МИНИСТЕРСТВО СЕЛЬСКОГО ХОЗЯЙСТВА РОССИЙСКОЙ ФЕДЕРАЦИИ

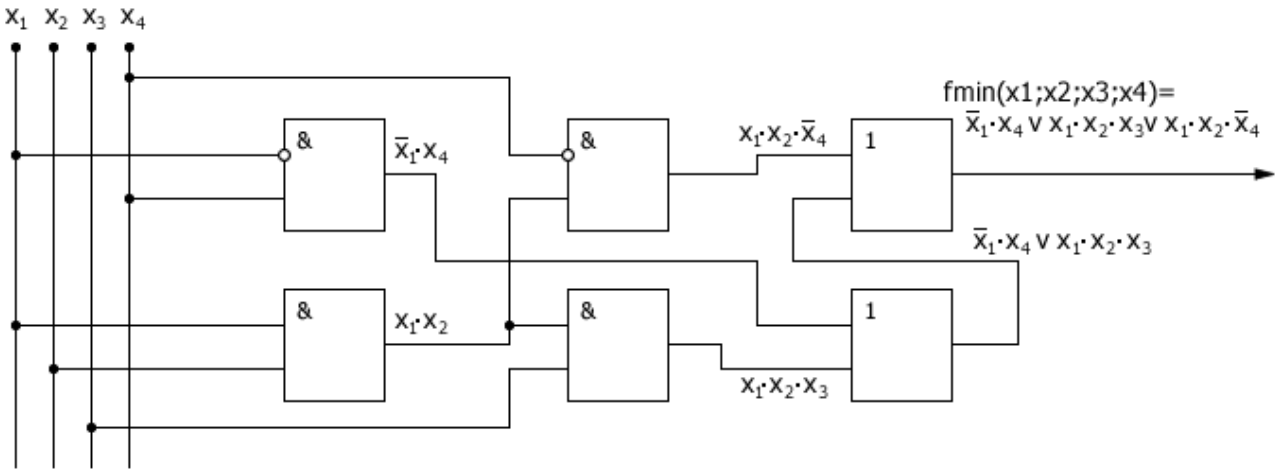
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Брянский государственный аграрный университет»

Кафедра математики, физики и информатики

Бычкова Т.В.

Дискретная математика

Учебное пособие
для бакалавров очной и заочной формы обучения
по направлению подготовки 15.03.04 «Автоматизация технологических процессов и производств»



Брянская область -2018

УДК 51 (07)
ББК 22.1
Б 95

Бычкова, Т. В. **Дискретная математика:** учебное пособие для бакалавров очной и заочной формы обучения по направлению подготовки 15.03.04 «Автоматизация технологических процессов и производств» / Т. В. Бычкова. - Брянск: Изд-во Брянский ГАУ, 2018. – 78 с.

Данное учебное пособие предназначено для бакалавров очной и заочной формы обучения, обучающихся по направлению подготовки 15.03.04 «Автоматизация технологических процессов и производств»

Пособие содержит теоретический материал, упражнения для практических занятий, примеры решений упражнений и задания для самостоятельной работы, необходимые при изучении дисциплины «Дискретная математика».

Рецензенты: к.т.н. Безик Д.А.

Рекомендовано к изданию учебно-методической комиссией института энергетики и природопользования Брянского ГАУ, протокол №8 от 28.06.2018 г.

Оглавление

ВВЕДЕНИЕ

Классическая («непрерывная») математика традиционно развивалась в интересах решения естественнонаучных задач и, прежде всего, физики. В отличие от нее «дискретная» математика преимущественно связана с исследованием и моделированием человеческого мышления.

Дискретная математика — раздел математики, в котором рассматриваются дискретные (не-непрерывные) множества, их организация, действия с ними и составляющими их объектами.

Это и определяет в качестве основных областей применения дискретной математики такие объекты, которые связаны с информатикой и вычислительной техникой. Для создания и эксплуатации интегрированных автоматизированных систем обработки информации и их компонент (математического обеспечения, пакетов прикладных программ, распределенных банков данных, сетей передачи данных, систем с распределением ресурсов и распределенной обработкой информации) необходимо знание дискретной математики.

В дисциплине «Дискретная математика» широко используются знания, полученные при изучении ряда разделов дисциплины «Математика», «Информатика». С другой стороны, «Дискретная математика» является базовой теоретической дисциплиной и является основой для изучения других математических, естественнонаучных и специальных дисциплин, таких как теория случайных процессов, численные методы, математическое моделирование, теория информации, моделирование систем, цифровые устройства и микропроцессоры, аппаратные средства вычислительной техники и др.

В данном пособии представлены теоретические сведения и практические задания по разделам: дискретное представление информации, алгебра логики, логические элементы.

Глава 1
Дискретное представление информации

Человек так устроен, что воспринимает информацию с помощью органов чувств. Свет, звук и тепло — это энергетические сигналы, а вкус и запах — это результат воздействия химических соединений, в основе которого тоже энергетическая природа. Человек испытывает энергетические воздействия непрерывно и может никогда не встретиться с одной и той же их комбинацией дважды. Мы не найдем двух одинаковых зеленых листьев на одном дереве и не услышим двух абсолютно одинаковых звуков — это информация аналоговая.

Если же разным цветам дать номера, а разным звукам — ноты, то аналоговую информацию можно превратить в цифровую. Музыка, когда мы ее слышим, несет аналоговую информацию, но стоит только записать ее нотами, как она становится цифровой.

Разница между аналоговой информацией и цифровой прежде всего в том, что аналоговая информация непрерывна, а цифровая — дискретна.

Преобразование информации различного вида из аналоговой формы в цифровую производится путем дискретизации, то есть разбиения непрерывного графического изображения или звукового сигнала на отдельные элементы. Дискретизация – это преобразование непрерывных изображений и звука в набор дискретных значений в форме кодов.

Приведем пример аналогового и дискретного представления информации. Рассмотрим положение тела на наклонной плоскости и на лестнице в системе координат *XOY* (рис. 1.1). При движении тела по наклонной плоскости его координаты могут принимать бесконечное множество непрерывно изменяющихся значений из опреленного диапазона, а при движении по лестнице – только определенный набор значений, причем меняющийся скачкообразно.

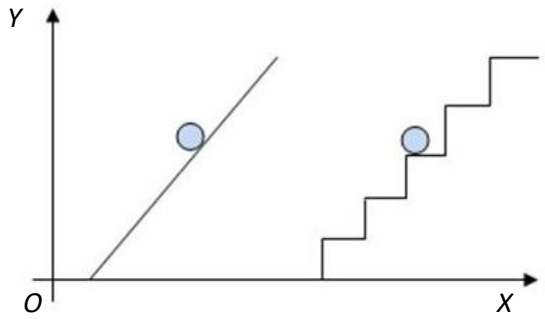


Рис.1.1. Аналоговая и дискретная форма представления информации

В настоящее время данные, изначально имеющие аналоговую форму (речь, телевизионное изображение), всё чаще передаются по каналам связи в дискретном виде, то есть в виде последовательности единиц и нулей.

Компьютеры имеют дело с дискретной информацией, на входе и выходе которых в качестве такой информации могут выступать любые последователь-

ности десятичных цифр, букв, знаков препинания и других символов. Внутри системы эта информация кодируется в виде последовательности сигналов, принимающих лишь два различных значения. Таким образом, любые данные для обработки компьютером представляются последовательностями двух чисел – единицы и нуля. Такая форма представления получила название двоичной. Для понимания процессов обработки двоичной информации рассмотрим теоретический аспект, изложенный в теории систем счисления.

1.1 Системы счисления

Система счисления – это совокупность приемов и правил представления чисел с помощью символов, имеющих определенное количественное значение.

Различают позиционные системы счисления и непозиционные.

Непозиционная системы счисления – система, в которой символы, обозначающие то или иное количество, не меняют своего значения в зависимости от местоположения (позиции) в изображении числа.

Непозиционной системой счисления является самая простая система с одним символом (палочкой). Для изображения какого-либо числа в этой системе надо записать количество палочек, равное данному числу. Это система самая неэффективная, так как форма записи очень громоздка.

К непозиционной системе относится и римская, табл. 1.

Так, например, в римской системе счисления в числе XXXII (тридцать два) значение цифры X в любой позиции равно десяти.

Запись чисел в данной системе счисления осуществляется по правилам:

1) если цифра слева меньше, чем цифра справа, то левая цифра вычитается из правой (IX: 1<10, следовательно, 10 – 1 = 9; XC: 10<100, следователь-но, 100 – 10 = 90);

2) если цифра справа меньше или равна цифре слева, то эти цифры складываются (VII: 5+1+1=7; XXXV: 10+10+10+5=35).

3) В римской системе нельзя записывать подряд 4 одинаковые цифры.

Таблица 1

Римские цифры	Значение в десятичной системе счисления
I	1
V	5
X	10
L	50
C	100
D	500
M	1000

Так, число 1984 в римской системе счисления имеет вид MCMLXXXIV (M – 1000, CM – 900, LXXX – 80, IV – 4).

В общем случае непозиционные системы счисления характеризуются сложными способами записи чисел и правилами выполнения арифметических

операций.

Позиционная система счисления – это система счисления, в которой значение цифры определяется ее местоположением (позицией) в изображении числа.

Алфавит позиционной системы счисления – упорядоченный набор символов (цифр) $\{a_0, a_1, \ldots, a_n\}$, используемый для представления чисел в данной системе счисления, где *0, 1,…, n* – порядковые номера символов.

Основание позиционной системы счисления – количество символов (цифр) алфавита $q = n + 1$, используемых для изображения чисел в данной системе счисления.

Примером позиционной системы счисления является десятичная система счисления. Ее алфавит {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}. Основание q = 10.

Например, в десятичной системе счисления число 333 записывается с помощью одной цифры 3, но значение каждой цифры определяется ее местоположением в числе: первая тройка – число сотен в числе, вторая тройка – число десятков, последняя – число единиц.

За основание системы счисления можно принять любое натуральное число – два, три, четыре и т.д.

Обычно в качестве алфавита берутся последовательные целые числа от 0 до (q – 1) включительно. В тех случаях, когда общепринятых (арабских) цифр не хватает для обозначения всех символов алфавита системы счисления с основанием q > 10, используются буквенные обозначения цифр. В табл. 2 приведены алфавиты некоторых систем счисления.

Таблица 2

Система счисления	Основание	Алфавит
Двоичная	2	0, 1
Восьмеричная	8	0, 1, 2, 3, 4, 5, 6, 7
Десятичная	10	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
Шестнадцатеричная	16	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

Для позиционной системы счисления справедливо равенство, которое называют развернутой формой записи числа:

$$A_q = a_n q^n + a_{n-1} q^{n-1} + \cdots + a_1 q^1 + a_0 q^0 + a_{-1} q^{-1} + \cdots + a_{-m} q^{-m} \qquad (1)$$

где A_q ($A_q = a_n a_{n-1} a_{n-2} \ldots a_1 a_0, a_{-1} a_{-2} \ldots a_{-m}$) – любое число, записанное в системе счисления с основанием *q*;

a_i - цифры числа ($i = n, n-1, \ldots, 1, 0, -1, -2, -m$);

$n + 1$ – число целых разрядов;

m – число дробных разрядов.

Пример 1.1

Записать числа 386,15410; 101,112; 561,428, 6BF,A16 в развернутой форме. Согласно (1) имеем:

$386{,}15_{10} = 3\cdot10^2 + 8\cdot10^1 + 6\cdot10^0 + 1\cdot10^{-1} + 5\cdot10^{-2}$
$101{,}11_2 = 1\cdot2^2 + 0\cdot2^1 + 1\cdot2^0 + 1\cdot2^{-1} + 1\cdot2^{-2}$
$561{,}423_8 = 5\cdot8^2 + 6\cdot8^1 + 1\cdot8^0 + 4\cdot8^{-1} + 2\cdot8^{-2} + 3\cdot8^{-3}$
$6BF{,}A_{16} = 6\cdot16^2 + B\cdot16^1 + F\cdot16^0 + A\cdot16^{-1}$

Пример 1.2

Упорядочить по убыванию числа: 43_5, 43_{16}, 43_8.

Решение. Переведем все числа в одну систему счисления, например, в десятичную:

$43_5 = 4\cdot5^1 + 3\cdot5^0 = 23,$
$43_{16} = 4\cdot16^1 + 3\cdot16^0 = 67,$
$43_8 = 4\cdot8^1 + 3\cdot8^0 = 35.$

Теперь упорядочим их по убыванию: 67 (43_{16}), 35 (43_8), 23 (43_5).

Ответ: 43_{16}, 43_8, 43_5.

Пример 1.3

В какой системе счисления выполнены действия: 123 + 4 = 132?

Решение. Обозначим за *x* основание системы счисления, в которой выполнены указанные действия. Складывая последние разряды чисел, имеем:

$3_x + 4_x = 12_x$. Используя представление чисел в развернутой форме, получим:

$3 + 4 = 1\cdot x + 2$. Решив данное уравнение, получим $x = 5$.

Ответ: действия выполнены в пятеричной системе счисления.

Пример 1.4

Число 35_x из системы счисления с основанием x перевели в десятичную систему счисления и получили 29_{10}. Найти основание системы счисления *x*.

Решение. Используя представление чисел в развернутой форме, запишем:

$3\cdot x + 5 = 29$. Решив данное уравнение, получим $x = 8$.

Ответ: Основание системы $x = 8$.

Приведем таблицу для перевода первых 16 чисел в различные системы счисления (табл. 3).

Таблица 3

Десятичные числа	Двоичные числа	Восьмеричные числа	Шестнадцатеричные числа
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

1.2 Двоичная система счисления

В технике, большое распространение получила двоичная система счисления. Основание двоичной системы равно двум, следовательно, в ее алфавите имеется только две цифры: 0 и 1.

Пример 1.5

Перевод десятичного числа в двоичную систему поясним на примере числа 27:

$$\begin{array}{r|l} 27:2=13 & \text{остаток } 1 \\ 13:2=6 & \text{остаток } 1 \\ 6:3=2 & \text{остаток } 0 \\ 2:2=1 & \text{остаток } 0 \\ 1:2=0 & \text{остаток } 1 \end{array} \uparrow$$

Читая снизу вверх цифры правой колонки, получаем искомое двоичное число: $27_{10} = 10011_2$.

Итак, для перевода целого числа из десятичной системы счисления в двоичную необходимо число разделить с остатком (нацело) на 2. Затем неполное частное, полученное от деления, нужно снова разделить с остатком на 2 и т. д., пока последнее полученное неполное частное не станет равным нулю. Представлением числа в двоичной системе счисления будет последовательность остатков деления, записанных в порядке, обратном порядку их получения.

Пример 1.6

Перевод десятичного вещественного числа в двоичную систему поясним на примере числа 0,61:

*	0,	61 2
*	1,	22 2
*	0,	44 2
*	0,	88 2
*	1,	76 2
	1,	53

↓

Читая сверху вниз, начиная со второй цифры значения целой части, полученной в результате умножения, получаем искомое значение.

Тогда, $0{,}61_{10} = 0{,}10011_2$.

Итак, для перевода вещественного числа в двоичную систему счисления требуется отдельно перевести целую часть (см. пример1.5), и отдельно перести дробную часть числа (см. пример1.6). Дробная часть числа умножается на 2 в столбик. Умножение продолжается до тех пор, пока дробная часть не станет нулевой или не будет обеспечена заданная точность, т.е. найдено необходимое количество знаков после запятой (для нахождения пяти знаков после запятой требуется провести умножение на 2 пять раз).

Перевод в десятичную систему числа, записанного в двоичной системе счисления, сводится к вычислению значения многочлена (1) средствами десятичной арифметики. Для двоичной системы счисления этот многочлен выглядит следующим образом:

$$A_q = a_n 2^n + a_{n-1} 2^{n-1} + \cdots + a_1 2^1 + a_0 2^0 + a_{-1} 2^{-1} + \cdots + a_{-m} 2^{-m} \quad (2)$$

где A_q ($A_q = a_n a_{n-1} a_{n-2} \ldots a_1 a_0, a_{-1} a_{-2} \ldots a_{-m}$) – любое число, записанное в двоичной системе счисления;

a_i - цифры числа;

$n+1$ – число целых разрядов;

m – число дробных разрядов

Пример 1.7

Перевести число $1001{,}1101_2$ в десятичную систему.

Применим формулу (2) $n = 3$, $m = 4$.

$1001{,}1101_2 = 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 + 1 \cdot 2^{-1} + 1 \cdot 2^{-2} + 0 \cdot 2^{-3} + 1 \cdot 2^{-4} =$

$= 8 + 0 + 0 + 1 + 0{,}5 + 0{,}25 + 0 + 0{,}0625 = 9{,}8125_{10}$.

Над двоичными числами можно выполнять те же операции, что и над десятичными. Главной из них является операция сложения. Сложение двоичных чисел осуществляется поразрядно, с запоминанием единиц переноса, точно так же, как и в десятичной системе. Поясним это на примере.

Пример 1.8

Пусть $a = 101011$, $b = 101110$, найдём их сумму $a + b$. Запишем числа a и b одно под другим, совместив младшие разряды:

+	1	0	1	0	1	1	- число a
	1	0	1	1	1	0	- число b
1	0	1	1	0	0	1	- число $a+b$
	(1)	(0)	(1)	(1)	(1)	(0)	- переносы

Как и в десятичной системе, суммирование начинаем с младшего разряда:

а) 1 + 0 = 1, переноса нет, под цифрой 1 (младший разряд числа a + b) записываем в скобках нуль;

б) во втором разряде суммируются единицы: 1 + 1 = 10, т. е. сумма равна

нулю и есть единица переноса. Записываем её под результирующим нулём второго разряда суммы;

в) в третьем разряде 0 + 1 = 1, но ещё надо прибавить единицу переноса из второго разряда, тогда 0 + 1 + 1 = 10. Снова сумма равна нулю и есть единица переноса;

г) в четвёртом разряде суммируются две единицы и к ним прибавляется единица переноса из третьего разряда: 1 + 1 + 1 = 11. В результате сумма равна 1 и есть единица переноса;

д) в пятом разряде 0 + 0 + 1 = 1, т.е. сумма равна единице, переноса нет;

е) в шестом разряде 1 + 1 = 10. Сумма равна нулю, а единица переноса образует седьмой разряд суммы a + b. Это эквивалентно записи, когда числа *a* и *b* удлиняют путем приписывания слева нулей, т.е. *a* = 0101011, *b* = 0101110.

Рассмотрим операцию вычитания двоичных чисел. Примеры вычитания двоичных чисел:

10-1=1

1-0=1

1-1=0

0-0=0

Для многоразрядных чисел, вычитание, аналогично операции сложения двоичных чисел, выполняется поразрядно. А также, имеет сходство с вычитанием десятичных чисел: если необходимо, то занимают единицу у большего разряда.

Пример 1.9

Пусть *a* = 101011, *b* = 101110, найдём их разность *b – a (из большего вычитаем меньшее)*. Запишем числа *b* и *a* одно под другим, совместив младшие разряды:

	(0)	(0)	(0)	(1)	(1)		- занимаемые разряды
	1	0	1	1	1	0	- число *b*
-	1	0	1	0	1	1	- число *a*
	0	0	0	0	1	1	- число *b – a*

Как и в десятичной системе, вычитание начинаем с младшего разряда:

а) 0 - 1 = 1, т.к. 10-1=1 (занимаем 1 у соседнего старшего разряда), сверху над соседним разрядом записываем в скобках цифру 1;

б) во втором разряде вычитаем: 1 - 1, но с учетом занимаемого разряда: 0-1 = 1, т.е. т.к. 10-1=1 (занимаем 1 у соседнего старшего разряда), сверху над соседним разрядом записываем в скобках цифру 1;

в) в третьем разряде 1-0, но с учетом занимаемого разряда: 0-0=0, сверху над соседним разрядом записываем в скобках цифру 0 (ничего не занимали);

г) в четвёртом разряде вычитаются две единицы: 1 - 1 =0, сверху над соседним разрядом записываем в скобках цифру 0 (ничего не занимали);

д) в пятом разряде 0 - 0 = 0, т.е. разность равна нулю, занимаемых разрядов нет;

е) в шестом разряде 1 - 1 = 0. Разность равна нулю.

