МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Белгородский государственный технологический университет
им. В.Г. Шухова

Кафедра электроэнергетики и автоматики

Утверждено
научно-методическим советом
университета

Информатика

Методические указания к выполнению расчетно-графического задания
по дисциплине «Информатика»
для студентов направления бакалавриата
130302 – Электроэнергетика и электротехника,
профиль «Электроснабжение»,
профиль «Электропривод и автоматика»

Белгород
2018

УДК 004 (07)
ББК 32.81 я7
И74

Составители: ст. преп. И.А. Рыбин,
ассистент А.В. Рыбина
ассистент З.Г. Анисимова

Рецензент: канд. техн. наук, доцент Е.Н. Коробкова

Информатика: методические указания к выполнению
И74 расчетно-графического задания / сост.: И.А. Рыбин, А.В. Рыбина, З.Г. Анисимова. – Белгород: Изд-во БГТУ, 2018. – 54 с.

В методических указаниях приведены краткие теоретические сведения, примеры решения задач и задания для самостоятельного решения в рамках расчетно-графического задания по дисциплине «Информатика» в соответствии с государственным образовательным стандартом высшего образования.

Методические указания предназначены для студентов дневной формы обучения направления бакалавриата 130302 – Электроэнергетика и электротехника профилей «Электроснабжение», «Электропривод и автоматика».

Издание публикуется в авторской редакции.

УДК 004 (07)
ББК 32.81 я7

Оглавление

Введение

Учебным планом для студентов, обучающихся по направлению бакалавриата 130302 – Электроэнергетика и электротехника предусмотрено расчетно-графическое задание. Целью расчетно-графического задания является ознакомление студентов с логическими и арифметическими основами функционирования компьютерной техники.

Методические указания состоят из двух разделов и приложения. В первом разделе приведены теоретические сведения, примеры решения задач и задачи для самостоятельной работы по основам систем счисления: перевод из одной системы счисления в другую, получение прямого, обратного и дополнительного кода числа, представление числа в формате с фиксированной и плавающей запятой. Второй раздел включает краткие теоретические сведения, примеры решения задач и задачи для самостоятельной работы, посвященные логическим аспектам функционирования персонального компьютера: построение таблиц истинности, преобразование логических функций, построение логических выражений, составление логических схем. В приложении приведен образец оформления титульного листа. Знания, полученные в ходе выполнения заданий, будут полезны при дальнейшем изучении электроники и схемотехники.

Расчетно-графическое задание выполняется в соответствии с вариантом по журналу и оформляется в форме отчета.

Часть 1. Системы счисления. Представление числовой информации в персональном компьютере

Теоретические сведения

Система счисления — совокупность приёмов и правил наименования и обозначения чисел, позволяющих установить взаимно однозначное соответствие между любым числом и его представлением в виде конечного числа символов.

В любой системе счисления выбирается *алфавит*, представляющий собой совокупность некоторых символов (слов или знаков), с помощью которого в результате каких-либо операций можно представить любое количество. Изображение любого количества называется *числом*, а символы алфавита — *цифрами* (от лат. cifra).

Все системы счисления можно разделить на позиционные и непозиционные.

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

В настоящее время все наиболее распространённые системы счисления относятся к разряду позиционных. *Позиционная система счисления* — система, в которой значение цифры определяется её местоположением (позицией) в изображении числа. Упорядоченный набор символов (цифр) $\{a_0,a_1,\ldots a_n\}$, используемый для представления любых чисел в заданной позиционной системе счисления, называют её алфавитом, число символов (цифр) алфавита $p=n+1$ — её основанием, а саму систему счисления называют p-ичной. Основание позиционной системы счисления — количество различных цифр, используемых для изображения чисел в данной системе счисления.

Самой привычной для нас является десятичная система счисления. Её алфавит — $\{0,1,2,3,4,5,6,7,8,9\}$, а основание $p=10$, т.е. в этой системе для записи любых чисел используется только десять разных символов (цифр). Эти цифры введены для обозначения первых десяти последовательных чисел, а все последующие числа, начиная с 10 и т.д., обозначаются уже без использования новых цифр. Десятичная система счисления основана на том, что десять единиц каждого разряда объединяются в одну единицу соседнего старшего разряда, поэтому каждый разряд имеет вес, равный степени 10. Следовательно, значение одной и той же цифры определяется её местоположением в изображении числа, характеризуемым степенью числа 10.

Таким образом, любое число можно представить в виде полинома путём разложения его по степеням числа 10:

$$A_{10} = a_n \cdot 10^n + a_{n-1} \cdot 10^{n-1} + ... + a_1 \cdot 10^1 + a_0 \cdot 10^0 +$$
$$+ a_{-1} \cdot 10^{-1} + ... + a_{-m} \cdot 10^{-m} + ...,$$

последовательность из коэффициентов которого представляет собой десятичную запись числа A_{10}:

$$A_{10} = a_n a_{n-1} ... a_1 a_0 a_{-1} ... a_{-m} ...$$

Точка, отделяющая целую часть числа от дробной, служит для фиксации конкретных значений каждой позиции в этой последовательности цифр и является началом отсчёта.

