, *б*.

Схема оказалась реализованной на 16 базовых логических элементах, т.е. неэкономичной. Поэтому нужно преобразовать совокупность данных логических функций к такому виду, чтобы реализующие их схемы содержали общие части, а схема со многими выходами представляла собой единое целое.

На следующем этапе преобразований целесообразно более простую реализацию функции $P_{i+1}\left(x_1, x_2, P_i\right)$ использовать в качестве составной части другой функции $S_i\left(x_1, x_2, P_i, P_{i+1}\right)$. Табл. 7.5 показывает логику работы устройства.

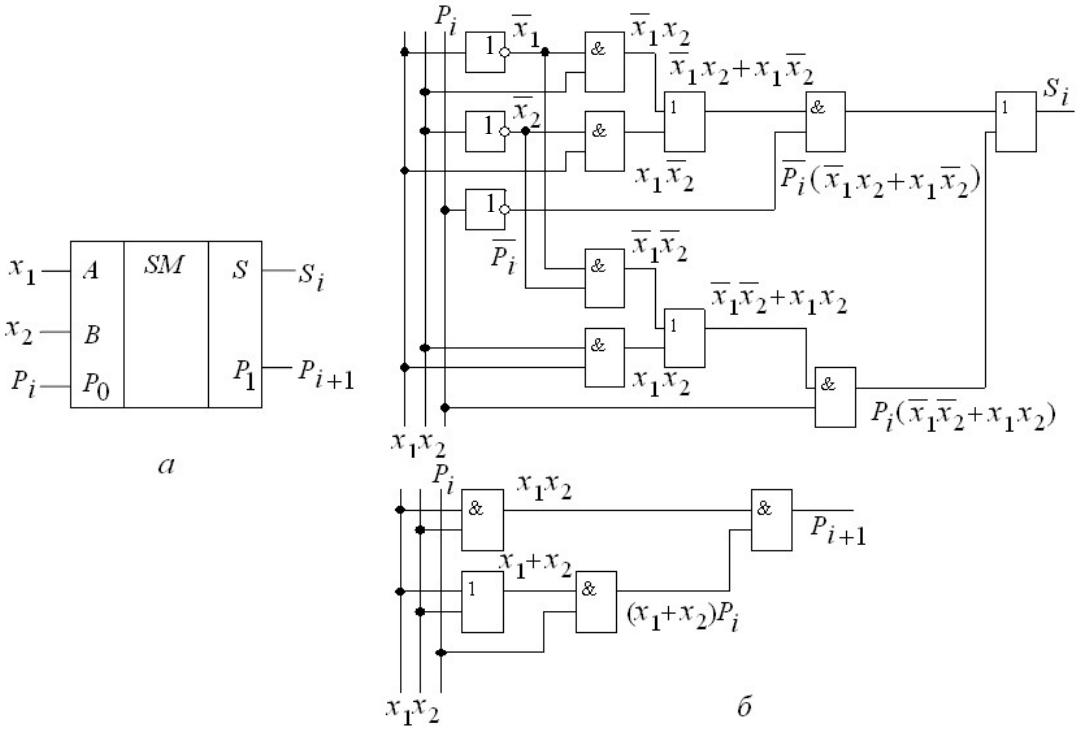


Рис. 7.4. Полный одноразрядный сумматор: *а* – УГО; *б* – функциональная схема

Таблица 7.5

Таблица истинности полного одноразрядного сумматора

x_1	x_2	P_i	P_{i+1}	S_i
0	0	0	0	0
0	0	0	1	*
0	0	1	0	1
0	0	1	1	*
0	1	0	0	1
0	1	0	1	*
0	1	1	0	*
0	1	1	1	0
1	0	0	0	1
1	0	0	1	*
1	0	1	0	*
1	0	1	1	0
1	1	0	0	*
1	1	0	1	0
1	1	1	0	*
1	1	1	1	1

Таблица истинности для функции S_i теперь содержит избыточные наборы переменных (т. е. такие, которые не могут встретиться при работе сумматора), которые отмечены крестиками ×, т.е. функция оказывается частично (не полностью) определенной. Используем для минимизации частично определенной функции S_i карту Карно (рис. 7.5) и доопределим данную функцию.