Для перевода чисел из двоичной системы счисления в шестнадцатеричную, нужно разбить двочное число по четыре двоичные цифры справа налево (недостающие цифры заменить нулями), и заменить каждую четверку соответствующей шестнадцатеричной цифрой (см. таблица 3). Обратная замена позволяет переводить шестнадцатеричные числа в двоичные.

Пример 1.10

Число $5F7,A23_{16}$ перевести в двоичную систему счисления.

Решение. Заменим каждую цифру числа $5F7,A23_{16}$ двоичным числом длиной четыре разряда, т.е. 5→0101, F→1111, 7→0111, A→1010, 2→0010, 3→0011. Тогда получим $5F7,A23_{16} = 010111110111,101000100011_2$

1.3 Представление данных в компьютере

Любой компьютер предназначен для обработки, хранения, преобразования данных. Для выполнения перечисленных функций компьютер должен обладать некоторыми свойствами представления этих данных. Представление данных заключается в их преобразовании к виду, удобному для последующей обработки либо пользователем, либо компьютером. В зависимости от этого данные имеют внешнее и внутреннее представление (рис. 1.2.).

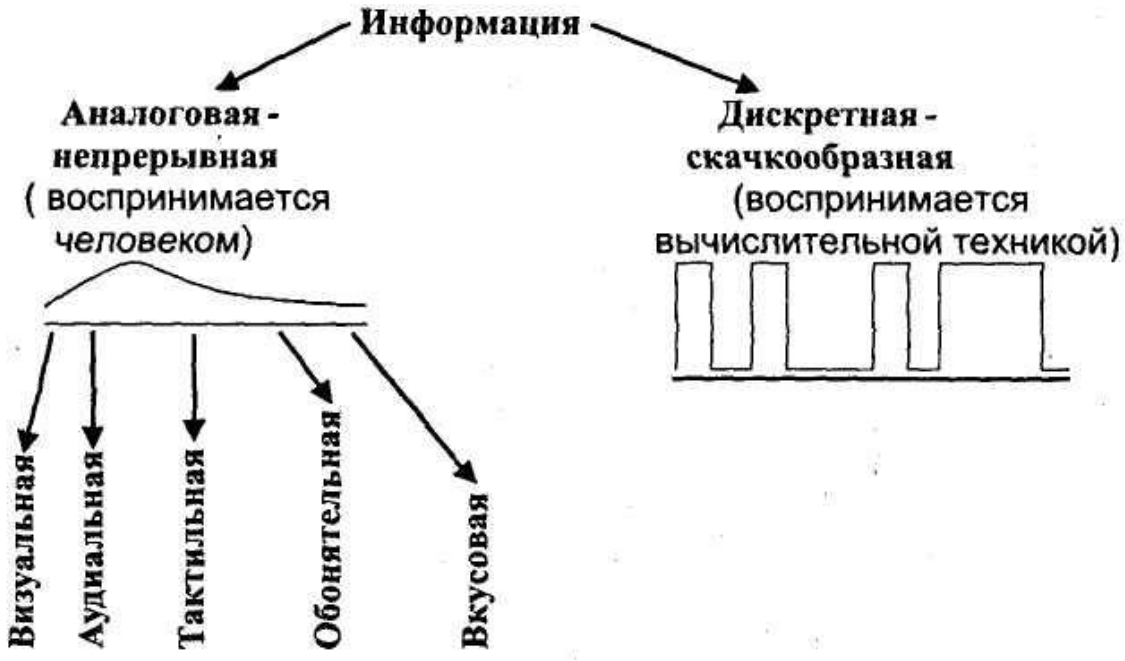


Рис.1.2. Виды информации по способу представления

Внешнее представление данных – это естественный и понятный для пользователя формат, в котором он вводит данные в компьютер и получает результат их обработки. Благодаря такому представлению пользователю легко и удобно работать с компьютером. Основными форматами данных, с которыми

работает пользователь, являются:

- числовые данные (целые и вещественные);
- текст (последовательность символов);
- изображение (графика, фотографии, рисунки, схемы);
- звук.

Внутреннее представление данных – это формат, в котором данные хранятся и обрабатываются внутри. Внутреннее представление данных определяется логикой работы компьютера, принципами организации его памяти, физическими принципами, по которым происходит обмен сигналами между аппаратными компонентами компьютера.

Для автоматизации работы с данными, относящимися к различным типам важно унифицировать их форму представления – для этого обычно используется прием кодирования, то есть выражение данных одного типа через данные другого типа: числовую, текстовую, графическую, звуковую информации переводят в цифровую. В ЭВМ применяется система двоичного кодирования, основанная на представлении данных последовательностью двух знаков: 1 и 0, которые называются двоичными цифрами (binary digit – сокращенно bit). Таким образом, единицей информации в компьютере является один бит, т.е. двоичный разряд, который может принимать значение 0 или 1. Восемь последовательных битов составляют байт. Байт - наименьшая единица обработки и передачи информации. В одном байте можно закодировать значение одного символа из 256 возможных ($256 = 2^8$). Более крупной единицей информации является килобайт (Кбайт), равный 1024 байтам ($1024 = 2^{10}$). Еще более крупные единицы измерения данных: мегабайт, гигабайт, терабайт (1 Мбайт = 1024 Кбайт; 1 Гбайт = 1024 Мбайт; 1 Тбайт = 1024 Гбайт).

Память компьютера имеет байтовую структуру, к ней привязывается представление любых данных. Более точным является утверждение, что для представления значений элементарных данных в памяти компьютера используется машинное слово; этот термин в информатике применяется в двух значениях:

Машинное слово – *это, во-первых совокупность двоичных элементов, обрабатываемая как единое целое в устройствах и памяти компьютера; а также, объем данных, содержащиеся в одной ячейке памяти компьютера.*

С технической точки зрения машинное слово объединяет запоминающие элементы, служащие для записи 1 бит информации, в единую ячейку памяти. Количество таких объединяемых элементов кратно 8, т.е. целому числу байт.

На ранних компьютерах размер слова совпадал также с минимальным размером адресуемой информации (разрядностью данных, расположенных по одному адресу); на современных компьютерах минимальным адресуемым блоком информации обычно является байт, а слово состоит из нескольких байтов. На ранних компьютерах слово было минимально адресуемой ячейкой памяти; сейчас чаще всего минимально адресуемой ячейкой памяти является байт, а слово состоит из нескольких байт. Это приводит к неоднозначному толкованию размера слова. Например, на процессорах 80386 и их потомках «словом» традиционно называют 16 бит (2 байта), хотя эти процессоры могут одновременно обрабатывать и более крупные блоки данных.

Кодирование целых чисел

Целые числа (от англ. *Integer*) хранятся и обрабатываются в компьютере в двоичном формате. При вводе число записывается в привычной для нас десятичной системе счисления, а компьютер переводит его в двоичный код.

В математике, как известно, целыми числами называют множество из натуральных чисел, противоположных им по знаку чисел и числа нуль. В вычислительной технике и программировании в связи с разным внутренним представлением различают целые числа **короткие** (англ. *short*), **длинные** (англ. *long*) и целые стандартной длины. Длина стандартного целого типа, как правило, совпадает с размером машинного слова. Для 16-разрядных операционных систем — этот тип (int) составляет 2 байта и совпадает с типом short int (можно использовать как short, опуская слово int), для 32-разрядных операционных систем он будет равен 4 байтам и совпадает с длинным целым long int (можно использовать как long, опуская слово int), и в этом случае будет составлять 4 байта. Короткое целое short int, для 16-разрядных операционных систем, 32-разрядных операционных систем, и для большинства 64-разрядных операционных систем составляет — 2 байта. Также в некоторых языках может использоваться тип данных двойное длинное long long, который составляет 8 байт. Для 64-разрядных операционных систем учитывая разность моделей данных, представление целого типа на разных моделях данных может отличаться между собой. Тип int и long может составлять как 4, так и 8 байт.

Каждый язык программирования реализует свою сигнатуру представления целых чисел, и может занимать 8, 16, 32, 64 бита. По аналогии с математикой в программировании целые типы подразделяются на беззнаковые (от англ. unsigned) и знаковые (от англ. signed). В C и C++ для обозначения беззнаковых типов используется префикс unsigned. В C# в качестве показателя беззнаковости используется префикс *u* (*англ. unsigned*). Например, для объявления беззнакового целого, равного по размеру одному машинному слову используется тип uint. В некоторых языках, например Java, беззнаковые целые типы отсутствуют.

Целые числа без знака представляются в двоичной системе счисления, при этом диапазон значений изменяется от 0 до $2^n - 1$, где n – количество двоичных разрядов и максимальное число соответствует единичным значениям кода во всех разрядах: 111…111. Так, если под число отводится 1 байт (8 бит), то оно может изменяться в диапазоне от 0 до 255, а если 2 байта (16 бит), то от 0 до 65535.

Представление числа 58 при однобайтном размещении показано на рис. 1.3. Число 58 переводится в двоинуюсистуму счислени: 111010_2. При однобайтном размешении отводится 8 бит для записи числа в памяти, поэтому дописываю нули слева и тогда: $58_{10} = 00111010_2$.

Рис. 1.3. Представление числа 58 в формате без знака

Целые числа со знаком представляются в компьютере иначе. Один, старший, двоичный разряд обозначает знак числа: 0 – неотрицательное число, 1 – отрицательное. Для кодирования отрицательных значений существует прямой, обратный и дополнительный код.

Положительные значения изображаются одинаково в прямом, обратном и дополнительном кодах – двоичными кодами с цифрой 0 в знаковом разряде.

Прямой код

Для представления числа в прямом коде n-разрядного формата нужно перевести число в двоичную систему счисления и дополнить слева нулями до n знаков. Так как старший разряд числа отводится для знака, а оставшиеся n – 1 разрядов – для значащих цифр, то в знаковый разряд записать 1, если число отрицательное, и оставить 0, если число положительное.

Структура представления числа в прямом коде изображена на рис. 1.4.

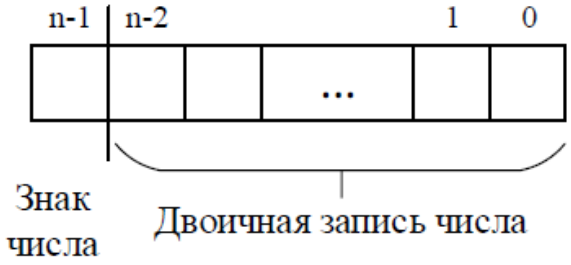


Рис. 1.4. Формат представления целого числа в прямом коде

Для примера, на рис. 1.5 и 1.6 показаны коды чисел 3_{10} и -3_{10} в однобайтном формате.

Рис. 1.5. Представление числа 3 в прямом коде в однобайтном фармате

Рис. 1.6. Представление числа -3 в прямом коде в однобайтном фармате

Прямой код имеет следующие недостатки. Во-первых, допускается существование как значения «плюс нуль» так и «минус нуль»: 00000000 и 10000000. Во-вторых, усложняется структура, так как операция сложения двух чисел с разными знаками, должна быть заменена на операцию вычитания меньшей величины из большей и присвоения результату знака большей величины.

Обратный код

Обратный код при суммировании двух чисел с разными знаками позволяет заменить вычитание на обычное сложение, но не решает проблему с «плюс нулем» и «минус нулем».

Для представления отрицательного числа в обратном коде n-разрядного формата нужно модуль отрицательного числа записать в прямом коде n двоичных разрядах (перевести число в двоичную систему счисления и дополнить слева нулями до n знаков). Значения всех знаков инвертировать (нули заменить единицами, единицы нулями).

Пример 1.11

Найти обратный код в однобайтном формате числа $x = -35_{10}$

Решение. Представим модуль числа x в двоичной системе и дополним нулями слева до 8 знаков: 00100011. Инвертируем значения всех знаков: 11011100.

Ответ: Обратный код в однобайтном формате числа x = 11011100.

Пример 1.12

Найти сумму чисел 5 и -7.

Решение. Найдем обратный код для обоих чисел:

$5_{10}=00000101_2$ (для положительного числа прямой и обратный код совпадают);

$-7_{10}=11111000_{2.}$

Найдем сумму двоичных чисел:

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

Полученная сумма будет отрицательным числом, т.е. старший разряд равен 1. Инвертируем полученное значение, получаем 00000010_2. Переводя полученный результат в десятичную систему с помощью таблицы 3, получаем ответ -2_{10}.

Ответ: -2_{10}.

Дополнительный код

В современных компьютерах, как правило, отрицательные числа представляют в виде дополнительного кода. В дополнительном коде, как и в обратном, при суммировании двух чисел с разными знаками не требуется менять операцию сложения на вычитание, но в отличие от прямого и обратного кода нуль не кодируется двумя разными значениями. Диапазон значений целых чисел в дополнительном коде изменяется от -2^{n-1} до $2^{n-1}-1$. Например, если под число отводится 1 байт, то оно может изменяться от –128 до +127, а если 2 байта, то от – 32768 до 32767.

Для представления отрицательного числа в дополнительном коде n-разрядного формата нужно представить его в обратном коде и прибавить 1 к последнему разряду числа.

Пример 1.13

Найти дополнительный код в двухбайтном формате числа $x = -562_{10}$

Решение. Представим модуль числа x в двоичной системе и дополним нулями слева до 16 знаков: 0000001000110010. Инвертируем значения всех знаков: 1111110111001101, прибавим к полученному обратному коду 1, получим: 1111110111001110.

Ответ: Дополнительный код в однобайтном формате числа x равен 1111110111001110.

Пример 1.14

Дополнительный код числа x имеет значение 11100111_2. Найти его значение в десятичной системе счисления.

Решение. Т.к. в первой позиции числа стоит 1, то искомое число будет отрицательным. Вычтем из заданного значения 1 (11100111 – 1 = 11100110). Инвертируем значения всех знаков: 00011001. Переведем полученное число в

десятичную систему $00011001_2 = 25_{10}$ и не забудем, что число является отрицательным.

Ответ: $x = –25$.

Пример 1.15

Найти сумму чисел -5 и -7.

Решение. Найдем дополнительный код для обоих чисел в однобайтном фармате:

-5_{10}: Переведем число 5 в двоичную систему счисления, получим 00000101_2 , инвертируем значения всех знаков: 11111010. Прибавим к полученному обратному коду 1, получим: 11111011

-7_{10}: Переведем число 7 в двоичную систему счисления, получим 00000111_2 , инвертируем значения всех знаков: 11111000. Прибавим к полученному обратному коду 1, получим: 11111001

Найдем сумму двоичных чисел:

+	1	1	1	1	1	0	1	1
	1	1	1	1	1	0	0	1
1	1	1	1	1	0	1	0	0

Полученная сумма будет отрицательным числом, т.к. старший разряд равен 1. Вычтем из полученного значения 1 (111110100 – 1 = 11110011). Инвертируем полученное значение, получаем 00001100_2. Переводя полученный результат в десятичную систему с помощью таблицы 3, получаем ответ -12_{10}.

Ответ: -12_{10}.

Кодирование вещественных чисел

Вещественные числа обычно представляются в виде чисел с плавающей запятой. Числа с плавающей запятой — один из возможных способов представления действительных чисел, который является компромиссом между точностью и диапазоном принимаемых значений, его можно считать аналогом экспоненциальной записи чисел, но только в памяти компьютера (нормализованное число). Представление вещественных чисел утверждено в нескольских стандартах, самым распространенным является IEEE 754 (*IEEE Standart for Binary Floating-Point Arithm etic*). Он широко используется в программных (компиляторы с разныхязыков программирования) и аппаратных (CPU и FPU) реализациях арифметических действий. Далее будем придерживаться данных стандарта IEEE 754.

При этом предполагается запись вещественного числа в виде:

$$x = \pm m \cdot q^{\pm p} \qquad (3)$$

где m – мантисса числа ($|m| < 1$);

q – основание системы счисления;

p – порядок числа (p – целое число).

Пусть нам дано число 666,66 в десятичной системе счисления, приведём его к экспоненциальной форме: $x = 0{,}66666 \cdot 10^3$ ($q = 10$, $p = 3$).

Пример в двоичной системе счисления:

$10110{,}01 = 0{,}1011001 \cdot 2^{101}$ ($q = 2$, $p = 101_2 = 5_{10}$)

Изобразим структуру представления вещественного числа (рис. 1.7). Здесь порядок p числа задается в смещенной форме. Смещенный порядок нормализованного числа вычисляется следующим образом: если для задания порядка выделено k разрядов, то к истинному значению порядка, представленного в дополнительном коде, прибавляют смещение, равное $(2^{k-1}-1)$. Таким образом, порядок, принимающий значения в диапазоне от -128 до +127, преобразуется в смещенный порядок в диапазоне от 0 до 255. Смещенный порядок хранится в виде беззнакового числа, что упрощает операции сравнения, сложения и вычитания порядков, а также упрощает операцию сравнения самих нормализованных чисел.

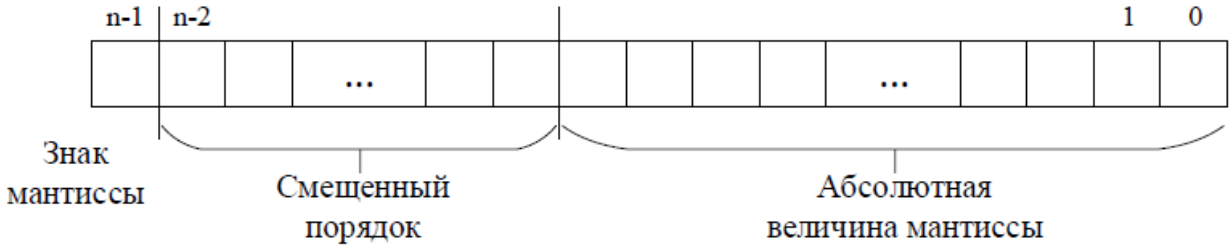


Рис. 1.7. Формат представления вещественного числа

Для хранения вещественного числа выделяется 4 байта (формат одинарной точности), 8 байт (формат двойной точности) или 10 байт (расширенный формат). В формате одинарной точности под смещенный порядок отводится 8 бит, а под мантиссу 23 бита; в формате двойной точности под смещенный порядок – 11 бит, под мантиссу – 52 бита; в расширенном формате – 15 бит и 64 бита соответственно.

Организация хранения значения чисела с плавающей запятой с обычной точностью: из 4 байт, выделенных под хранение цифр, 1 байт (8 разрядов) отдается под данные о порядке и его знаке, а 3 байта (24 разряда) уходят на хранение мантиссы и её знака по тем же принципам, что и для целочисленных значений. Зная это, мы можем провести нехитрые расчеты.

Максимальное значение $p = 1111111_2 = 127_{10}$.

Исходя из него, мы можем получить максимальный размер числа, которое может храниться в памяти компьютера. $x = 2^{127}$.

Теперь мы можем вычислить максимально возможную мантиссу. Она будет равна

$$2^{23} - 1 \geq 2^{23} = 2^{(10 \times 2{,}3)} \geq 1000^{2{,}3} = 10^{(3 \times 2{,}3)} \geq 10^7.$$

В итоге, мы получили приближенное значение. Если теперь мы объединим оба расчета, то получим значение, которое может быть записано без потерь в 4 байта памяти. Оно будет равно $x = 1{,}701411 * 10^{38}$. Остальные цифры были

отброшены, поскольку именно такую точность позволяет иметь данный способ записи.

Для чисел с двойной точностью обычно выделяется 11 разрядов для порядка и его знака, а также 53 разряда для мантиссы.