В общем случае для задания p-ичной системы счисления необходимо определить основание и алфавит, состоящий из p различных символов (цифр) a_i, $i=1,...,p$. За основание системы можно принять любое натуральное число — два, три, четыре и т.д. Обычно в качестве алфавита берутся последовательные целые числа от 0 до $(p-1)$ включительно. В тех случаях, когда общепринятых (арабских) цифр не хватает для обозначения всех символов алфавита системы счисления с основанием $p>10$, используют буквенное обозначение цифр *a, b, c, d, e, f*. Таким образом, возможно бесчисленное множество позиционных систем: двоичная, троичная, четверичная и т.д. Запись чисел в каждой из систем счисления с основанием означает сокращённую запись выражения:

$$A_p = a_n \cdot p^n + a_{n-1} \cdot p^{n-1} + ... + a_1 \cdot p^1 + a_0 \cdot p^0 +$$
$$+ a_{-1} \cdot p^{-1} + ... + a_{-m} \cdot p^{-m} + ...,$$

где a_i — цифры системы счисления; n и m — число целых и дробных разрядов соответственно, A_p — запись числа A в p-ичной системе счисления. Изображением числа A в p-ичной системе счисления является последовательность цифр a_k.

Все известные позиционные системы счисления являются аддитивно-мультипликативными. Арифметические действия над числами в любой позиционной системе счисления производятся по тем же правилам, что и в десятичной системе, так как все они основываются на правилах выполнения действий над соответствующими полиномами. При этом нужно только пользоваться теми таблицами сложения и умножения, которые имеют место при данном основании p системы счисления. Во всех позиционных системах счисления с любым основанием p умножение на числа вида p^m, где m — целое число, сводится просто к перенесению запятой у множимого на m разрядов вправо или влево (в зависимости от знака

m), так же как и в десятичной системе.

Системы счисления используются для построения на их основе различных кодов в системах передачи, хранения и преобразования информации. *Код* (от лат. codex) — система условных знаков (символов) для представления различной информации.

Любому дискретному сообщению или знаку сообщения можно приписать какой-либо порядковый номер. Измерение аналоговой величины, выражающееся в сравнении её с образцовыми мерами, также приводит к числовому представлению информации. Передача или хранение сообщений при этом сводится к передаче или хранению чисел. Числа можно выразить в какой-либо системе счисления. Таким образом будет получен один из кодов, основанный на данной системе счисления. Каждому разряду числа можно поставить в соответствие какой-либо параметр электрического сигнала, например амплитуду.

Анализ систем счисления и построенных на их основе кодов с позиций применения в системах передачи, хранения и преобразования информации показывает, что чем больше основание системы счисления, тем меньшее число разрядов требуется для представления данного числа, а следовательно, и меньшее время для его передачи. Однако с ростом основания существенно повышаются требования к аппаратуре формирования и распознавания элементарных сигналов, соответствующих различным символам. Логические элементы вычислительных устройств в этом случае должны иметь большее число устойчивых состояний. С учётом этих обстоятельств в качестве показателя эффективности системы может быть выбрано число, равное произведению количества различных символов на количество разрядов для выражения любого числа. Тогда наиболее эффективной будет система, обеспечивающая минимум значения данного показателя.

Обозначим произведение основания системы p на длину разрядной сетки N, выбранную для записи чисел в этой системе, через C:

$$C = p \cdot N.$$

Если принять, что каждый разряд числа представлен не одним элементом с p устойчивыми состояниями, а p элементами, каждый из которых имеет одно устойчивое состояние, то показатель C определит условное количество оборудования, которое необходимо затратить на представление чисел в этой системе. В связи с этим показатель C называют показателем экономичности системы.

Максимальное число, которое можно изобразить в системе с основанием p:

$$A_{p\max} = p^N - 1.$$

Отсюда можно найти требуемую длину разрядной сетки:

$$N = \log_p(A_{p\max} + 1).$$

Тогда для любой системы счисления:

$$C = p\log_p(A_{p\max} + 1).$$

Допустим, что величина *p* является непрерывной величиной. При этом будем рассматривать величину *C* как функцию от величины *p*. Теперь если за единицу измерения оборудования принять условный элемент с одним устойчивым состоянием, то для сравнения двух систем счисления можно ввести относительный показатель экономичности:

$$F = \frac{p\log_p(A_{p\max} + 1)}{2\log_2(A_{2\max} + 1)},$$

позволяющий сравнить любую систему счисления с двоичной. Функция *F* имеет минимум, определяемый из условия $\frac{\partial F}{\partial p} = 0$, что соответствует значению $p = e \approx 2{,}72$. Следовательно, с точки зрения минимальных затрат условного оборудования наиболее экономичной является система счисления с основанием 3. Незначительно уступают ей двоичная и четверичная. Системы с основанием 10 и более существенно менее эффективны. Сравнивая эти системы с точки зрения удобства физической реализации соответствующих им логических элементов и простоты выполнения в них арифметических и логических действий, предпочтение в настоящее время отдаётся двоичной системе счисления. Действительно, логические элементы, соответствующие этой системе, должны иметь всего два устойчивых состояния. Задача различения сигналов сводится в этом случае к задаче обнаружения (есть импульс или его нет), что значительно проще. Арифметические и логические действия также легче осуществляются в двоичной системе.