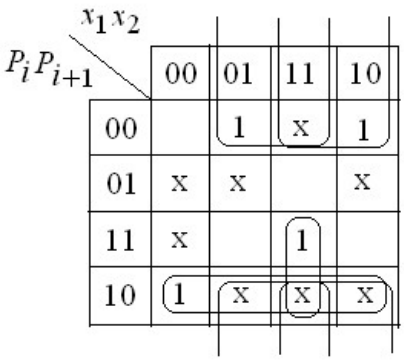


Рис. 7.5. Карта Карно

Минимальному покрытию соответствует логическая функция:

$$S_i = x_1\overline{P}_{i+1} + x_2\overline{P}_{i+1} + P_i\overline{P}_{i+1} + x_1x_2P_i.$$

После вынесения за скобки $\overline{P}_{i+1}$ получают готовую для реализации логическую функцию:

$$\begin{aligned} S_i &= \left(x_1 + x_2 + P_i\right)\overline{P}_{i+1} + x_1x_2P_i; \\ P_{i+1} &= x_1x_2 + \left(x_1 + x_2\right)P_i. \end{aligned} \qquad (7.5)$$

Функциональная схема для логической функции (7.5) в логическом базисе И, ИЛИ, НЕ показана на рис. 7.6

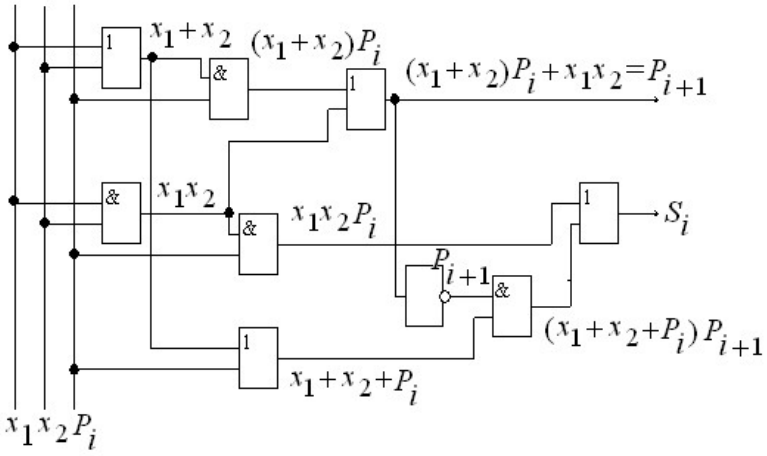


Рис. 7.6. Минимизированная функциональная схема
полного одноразрядного сумматора

Схема (рис. 7.6) реализована на 9 базовых логических элементах, что почти в два раза меньше, чем в схеме на рис. 7.4, *б*.

Для реализации схемы в базисах И-НЕ и ИЛИ-НЕ для логической функции следует применить формулу Де Моргана.

Пример 7.1. Составить схему полного сумматора, используя полусумматоры.

Р е ш е н и е . Полный сумматор осуществляет сложение трех цифр: двух цифр x_1 и x_2, принадлежащих одному разряду складываемых чисел, а также цифры переноса из предыдущего разряда P_i. В результате суммирования этих трех цифр получается сумма S_i и цифра переноса в старший разряд P_{i+1}. Таким образом, это устройство с тремя входами и двумя выходами.

Полусумматоры имеют два входа для x_1 и x_2, два выхода для S_i и P_{i+1}.

Используя сочетательный закон, получаем

$$S_i = x_1 + x_2 + P_i = (x_1 + x_2) + P_i = S_i' + P_i,$$

т.е. можно сначала сложить две цифры x_1 и x_2, а затем к промежуточной сумме S_i' прибавить P_i.