Тогда p=1111111111$_2$ = 1023$_{10}$;

$m = 2^{52} - 1 = 2^{(10*5,2)} = 1000^{5,2} = 10^{15,6}$.

Округляем в большую сторону и получаем максимальное число $x = 2^{1023}$.

Пример 1.15

Как храниться в памяти число 37,16$_{10}$.

Решение. Переведем число в двоичную систему счисления. При переводе в двоичное число не получается точного перевода 100101,(00101000111101011100) - дробная часть, заключенная в скобках, повторяется в периоде.

Переводим число в нормализованный вид:

0,100101(00101000111101011100) $\cdot 2^{110}$ (т.к. 110$_2$=6$_{10}$)

Представим вещественное число в 32-разрядном формате (таблица 4):

1. Знак числа «+», поэтому в знаковый разряд (31) заносим 0;

2. Для задания порядка выделено 8 разрядов, к истинному значению порядка, представленного в дополнительном коде, прибавляем смещение

(2^7—1)=127.

Так как порядок положительный, то прямой код порядка совпадает с дополнительным, вычислим смещенный порядок:

00000110 + 01111111=10000101

Заносим полученный смещенный порядок.

3. Заносим мантиссу, при этом старший разряд мантиссы убираем (он всегда равен 1).

Таблица 4

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

Пример 1.16

Задано число в шестнадцатеричной системе счисления F023A9,12C4. Как изменится число, если в его представлении запятую перенести на два знака влево? На три знака вправо?

Решение. Если задано число в системе счисления с основанием x, то перенос запятой в представлении числа на n знаков влево уменьшает его в x^n раз, а перенос запятой в представлении числа на n знаков вправо увеличивает его в x^n раз.

Ответ: Число F023A9,12C4 уменьшится в 16^2 (256) раз, если запятую перенести на два знака влево. Число F023A9,12C4 увеличится в 16^3 (4096) раз, если запятую перенести на три знака вправо.

Кодирование логических данных

Логические данные (от англ.*Boolean* или *logical data type)* принимают два значения: «Истина» («True», 1) или «Ложь» («False», 0). Даннай тип данных может быть реализован и храниться в памяти с использованием только одного бита, но обычно используется минимальная адресуемая ячейка памяти (обычно байт или машинное слово), как более эффективная с точки зрения быстродействия единица хранения при работе с регистрами процессора и оперативной памятью.

В некоторых языках программирования за значение *истина* полагается 1, за значение *ложь* — 0. Например, в Pascal для хранения логических переменных предусмотрен тип данных – BOOLEAN. Переменная типа BOOLEAN может принимать значения TRUE (истина) и FALSE (ложь). Логическая переменная занимает 1 байт в памяти компьютера.

Кодирование текстовых (символьных) данных

Правило кодирования символьных данных (букв алфавита и других символов) заключается в том, что каждому символу ставится в соответствие двоичный код.

Технически это выглядит просто, но существуют организационные сложности. В первые годы развития вычислительной техники эти сложности были связаны с отсутствием необходимых стандартов, а настоящее время вызваны, наоборот, изобилием одновременно действующих и противоречивых стандартов. Для того чтобы весь мир одинаково кодировал текстовые данные, нужны единые таблицы кодирования. Наиболее распространенный стандарт кодировки символов ASCII-код (American Standard Code for Information Interchange – американский стандартный код для обмена информацией) был введен институтом стандартизации США в 1963 г. и после модификации в 1977 г. был принят в качестве всемирного стандарта. Каждому символу в этой таблице поставлено в соответствие двоичное число от 0 до 255 (8-битовый двоичный код), например, A – 01000001, B – 01000010, C – 01000011, D – 01000100 и т. д.

В системе ASCII закреплены две таблицы кодирования – базовая и расширенная. Базовая таблица закрепляет значения кодов от 0 до 127 (рис. 1.8), а расширенная относится к символам с номерами от 128 до 255.

sp 32	! 33	« 34	# 35	$ 36	% 37	& 38	‘ 39	(40	) 41	* 42	+ 43	, 44	- 45	. 46	/ 47
0 48	1 49	2 50	3 51	4 52	5 53	6 54	7 55	8 56	9 57	: 58	; 59	< 60	= 61	> 62	? 63
@ 64	A 65	B 66	C 67	D 68	E 69	F 70	G 71	H 72	I 73	J 74	K 75	L 76	M 77	N 78	O 79
P 80	Q 81	R 82	S 83	T 84	U 85	V 86	W 87	X 88	Y 89	Z 90	[91	\ 92	] 93	^ 94	_ 95
` 96	a 97	b 98	c 99	d 100	e 101	f 102	g 103	h 104	i 105	j 106	k 107	l 108	m 109	n 110	o 111
p 112	q 113	r 114	s 115	t 116	u 117	v 118	w 119	x 120	y 121	z 122	{ 123	| 124	} 125	~ 126	del 127

Рис. 1.8. Базовая таблица ASCII

Первые 32 кода отданы производителям аппаратных средств. В этой области размещаются так называемые управляющие коды, которым не соответствуют никакие символы языков, и, соответственно, эти коды не выводятся ни на экран, ни на устройства печати, но ими можно управлять, например, тем, как производится вывод прочих данных.

Начиная с 32 кода по 127 размещены коды символов английского алфавита, знаков препинания, цифр, знаков арифметических действий, некоторые вспомогательные символы (код 32 – пробел, код 127 – delete). Национальные системы кодирования занимают расширенную часть, определяющую значения кодов с 128 до 255.

В России наиболее широкое применение нашли кодировки Windows 1251 (была введена компанией Microsoft), КОИ-8 (код обмена информации, восьмизначный), ISO (International Standard Organization – Международный институт стандартизации) – международная кодировка, в которой предусмотрена кодирование символов русского алфавита.

Организационные трудности, связанные с созданием единой системы кодирования текстовых данных, вызваны ограниченным набором кодов (256). Если, например, кодировать символы не восьмиразрядными двоичными числами, а числами с большим количеством разрядов, то и диапазон возможных значений кодов станет намного больше. Такая система, основанная на 16-разрядном кодировании символов, получила название UNICODE . Шестнадцать разрядов позволяют обеспечить уникальные коды для 65536 (2^{16}) различных символов – этого поля достаточно для размещения в одной таблице символов большинства языков планеты.

Несмотря на тривиальную очевидность такого подхода, простой механический переход на данную систему долгое время сдерживался из-за недостаточных ресурсов средств вычислительной техники (в системе кодирования UNICODE все тестовые документы автоматически становятся вдвое длиннее). Во второй половине 90-х г. прошлого столетия технические средства достигли необходимого уровня обеспеченности ресурсами, и сегодня, в основном, осуществлен переход документов и программных средств на универсальную систему кодирования.

Пример 1.17

Сообщение из 30 символов было записано в 8-битной кодировке Windows-1251. После вставки в текстовый редактор сообщение было перекодировано в 16-битный код Unicode. На сколько байт увеличилось при этом количество памяти?

Решение. При перекодировке из Windows-1251 в Unicode объем памяти увеличивается в два раза, т.е. если в кодировке Windows-1251 сообщение занимало 30·8 = 240 бит, то в кодировке Unicode сообщение займет 30·16 = 480 бит, т.е. количество памяти увеличилось на 480 – 240 = 240 бит или 240/8 = 30 байт.

Ответ: сообщение увеличилось на 30 байт.

Пример 1.18

Отправлено SMS-сообщение:

Чтоб мудро жизнь прожить, знать надобно немало,
Два важных правила запомни для начала:
Ты лучше голодай, чем что попало есть,
И лучше будь один, чем вместе с кем попало.

В мобильном телефоне адресата установлено ограничение размера входящего SMS-сообщения 64 байтами (при превышении этого размера сообщение автоматически делится на части). Каждый символ кодируется 16 битами. На сколько частей будет разбито сообщение?

Решение. Всего символов в сообщении 166. Т. к. каждый символ кодируется 16 битами (2 байтами), то сообщение занимает 166·2 = 332 байта. Теперь вычислим, на сколько частей будет разбито сообщение: 336/64 = 5,1875.

Ответ. Сообщение будет разбито на 6 частей.

Кодирование графических данных

Графические данные хранится и обрабатывается в двоичном коде.

Существуют два принципиально разных подхода к кодированию (представлению) графических данных: растровый и векторный.

При растровом представлении вся область данных разбивается на множество точечных элементов – пикселей, каждый из которых имеет свой цвет. Число пикселей по горизонтали и вертикали определяет *разрешение* изображения.

При растровом способе представления графических данных под каждый пиксель отводится определенное число битов, называемого *битовой глубиной* или *информационной емкостью одного пикселя*, и используемое для кодирования цвета пикселя. Каждому цвету соответствует двоичный код. Например, если битовая глубина равна 1, то под каждый пиксель отводится 1 бит. В этом случае 0 соответствует черному цвету, 1 – белому, а изображение может быть только черно-белым. Если битовая глубина равна 2, то каждый пиксель может быть закодирован цветовой гаммой из 4 цветов (2^2) и т. д. Для качественного представления графических данных в современных компьютерах используются цветовые схемы с битовой глубиной 8, 24, 32, 40, т.е. каждый пиксель может иметь 2^8, 2^{24}, 2^{32}, 2^{40} оттенков. Количество цветов N, отображаемых на экране монитора, может быть вычислено по формуле:

$$N = 2^i, \tag{3}$$

где i – битовая глубина.

Если известны размеры (в пикселях) рисунка по высоте X и ширине Y, а также битовая глубина i, то занимаемый объем V будет равен:

$$V = X \cdot Y \cdot i. \tag{4}$$

Основным недостатком растровой графики является большой объем памяти, необходимый для хранения изображения. Это объясняется тем, что запо-

минается цвет каждого пикселя, общее число которых задается разрешением.

При векторном представлении графических данных задается и впоследствии сохраняется математическое описание графического примитива – геометрического объекта (отрезка, окружности, прямоугольника и т.п.), из которых формируется изображение. Например, для воспроизведения окружности достаточно запомнить положение ее центра, радиус, толщину и цвет линии. Благодаря этому для хранения векторных графических данных требуется значительно меньше памяти.

Для представления цвета используются цветовые модели.

Цветовая модель – это правило, по которому может быть вычислен цвет. Самая простая цветовая модель – битовая. В ней для описания цвета каждого пикселя (черного или белого) используется всего один бит. Для представления полноцветных изображений используются более сложные модели, среди которых самые известные – модели RGB и CMYK.

Цветовая модель RGB используется в таких устройствах, как телевизионные кинескопы, компьютерные мониторы. Цветовая модель RGB (Red-Green-Blue, красный-зеленый-синий) основана на том, что любой цвет может быть представлен как сумма трех основных цветов: красного, зеленого и синего.

В основе цветовой модели лежит декартова система координат. Цветовое пространство представляет собой куб сочетаний трех базовых цветов (рис. 1.9).

Любой оттенок цвета при этом выражается набором из трех чисел. Обычно, на каждое число отводится один байт, поэтому интенсивность одного цвета имеет 256 значений (0-255), общее количество оттенков цвета – 16777216 (2^{24}). Белый цвет в RGB представляется как (255,255,255), черный – (0,0,0,0), красный – (255,0,0), зеленый – (0,255,0), синий – (0,0,255).

Цветовая модель CMYK используется в полиграфии.

Цветовая модель CMY является производной модели RGB и также построена на базе трех цветов: C – Cyan (голубого), M – Magenta (пурпурного), Y – Yellow (желтого), которые образуются следующим образом.

Голубой цвет C (0, 255, 255) является комбинацией синего и зеленого, желтый цвет Y (255, 255, 0) – зеленого и красного, а пурпурный цвет M (255, 0, 255) – красного и синего, иначе каждому из основных цветов ставится в соответствие дополнительный цвет (дополняющий основной до белого).

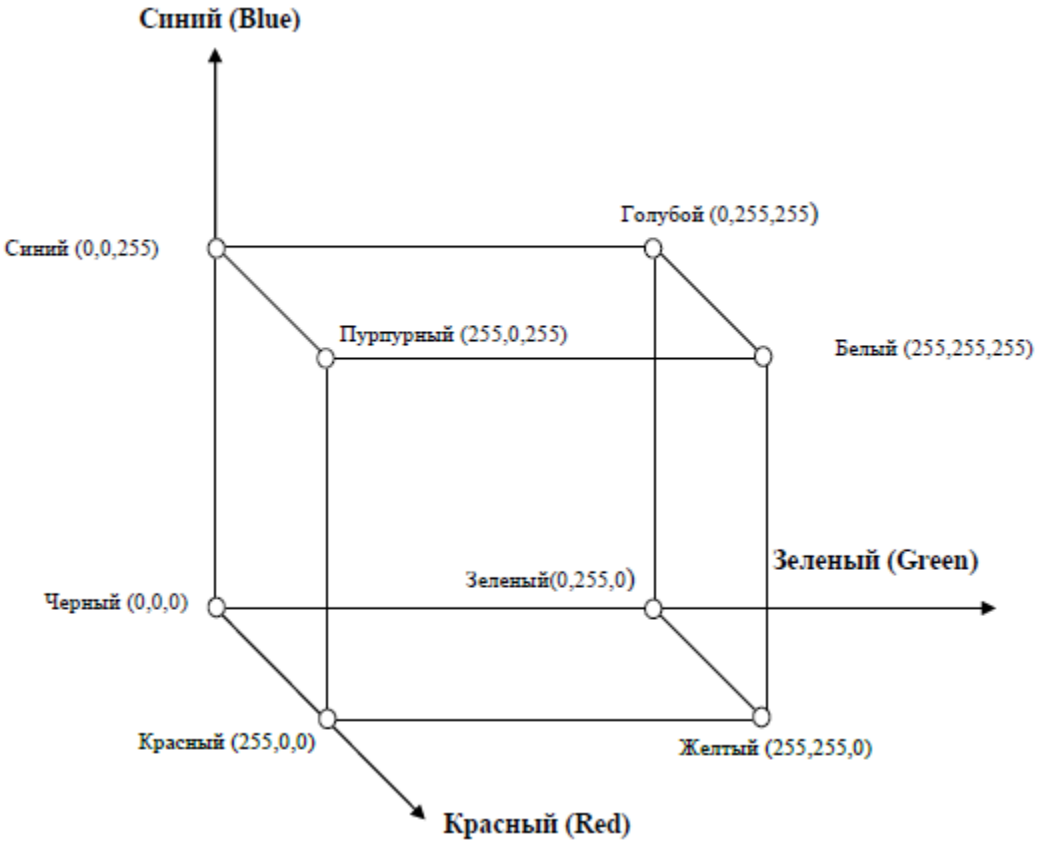


Рис. 1.9. Цветовая модель RGB

Дополнительными цветами для красного является голубой, для зеленого – пурпурный, для синего – желтый.

Смешение голубого, пурпурного и желтого цветов должно давать черный цвет, который, однако, выглядит осветленным по сравнению с оригиналом. Поэтому для получения чистого черного цвета при печати цветовая модель CMY расширяется до модели CMYK, содержащей четвертый основной цвет – черный (K – black).

Пример 1.19

Растровое графическое изображение 20×20 точек содержит не более 256 цветов. Сколько памяти потребуется для хранения изображения?

Решение. Для решения воспользуемся формулой (3) для вычисления количества цветов N, отображаемых на экране монитора при известной битовой глубине i. Одна точка может иметь один из 256 цветов (N = 256). Найдем сколько бит i, требуется для ее хранения (битовая глубина) из соотношения:

$256 = 2^i$

i = 8 (бит).

Для хранения изображения 20×20 точек требуется 20·20·8 = 3200 бит или 400 байт (3200/8 = 400).

Ответ: Для хранения изображения потребуется 400 байт.

Кодирование звуковой информации

Звук представляет собой звуковую волну с непрерывно меняющейся амплитудой и частотой. В процессе кодирования непрерывного сигнала производится его временная дискретизация и квантование.

Дискретизация заключается в замерах величины аналогового сигнала огромное множество раз в секунду (рис. 1.10). Полученной величине аналогового сигнала сопоставляется определенное значение из заранее выделенного диапазона: 256 (8 бит) или 65536 (16 бит). Привидение в соответствие уровня сигнала определенной величине диапазона называется квантованием.

Как бы часто не проводились измерения, все равно часть информации будет теряться. Но чем чаще проводятся замеры, тем точнее будет соответствовать цифровой звук своему аналоговому оригиналу. Также, чем больше бит отведено под кодирование уровня сигнала (квантование), тем точнее соответствие.

С другой стороны, звук хорошего качества будет содержать больше данных и, следовательно, больше занимать места на цифровом носителе информации.

Определить информационный объем V цифрового аудио файла, длительность звучания которого составляет t секунда при частоте дискретизации H и разрешении i битов (квантуют i битами) можно по формуле:

$$V = H \cdot t \cdot i \tag{5}$$

Если требуется определить информационный объем стерео аудио файл, то полученные вычисления умножаются на 2:

$$V = H \cdot t \cdot i \cdot 2 \tag{6}$$

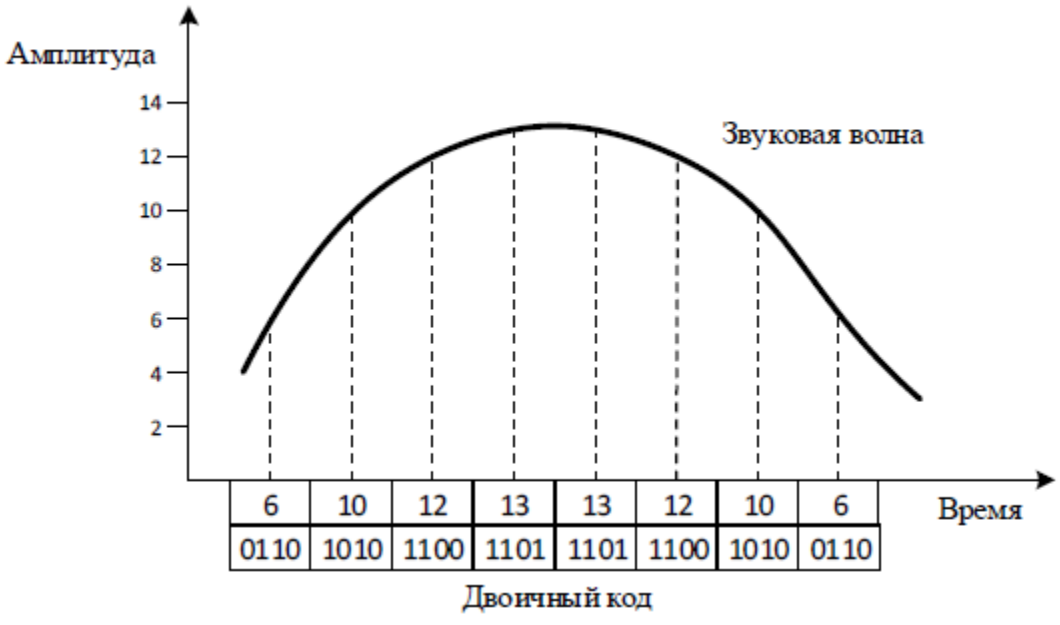


Рис. 1.10. Дискретизация звуковой волны

Пример 1.20

Определить информационный объем цифрового стерео-аудио файла, длительность звучания которого составляет 10 секунд при частоте дискретизации 22,05 кГц и разрешении 8 битов (квантуется 8 битами).

Решение. Для определения информационного объема цифрового стерео-аудио файла воспользуемся формулой (6):

V = 22050·8·10·2 = 3528000(бит) = 3528000/8/1024/1024 (Мбайт) = 0,42 (Мбайт).

Ответ. Информационный объем цифрового стерео-аудио файла составляет 0,42 Мбайт.

1.4 Упражнения для самостоятельного решения к главе 1

1.1. Переведите в десятичную систему счисления двоичные числа: 10010; 10011110; 1011100; 10000000; 1110001; 10001000; 10000001; 11111111; 11010001; 11111100.