Перевод чисел из одной системы счисления в другую

Для перевода целого числа A_p из *p*-ичной системы счисления в систему счисления с основанием *d* необходимо A_p разделить с остатком («нацело») на число *d*, записанное в той же *p*-ичной системе. Затем неполное частное, полученное от такого деления, нужно снова разделить с остатком на *d* и т.д., пока последнее полученное неполное частное не станет равным нулю. Представлением числа A_p в новой системе счисления будет последовательность остатков деления, изображённых d-ичной цифрой и записанных в порядке, обратном порядку их получения.

Пример 1.1. Перевести целое десятичное число 18 в двоичную систему счисления.

Решение.

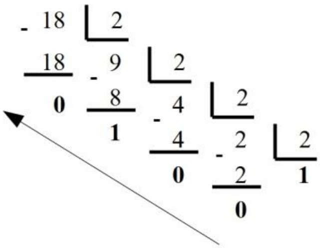


Рис. 1.1. Иллюстрация перевода целого числа из десятичной системы счисления в двоичную

Результат перевода: $(18)_{10} = (10010)_2$.

Перевод из десятичной системы счисления в двоичную можно осуществить с помощью таблицы степеней числа 2 (табл. 1.1).

Таблица 1.1

Таблица степеней числа 2

n степень	…	6	5	4	3	2	1	0	-1	-2	-3
2^n		64	32	16	8	4	2	1	0,5	0,25	0,125

Для перевода правильной дроби A_p из p-ичной системы счисления в систему счисления с основанием d необходимо A_p умножить на d, записанное в той же p-ичной системе, затем дробную часть полученного произведения снова умножить на d и т.д. до тех пор, пока дробная часть очередного произведения не станет равной нулю, либо не будет достигнута требуемая точность изображения числа A_p в d-ичной системе. Правильной называется дробь, числитель которой меньше знаменателя.

Пример 1.2. Перевести правильную десятичную дробь 0,5625 в двоичную систему счисления.

Решение.

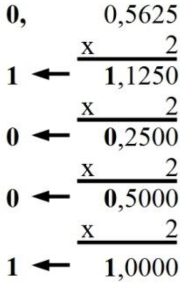


Рис. 1.2. Иллюстрация перевода десятичной дроби в двоичную систему счисления

Результат перевода: $0{,}5625_{10} = 0{,}1001_2$.

Нередко при переводе правильной дроби из десятичной системы счисления в двоичную результатом вычислений является иррациональная дробь. В этом случае число необходимых знаков после запятой определяет пользователь, исходя из необходимой погрешности.

При переводе неправильной дроби переводят отдельно целую и дробную части, руководствуясь соответствующими правилами.

Пример 1.3. Перевести десятичное число 9,1875 в двоичную систему счисления.

Решение.

Вначале необходимо перевести целую часть десятичного числа в двоичную систему счисления: $9_{10} = 1001_2$, а затем правильную дробь: $0{,}1875_{10} = 0{,}0011_2$. Окончательный ответ: $9{,}625_{10} = 1001{,}0011_2$.

Особого внимания заслуживает случай перевода чисел из одной системы счисления в другую, когда основания данных систем счисления p и d связаны равенством $p{=}d^k$, где k — целое положительное число. В этом случае перевод из d-ичной в p-ичную систему счисления может быть осуществлён по следующему правилу.
В исходной, d-ичной, записи числа разряды объединяются вправо и влево от точки в группы длины k (добавляя в случае необходимости левее старшей или правее младшей значащих цифр соответствующее количество нулей), и каждая такая группа записывается одной цифрой p-ичной системы счисления.

Пример 1.4. Перевести число 305,4 из восьмеричной системы счисления в двоичную.

Решение.

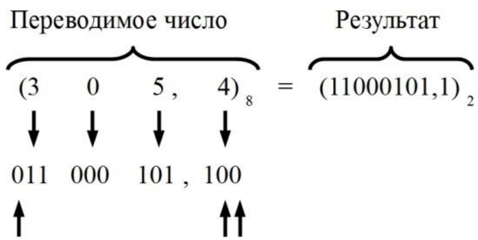


Рис. 1.3. Иллюстрация перевода числа из восьмеричной системы счисления в двоичную

Отмеченные символами «↑» нули следует отбросить.

Пример 1.5. Перевести число 7D2.E из шестнадцатеричной системы счисления в двоичную.

Решение.

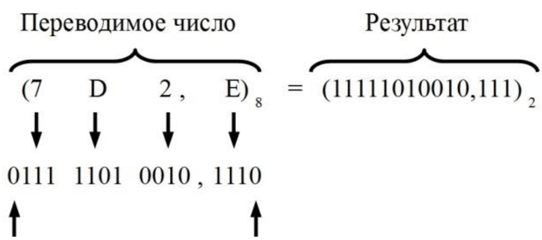


Рис. 1.4. Иллюстрация перевода числа из шестнадцатеричной системы счисления в двоичную

Отмеченные крайние нули следует отбросить.

Для обратного перевода из *p*-ичной в *d*-ичную систему счисления — каждая цифра числа, заданного в *p*-ичной системе счисления заменяется её *d*-ичным изображением.

Пример 1.6. Перевести число 111001100,001 из двоичной системы счисления в восьмеричную.

Решение.