Поэтому полный сумматор можно представить как объединение двух полусумматоров. Первый полусумматор служит для сложения двух цифр x_1 и x_2 и обеспечивает выход промежуточной суммы S_i' и переноса P_{i+1}'. Второй полусумматор складывает промежуточную сумму S_i' с цифрой переноса из предыдущего разряда P_i, формирует перенос P_{i+1}'' и сумму S_i. При этом

$$\begin{gathered} S_i' = x_1 + x_2; \quad S_i = S_i' + P_i = x_1 + x_2 + P_i; \\ P_{i+1}' = x_1 x_2; \quad P_{i+1}'' = P_i (x_1 + x_2). \end{gathered} \tag{7.6}$$

Из анализа таблицы истинности для полусумматора (табл. 7.2) следует, что при сложении трех цифр двумя полусумматорами цифра переноса может образоваться только в одном полусумматоре: P_{i+1}' или P_{i+1}''. Поэтому для получения P_{i+1}' эти переносы следует объединить логической ячейкой ИЛИ:

$$P_{i+1} = P_{i+1}' + P_{i+1}'' = x_1 x_2 + P_i (x_1 + x_2). \tag{7.7}$$

Выражение (7.7) совпадает с выражением (7.5) для полного сумматора. Для S_i из выражения (7.6) получаем

$$S_i = x_1 \oplus x_2 \oplus P_i. \tag{7.8}$$

Так как операция $\oplus$ в выражении (7.8) коммутативна (переменные можно менять местами), следовательно, три входа полного двоичного сумматора абсолютно равноправны и на любой из них можно подавать любую входную переменную.

Функциональная схема полного сумматора, синтезированного из двух полусумматоров, представлена на рис. 7.7.

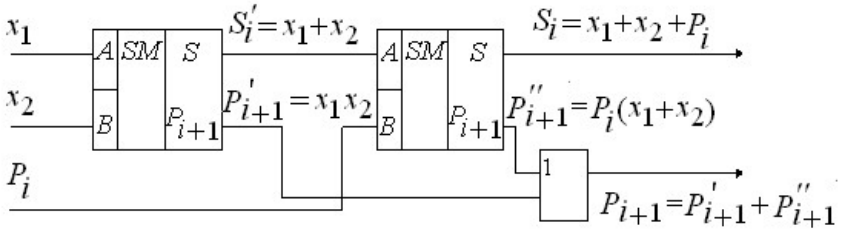


Рис. 7.7. Полный сумматор, синтезированный из двух полусумматоров

Для сложения двух многоразрядных чисел объединяют соответствующее количество одноразрядных сумматоров. Схема многоразрядного сумматора приведена на рис.7.8.

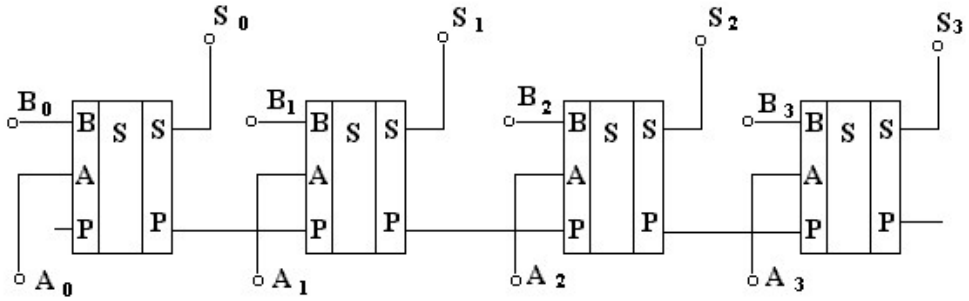


Рис. 7.8. Схема многоразрядного сумматора

В таком сумматоре $B_0 A_0, \ldots, B_{n-1} A_{n-1}$ представляют входы для подачи значений соответствующих разрядов слагаемых чисел, а $S_0, \ldots, S_n$ – результат сложения. Например, производится сложение двух чисел A = 1011 (11) или B =1101 (13). Тогда на входы подаются: A_0 – 1, A_1 – 1, A_2 – 0, A_3 – 1, B_0 – 1, B_1 – 0, B_2 – 1, B_3 – 1. На выходах получаем S_0 – 0, S_1 – 0, S_2 – 0, S_3 – 1, S_4 – 1. Результат сложения – число 11000 (24).