1.2. Переведите в двоичную систему десятичные числа: 12; 25; 64; 10; 32; 60; 16; 30; 31; 17; 49; 63.

1.3. Представьте сумму двоичных чисел в двоичной системе: 1010 + 1101; 1111 + 100; 1100 + 1000; 11111 + 1; 1001 + 10000; 10 + 10100.

1.4. Вместо крестиков поставьте двоичные знаки, если:

$11\times0\times\times0_2 = 112_{10}$;

$\times\times\times0000_2 = 80_{10}$;

$1\times\times1\times\times11_2 = 255_{10}$;

$\times0\times\times0\times1_2 = 67_{10}$;

$\times\times000\times\times0_2 = 128_{10}$;

$\times\times000\times\times_2 = 96_{10}$.

1.5. Перечислите все двоичные четырёхзначные числа, содержащие точно одну единицу.

1.6. Представьте в десятичной системе двоичные числа:

0110; 0111; 1001; 0001; 1110;

1101; 1010; 0100; 1000; 0011;

0001; 1000; 0100; 1011; 0101.

1.7. Укажите десятичные числа, двоичные эквиваленты которых содержат точно две единицы:

3 7 9 12 15;

3 8 9 14 18;

1 4 6 13 14;

6 10 13 17 19;

2 3 5 8 12;

3 10 20 24 28.

1.8. Сколько существует 10-разрядных двоичных чисел?

1.9. В какой системе счисления выполнены действия:

122 + 2 = 201?

1.10. В какой системе счисления выполнены действия:

127 + 2 = 131?

1.11. Число 23_x из системы счисления с основанием x перевели в десятичную систему счисления и получили 19_{10}. Найти основание системы счисления x.

1.12. Число 135_x из системы счисления с основанием x перевели в десятичную систему счисления и получили 59_{10}. Найти основание системы счисления x.

1.13. Обратный код числа x имеет значение 11100111_2. Найти его значение в десятичной системе счисления.

1.14. Найти дополнительный код в однобайтном формате числа 11_{10}.

1.15. Найти дополнительный код для числа $x = -24_{10}$ в однобайтном формате.

1.16. Дополнительный код числа x имеет значение 10101101. Найти его значение в десятичной системе счисления.

1.17. Даны три числа 33, 66, 88 в различных системах счисления. К этим числам прибавили по единице и получили во всех системах счисления 100. Найти значения всех этих чисел в десятичной системе счисления.

1.18. Изобразите структуры представления вещественных чисел:

3,00057

25,37;

210,125.

1.19. Для хранения растрового изображения размером 128x128 пикселей отвели 4 Кбайта памяти. Каково максимально возможное число цветов в палитре изображения?

1.20.Укажите минимальный объем памяти (в килобайтах), достаточный для хранения любого растрового изображения размером 64×64 пикселя, если известно, что в изображении используется палитра из 256 цветов. Саму палитру хранить не нужно.

1.21. Системный администратор ограничил длительность непрерывного подключения компьютеров сотрудников организации к сети Интернет 10 минутами. Сотруднику требуется переслать файл размером 100 Мбайт. Скорость передачи информации с рабочего места (компьютера) сотрудника в среднем составляет 512 Кбит/с. На сколько частей необходимо разделить файл для пересылки?

1.22. Для регистрации на сайте пользователю необходимо придумать пароль длиной ровно 11 символов. В пароле можно использовать десятичные цифры и 32 различных символа местного алфавита, причем все буквы используются в двух начертаниях – строчные и прописные. Каждый символ кодируется одинаковым и минимально возможным количеством бит, а каждый пароль – одинаковым и минимально возможным количеством байт. Каков объем памяти, необходимый для хранения 50 паролей?

1.23. Каков объем памяти для хранения цифрового аудиофайла, время звучания которого составляет две минуты при частоте дискретизации 44,1 кГц и разрешении 16 бит.

1.24. Автоматическое устройство осуществило перекодировку информационного сообщения на русском языке, первоначально записанного в 16-битном коде Unicode, в 8-битную кодировку КОИ-8. При этом информационное сообще-ние уменьшилось на 480 бит. Какова длина сообщения в символах?

Индивидуальные задания на тему «Дискретное представление информации»

Вариант 1

1. Запишите дополнительные коды чисел: –34; 119 в однобайтном формате.

2. Даны однобайтные коды двух целых чисел: 00101010 и 10011000. Известно, что отрицательное число представлено в дополнительном коде. Запишите значения целых чисел в десятичной системе.

3. За 45 секунд был распечатан текст. Подсчитать количество страниц в тексте, если известно, что в среднем на странице 50 строк по 75 символов в каждой, скорость печати лазерного принтера 8 Кбит/с., 1 символ – 1 байт. Ответ округлить до целой части.

Вариант 2

1. Запишите дополнительные коды чисел: –42; 83 в однобайтном формате.

2. Даны однобайтные коды двух целых чисел: 00101001 и 10101010. Известно, что отрицательное число представлено в дополнительном коде. Запишите значения целых чисел в десятичной системе.

3. Лазерный принтер печатает со скоростью в среднем 7 Кбит в секунду. Сколько времени понадобится для распечатки 12-страничного документа, если известно, что на одной странице в среднем по 45 строк, в строке 60 символов (1 символ – 1 байт). Результат округлить до целой части.

Вариант 3

1. Запишите дополнительные коды чисел: –11; 97 в однобайтном формате.

2. Даны однобайтные коды двух целых чисел: 00010010 и 10000001. Известно, что отрицательное число представлено в дополнительном коде. Запишите значения целых чисел в десятичной системе.

3. Автоматическое устройство осуществило перекодировку информационного сообщения на русском языке, первоначально записанного в 16-битном коде Unicode, в 8-битную кодировку КОИ-8. При этом информационное сообщение уменьшилось на 480 бит. Какова длина сообщения в символах?

Вариант 4

1. Запишите дополнительные коды чисел: –60; 88 в однобайтном формате.

2. Даны однобайтные коды двух целых чисел: 01011000 и 10001001. Известно, что отрицательное число представлено в дополнительном коде. Запишите значения целых чисел в десятичной системе.

3. Скорость передачи данных через модемное соединение 28 Кбит/с. Передача текстового файла заняла 10 с. Определите, сколько символов содержал переданный текст, если известно, что он был представлен в кодировке Unicode.

4. Запишите дополнительные коды чисел в однобайтном формате.

Вариант 5

1. Запишите дополнительные коды чисел: –49; 64 в однобайтном формате.

2. Даны однобайтные коды двух целых чисел: 00000100 и 10011000. Известно, что отрицательное число представлено в дополнительном коде. Запишите значения целых чисел в десятичной системе.

3. Для хранения растрового изображения размером 128 на 128 пикселей отвели 4 Кбайта памяти. Каково максимально возможное число цветов в палитре изображения?

Вариант 6

1. Запишите дополнительные коды чисел: –26; 57 в однобайтном формате.

2. Даны однобайтные коды двух целых чисел: 00010110 и 10010001. Известно, что отрицательное число представлено в дополнительном коде. Запишите значения целых чисел в десятичной системе.

3. Скорость передачи данных через ADSL-соединение равна 128000 бит/с. Через данное соединение передают файл размером 625 Килобайт. Определите время передачи файла в секундах.

Вариант 7

1. Запишите дополнительные коды чисел: –48; 20 в однобайтном формате.

2. Даны однобайтные коды двух целых чисел: 01001001 и 10000110. Известно, что отрицательное число представлено в дополнительном коде. Запишите значения целых чисел в десятичной системе.

3. Системный администратор ограничил длительность непрерывного подключения компьютеров сотрудников организации к сети Интернет 10 мин. Сотруднику требуется переслать файл размером 100 Мбайт. Скорость передачи информации с рабочего места (компьютера) сотрудника в среднем составляет 512 Килобит/с. На сколько частей необходимо разделить файл для пересылки?

Вариант 8

1. Запишите дополнительные коды чисел: –22; 70 в однобайтном формате.

2. Даны однобайтные коды двух целых чисел: 00110010 и 10010110. Известно, что отрицательное число представлено в дополнительном коде. Запишите значения целых чисел в десятичной системе.

3. Определить объем памяти для хранения цифрового аудиофайла, время звучания которого составляет две минуты при частоте дискретизации 44,1 кГц и разрешении 16 битов.

Вариант 9

1. Запишите дополнительные коды чисел: –41; 74 в однобайтном формате.

2. Даны однобайтные коды двух целых чисел: 00010001 и 10001111. Известно, что отрицательное число представлено в дополнительном коде. Запишите значения целых чисел в десятичной системе.

3. Укажите минимальный объем памяти (в килобайтах), достаточный для хранения любого растрового изображения размером 64 на 64 пикселя, если известно, что в изображении используется палитра из 256 цветов. Саму палитру хранить не нужно.

Вариант 10

1. Запишите дополнительные коды чисел: –44; 101 в однобайтном формате.

2. Даны однобайтные коды двух целых чисел: 00000101 и 10000100. Известно, что отрицательное число представлено в дополнительном коде. Запишите значения целых чисел в десятичной системе.

3. На магнитном диске объемом 30 Мбайт записана книга. В книге 1552 страницы. Из них страниц с текстом на 752 больше, чем страниц с рисунками. Страница с текстом содержит 640 символов. Все рисунки восьми цветные и имеют единый формат. Определите размер рисунков.

Глава 2
Алгебра логики

Традиционная логика берет начало в древней Греции и Риме. Недаром ее называют Аристотелевой. До начала XIX в. формальная логика практически не выходила за рамки силлогических умозаключений. Однако, начиная с работ Джона Буля, можно говорить о превращении ее в математическую логику. Особенностью математической логики является математический аппарат.

В 1910 г. появились первые работы, связанные с применением логики высказываний для описания переключательных цепей в телефонной связи. А в 1938–1940 гг. почти одновременно в СССР, США и Японии появились работы о применении математической логики в цифровой технике.

Основу ЭВМ и других цифровых устройств составляют элементарные логические схемы, которые работают в строгом соответствии с законами и правилами алгебры логики. Знание и понимание этих законов и правил помогает лучше разобраться с принципами работы ЭВМ.

Алгебра логики (булева алгебра – по фамилии ученого Д. Буля) является частью раздела математики под названием математическая логика, посвященного изучению математических доказательств и вопросов оснований математики. Построенная Д. Булем алгебра служила для описания логических действий над высказываниями. В честь ученого переменные логического типа в языках программирования назвали булевскими переменными (тип Boolean в языках Basic и Pascal, bool в C, C++).

2.1 Понятие высказывания

Высказывание - это некоторое утверждение в виде повествовательного предложения, по содержанию которого можно сказать, истинно оно или ложно. Примеры истинных высказываний: «Число 14 делится на 2 и 7»; «Существуют чётные числа, делящиеся на 3»; «Хабаровск стоит на Амуре». Примеры ложных высказываний: «Париж – столица Испании»; «3 < 1».

Существуют утверждения, которые меняли свою истинность по мере развития науки. Например: «Солнце вращается вокруг Земли». Это высказывание длительное время считалось истинным. Теперь же оно ложно.

Встречаются утверждения, относительно истинности которых невозможно сказать что-либо определённое, ввиду отсутствия способов их доказательства или опровержения. Например: «Между людьми существует телепатическая связь». По мере развития науки это утверждение может стать либо истинным, либо ложным.

В некоторых случаях утверждения объявляются истинными без каких-либо объяснений и доказательств. Например: «На плоскости через точку, лежащую вне прямой, можно провести только одну прямую, не пересекающую данной». Это утверждение Евклида. А Н.И. Лобачевский о том же утверждает: «На плоскости через точку, лежащую вне прямой, можно провести сколько угодно прямых, не пересекающих данной». Во втором высказывании утверждается нечто, противоположное первому. Однако оба высказывания истинны!

Возможно ли это? Да. Оба высказывания являются аксиомами, которые, как известно, принимаются истинными без доказательств. Мы в дальнейшем будем рассматривать только такие утверждения, которые определяются однозначно либо истинными, либо ложными.

Для удобства высказывания условимся обозначать латинскими буквами. Например, можно считать, что А — это высказывание «Идёт дождь». Если оно является истинным, то пишут А = 1. Соответственно запись А = 0 обозначает: высказывание «Идёт дождь» ложно. Всякая буква, обозначающая некоторое высказывание, - это переменная величина, принимающая одно из двух значений — либо 0, либо 1. Такую переменную называют двоичной. Таким образом, высказывания фактически являются двоичными объектами.

Высказывание, представляющее собой одно утверждение, принято называть *простым* или элементарным. Высказывания, которые получаются из простых с помощью логических операций, принято называть *сложными* или составными. Из вышеприведенных высказываний, высказывание «Число 14 делится на 7», является сложным.

Простые высказывания соответствуют логическим переменным, а сложные – логическим функциям или выражениям. Рассмотрим пример построения логической формулы. Для высказывания «Если при выполнении программы отклонение контролируемых параметров превышает предусмотренные нормы, то требуется оперативная корректировка программы или уточнение стандартов», при обозначениях: *A* – «отклонение контролируемых параметров превышает предусмотренные нормы»; *B* – «требуется оперативная корректировка программы»; *C* – «требуется уточнение стандартов», логическую формулу $f(A,B,C)$ можно записать в виде «если *A*, то *B* или *C*».

2.2 Логические операции

Основными операциями алгебры логики являются отрицание, конъюнкция, дизъюнкция, импликация и эквиваленция. В вычислительной технике также часто используется операция исключающее ИЛИ.

Отрицание

Операция отрицания является унарной, т.к. имеет один аргумент. Иначе ее называют *инверсией*, *дополнением*, *НЕ* и обозначают: ¬X, $\overline{X}$, NOT *X*.

Отрицанием $\overline{X}$ некоторого высказывания *X* называется такое высказывание, которое истинно, когда ложно исходное высказывание *X*, и ложно, когда истинно исходное высказывание *X*.

Определение отрицания может быть записано с помощью таблицы истинности (табл. 5):

Таблица 5

X	$\overline{X}$
1	0
0	1

Конъюнкция

Операцию конъюнкции еще называют *логическим умножением, логическим И.* Для обозначения данной операции используют символы ∧, &, точку, которую можно опускать, *AND.*

Конъюнкцией двух высказываний X и Y называется такое высказывание, которое истинно тогда и только тогда, когда истинны оба высказывания X и Y.

Определение конъюнкции может быть записано в виде таблицы истинности:

Таблица 6

X	Y	$X \wedge Y$
1	1	1
1	0	0
0	1	0
0	0	0

Определение конъюнкции двух высказываний естественным образом распространяется на любое конечное число составляющих: конъюнкция истинна тогда и только тогда, когда истинны все высказывания, следовательно, принимает значение «ложь», когда ложно хотя бы одно из этих высказываний.

Дизъюнкция

Операцию дизъюнкции иначе называют *логическим сложением, логическим ИЛИ.* Для обозначения логического сложения используют символы ∨, +, *OR.*

Дизъюнкцией двух высказываний и называется такое высказывание, которое истинно тогда и только тогда, когда истинно хотя бы одно из этих высказываний. Определение дизъюнкции может быть записано в виде таблицы истинности:

Таблица 7

X	Y	$X \wedge Y$
1	1	1
1	0	1
0	1	1
0	0	0

Определение дизъюнкции двух высказываний естественным образом распространяется на любое конечное число составляющих: дизъюнкция истинна тогда и только тогда, когда истинно хотя бы одно из этих высказываний, а, следовательно, принимает значение «ложь», когда все высказывания ложны.

Импликация

Операцию импликации иначе называют логическим следованием и для обозначения используют символ →. Импликацией двух высказываний и называется такое высказывание, которое ложно тогда и только тогда, когда истинно первое и ложно второе. Определение импликации может быть записано в виде таблицы истинности:

Таблица 8

X	Y	$X \rightarrow Y$
1	1	1
1	0	0
0	1	1
0	0	1

Эквиваленция

Операцию эквиваленции иначе называют *логическим тождеством, эквивалентностью* и для обозначения используют символы =,↔ , ~.

Эквиваленцией двух высказываний и называется такое высказывание, которое истинно тогда и только тогда, когда оба эти высказывания истинны или оба ложны.

Определение эквиваленции может быть записано в виде таблицы истинности:

Таблица 9

X	Y	$X \leftrightarrow Y$
1	1	1
1	0	0
0	1	0
0	0	1

Исключающее ИЛИ

Операция исключающее ИЛИ (*неравнозначность, сложение по модулю два*) обозначается символом $\oplus$.

Таким образом, неравнозначностью двух высказываний называют такое высказывание, которое истинно тогда и только тогда, когда одно из этих высказываний истинно, а другое ложно.

Определение данной операции может быть записано в виде таблицы истинности:

Таблица 10

X	Y	$X \oplus Y$
1	1	0
1	0	1
0	1	1
0	0	0

Операция исключающее ИЛИ фактически сравнивает на совпадение два двоичных разряда.

2.3 Логические формулы

Всякое сложное выражение, которое может быть получено из булевых переменных посредством применения логических операций отрицания, конъюнкции, дизъюнкции, импликации и эквиваленции, исключающего ИЛИ называется формулой алгебры логики.

Логическое значение формулы алгебры логики полностью определяется логическими значениями входящих в неё булевых переменных. С помощью логических операций над булевыми переменными можно строить различные сложные выражения. При вычислении значения логического выражения принято следующее старшинство (приоритет) логических операций:

- инверсия;
- конъюнкция;
- дизъюнкция;
- импликация;
- эквиваленция.

Для изменения указанного порядка используют скобки.

Как и в случае с логическими операциями все возможные логические значения формулы, в зависимости от значений входящих в неё булевых переменных, могут быть описаны полностью с помощью таблицы истинности.

Пример 2.1

Составить таблицу истинности для формулы $\bar{X} \vee Y \to X \wedge \bar{Y}$

Решение:

Таблица 11

X	Y	$\bar{X}$	$\bar{Y}$	$\bar{X} \vee Y$	$X \wedge \bar{Y}$	$\bar{X} \vee Y \to X \wedge \bar{Y}$
1	1	0	0	1	0	0
1	0	0	1	0	1	1
0	1	1	0	1	0	0
0	0	1	1	1	0	0

Легко видеть, что если формула содержит n элементарных высказываний (которые называют переменными), то она принимает 2^n значений, состоящих из нулей и единиц, или, что то же, таблица содержит 2^n строк. Другими словами, для формулы, содержащей два высказывания таблица истинности содержит 4 строки, а для формулы, содержащей три высказывания 8 строк.

Две формулы алгебры логики называются *равносильными*, если они принимают одинаковые логические значения на любом наборе значений, входящих в формулы элементарных высказываний. Обозначается равносильность ≡.

Следующие формулы являются равносильными:

$$\begin{gathered} \bar{\bar{x}} \equiv x, \\ x \vee x \equiv x, \\ (x \wedge \bar{x}) \vee y \equiv y. \end{gathered} \tag{7}$$

Формула называется *тождественно истинной* (или *тавтологией*), если она принимает значение 1 (истина) при всех значениях входящих в нее переменных.

Следующие формулы являются тавтологиями:

$$\begin{gathered} x \vee \bar{x}, \\ x \to (y \to x). \end{gathered} \tag{8}$$

Формула называется *тождественно ложной*, если она принимает значение 0 (ложна) при всех значениях входящих в нее переменных.

Тождественно ложной является формула:

$$x \wedge \bar{x}. \tag{9}$$

Логические операции инверсии, дизъюнкции, конъюнкции образуют **полную систему логических операций**, из которых можно построить сколь угодно сложное логическое выражение.

Формулы для выражения логических операций импликации, эквиваленции, исключающее ИЛИ через операции инверсии, дизъюнкции, конъюнкции имеют вид:

$$x \to y \equiv \bar{x} \vee y, \tag{10}$$

$$x \leftrightarrow y \equiv (x \to y) \wedge (y \to x) \equiv (\bar{x} \vee y) \wedge (\bar{y} \vee x), \tag{11}$$

$$x \oplus y \equiv (\bar{x} \wedge y) \vee (\bar{y} \wedge x). \tag{12}$$

Формулы, содержащие только операции конъюнкции, дизъюнкции и отрицания называются **булевыми**. Более того, всякая логическая формула может быть представлена булевой формулой.