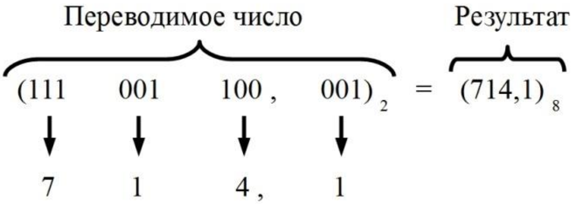


Рис. 1.5. Иллюстрация перевода числа из двоичной системы счисления в восьмеричную

Перевод в десятичную систему числа A, записанного в p-ичной системе счисления в виде $A_p=(a_n a_{n-1}\ldots\ a_0,a_{-1}a_{-2}\ldots a_{-m})_p$ сводится к вычислению значения многочлена $A_{10}=a_n\cdot 10^n+a_{n-1}\cdot 10^{n-1}+\ldots+$ $+a_1\cdot 10^1+a_0\cdot 10^0+a_{-1}\cdot 10^{-1}+\ldots+a_{-m}\cdot 10^{-m}+\ldots,$ средствами десятичной арифметики.

Пример 1.7. Перевести число 11011,11 из двоичной системы счисления в десятичную.

Решение.

$$11011{,}11_2 = 1\cdot 2^4 + 1\cdot 2^3 + 0\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0 + 1\cdot 2^{-1} + 1\cdot 2^{-2} =$$
$$= 16 + 8 + 0 + 2 + 1 + 0{,}5 + 0{,}25 = 27{,}75_{10}.$$

Пример 1.8. Перевести шестнадцатеричное число 2E5,A в десятичную систему счисления.

Решение.

$$2E5{,}A_{16}=2\cdot 16^2+14\cdot 16^1+5\cdot 16^0+10\cdot 16^{-1}=741{,}625_{10}.$$

Двоично-десятичная система счисления получила большое распространение в современных ЭВМ ввиду легкости перевода в десятичную систему счисления и обратно. Она используется при решении задач учетно-статистического характера. В этой системе счисления все десятичные цифры отдельно кодируются четырьмя двоичными цифрами (тетрадами) и в таком виде записываются последовательно друг за другом.

Пример 1.9. Записать десятичное число 37 в двоично-десятичной системе счисления.

Решение.

$$\overbrace{(3\qquad 7)_{10}}^{\text{Переводимое число}} = \overbrace{(00110111)_{2\text{-}10}}^{\text{Результат}}$$

$$\begin{array}{cc} \downarrow & \downarrow \\ 0011 & 0111 \\ \uparrow\uparrow & \end{array}$$

Рис. 1.6. Иллюстрация перевода числа из десятичной системы счисления в двоично-десятичную

Этот перевод напоминает перевод шестнадцатеричных чисел в двоичную систему счисления.

Арифметические операции в позиционных системах счисления

Правила арифметики во всех позиционных системах счисления аналогичны.

Пример 1.10. Выполнить операцию арифметического сложения в двоичной системе счисления.

Решение.

$$\left[\begin{array}{r} \overset{\bullet}{1}3 \\ +\,7 \\ \hline 20 \end{array}\right]_{10} \rightarrow \left[\begin{array}{r} \overset{\bullet}{0}\overset{\bullet}{1}\overset{\bullet}{1}\overset{\bullet}{0}1 \\ +00111 \\ \hline 10100 \end{array}\right]_{2}$$

Рис. 1.7. Сложение целых чисел в двоичной системе счисления

Точками показаны переносы.

В устройствах, реализующих операцию арифметического сложения двоичных чисел, операнды представляют числами определенной разрядности (одинаковой для обоих операндов). При этом неиспользуемые старшие разряды заполняются нулями.

Пример 1.11. Выполнить операцию арифметического сложения двух вещественных чисел в двоичной системе счисления.

Решение.

$$\left[\begin{array}{r} 53{,}25 \\ +\ 11{,}5 \\ \hline 64{,}75 \end{array}\right]_{10} \rightarrow \left[\begin{array}{r} 0110101{,}01 \\ +\ 0001011{,}10 \\ \hline 1000000{,}11 \end{array}\right]_{2}$$

Рис. 1.8. Сложение вещественных чисел в двоичной системе счисления

При сложении вещественных чисел в общем случае перенос осуществляется и из дробной части числа в целую часть.

Умножение двоичных многоразрядных чисел производится путем образования частичных произведений и последующего их суммирования. Каждое частичное произведение равно нулю, если в соответствующем разряде множителя стоит 0, или равно множимому, сдвинутому на соответствующее число разрядов влево, если в разряде множителя стоит 1. Таким образом, операция умножения многоразрядных двоичных чисел внутри ЭВМ сводится к операции сдвига и сложения. Положение точки, отделяющей целую часть от дробной части, определяется так же, как и при умножении десятичных чисел.

Пример 1.12. Перемножить десятичные числа 7,5 и 5 в двоичной системе счисления.

Решение.

$$\left[\begin{array}{r} 7{,}5 \\ \mathrm{x}\ 5 \\ \hline 37{,}5 \end{array}\right]_{10} \rightarrow \left[\begin{array}{r} 111{,}1 \\ \mathrm{x}\ 101 \\ \hline 1111 \\ +\ 0000 \\ 1111 \\ \hline 100101{,}1 \end{array}\right]_{2}$$

Рис. 1.9. Умножение вещественных чисел в двоичной системе счисления

Прямой, обратный и дополнительный коды

В вычислительной технике, с целью упрощения реализации арифметических операций, применяют специальные коды. За счет этого облегчается определение знака результата операции, а операция вычитания чисел сводится к арифметическому сложению. В результате упрощаются устройства, выполняющие арифметические операции.