Задания для самостоятельной работы

1. Реализуйте четвертьсумматор в базисах И-НЕ и ИЛИ-НЕ.
2. Реализуйте четвертьсумматор в базисе И, ИЛИ, НЕ.
3. Синтезируйте узел, осуществляющий суммирование двух одноразрядных двоичных чисел (полусумматор): а) на элементах И, ИЛИ, НЕ; б) на элементах И-НЕ; в) на элементах ИЛИ-НЕ.
4. Синтезируйте схему полного одноразрядного двоичного сумматора, используя двухвходовые логические элементы, реализующие логическую функцию И-НЕ.

5. Составьте на элементах И-ИЛИ-НЕ схему сумматора.

6. Синтезируйте одноразрядный четверичный сумматор.

7. На логическом элементе 2И-2И-ИЛИ реализуйте схему полусумматора.

8. Реализуйте схему полного двоичного сумматора на логических элементах 4И-ИЛИ.

9. Реализуйте схему полного двоичного сумматора на логических элементах И-ИЛИ-НЕ.

10. Синтезируйте схему четырехразрядного полного сумматора с одновременным переносом, реализованную на логических элементах И-ИЛИ-НЕ.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Смирнов, Ю. А. Основы микроэлкетроники и микропроцессорной техники: учебное пособие / Ю. А. Смирнов, С. В. Соколов, Е. В. Титов. – 2-е изд., испр. – СПб.: Лань, 2013. – 496 с.

2. Смирнов, Ю. А. Физические основы электроники: учебное пособие / Ю. А. Смирнов, С. В. Соколов, Е. В. Титов. – 2-е изд., испр. – СПб.: Лань, 2013. – 560 с.

3. Белов, Н. В. Электротехника и основы электроники: учебное пособие / Н. В. Белов, Ю. С. Волков. – СПб.: Лань. 2012, – 430 с.

4. Иванов, И. И. Электротехника и основы электроники: учебник / И. И. Иванов, Г. И. Соловьев, В. Я. Фролов. – 7-е изд., перераб. и доп. – СПб.: Лань, 2012. – 736 с.

5. Кулагина, Л. Г. Физические основы электроники: учебное пособие / Л. Г. Кулагина, Г. Р. Еникеева. – Казань: Казан. гос. энерг. ун-т, 2015. – 148 с.

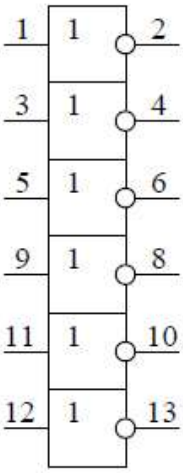
6. Гусев, В. Г. Электроника и микропроцессорная техника: учебник для вузов / В. Г. Гусев, Ю. М. Гусев. – 5 изд., доп. – М.: Высшая школа, 2008. – 798 с.

7. Ахметвалеева, Л. В. Цифровые устройства: учебное пособие / Л. В. Ахметвалеева. – Казань: Казан. гос. энерг. ун-т, 2002. – 172 с.