Преобразования к виду булевой формы и обратно производятся применяя законы логики. Законы логики записываются в виде формул, которые позволяют производить эквивалентные преобразования логических выражений. Пред-

ставим основные законы алгебры логики в таблице 12.

Логические функции трех и более переменных также могут задаваться таблицами истинности или формулами, состоящими из символов переменных и знаков логических операций. Таким образом, формула наряду с таблицей служит способом задания и вычисления функций. В общем случае формула описывает логическую функцию как суперпозицию других, более простых, функций.

Таблица 12

№	Закон	Конъюнкция(·)	Дизъюнкция(+)
1	Двойного отрицания	$\bar{\bar{x}} \equiv x$	
2	Перемести-тельный	$x \wedge y \equiv y \wedge x$	$x \vee y \equiv y \vee x$
3	Сочетательный	$x \wedge (y \wedge z) \equiv (x \wedge y) \wedge z$	$x \vee (y \vee z) \equiv (x \vee y) \vee z$
4	Распредели-тельный	$x \vee (y \wedge z) \equiv (x \vee y) \wedge (x \vee z)$	$x \wedge (y \vee z) \equiv (x \wedge y) \vee (x \wedge z)$
5	Де Моргана	$\overline{x \wedge y} \equiv \bar{x} \vee \bar{y}$	$\overline{x \vee y} \equiv \bar{x} \wedge \bar{y}$
6	Исключения третьего	$x \wedge \bar{x} \equiv 0$	$x \vee \bar{x} \equiv 1$
7	Операции с константами	$x \wedge 0 \equiv 0$ $x \wedge 1 \equiv x$	$x \vee 0 \equiv x$ $x \vee 1 \equiv 1$
8	Повторения	$x \wedge x \equiv x$	$x \vee x \equiv x$
9	Поглощения	$x \wedge (x \vee y) \equiv x$	$x \vee (x \wedge y) \equiv x$
10	Склеивания	$(x \vee y) \wedge (x \vee \bar{y}) \equiv x$	$(x \wedge y) \vee (x \wedge \bar{y}) \equiv x$

Пример 2.2

Докажем тавтологию с помощью равносильных логических преобразований:

$x \to (y \to x) \equiv |$ применяем формулу 10$| \equiv \bar{x} \vee (\bar{y} \vee x) \equiv$

$|$применяем переместительный и сочетательный законы$| \equiv (\bar{x} \vee x) \vee y \equiv$

$|$применяем закон исключения третьего$| \equiv 1 \vee y \equiv$

$|$применяем операции с константами$| \equiv 1$

Пример 2.3

Составить таблицу истинности функции трех переменных, заданной формулой $f\ (x_1, x_2\ , x_3)\ =\ (\overline{x_1}\ \vee\ x_2\ ) \to (x_1\ \wedge\ x_3)$.

Для построения таблицы истинности f вычислим ее значение на каждом из 8 наборов значений (табл. 13).

Таблица 13

x_1	x_2	x_3	$\overline{x_1}$	$\overline{x_1}\ \vee\ x_2$	$x_1\ \wedge\ x_3$	$(\overline{x_1}\ \vee\ x_2\ ) \to (x_1\ \wedge\ x_3)$
1	1	1	0	1	1	1
1	1	0	0	1	0	0
1	0	1	0	0	1	1
1	0	0	0	0	0	1
0	1	1	1	1	0	0
0	1	0	1	1	0	0
0	0	1	1	1	0	0
0	0	0	1	1	0	0

2.4 Совершенная дизъюнктивная нормальная форма (СДНФ) и совершенная конъюнктивная нормальная форма (СКНФ)

Как уже отмечалось выше, каждая логическая формула может быть представлена булевой формулой, т.е. содержать в своей записи только операции конъюнкцию, дизъюнкцию и отрицание. Сформулируем очень важный для практики способ перехода от табличного задания логической функции к булевой формуле. Он включает следующие действия:

1. для каждого набора значений переменных x_1, x_2 ,..., x_n, на котором функция $f(x_1, x_2, ..., x_n)$ равна 1, выписываются конъюнкции всех переменных;
2. над теми переменными, которые на этом наборе равны 0, ставятся отрицания;
3. все такие конъюнкции соединяются знаками дизъюнкции.

Полученная таким образом формула называется **совершенной дизъюнктивной нормальной формой** (СДНФ) логической функции.

Для каждой функции СДНФ единствена.

Таким образом, СДНФ функции $f(x_1, x_2, ..., x_n)$ представляет собой дизъюнкцию элементарных конъюнкций:

$$D = K_1 \vee K_2 \vee \ldots \vee K_m,$$

где все конъюнкции имеют одинаковое число сомножителей, равное числу логических переменных, а число конъюнкций равно числу наборов значений переменных x_1, x_2 ,..., x_n, на которых функция $f(x_1, x_2, ..., x_n)$ равна 1. Любые другие записи логической функции, вида $D = K_1 \vee K_2 \vee \ldots \vee K_m$,, не отвечающие этим условиям, называются **дизъюктивными нормальными формами** (ДНФ) этой функции.

Введем **понятие совершенной конъюнктивной нормальной формы** (СКНФ):

1. для каждого набора значений переменных x_1, x_2 ,..., x_n, на котором функция $f(x_1, x_2, ..., x_n)$ равна 0, выписываются дизъюнкции всех переменных;
2. над теми переменными, которые на этом наборе равны 1, ставятся отрицания;
3. все такие дизъюнкции соединяются знаками конъюнкции.

Таким образом, СКНФ функции $f(x_1, x_2, ..., x_n)$ представляет собой конъюнкцию элементарных дизъюнкций:

$$К = D_1 \vee D_2 \vee \ldots \vee D_m,$$

где все дизъюнкции имеют одинаковое число сомножителей, равное числу логических переменных, а число дизъюнкций равно числу наборов значений переменных x_1, x_2 ,..., x_n, на которых функция $f(x_1, x_2, ..., x_n)$ равна 0. Любые другие записи логической функции, вида $К = D_1 \vee D_2 \vee \ldots \vee D_m$, не отвечающие этим условиям, называются **конъюнктивными нормальными формами** (КНФ) этой функции.

Пример 2.4

Составить СДНФ и СКНФ для логической функции

$f\ (x_1, x_2\ , x_3)\ =\ (\overline{x_1}\ \vee\ x_2\ ) \rightarrow (x_1\ \wedge\ x_3)$, таблица истинности которой задана в табл. 13.

Решение.

СДНФ имеет вид:

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

СКНФ имеет вид:

$f\ (x_1, x_2\ , x_3) =\ (\overline{x_1} \vee\ \overline{x_2}\ \vee x_3)\ \wedge (x_1 \vee \overline{x_2} \vee\ \overline{x_3})\ \wedge (x_1 \vee \overline{x_2}\ \vee x_3) \wedge$
$\wedge\ (x_1 \vee\ x_2\ \vee \overline{x_3}) \wedge (x_1 \vee\ x_2\ \vee x_3)$

Переход от логической формулы произвольного вида или формулы, записанной в некоторой не булевой алгебре, возможен не только через таблицу истинности, но и на основе применения законов алгебры логики. Для этого необходимо лишь выразить элементы через дизъюнкцию, конъюнкцию и отрицание.

Пример 2.5

Представить в виде булевой формулы следующую логическую формулу: $((x \rightarrow (y\ \wedge\ (z\ \vee k)))\ \wedge\ x) \rightarrow (z\ \vee\ k)$.

Для решения задачи воспользуемся соотношением *(10)*, правильность которого легко проверить через построение таблицы истинности. Последовательно проводя преобразования, будем получать:

$$\Big(\Big(x \rightarrow \big(y\ \wedge\ (z\ \vee k)\big)\Big) \wedge\ x\Big) \rightarrow (z\ \vee\ k) = \overline{\Big(\Big(\bar{x} \vee \big(y\ \wedge\ (z\ \vee k)\big)\Big) \wedge\ x\Big)} \vee (z\ \vee\ k) =$$
$$\Big(\overline{\Big(\bar{x} \vee \big(y\ \wedge\ (z\ \vee k)\big)\Big)} \vee\ \bar{x}\Big) \vee (z\ \vee\ k) = \Big(\bar{\bar{x}} \wedge \overline{\big(y\ \wedge\ (z\ \vee k)\big)}\Big) \vee \bar{x} \vee z \vee k = \Big(x \wedge \big(\bar{y}\ \vee$$
$$\overline{(z\ \vee k)}\big)\Big) \vee \bar{x} \vee z \vee k = \Big(x \wedge \Big(\bar{y}\ \vee\ \big(\bar{z} \wedge \bar{k}\big)\Big)\Big) \vee \bar{x} \vee z \vee k.$$

Замечание. При описании булевых операций отмечалось, что наряду с символами ∧,∨ для обозначения конъюнкции и дизъюнкции используются традиционные символы умножения и сложения: ·,+ , поэтому далее, там, где это ясно из контекста, будут преимущественно использоваться знаки ·,+ . Таким образом, записи

$x \vee (x \wedge z)\ \vee\ (x\ \wedge y\ \wedge\ z)\ \vee\ (y\ \wedge\ z\ )$
и $x + xz\ +\ xyz\ +\ yz$

эквивалентны, однако, последняя значительно короче.

Пример 2.6

Какой логической операции соответствует таблица истинности, приведенная ниже?

X	F
0	1
1	0

$F = \bar{X}$. Таблица истинности соответствует операции отрицания.

Пример 2.7

Найти значение логического выражения $A\ V\ \text{B} \wedge \bar{\text{C}}$ при A=0, B=1, C=0.

Решение: подставим значения переменных в выражение и вычислим его согласно приоритету выполнения операций:

$$A\ V\ \text{B} \wedge \bar{\text{C}} = 0\ V\ 1 \wedge \bar{0} = 0\ V\ (1 \wedge 1) = 0\ V\ 1 = 1.$$

Ответ: заданное логическое выражение принимает значение 1.

Пример 2.8

Найти значение логического выражения $(a < z) \vee (z > 2) \vee (a \neq 5)$ при $a = 5, z = -4$.

Решение: подставим значения переменных в выражение и вычислим его согласно приоритету выполнения операций:

$$(5 < -4) \vee (-4 > 2) \vee (5 \neq 5) = 0 \vee 0 \vee 0 = 0.$$

Ответ: заданное логическое выражение принимает значение 0.

Пример 2.9

Упростить логическое выражение: $(\text{A} \wedge \text{B}) \vee (\text{A} \wedge \bar{B})$. Правильность упрощения проверить с помощью таблиц истинности.

Решение: воспользуемся законом дистрибутивности и вынесем за скобки A: $(A \wedge B) \vee (A \wedge \bar{B}) = \text{A} \wedge (B \vee \bar{B})$. По закону исключенного третьего $\text{B} \vee \bar{B} = 1$, следовательно, $\text{A} \wedge (\text{B} \vee \bar{B}) = \text{A} \wedge 1 = \text{A}$.

Таким образом, в результате упрощения получили

$(\text{A} \wedge \text{B}) \vee (\text{A} \wedge \bar{B}) = \text{A}$.

Составим таблицу истинности (табл. 14) для выражения $(\text{A} \wedge \text{B}) \vee (\text{A} \wedge \bar{B})$.

Таблица 14

A	B	$(\text{A} \wedge \text{B})$	$\bar{B}$	$(\text{A} \wedge \bar{B})$	$(\text{A} \wedge \text{B}) \vee (\text{A} \wedge \bar{B})$
1	1	1	0	0	1
1	0	0	1	1	1
0	1	0	0	0	0
0	0	0	1	0	0

Из таблицы истинности видно, что справедливо

$(\text{A} \wedge \text{B}) \vee (\text{A} \wedge \bar{B}) = \text{A}$.

Ответ: $(\text{A} \wedge \text{B}) \vee (\text{A} \wedge \bar{B}) = \text{A}$.

2.5 Упражнения для самостоятельного решения к главе 2

2.1 Пользуясь законами алгебры логики, найдите номера выражений, равных единице:

1) 0·0 + 0·0 +1+1·0 ;
2) 1·1·1+1·1·1+1·0·1·0 ;
3) 1·1·1+ 0·1+ 0·1·1;
4) 0·0·1+1·1·0 + 0·1·1 ;
5) 0·1·1+1·1·0 + 0·0·1 ;
6) 1·1·0 + 0·1·0 + 0·1·1 .

2.2 Пользуясь законами алгебры логики, найдите номера выражений, равных нулю:

1) 0·1·0 +1·0 + 0·1;
2) 1·0 + 0·1·0 + 0·1;
3) 1·0·1+ 0·1+1·0 ;
4) 0·1·1+ 0·1·1+ 0·1·0 ;
5) 0·1·0 ·1+1·0·1·0 + 0 ·1·1 ;
6) 1·1·0·0 +1·1·1+ 0·0·1·1 .

2.3 Пользуясь законами алгебры логики, найдите значение выражения:
$A+ A\cdot A +1\cdot A + 0\cdot A\cdot A + A\cdot A\cdot 1\cdot A.$

2.4 Пользуясь законами алгебры логики, найдите номера выражений, равных нулю:

1) $A\cdot A\cdot A+1\cdot 0\cdot A+ A\cdot 0\cdot 1$;
2) $A\cdot A\cdot 1+ A\cdot A\cdot A+ A\cdot 1$;
3) $1\cdot 1\cdot A + 0\cdot A\cdot 1 + A\cdot A\cdot 0$;
4) $0 +1\cdot 0 + A\cdot 0 + A\cdot 0 + A\cdot A$;
5) $A\cdot A+ A\cdot A+ A\cdot 1+ A\cdot 1$;
6) $0\cdot A\cdot A\cdot 1+ 0\cdot A\cdot 1\cdot A+1\cdot A\cdot A$.

2.5 Примените теорему поглощения:
$A + AB$;
$K + KP$.

2.6 Упростите выражения:
$(P \wedge Q) \vee (S \wedge P \wedge Q) \vee (P \wedge Q \wedge R \wedge S)$;
$X \wedge Y \wedge Z \ \vee \ X \wedge Z \ \vee \ X \wedge Z \wedge \bar{Y}$;
$ABCD + ABCD + ABC$.

2.7 Упростите:
$(B +C)(B +C)$;
$(BC + D)(BC + D)$;

$(B + C)(B + C)D$;
$V(X + YZ)(X + YZ)$.

2.8 Найдите инверсию (отрицание):
BCD ;
$B + C + D$.

2.9 Упростите:
$A + B \cdot C + D \cdot A \cdot C$;
$P + Q \cdot (P + Q)$;
$A + B + C \cdot (A + B + C) + D$;
$RST \cdot (R + S + T) \cdot RST$;
$P + Q + PQRS$.

2.10. Докажите истинность формул (7)-(12) с помощью таблиц истинности.

2.11 Какой логической операции соответствуют таблицы истинности (табл. 15, 16)?

Таблица 15

X	Y	F
0	0	0
0	1	1
1	0	0
1	1	1

Таблица 16

X	Y	F
0	0	1
0	1	1
1	0	0
1	1	1

2.12 Найти значения приведенных ниже логических выражений:
1. $A \wedge \text{B}\, V\, \overline{\text{C}}$ при A = 0, B = 1, C = 0.
2. $(\text{x} = \text{y}) \vee (z < 4)$ при x = 5, y = 7, z = 0.
3. $A \vee \overline{B}$ при A = 0, B = 0.
4. $(a < z) \vee (z > -10) \wedge (a \neq 5)$ при a = 8, z = – 6.
5. $\overline{A \vee B} \wedge \text{C}$ при A = 0, B = 0, C = 1.

2.13 Упростить логическое выражение $\overline{\overline{A} \vee B} \vee \overline{\overline{\mathrm{A}} \vee \mathrm{B}}$. Правильность упрощения проверить с помощью таблицы истинности.

2.14 Доказать данное равенство с помощью таблицы истинности:
$X \leftrightarrow Y = (\bar{X} \wedge \bar{Y}) \vee (\mathrm{X} \wedge Y)$.

2.15 Докажите тавтологии:

1. $\mathrm{A} \rightarrow (\mathrm{B} \rightarrow \mathrm{A})$
2. $((\mathrm{A} \rightarrow \mathrm{B}) \rightarrow \mathrm{A}) \rightarrow \mathrm{A}$
3. $(\overline{\mathrm{B}} \rightarrow \overline{\mathrm{A}}) \rightarrow (\mathrm{A} \rightarrow \mathrm{B})$
4. $((\mathrm{A} \rightarrow \mathrm{C}) \wedge (\mathrm{B} \rightarrow \mathrm{C})) \rightarrow ((\mathrm{A} \vee \mathrm{B}) \rightarrow \mathrm{C})$

2.16 Найдите СДНФ и СКНФ для следующих логических формул:

1. $\overline{X \vee Z} \wedge (X \rightarrow Y)$
2. $(X \leftrightarrow Z) \rightarrow (X \wedge \bar{Y})$

Глава 3
Логические элементы

Алгебра логики дала в руки конструкторам мощное средство разработки, анализа и совершенствования логических схем. В самом деле, гораздо проще, быстрее и дешевле изучать свойства и доказывать правильность работы схемы с помощью выражающей её формулы, чем создать реальное техническое устройство. Благодаря использованию формализованных методов, достигается многовариантность в решении прикладных задач, появляется возможность оптимального выбора схемотехнических решений по тем или иным критериям.

При всей сложности устройства электронных блоков современных компьютеров выполняемые ими действия осуществляются комбинацией относительно небольшого числа типовых электронных узлов. Перечислим основные из них:

- регистры для хранения данных (от лат. regestum – внесенное, записанное);
- комбинационные преобразователи кодов (шифратор, дешифратор, мультиплексор и др.);
- счетчики (кольцевой, синхронный, асинхронный);
- арифметико-логические узлы (сумматор, узел сравнения) и др.

Из этих узлов строятся интегральные микросхемы высокого уровня интеграции: микропроцессоры, модули ОЗУ, контроллеры внешних устройств и т. д. Эти функциональные узлы состоят из простейших логических элементов, которые, в свою очередь состоят из полупроводниковых транзисторов, диодов и резисторов. При конструировании простых триггерных и других электронных импульсных схем, приходят на помощь – **логические элементы**.

3.1. Связь логических функций и логических элементов

Логический элемент – часть электронной логической схемы, которая реализует элементарную логическую операцию или функцию. При этом цифровая микросхема может содержать в себе от одного, до нескольких единиц, десятков, …и до нескольких сотен тысяч логических элементов в зависимости от степени интеграции. Рассматрим простейшие логические элементы.

Условные обозначения основных логических элементов приведены в табл.17. Мы будем придерживаться стандарта ГОСТ 2.743-91. Логический элемент на схемах изображают прямоугольником (рис. 3.1). Линии, которые находятся с левой стороны этого прямоугольника, называются входами, а с правой - выходами элемента. Поскольку простейший логический элемент это электронное устройство, то это означает, что у него есть входы (входные выводы) и выходы (выходные выводы). И входов и выходов может быть один, а может быть и больше. Простейшие логические элементы всегда имеют лишь один выход.

Внутри самого прямоугольника изображен указатель логической функции, которую выполняет данный элемент.