Для этого применяют прямой, обратный и дополнительный коды.

Прямой двоичный код $P_{пр}(x)$ — это такое представление двоичного числа x, при котором знак «+» кодируется нулем в старшем разряде числа, а знак «–» — единицей. При этом старший разряд называется знаковым. Остальные разряды двоичного числа называются значащими. Например, числа $+5_{10}$ и $–5_{10}$, представленные в прямом четырехразрядном двоичном коде, выглядят так: $+5_{10} = 0'101_2$; $–5_{10} = 1'101_2$.

Примечание. Апострофом условно (для удобства определения знака) отделены знаковые разряды.

Обратный код $P_{обр}(x)$ получается из прямого кода по следующему правилу:

$$P_{обр}(x) = \begin{cases} 0'\, P_{пр}(x), & \text{при}\, x \geq 0, \\ 1'\, \overline{P_{пр}(x)}, & \text{при}\, x \leq 0. \end{cases}$$

Из приведенного выражения видно, что обратный код для положительных чисел совпадает с прямым кодом. Чтобы представить отрицательное двоичное число в обратном коде, нужно поставить в знаковом разряде 1, во всех значащих разрядах заменить 1 на 0, а 0 на 1. Такая операция называется инверсией и обозначается горизонтальной чертой над инвертируемым выражением (см. часть 2 «Алгебра логики»).

Пример 1.13. Получить обратный код для числа $x = –11_{10}$.

Решение.

$P_{пр}(x) = 1'1011_2$ $P_{обр}(x) = 1'0100_2$.

Считается, что здесь числа представлены пятью разрядами.

Дополнительный код $P_{доп}(x)$ образуется следующим образом: дополнительный код положительного числа совпадает с прямым кодом, а для отрицательного числа получается инверсией всех значащих разрядов и добавлением единицы к младшему разряду результата.

Дополнительный код отрицательного числа может быть получен из обратного кода путем прибавления 1 к младшему разряду обратного кода (с учетом переносов между разрядами).

$$P_{доп}(x) = P_{обр}(x) + 1.$$

Пример 1.14. Получить дополнительный код для числа $x = –14_{10}$.

Решение.

Прямой код — $P_{пр}(x)$ = $1'1110_2$, обратный код — $P_{обр}(x)$ = $1'0001_2$, дополнительный код — $P_{доп}(x)$ = $1'0010_2$.

При алгебраическом сложении двоичных чисел положительные слагаемые представляют в прямом коде, а отрицательные слагаемые — в дополнительном коде, производят арифметическое суммирование этих кодов, включая разряды знаков, которые при этом рассматривают как старшие разряды. При возникновении переноса из разряда знака единицу переноса отбрасывают. В результате получают алгебраическую сумму в прямом коде, если эта сумма положительная, и в дополнительном коде, — если сумма отрицательная.

Пример 1.15. Выполнить алгебраическое сложение с использованием дополнительного кода для чисел $x_1 = 7_{10}$ и $x_2 = -3_{10}$.

Решение.

Необходимо найти сумму: $y = x_1 + x_2$. Учитывая, что $x_1 > 0$, это число нужно представить в прямом коде, а так как $x_2 < 0$, то число x_2 нужно перевести в дополнительный код.

$$P(y) = P_{пр}(x_1) + P_{доп}(x_2).$$

$$P_{пр}(x_1) = 0'111,\ P_{пр}(x_2) = 1'011,\ P_{обр}(x_2) = 1'100,\ P_{доп}(x_2) = 1'101.$$

$$P(y) = \left[\begin{array}{r} \bullet\bullet\bullet\ \ \\ 0'111 \\ +\ 1'101 \\ \hline 0'100 \end{array}\right]_2$$

Рис. 1.10. Сложение двоичных чисел с использованием дополнительного кода

Так как результат положителен (в знаковом разряде $P(y)$ — 0), значит, он представлен в прямом коде $P_{пр}(y) = 0'100_2$, а в десятичной системе счисления: $y = +4_{10}$.

Пример 1.16. Выполнить алгебраическое сложение с использованием дополнительного кода для чисел $x_1 = -6_{10}$ и $x_2 = -17_{10}$.

Решение.

Необходимо найти сумму: $y = x_1 + x_2$. Числа x_1 и x_2 нужно представить в дополнительном коде.

$$P(y) = P_{доп}(x_1) + P_{доп}(x_2).$$

$P_{пр}(x_1) = 1'00110,\ P_{обр}(x_1) = 1'11001,\ P_{доп}(x_1) = 1'11010,\ P_{пр}(x_2) = 1'10001,$
$P_{обр}(x_2) = 1'01110,\ P_{доп}(x_2) = 1'01001.$

$$P(y) = \left[\begin{array}{r} 1'11010 \\ +\ 1'01111 \\ \hline 1'01001 \end{array}\right]_2$$

Рис. 1.11. Сложение двоичных чисел с использованием дополнительного кода

В знаковом разряде стоит единица, и, значит, результат получен в дополнительном коде. Для перехода от дополнительного кода $P_{доп}(y) = 1'01001$, к прямому коду $P_{пр}(y)$ необходимо применить следующие преобразования: $P_{обр}(y) = P_{доп}(y) - 1 = 1'01001 - 1 = 1'01000,$ $P_{пр}(y) = \overline{P_{обр}(y)} = \overline{1'01000} = 1'10111,$ или в десятичной системе счисления: $y = -23_{10}$.