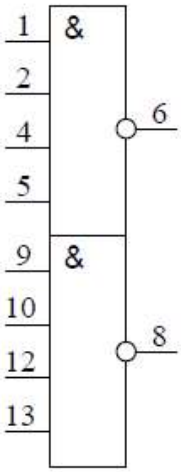
Приложение

Базовые логические элементы серии К155

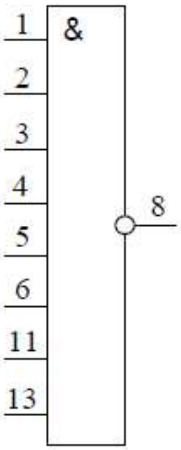
К155ЛН1



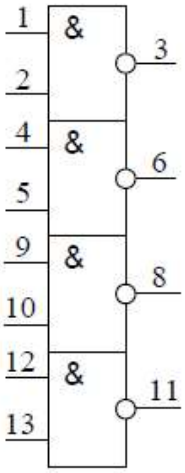
К155ЛА1



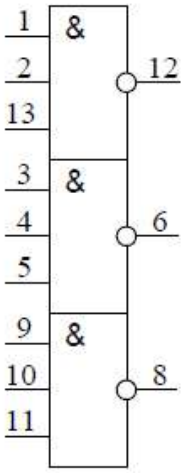
К155ЛА2



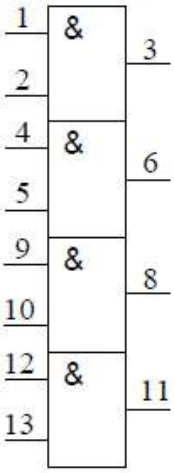
К155ЛА3



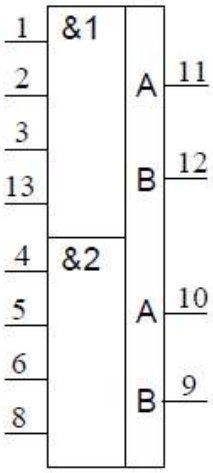
К155ЛА4



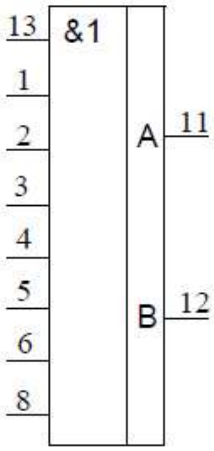
К155ЛИ1



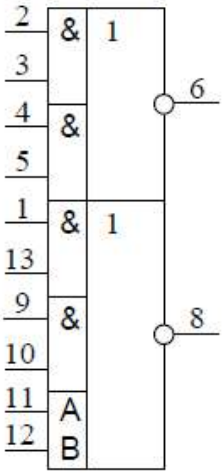
К155ЛД1



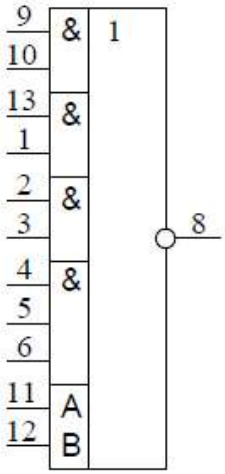
К155ЛД3



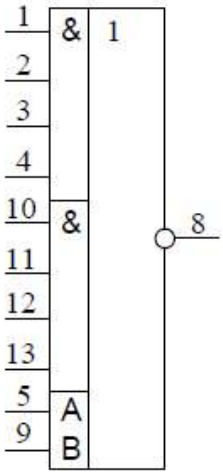
К155ЛР1



К155ЛР3



К155ЛР4



СОДЕРЖАНИЕ

ДЛЯ ЗАМЕТОК

Учебное издание

Ахметвалеева Ляля Вахитовна,
Кулагина Людмила Георгиевна

ОСНОВЫ ЦИФРОВОЙ ЭЛЕКТРОНИКИ

Учебно-методическое пособие

Кафедра промышленной электроники и светотехники КГЭУ

Редактор издательского отдела И.В. Краснова
Компьютерная верстка И.В. Краснова

Подписано в печать 20.02.18.
Формат 60×84/16. Бумага ВХИ. Гарнитура «Times». Вид печати РОМ.
Усл. печ. л. 5,81. Уч.-изд. л. 2,67. Тираж 500. Заказ № 180/эл.

Редакционно-издательский отдел КГЭУ
420066, г. Казань, ул. Красносельская, 51