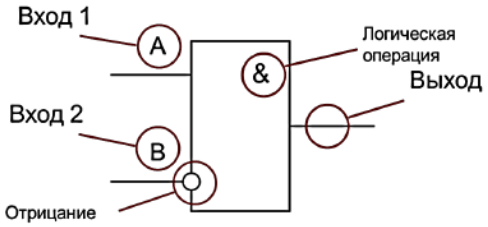
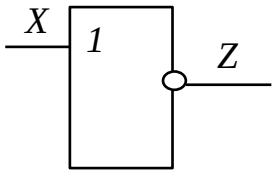
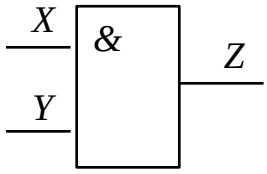
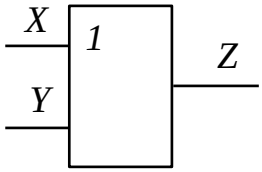
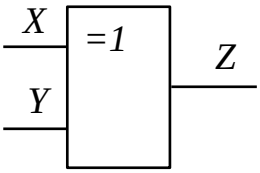
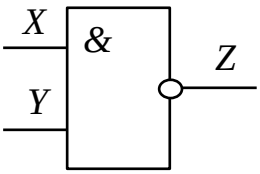
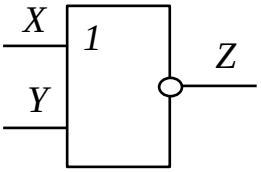


Рис. 3.1. Расшифровка графической схемы логического элемента

При описании работы логических элементов *выходным сигналам* ставят в однозначное соответствие **функции**, а *входным сигналам* - **аргументы** этих функций.

Таблица 17

Логическая функция	Условное обозначение логического элемента
Инверсия (НЕ) $Z = \bar{X}$	
Конъюнкция (И) $Z = X \wedge Y$	
Дизъюнкция (ИЛИ) $Z = X \vee Y$	
Исключающее ИЛИ $Z = X \oplus Y$	
Инверсия конъюнкции (И – НЕ) $Z = \overline{X \wedge Y}$	
Инверсия дизъюнкции (ИЛИ – НЕ) $Z = \overline{X \vee Y}$	

Остановимся подробнее на каждой логической функции.

Функция отрицания (инверсии сигнала) **«НЕ» (NOT)**, часто ее называют - **«инвертор»**. Графически, инверсия обозначается пустым кружочком вокруг вывода элемента (микросхемы). Обычно кружок инверсии ставится у выхода, но в более сложных логических элементах, он может стоять и на входе. Графическое обозначение элемента «НЕ» приведено в таблице 17, а таблицей истинности является таблица 18.

Таблица 18

X	Z
1	0
0	1

У элемента «НЕ» всегда один вход и один выход. По таблице истинности следует, что при наличии на входе элемента логического нуля, на выходе будет логическая единица. И наоборот, при наличии на входе логической единицы, на выходе будет логический ноль. Цифра «1» внутри прямоугольника обозначает функцию «ИЛИ», её принято рисовать и внутри прямоугольника элемента «НЕ», но это ровным счётом ничего абсолютно не значит.

Самой распространённой микросхемой «транзисторно-транзисторной логики» (ТТЛ), выполняющей функцию «НЕ», является интегральная микросхема (ИМС) К155ЛН1, внутри которой имеется шесть элементов «НЕ».

Функция сложения «И» (AND) – если на всех входах единица, то на выходе будет единица, в противном случае, если хотя бы на одном входе ноль, то и на выходе всегда будет ноль. В алгебре-логике элемент «И» называют «конъюнктор». Графическое обозначение элемента «2И» приведено в таблице 17, его таблица истинности – таблица 19.

Таблица 19

X	Y	Z
0	0	0
1	0	0
0	1	0
1	1	1

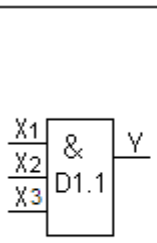
Название элемента **«2И»** обозначает, что у него два входа, и он выполняет функцию **«И»**. На схеме внутри прямоугольника микросхемы рисуется значок **«&»** (амперсант), что на английском языке означает «AND» (в переводе на русский - И).

По таблице истинности следует, что на выходе элемента «И» будет логическая единица только в одном случае - когда на обоих входах будет логическая единица. Если хотя бы на одном входе ноль, то и на выходе будет ноль.

Самой распространённой микросхемой ТТЛ, выполняющей функцию «2И», является ИМС К155ЛИ1, внутри которой имеется четыре элемента «2И». Нумерация выводов этой микросхемы показана на рисунке 3.2.

Рис.3.2. Интегральная микросхема К155ЛИ1, содержащая четыре элемента «2И»

Для того, чтобы было понятнее что такое «2И», «3И», «4И», и т.д., рассмотрим графическое обозначение и таблицу истинности элемента «3И» (рис.3.3).



X1	X2	X3	Y
0	0	0	0
1	0	0	0
0	1	0	0
1	1	0	0
0	0	1	0
1	0	1	0
0	1	1	0
1	1	1	1

Рис.3.3. Графическое обозначение и таблицу истинности элемента «3И»

По таблице истинности следует, что на выходе элемента «3И» будет логическая единица только в том случае - когда на всех трёх входах будет логическая единица. Если хотя бы на одном входе будет логический ноль, то и на выходе элемента также будет логический ноль. Самой распространённой микросхемой ТТЛ, выполняющей функцию «3И», является микросхема К555ЛИ3, внутри которой имеется три элемента «3И».

Если рассматривать элемент 8И, то составлять для него таблицу истинности будет проблематично, т.к. она будет содержать $2^8 = 256$ различных состояний, а на самом деле описать его работу можно думя строчками:

на выходе будет логическая единица лишь в том еслучае, когда на всех входах будут единицы.

Функция сложения с отрицанием «И-НЕ» (NAND) – если на всех входах единица, то на выходе будет ноль, в противном случае на выходе всегда будет единица. Графическое обозначение элемента «2И-НЕ» приведено в таблице 17, его таблица истинности – таблица 20.

По таблице истинности следует, что на выходе элемента «2И-НЕ» будет логический ноль только в том случае, если на обоих входах будет логическая единица. Если хотя бы на одном входе ноль, то на выходе будет единица.

Таблица 20

X	Y	Z
0	0	1
1	0	1
0	1	1
1	1	0

Самой распространённой микросхемой ТТЛ, выполняющей функцию «2И-НЕ», является ИМС К155ЛА3, а микросхемами **КМОП (комплементарный металлооксидный полупроводник)** – ИМС К561ЛА7 и К176ЛА7, внутри которых имеется четыре элемента «2И-НЕ» (рис.3.4).

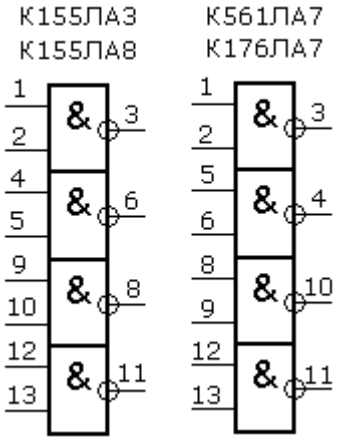


Рис. 3.4 ИМС К155ЛА3, К561ЛА7, К176ЛА7, содержащие четыре элемента «2И-НЕ»

Сравнив таблицы истинности элемента «2И-НЕ» и элемента «2И» можно догадаться об эквивалентности схем:

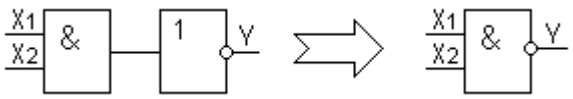


Рис. 3.5. Эквивалентность схем

Добавив к элементу «2И» элемент «НЕ» мы получили элемент «2И-НЕ». Так можно собрать схему, если нам необходим элемент «2И-НЕ», а у нас в распоряжении имеются только элементы «2И» и «НЕ» (рис. 3.5).

И наоборот:

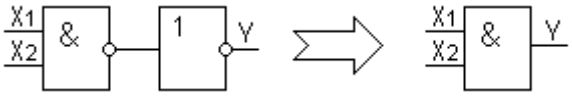


Рис. 3.6. Эквивалентность схем

Добавив к элементу «2И-НЕ» элемент «НЕ» мы получили элемент «2И». Так можно собрать схему, если нам необходим элемент «2И», а у нас в распоряжении имеются только элементы «2И-НЕ» и «НЕ» (рис. 3.6).

Аналогичным образом, путём соединения входов элемента «2И-НЕ» мы можем получить элемент «НЕ» (рис. 3.7):

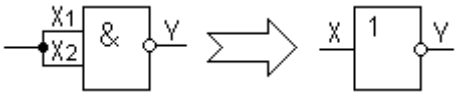


Рис. 3.7. Эквивалентность схем

Функция выбора «ИЛИ» (OR) – если хотя бы на одном из входов – единица, то на выходе – единица, в противном случае на выходе всегда будет ноль. В алгебре логики, элемент «ИЛИ» называют «дизъюнктор». Графическое обозначение элемента «2ИЛИ» приведено в таблице 17, его таблица истинности – таблица 21.

Таблица 21

X	Y	Z
0	0	0
1	0	1
0	1	1
1	1	1

Самой распространённой микросхемой ТТЛ, выполняющей функцию «2ИЛИ», является ИМС К155ЛЛ1, внутри которой имеется четыре элемента «2ИЛИ». Нумерация выводов этой микросхемы показана на рисунке 3.8.

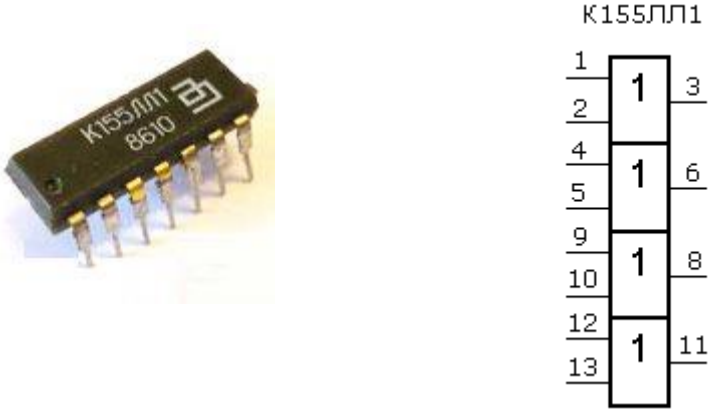


Рис.3.8. ИМС К155ЛЛ1, выполняющая функцию «2ИЛИ»

Предположим, что нам в схеме необходим элемент, выполняющий функцию «2ИЛИ», но у нас есть в распоряжении только элементы «НЕ» и «2И-НЕ», тогда можно собрать схему, которая будет выполнять функцию «2ИЛИ» (рис. 3.9)

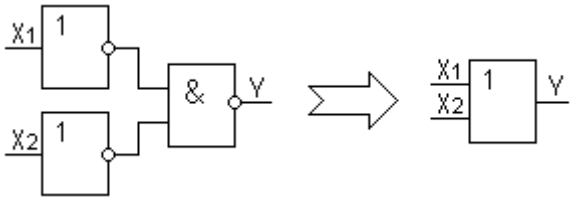


Рис.3.9. Эквивалентность схем

Функция выбора «ИЛИ-НЕ» (NOR) – если хотя бы на одном из входов – единица, то на выходе – ноль, в противном случае на выходе всегда будет единица. Элемент «ИЛИ-НЕ» выполняет функцию «ИЛИ», а потом инвертирует его функцией «НЕ». Графическое обозначение элемента «2ИЛИ-НЕ» приведено в таблице 17, его таблица истинности – таблица 22.

Таблица 22

X	Y	Z
0	0	1
1	0	0
0	1	0
1	1	0

Самой распространённой микросхемой ТТЛ, выполняющей функцию «2ИЛИ-НЕ», является ИМС К155ЛЕ1, а микросхемами КМОП – К561ЛЕ5 и К176ЛЕ5, внутри которых имеется четыре элемента «2ИЛИ-НЕ». Нумерация выводов этих микросхем показана на рисунке 3.10.

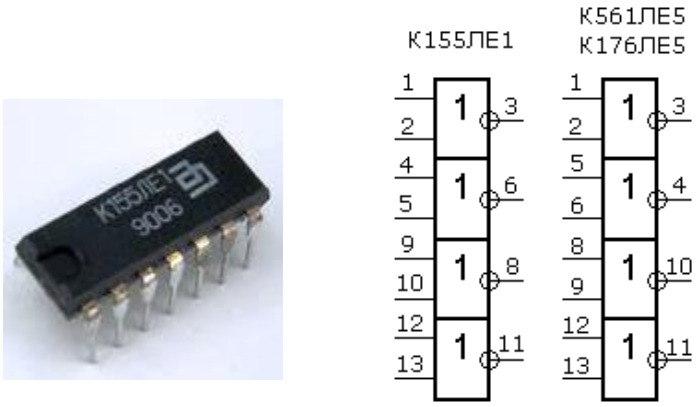


Рис.3.10. ИМС К155ЛЕ1, К561ЛЕ5, К176ЛЕ5, содержащие четыре элемента «2ИЛИ-НЕ»

Предположим, что нам в схеме необходим элемент, выполняющий функцию «2ИЛИ-НЕ», но у нас есть в распоряжении только элементы «НЕ» и «2И-НЕ», тогда можно собрать следующую схему, которая будет выполнять функцию «2ИЛИ-НЕ» (рис. 3.11):

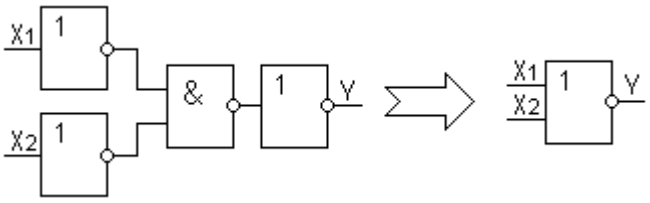


Рис.3.11. Эквивалентность схем

По аналогии с элементом «2И-НЕ», путём соединения входов элемента «2ИЛИ-НЕ» мы можем получить элемент «НЕ» (рис. 3.12):

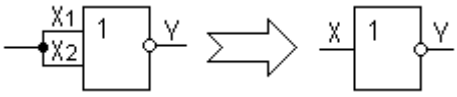


Рис.3.12. Эквивалентность схем

Функция неравенства двух входов «Исключающее ИЛИ» (XOR) - если на обоих входах элемента одинаковые сигналы, то на выходе – ноль, в противном случае на выходе всегда будет единица. Операция, которую он выполняет, часто называют «сложение по модулю 2».

Графическое обозначение элемента «Исключающее ИЛИ» приведено в таблице 17, его таблица истинности – таблица 23.

Таблица 23

X	Y	Z
0	0	0
1	0	1
0	1	1
1	1	0

Самой распространённой микросхемой ТТЛ, выполняющей функцию «Исключающее ИЛИ», является ИМС К155ЛП5, а микросхемами КМОП – К561ЛП2 и К176ЛП2, внутри которых имеется четыре элемента «Исключающее ИЛИ». Нумерация выводов этих микросхем показана на рисунке 3.13.

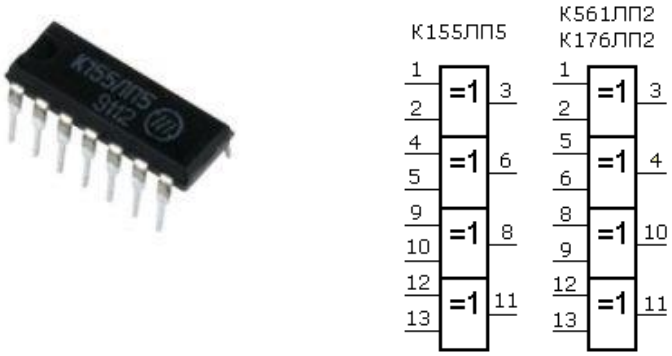


Рис.3.13. ИМС К155ЛП5, К561ЛП2, К176ЛП2, содержащие четыре элемента «Исключающее ИЛИ»

Предположим, что нам в схеме необходим элемент, выполняющий функцию «Исключающее ИЛИ», но у нас есть в распоряжении только элементы «2И-НЕ», тогда можно собрать следующую схему, которая будет выполнять функцию **«Исключающее ИЛИ»** (рис. 3.14):

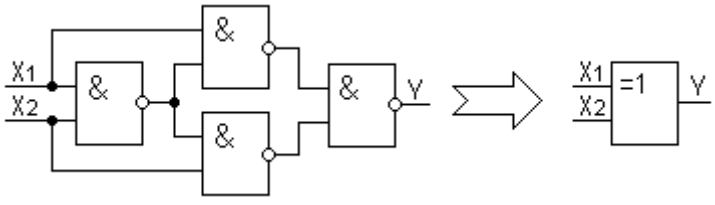


Рис. 3.14. Эквивалентность схем

Пример 3.1

Имея в распоряжении логические элементы 2И-НЕ, сконструировать устройство, реализующее операцию 3ИЛИ-НЕ.

Решение. Сначала сконструируем устройство, выполняющее операцию 3И-НЕ (рис. 3.15,а), а потом на входах и выходе элемента 3И-НЕ установим логические элементы НЕ (рис. 3.15,б).

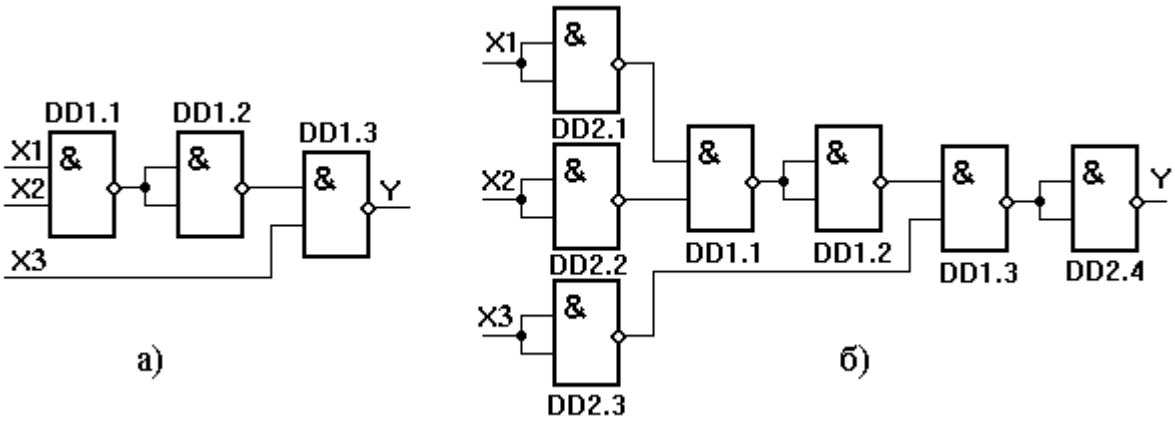
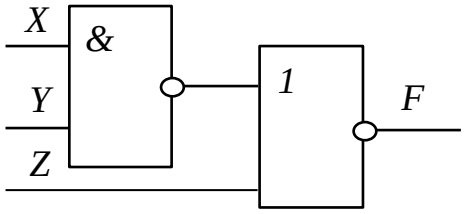


Рис.3.15. Реализация операции 3ИЛИ-НЕ на логических элементах 2И-НЕ

Пример 3.2

По заданной логической схеме составить логическое выражение и заполнить для него таблицу истинности:



Решение: используя обозначения логических элементов, составим логическое выражение

$$F = \overline{\overline{X \wedge Y} \vee Z}.$$

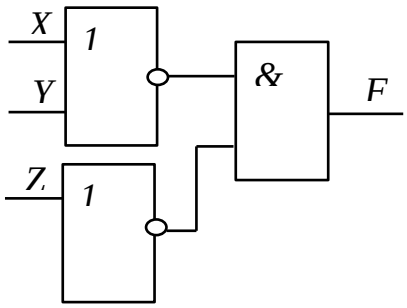
Заполним для F таблицу истинности (табл. 24).

Таблица 24

X	Y	Z	$X \wedge Y$	$\overline{X \wedge Y}$	$\overline{X \wedge Y} \vee Z$	$\overline{\overline{X \wedge Y} \vee Z}$
1	1	1	1	0	1	0
1	1	0	1	0	0	1
1	0	1	0	1	1	0
1	0	0	0	1	1	0
0	1	1	0	1	1	0
0	1	0	0	1	1	0
0	0	1	0	1	1	0
0	0	0	0	1	1	0

Пример 3.3

Задано логическое выражение $F = \overline{X \vee Y} \wedge \bar{Z}$. Составить логическую схему для данного выражения и заполнить таблицу истинности.