Представление данных в персональном компьютере

При проведении математических расчетов числа в персональном компьютере могут быть представлены с помощью естественной и нормальной форм записи. Примером записи в естественной форме может служить вещественное число 173,856. Для записи числа в естественной форме машинное слово (операнд) делится на две части (на два поля). Первое поле отводится для записи целой части числа, второе — для записи дробной части числа. Старший разряд машинного слова используется для указания знака числа. Знак положительного числа кодируется двоичной цифрой 0, а знак отрицательного числа – цифрой 1.

Разряды машинного слова нумеруются справа – налево, начиная с нуля. В вычислительной технике принято отделять целую часть числа от дробной части точкой. Так как положение точки между целой и дробной частями числа четко определено, то такое представление чисел называют *формой с фиксированной точкой*. На рис. 1.12 дана иллюстрация формата чисел с фиксированной точкой.

Недостатком формы с фиксированной точкой является малый диапазон представления чисел. Поскольку некоторые вещественные числа нельзя точно представить в виде конечного числа двоичных разрядов, то всегда существует некоторая *точность представления* числа, зависящая от количества бит, отведенных под дробную часть. В современных персональных компьютерах в этой форме записывают только целые числа. При этом отпадает необходимость отводить поле для записи дробной части числа.

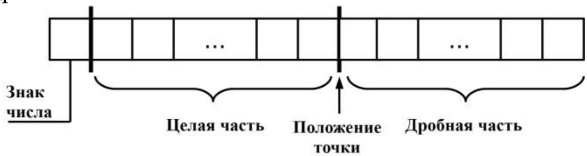


Рис. 1.12. Формат чисел с фиксированной точкой

Нормальная форма записи числа *n* имеет следующий вид:

$$n = m \cdot d^p,$$

где *m* — мантисса числа; *p* — порядок; *d* — основание системы счисления. Приведем пример записи числа в нормальной форме: $n=1{,}23 \cdot 10^2$.

Порядок *p* изменяет местоположение точки в мантиссе. В зависимости от значения порядка *p* точка перемещается (плавает) по мантиссе. Например, пусть $m = 0{,}5$, основание системы счисления $d = 10$, а порядок *p* будем брать разным: $0{,}5 \cdot 10^{-1}=0{,}05$; $0{,}5 \cdot 10^{-2}=0{,}005$; $0{,}5 \cdot 10^{2}=50$; $0{,}5 \cdot 10^{3}=500$.

Рассматриваемая форма представления чисел называется *формой с плавающей точкой*. Из приведенного примера видно, что благодаря изменению порядка точка перемещается (плавает) по мантиссе. При этом если порядок отрицательный, точка смещается по мантиссе влево, а если положительный, то — вправо. В нормальной форме машинное слово делится на два поля. В одном поле записывается мантисса числа, а во втором — указывается порядок числа. Следующий рисунок иллюстрирует форму числа с плавающей точкой на примере 32-х разрядного машинного слова.

Рис. 1.13. Формат чисел с плавающей точкой

Диапазон представления чисел с плавающей точкой значительно больше диапазона представления чисел с фиксированной точкой. Однако быстродействие ЭВМ при обработке чисел с плавающей точкой гораздо ниже, чем при обработке чисел с фиксированной точкой. Этим объясняется одновременное существование двух форм чисел. Последовательность нескольких битов или байтов называют полем данных. В ЭВМ используют поля постоянной (полуслово — 1 байт, слово — 2 байта, двойное слово — 4 байта, расширенное слово — 8 байт) и переменной длины. Поля переменной длины могут иметь любой размер от 1 до 256 байт. При этом поле должно состоять из целого числа байтов.

Пример 1.17. Записать число $-193_{10} = -11000001_2$ в формате слова со знаком и фиксированной точкой.

Решение.

В старшем разряде (пятнадцатом) указан знак числа.

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

Для представления как положительных, так и отрицательных порядков применяют *смещенный порядок*. При этом машинный порядок M_p формируют со смещением на 64 разряда по отношению к фактическому порядку p: $M_p = p + 64$, чтобы не отводить отдельный разряд под знак порядка. При фактическом порядке p, равном нулю, машинный порядок M_p равен 64, а при фактическом порядке равном $+63_{10}$ машинный порядок равен своему максимальному значению 127_{10}. Для отрицательного порядка -64_{10} машинный порядок равен нулю. Таким образом, и положительные и отрицательные порядки представляют только положительными числами.

Пример 1.18. Записать число -193_{10} в формате двойное слово и плавающей точкой.

Решение.

Переведем десятичное число в двоичную систему счисления: $-193_{10} = -11000001_2 = -0{,}11000001 \cdot 2^8$.

	Знак числа	Порядок 1000_2							Мантисса $0{,}11000001_2$											
N	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	…	1	0
Число	1	1	0	0	1	0	0	0	1	1	0	0	0	0	0	1	0		0	0

В *нормализованной форме* целая часть мантиссы должна быть равна нулю, а первый знак после точки должен быть равен единице. Здесь мантисса $m = -0{,}11000001$, фактический порядок $p = 8_{10} = 1000_2$. Машинный порядок: $M_p = p + 64 = 1000_2 + 1000000_2 = 1001000_2$.