Решение: используя обозначения логических элементов, составим логическую схему:

Заполним для F таблицу истинности (табл. 25).

Таблица 25

X	Y	Z	$X \vee Y$	$\overline{X \vee Y}$	$\bar{Z}$	$\overline{X \vee Y} \wedge \bar{Z}$
1	1	1	1	0	0	0
1	1	0	1	0	1	0
1	0	1	1	0	0	0
1	0	0	1	0	1	0
0	1	1	1	0	0	0
0	1	0	1	0	1	0
0	0	1	0	1	0	0
0	0	0	0	1	1	1

Пример 3.4

Составить логическую схему на основании приведённой ниже таблице истинности (табл. 26).

Таблица 26

№	X_1	X_2	X_3	Y
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1

Решение: значение выходной функции равно единице для строк №3, 5, 6, 7. Конъюнкция для строки №3 равна $\overline{X}_1 X_2 X_3$, так как X_1 для строки №3 равна «0», а для X_2 и X_3 равна «1». Аналогично составляются логические произведения для строк №5, 6, 7. Затем составляется СДНФ:

$$Y = \overline{X}_1 X_2 X_3 + X_1 \overline{X}_2 X_3 + X_1 X_2 \overline{X}_3 + X_1 X_2 X_3 .$$

Получившееся логическое выражение можно реализовать и непосредственно с помощью логических элементов, но это будет нерационально. Поэтому упростим полученное выражение:

$$Y = \overline{X}_1 X_2 X_3 + X_1 X_2 X_3 + X_1 \overline{X}_2 X_3 + X_1 X_2 X_3 + X_1 X_2 \overline{X}_3 + X_1 X_2 X_3 =$$
$$= \left(\overline{X}_1 + X_1\right) X_2 X_3 + \left(\overline{X}_2 + X_2\right) X_1 X_3 + \left(\overline{X}_3 + X_3\right) X_1 X_2 = X_2 X_3 + X_1 X_3 + X_1 X_2$$

Здесь добавлено два слагаемых равных четвёртому, а затем, после вынесения за скобки, учтено правило $a + \overline{a} = 1$.

Итак, заданную логическую функцию можно реализовать с помощью трёх умножений (логических элементов, реализующих функцию «И») и одного логического сложения (логического элемента, реализующего функцию «ИЛИ») (см.рис. 3.16).

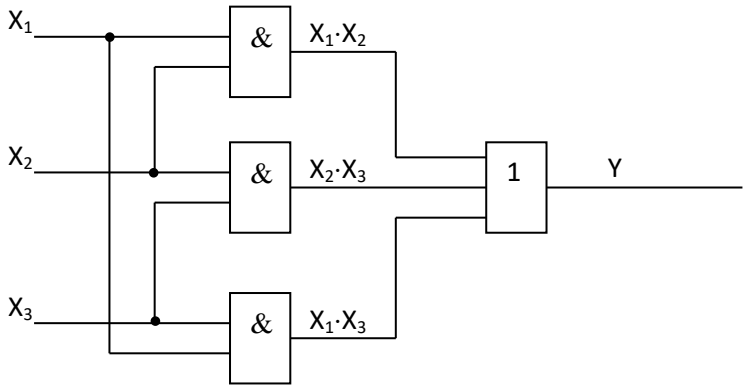


Рис.3.16. Реализация задания с помощью трех элементов 2И и одного элемента ЗИЛИ

Эта же схема может быть реализована и другим способом. Преобразуем логическую функцию:

$$Y = X_1 X_2 + X_2 X_3 + X_1 X_3 = \overline{\overline{X_1 X_2} \cdot \overline{X_2 X_3} \cdot \overline{X_1 X_3}}.$$

Теперь для реализации данной логической функции надо использовать элементы «И-НЕ» (см.рис. 3.17).

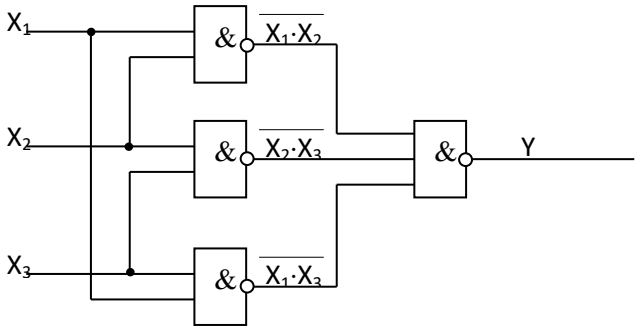


Рис.3.17. Реализация задания с помощью трех элементов 2И-НЕ и одного элемента ЗИЛИ-НЕ

3.2 Сумматоры

В цифровой схемотехнике процессоров главная функция - «Суммирование двоичных чисел», поэтому сложный логический элемент –«Сумматор», является неотъемлемой частью арифметико-логического устройства любого, без исключения процессора.

Узлы сумматора и полусумматора лежат в основе арифметического устройства ЭВМ и иллюстрируют некоторые принципы выполнения вычислительных операций в компьютере.

В основе каждой из элементарных операций лежит некоторая последовательность логических действий. Рассмотрим, например, операцию сложения двух чисел: $3_{10} + 6_{10}$. Имеем:

$$\begin{array}{r} 011_2 \\ +\ \underline{110_2} \\ 1011_2 \end{array}$$

На каждом простейшем шаге операции сложения двум двоичным цифрам сопоставляется одно- или двузначное двоичное число по правилам: $(0,0) \rightarrow 0$; $(0,1) \rightarrow 1$; $(1,0) \rightarrow 1$; $(1,1) \rightarrow 10$. Таким образом, сложение цифр можно описать логической функцией. Если дополнить это логическим правилом переноса единицы в старший разряд для $(1,1) \rightarrow 10$, то сложение полностью сведется к цепочке логических операций.

Составной частью сумматора является набор логических элементов, выполняющих функцию **«Исключающее ИЛИ с переносом остатка»**. В результате сложения двух двоичных чисел, две единицы одного разряда дают ноль, при этом формируется «единица переноса» в следующий старший разряд, который участвует в операции суммирования в старшем разряде. Для этого в логическую схему добавляется ещё один вывод «переноса» - *Co* (от английских слов Carry out – выходной перенос).

Полусумматор реализует сложение двух одноразрядных двоичных чисел. Рассмотрим таблицу истинности сложения одноразрядных двоичных чисел *A* и *B*. Младшую цифру результата сложения обозначим *S*, а старшую, которая при сложении многоразрядных чисел будет перенесена в старший разряд, *Co* (от английских слов Carry out – выходной перенос). Таблица истинности для полусумматора имеет вид (таблица 27):

Таблица 27

A	*B*	*S*	*Co*
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Рассмотрим три первые столбца *A, B, S* и сравним их с рассмотренными таблицами истинности логических элементов. Видно, что данные столбцы соответствуют логическому элементу исключающее ИЛИ:

$$S = A \oplus B = (\bar{A} \wedge B) \vee (A \wedge \bar{B}).$$

Аналогично рассмотрим столбцы *A, B, C_0*. Полученная таблица соответствует базовому логическому элементу И:

$$C_0 = A \wedge B.$$

Такая функция сложения одноразрядных чисел в простых устройствах обычно не используется, и как правило, интегрирована в состав одной микросхемы – сумматора, с минимальным количеством разрядов – четыре, для сложения четырехбитных чисел. По причине слабого спроса, промышленность таких логических элементов не выпускает. Поэтому, в случае необходимости, функцию «Исключающее ИЛИ с переносом» можно собрать по следующей схеме из элементов «2И-НЕ» и «2ИЛИ-НЕ», которая активно применяется как внутри простых сумматоров, так и во всех сложных процессорах (в том числе Pentium, Intel-Core, AMD и других, которые появятся в будущем):Таким образом, полусумматор реализуется с помощью базовых логических элементов (рис. 3.18 и рис. 3.19).

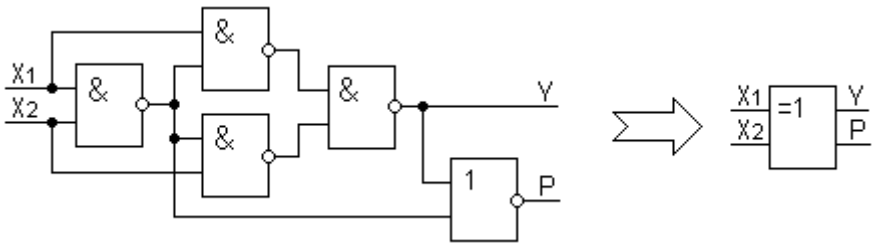


Рис. 3.18. Логическая схема полусумматора. Эквивалентность схем

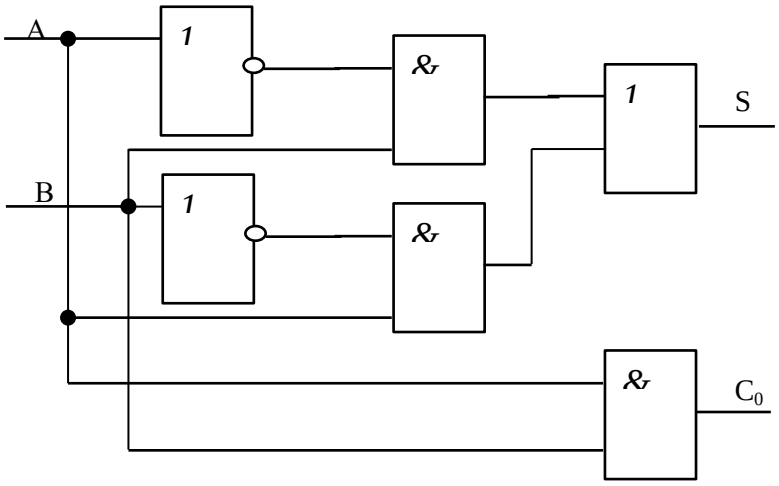


Рис. 3.19. Логическая схема полусумматора

Полусумматор является звеном сумматора, который при сложении двух чисел учитывает возможное наличие единицы, переносимой из старшего разряда. Полный одноразрядный сумматор удобно представить в виде двух полусумматоров (рис. 3.19): первый суммирует разряды A и B, а второй к полученному результату прибавляет бит переноса Ci (от английского Carry in – входной перенос).

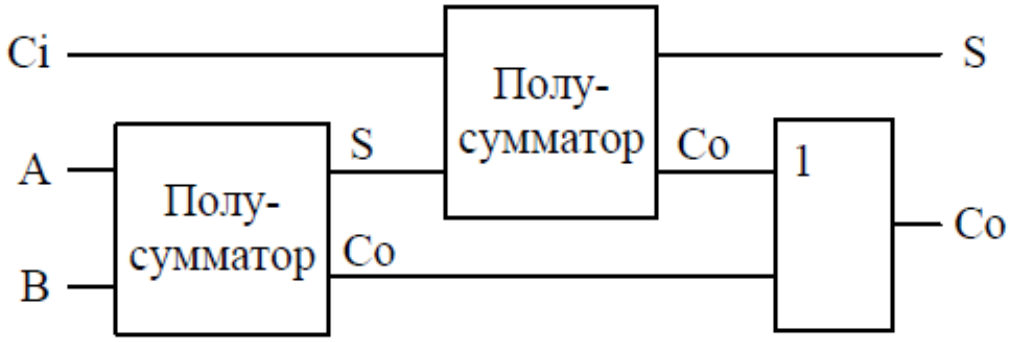


Рис. 3.19. Одноразрядный сумматор

Приведем таблицу истинности для сумматора (таблица 28):

Таблица 28

Входы			Выходы	
A	B	Ci	S	Co
0	0	0	0	0
0	0	1	1	0
0	1	1	1	0
0	1	0	0	1
1	0	1	1	0
1	0	0	0	1
1	1	0	0	1
1	1	1	1	1

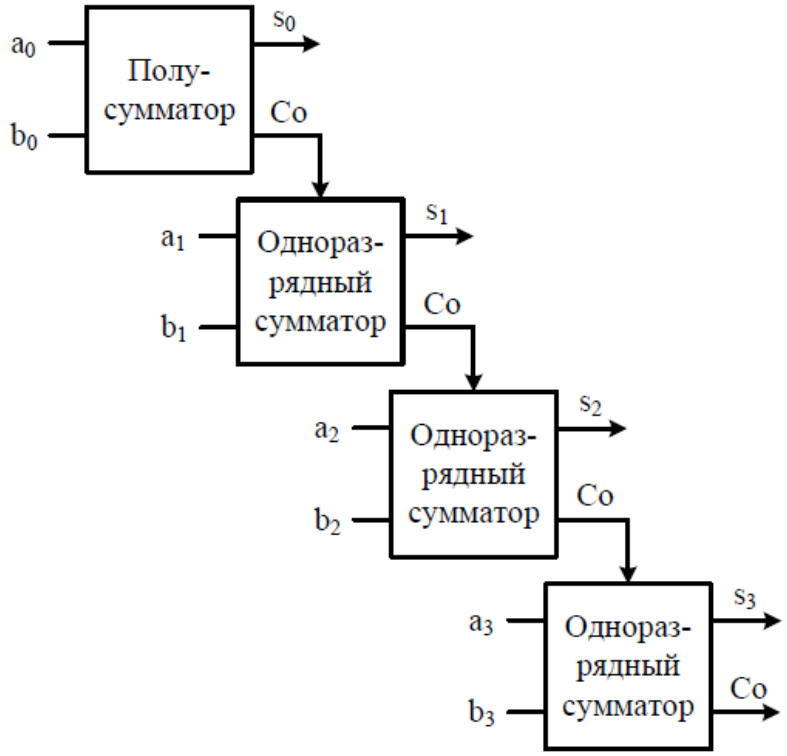


Рис. 3.20. Схема суммирования двух четырехразрядных чисел

Перейти к сложению многоразрядных чисел можно путем последовательного соединения соответствующего количества сумматоров. Схема суммирования двух четырехразрядных двоичных чисел $A = a_3a_2a_1a_0$ и $B = b_3b_2b_1b_0$ представлена на рис. 3.20. Как видно из рисунка, для суммирования младших разрядов a_0 и b_0 достаточно полусумматора, так как отсутствует сигнал входного переноса.

3.3 Триггеры

Вышеперечисленные логические элементы выполняют статические функции, а на основе них строятся более сложные статические и динамические элементы (устройства): триггеры, регистры, счётчики, шифраторы, дешифраторы, сумматоры, мультиплексоры.

На рис. 3.21 приведена логическая схема триггера. Триггер (от англ. – спусковой крючок) устройство переключаемое из одного устойчивого состояния в другое под воздействием внешнего управляемого фактора. Это устройство может хранить 1 бит информации. Триггеры используются как разряды оперативной памяти и памяти процессора.

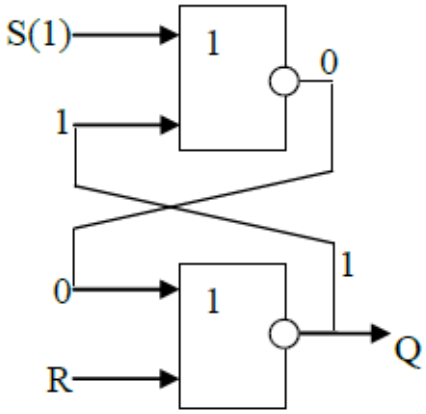


Рис. 3.21. Логическая схема триггера

Триггеры могут иметь два выхода: прямой и инверсный. Если на оба входа (R и S) подать лог. нули, то состояние выходов определить невозможно, это называется запрещенным состоянием, поскольку триггер установится как ему заблагорассудится, т. е. в произвольное состояние. Допустим, на выходе Q присутствует 1, тогда на выходе $\bar{Q}$ обязательно будет 0. И наоборот, чтобы установить триггер в нулевое состояние достаточно на вход R подать напряжение высокого уровня (1). Если высокий уровень подать на вход S, то это переведет его в состояние 1, или как говорят, в единичное состояние (на выходе 0). И в том, и в другом случаях напряжение соответствующего уровня может быть очень коротким импульсом - на грани физического быстродействия микросхемы. То есть, триггер обладает двумя устойчивыми состояниями, причем эти состояния зависят от ранее воздействующих сигналов, что позволяет сделать следующий вывод - **триггер является простейшим элементом памяти**. Буквы R и S сокращения от английсских слов set - установка, reset - сброс (предустановка).

Регистр - последовательное логическое устройство, используемое для хранения n-разрядных двоичных чисел и выполнения преобразований над ними. Регистр представляет собой упорядоченную последовательность триггеров, число которых соответствует числу разрядов в числе.

Фактически любое цифровое устройство можно представить в виде совокупности регистров, соединённых друг с другом при помощи комбинационных цифровых устройств.

Основой построения регистров являются D-триггеры, RS-триггеры.

Счетчик – увеличивает или уменьшает свое значение при поступлении на специальный вход импульсов.

Счётчиком называется цифровой автомат, служащий для формирования многоразрядных двоичных слов (кодов) соответствующих количеству входных импульсов. По мере поступления входных импульсов счётчик последовательно перебирает свои состояния в определённом для данной схемы порядке. Длина списка используемых состояний называется модулем пересчёта M, основанием пересчёта или ёмкостью счётчика. Одно из возможных состояний счётчика принимается как начальное или нулевое. Если счётчик начал считать с началь-

ного состояния и через каждые М входных сигналов в нём снова устанавливается начальное состояние, а на выходе счётчика формируется сигнал М–ичного переноса Р, то такой счётчик называется счётчиком–делителем частоты. Частота следования М-ичного переноса Р в таком счётчике меньше частоты входных импульсов ровно в М раз. Различные схемы счётчиков могут перебирать свои состояния в самом различном порядке. Порядок перебора кодов состояний счётчика определяется системой кодирования состояний счётчика как цифрового автомата. Чаще всего применяются двоичные счётчики, у которых порядок смены состояний триггеров соответствует последовательности двоичных чисел. Счетчик может перебирать свои состояния в возрастающем порядке чисел и тогда он называется суммирующим. Если счётчик перебирает свои состояния в порядке убывания двоичных чисел, то он называется вычитающим. Если под действием сигналов управления порядок перебора может изменяться на противоположный, то счётчик называется реверсивным. Счётчики строятся с использованием синхронизированных триггеров. Если несколько триггеров блока памяти имеют общий синхросигнал, то они образуют синхронный счётчик. Несколько синхронных счётчиков соединённых по схеме подачи синхросигналов последовательно образуют асинхронный счётчик. В этом случае сигнал выходного переноса предыдущего счётчика используется как сигнал синхронизации последующего счётчика.

3.4 Шифраторы, дешифраторы, мультиплексоры, демультиплексоры

Шифратор (кодер) преобразует сигнал на одном из входов в *n*-разрядное двоичное число. Функциональная схема шифратора, преобразующего десятичные цифры в 4-разрядное двоичное число, приведена на рисунке 3.22,а, а его условное обозначение – на рисунке 3.22,б. При появлении сигнала логической единицы на одном из десяти входов на четырех выходах шифратора будет присутствовать соответствующее двоичное число. Пусть сигнал логической единицы подан на вход 7. Тогда на выходах логических элементов DD1.1, DD1.2, DD1.3 будут сигналы логических единиц, а на выходе элемента DD1.4 – сигнал логического нуля. Таким образом, на выходах 8, 4, 2, 1 шифратора мы получим двоичное число 0111.