Скорость выполнения компьютером операций с числами, представленными в форме с плавающей запятой, измеряется во FLOPS (от англ. floating-point operations per second — «количество операций с плавающей запятой в секунду») и является одной из основных единиц измерения быстродействия вычислительных систем.

Задания для самостоятельного решения

1. Перевести десятичный номер зачетной книжки в двоичную систему счисления, полученное двоичное число перевести в восьмеричную и шестнадцатеричную систему счисления.
2. Перевести десятичный номер зачетной книжки в двоично-десятичную систему счисления.
3. Перевести число из десятичной системы счисления в двоичную систему счисления (см. таблицу ниже).
4. Перевести число из двоичной системы счисления в десятичную систему счисления (см. таблицу ниже).
5. Перевести число из десятичной системы счисления в двоичную с точностью до седьмого знака после запятой (см. таблицу ниже).
6. Выполнить операции сложения и умножения с двоичным числом из задания 4 (см. таблицу ниже).

Вариант	№ задания			
	3	4	5	6
1.	123.250	1111111.101	0.345	11001.011
2.	185.750	1010110.011	0.456	10110.101
3.	112.125	1110110.101	0.567	10111.111
4.	192.625	1010110.001	0.678	11100.110
5.	116.750	1110100.111	0.789	11001.001
6.	115.375	1101110.101	0.234	11010.011
7.	178.750	1100101.011	0.123	10111.101
8.	159.125	1110101.101	0.246	11110.111
9.	145.625	1100100.011	0.357	10101.110
10.	129.750	1011011.101	0.682	11010.001
11.	103.250	1110111.100	0.343	11011.011
12.	105.750	1011110.010	0.121	11101.001
13.	102.125	1110100.101	0.122	10100.101
14.	182.625	1010110.011	0.232	11111.011

Вариант	№ задания			
	3	4	5	6
15.	106.750	1111100.111	0.353	10101.101
16.	125.375	1101010.110	0.454	10101.110
17.	146.250	1111111.001	0.565	10110.110
18.	114.125	1111111.010	0.676	11100.011
19.	162.875	1111111.100	0.787	11101.111
20.	133.250	1010101.001	0.898	10111.110
21.	188.625	1010101.010	0.231	11110.011
22.	121.375	1010101.011	0.342	10110.110
23.	140.500	1010101.100	0.453	11100.111
24.	150.250	1111101.001	0.564	11101.011
25.	139.750	1111101.010	0.675	10000.101

7. Сформировать дополнительный код для пятиразрядного отрицательного десятичного числа –ABCDE. Цифры A, B, C, D, E являются последними цифрами зачетной книжки (студенческого билета). Например, номер зачетной книжки 4376534567. Значит, десятичное число -34567 (со знаком минус). Нужно получить дополнительный код этого числа.
8. Сформировать число x_1 из четырех последних цифр зачетной книжки. Сформировать отрицательное число x_2 из четырех последних цифр зачетной книжки, взятых в обратном порядке. Например, x_1 = 1234, x_2 = -4321. Выполнить алгебраическое сложение чисел в дополнительном коде. Самостоятельно выбрать необходимую разрядность двоичных чисел.
9. Записать десятичное число A (см. таблицу ниже) в формате слова со знаком и фиксированной точкой в машинное слово заданной разрядности.

Вариант	Число A	Разрядность машинного слова
1.	100	8
2.	-110	8
3.	120	8
4.	-80	8
5.	60	8
6.	-111	8
7.	101	8
8.	-91	8
9.	25	8

Вариант	Число А	Разрядность машинного слова
10.	-56	8
11.	82	8
12.	-100	8
13.	96	8
14.	87	16
15.	-8000	16
16.	9000	16
17.	-10000	16
18.	16000	16
19.	-777	16
20.	2017	16
21.	-999	16
22.	2033	16
23.	-5000	16
24.	4444	16
25.	-3333	16

10. Записать десятичное число А в формате с плавающей точкой со смещенным порядком. Для записи порядка использовать 7 разрядов, а для записи мантиссы – 23 разряда. Один разряд использовать для записи знака числа.

Вариант	Число А	Разрядность машинного слова
1.	100	32
2.	-110	32
3.	120	32
4.	-80	32
5.	60	32
6.	-111	32
7.	101	32
8.	-91	32
9.	25	32
10.	-56	32
11.	82	32
12.	-100	32
13.	96	32
14.	87	32

Вариант	Число А	Разрядность машинного слова
15.	-80	32
16.	90	32
17.	-10	32
18.	160	32
19.	-77	32
20.	20	32
21.	-99	32
22.	203	32
23.	-50	32
24.	44	32
25.	-33	32

Часть 2. Алгебра логики. Применение средств алгебры логики для описания функционирования персонального компьютера

Теоретические сведения

Для описания работы аппаратных и программных средств персонального компьютера используется алгебра логики или булева алгебра (основоположником этого раздела математики был Джордж Буль). Булева алгебра оперирует с логическими переменными, которые могут принимать только два значения: истина или ложь, обозначаемые соответственно 1 и 0. Логические операции широко используются при аппаратной реализации самой ЭВМ и при программной реализации многих приложений (например, слияние объектов в векторных графических редакторах осуществляют с помощью логических операций И, ИЛИ, И-НЕ). Кроме того, современные языки программирования немыслимы без встроенных в них логических функций.