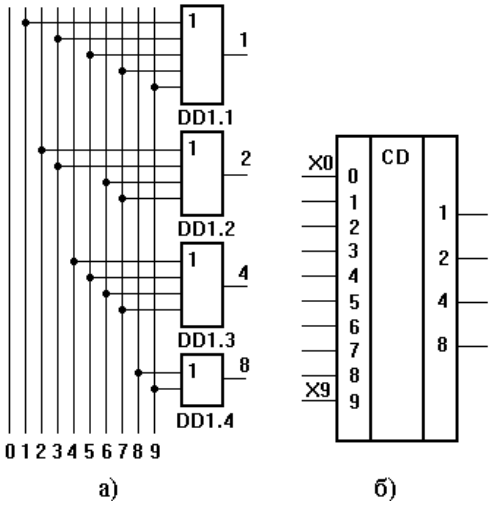


Рис. 3.22. Функциональная схема шифратора и его условное обозначение

Некоторые из шифраторов снабжаются входом стробирования. Наличие входа стробирования позволяет выделять сигнал в определенный момент времени.

Дешифратор (декодер) преобразует код, поступающий на его входы, в сигнал только на одном из его выходов. Дешифратор n-разрядного двоичного числа имеет 2^n выходов. Функциональная схема дешифратора на 16 выходов приведена на рисунке 3.23,а. По такой функциональной схеме построена микросхема К155ИД3. Условное обозначение этой микросхемы на принципиальных схемах приведено на рисунке 3.23,б. Для преобразования сигнала необходимо на входы V1 и V2 микросхемы подать сигналы логических нулей.

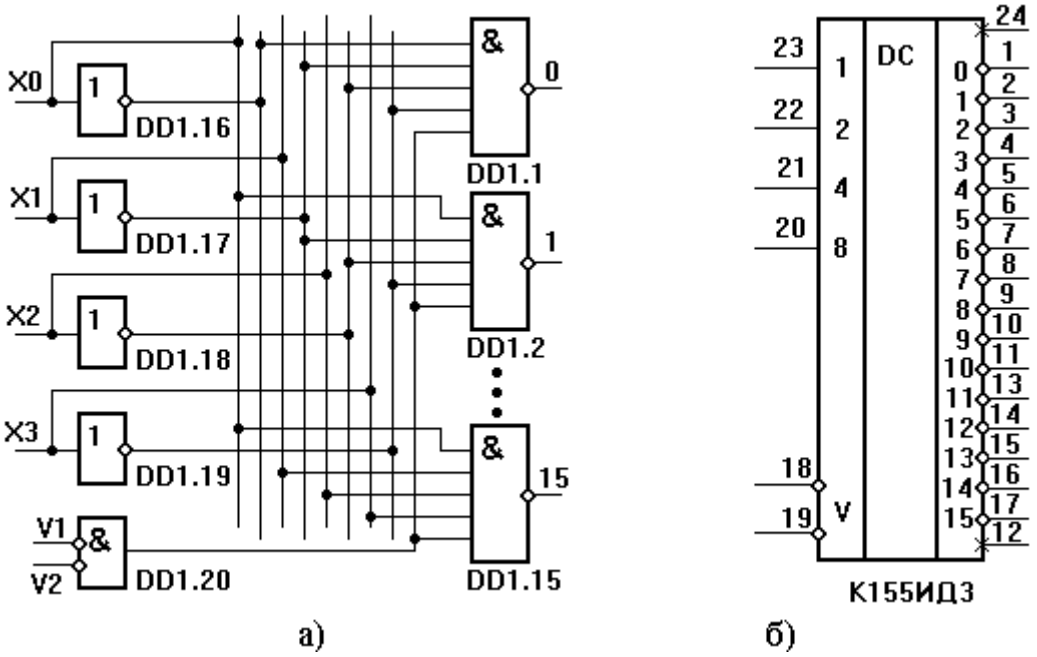


Рис. 3.23. Функциональная схема дешифратора и его условное обозначение

Пусть на входе дешифратора присутствует двоичное число 1111. В этом случае на всех пяти входах элемента DD1.15 будут сигналы логических единиц, а на выходе этого элемента будет логический нуль. На выходах всех остальных 15 элементов будут сигналы логических единиц. Если хотя бы на одном из входов V логическая единица, то единицы будут на всех 16 выходах.

В цифровой технике широко применяются мультиплексоры и демультиплексоры. Мультиплексор это устройство, обеспечивающее соединение одного из информационных входов с выходом. Номер информационного входа, который соединяется с выходом, задается в двоичном коде на адресных входах. Если мультиплексор имеет n адресных входов, то в нем может быть 2^n информационных входов.

Демультиплексор это устройство, обеспечивающее соединение одного из информационных выходов с одним входом. Номер информационного выхода, который соединяется с входом, задается в двоичном коде на адресных входах. Если демультиплексор имеет n адресных входов, то в нем может быть 2^n информационных выходов.

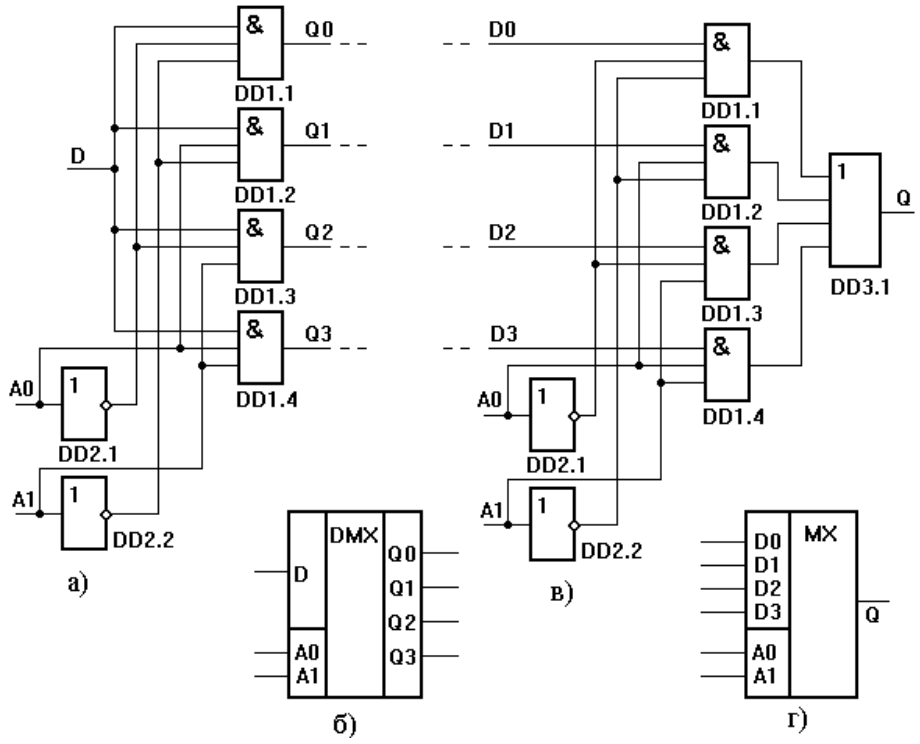


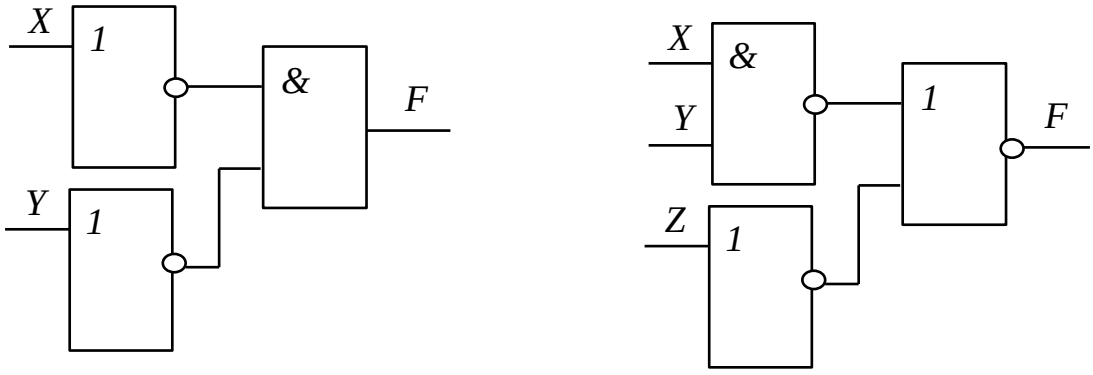
Рис. 3.24. Функциональные схемы демультипликсора и мультиплексора и их условные обозначения

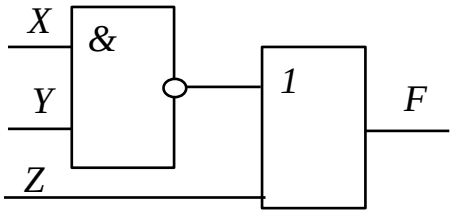
Функциональная схема демультиплексора, имеющего четыре выхода, приведена на рисунке 3.24,а, а его условное обозначение на принципиальных схемах – на рисунке 3.24,б.

Функциональная схема мультиплексора, имеющего четыре входа, приведена на рисунке 3.24,в, а его условное обозначение на принципиальных схемах – на рисунке 3.24,г. Мультиплексоры могут снабжаться дополнительным входом – входом разрешения передачи информации с входов на выход.

3.4 Упражнения для самостоятельного решения к главе 3

3.1 По заданным логическим схемам составить логические выражения и заполнить для них таблицы истинности.





3.2 По заданным логическим выражениям составить логические схемы и заполнить таблицы истинности.

$$\overline{\overline{X \wedge \overline{Y}} \vee Z} = F;$$
$$X \vee \overline{\overline{Y} \wedge Z} = F;$$
$$\overline{\overline{X} \vee Y \wedge \overline{Z}} = F.$$

3.3 Составить логическую схему на основании приведённой ниже таблице истинности:

X_1	X_2	X_3	Y
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	0
1	1	1	1

3.4 Составить логическую схему на основании приведённой ниже таблице истинности:

X_1	X_2	X_3	X_4	Y
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	0	1
1	0	0	0	0
1	0	1	0	0
1	1	0	0	0
1	1	1	0	1
0	0	0	1	1
0	0	1	1	0
0	1	0	1	0
0	1	1	1	0
1	0	0	1	1
1	0	1	1	1
1	1	0	1	0
1	1	1	1	1

Индивидуальные задания на тему «Алгебра логики»

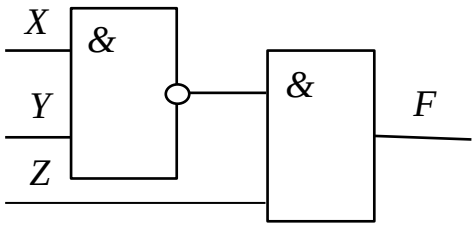
Вариант 1

1. Найдите значение логического выражения:
   $AB + C$ при *A=False, B=True, C= False.*
2. Упростите логическое выражение. Правильность упрощения проверьте с помощью таблицы истинности: $\overline{A \vee B} \vee \overline{\bar{A} \vee B}$
3. По заданному логическому выражению составьте логическую схему и заполните таблицу истинности: $\overline{\bar{X}} \vee Y \vee Z = F$
4. Составить логическую схему на основании приведённой ниже таблице истинности:

X_1	X_2	X_3	Y
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Вариант 2

1. Найдите значение логического выражения:
   $\overline{(A+B)}C$ при *A=False, B= False, C=True.*
2. Упростите логическое выражение. Правильность упрощения проверьте с помощью таблицы истинности: $X \to Y = (\bar{X} \wedge \bar{Y}) \wedge (X \wedge Y)$
3. По заданной логической схеме составьте логическое выражение и заполните для него таблицу истинности:



4. Составить логическую схему на основании приведённой ниже таблице истинности:

X_1	X_2	X_3	Y
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

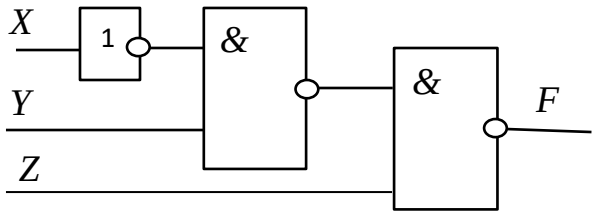
Вариант 3

1. Найдите значение логического выражения:
   $A + \bar{B} \cdot \bar{A}$ при *A=False, B= False* .
2. Упростите логическое выражение. Правильность упрощения проверьте с помощью таблицы истинности: $((X \vee Y) \wedge \bar{X}) \vee ((\overline{X \vee Y}) \wedge X)$
3. По заданному логическому выражению составьте логическую схему и заполните таблицу истинности: $\overline{X \vee Y} \wedge \bar{Z} = F$.
4. Составить логическую схему на основании приведённой ниже таблице истинности:

X_1	X_2	X_3	Y
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	0
1	1	1	0

Вариант 4

1. Найдите значение логического выражения:
   $\overline{B \vee A} \cdot \bar{B}$ при $A=False$, $B= True$.
2. Упростите логическое выражение. Правильность упрощения проверьте с помощью таблицы истинности: $\overline{((X \wedge Y) \vee (\bar{X} \wedge \bar{Y}) \wedge (X \vee \bar{Y})}$
3. По заданной логической схеме составьте логическое выражение и заполните для него таблицу истинности:



4. Составить логическую схему на основании приведённой ниже таблице истинности:

X_1	X_2	X_3	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

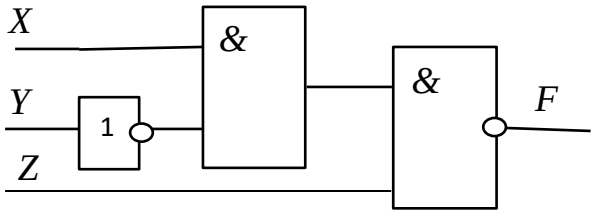
Вариант 5

1. Найдите значение логического выражения:
   $A \vee \bar{B} \cdot \bar{C}$ при *A=False, B= False, C= True* .
2. Упростите логическое выражение. Правильность упрощения проверьте с помощью таблицы истинности: $(X \vee Y) \cdot (\bar{X} \vee Y) \cdot (\bar{X} \vee \bar{Y})$
3. По заданному логическому выражению составьте логическую схему и заполните таблицу истинности:$\bar{X} \vee \overline{Y \wedge \bar{Z}} = F$
4. Составить логическую схему на основании приведённой ниже таблице истинности:

X_1	X_2	X_3	Y
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

Вариант 6

1. Найдите значение логического выражения:
   $(x = y) \vee (z < 4)$ при $x=5, y=7, z=0$.
2. Докажите данное равенство с помощью таблицы истинности: $X \to Y = \bar{X} \vee Y$
3. По заданной логической схеме составьте логическое выражение и заполните для него таблицу истинности:



4. Составить логическую схему на основании приведённой ниже таблице истинности:

X_1	X_2	X_3	Y
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

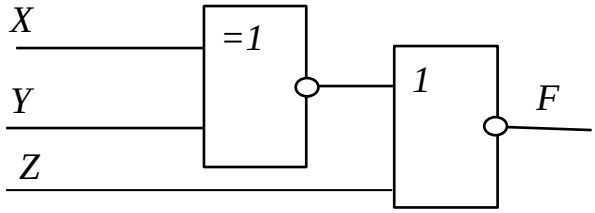
Вариант 7

1. Найдите значение логического выражения:
   $(a < z) \vee (z > -10) \wedge (a \neq 5)$ при $a=8$, $z=-6$.
2. Упростите логическое выражение. Правильность упрощения проверьте с помощью таблицы истинности: $\overline{(X \vee Y) \rightarrow (\overline{Y \vee Z})}$
3. По заданному логическому выражению составьте логическую схему и заполните таблицу истинности: $\bar{X} \wedge Y \vee \bar{Z} = F$
4. Составить логическую схему на основании приведённой ниже таблице истинности:

X_1	X_2	X_3	Y
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Вариант 8

1. Найдите значение логического выражения:
   $(x \neq z) \vee (z < 4) \wedge (y < 7)$ при x=-5, y=7, z=-5.
2. Докажите данное равенство с помощью таблицы истинности:
   $$A \oplus B = (\bar{A} \wedge B) \vee (A \wedge \bar{B})$$
3. По заданной логической схеме составьте логическое выражение и заполните для него таблицу истинности:



4. Составить логическую схему на основании приведённой ниже таблице истинности:

X_1	X_2	X_3	Y
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

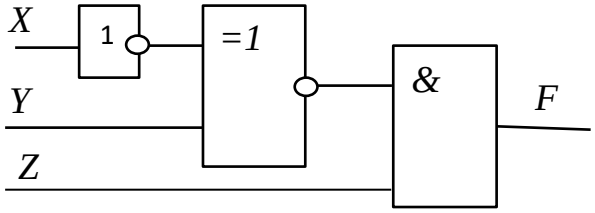
Вариант 9

1. Найдите значение логического выражения:
   $(a < b) \wedge (c < 2) \vee (a > c)$ при $a=8$, $b=-6$, $c=2$.
2. Упростите логическое выражение. Правильность упрощения проверьте с помощью таблицы истинности:
   $$X \wedge Y \vee \bar{X} \wedge Y \vee \bar{X} \wedge Z \vee X \wedge Z$$
3. По заданной логической схеме составьте логическое выражение и заполните для него таблицу истинности: $\overline{(X \vee Y) \wedge Z} = F$
4. Составить логическую схему на основании приведённой ниже таблице истинности:

X_1	X_2	X_3	Y
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

Вариант 10

1. Найдите значение логического выражения:
   $\bar{B} \vee C \wedge \overline{A \vee B}$ при $A=0$, $B=0$, $C=1$.
2. Упростите логическое выражение. Правильность упрощения проверьте с помощью таблицы истинности:
   $$(A \wedge C) \vee (A \wedge \bar{C}) \vee (\overline{\bar{A} \to \bar{B}})$$
3. По заданной логической схеме составьте логическое выражение и заполните для него таблицу истинности:



4. Составить логическую схему на основании приведённой ниже таблице истинности:

X_1	X_2	X_3	Y
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Темы для докладов, рефератов

1. Основные понятия теории множеств. Мощность множеств. Дискретные и непрерывные множества, примеры.
2. Круги Эйлера. Решение задач с помощью кругов Эйлера.
3. Элементы комбинаторики. Формулы и задачи комбинаторики.
4. Представление текстовой, графической информации в двоичном коде.
5. Кодирование звуковой и видео информации.
6. Функции алгебры логики. Представление процессов с помощью функций алгебры логики.
7. Минимизация функций алгебры логики. Релейно-контактные схемы.
8. Триггеры, их виды и принцип действия.
9. Шифраторы и дешифраторы.
10. Мультиплексоры и демультиплексоры.
11. Сравнение двоичной и троичной логики. Квантовые компьютеры.
12. Теория графов. Основные понятия. Деревья. Лес. Сеть.
13. Моделирование электрических сетей с помощью графов. Формула Мейзона.
14. Сети и потоковые модели.
15. Конечные автоматы.
16. Сети конечных автоматов.
17. Использование сетевых графиков для моделирования бизнеспроцессов.

Список литературы:

1. Бережной В.В., Шапошников А.В. Дискретная математика: учебное пособие (курс лекций). Ставрополь, Изд-во СКФУ, 2016. http://www.knigafund.ru/books/208367

2. Спирина М.С., Спирин П.А. Дискретная математика. М.: ФОРУМ, 2012.

3. Комогорцев В.Ф., Бардадын Н.Н. Дискретная математика. Брянск: Изд-во Брянская ГСХА, 2012.

4. Безик Д.А. Изучение микропроцессорной техники на примере микро-ЭВМ семейства МК51: учебно-метод. пособие с метод. указаниями к выполнению лабораторных работ. Брянск: Изд-во Брянская ГСХА, 2009.

Учебное издание

Татьяна Викторовна Бычкова

Дискретная математика
Учебное пособие

Редактор Лебедева Е.М.

Подписано к печати 16.07.2018 г. Формат 60х84. 1/16.
Бумага печатная Усл.п.л. 4,53. Тираж 100 экз. Изд. № 6186.
Издательство Брянского государственного аграрного университета
243365 Брянская обл., Выгоничский район, с. Кокино, Брянский ГАУ