Логической функцией от набора логических переменных (аргументов) $F(x_1, x_2, ..., x_n)$ называется функция, которая может принимать только два значения: истина или ложь (1 или 0). Область определения функции — это количество возможных наборов значений аргументов (или количество строк в таблице истинности). Оно равно $S = 2^N$, где N — количество переменных. Любая логическая функция может быть задана с помощью *таблицы истинности*, в левой части которой записываются возможные наборы аргументов, а в правой части — соответствующие им значения функции (например, функция «дизъюнкция»). Из логических функций широкое распространение имеют функции «конъюнкция», «дизъюнкция» и «инверсия», которые составляют функционально полную систему логических функций. С помощью этих трех функций можно представить (аналитически выразить) любую сколь угодно сложную логическую функцию.

Базовые логические операции

Логические операции инверсии, конъюнкции и дизъюнкции называют базовыми.

1. *Инверсия (отрицание, логическое «НЕ»).*

Инверсия делает истинное высказывание ложным и наоборот, ложное — истинным (см. табл. 2.1). Обозначение операции в алгебре логики: *не A*, $\neg A$, $\overline{A}$, *not A,* в языках программирования Pascal и Basic: *not A*, в языках С, С++: как *!A.*

Таблица 2.1

Таблица истинности операции НЕ

A	$F = \overline{A}$
0	*1*
1	*0*

2. *Конъюнкция (логическое умножение, логическое «И»).*

Операция логического умножения, соответствующая функции «И», выдает в качестве результата значение, называемое логическим произведением. Результат операции истинен тогда и только тогда, когда истинны все входящие в него простые высказывания. На формальном языке алгебры логики операция конъюнкции обозначается как *«&», «И», «and», «$\wedge$», «·»* (знаком умножения). Например, $F = A \& B$. В языках программирования используют обозначения «*A and B*» (Pascal, Basic) или «*A && B*» (C, C++). Аргументы могут принимать значения 1 или 0 и результат тоже только значения 1 или 0. Значение логической функции F можно определить из таблицы истинности этой функции (см. табл. 2.2).

Таблица 2.2

Таблица истинности операции И

A	B	$F = A \& B$
0	*0*	*0*
0	*1*	*0*
1	*0*	*0*
1	*1*	*1*

3. *Дизъюнкция (логическое сложение, логическое «ИЛИ»).*

Операция логического сложения, соответствующая функции «ИЛИ», выдает в качестве результата значение, называемое логической суммой. Результат операции истинен тогда, когда истинно хотя бы одно из входящих в него простых высказываний. На формальном языке алгебры логики операция дизъюнкции обозначается как *«+», «$\vee$», «ИЛИ», «or»* (например, $F = A + B$), а в языках программирования — «*A or B*» (Pascal, Basic), «*A || B*» (C, C++).. Аргументы могут принимать значения 1 или 0 и результат тоже только значения 1 или 0. Значение логической функции F можно определить из таблицы истинности этой функции (см. табл. 2.3).

Таблица 2.3

Таблица истинности операции ИЛИ

A	*B*	*F = A + B*
0	*0*	*0*
0	*1*	*1*
1	*0*	*1*
1	*1*	*1*

Любую функцию с помощью тождественных преобразований можно представить таким образом, чтобы она содержала только базовые логические операции.

Другие логические операции

1. *Сложение по модулю 2 (исключающее «ИЛИ»).*

Результат операции истинен тогда, когда истинно только одно из входящих в него простых высказываний, но не оба одновременно. На формальном языке алгебры логики операция исключающего «ИЛИ» обозначается как *«⊕»* или *«xor»* (например, $F = A \oplus B$), языках программирования — *«A xor B»* (Pascal), *«A ^ B»* (C). Значение логической функции *F* можно определить из таблицы истинности этой функции (см. табл. 2.4).

Таблица 2.4

Таблица истинности операции исключающее ИЛИ

A	*B*	*F = A ⊕ B*
0	*0*	*0*
0	*1*	*1*
1	*0*	*1*
1	*1*	*0*

Операция «исключающее ИЛИ» обладает интересными свойствами. По таблице истинности несложно проверить, что $A \oplus 0 = A$, $A \oplus 1 = A$, $A \oplus A = 0$. Для доказательства этих равенств можно просто подставить в них $A = 0$ и $A = 1$. Эту операцию можно представить через базовые операции («НЕ», «И», «ИЛИ») следующим образом: $A \oplus B = A \cdot \overline{B} + \overline{A} \cdot B$.

2. *Импликация (следование).*

Операция импликация равносильна логическому выражению ¬A + B (не A или B). Результат операции ложен тогда и только тогда, когда из истинной предпосылки (первого высказывания) следует ложный вывод (второе высказывание). На формальном языке алгебры логики операция импликации обозначается как *«→»*. Например, $F = A \rightarrow B$.

Значение логической функции F можно определить из таблицы истинности этой функции (см. табл. 2.5).

Таблица 2.5

Таблица истинности операции импликация

A	B	$F = A \rightarrow B$
0	0	1
0	1	1
1	0	0
1	1	1

Эту операцию можно представить через базовые операции следующим образом: $A \rightarrow B = \overline{A} + B$.

3. *Эквивалентность (равенство).*

Результат операции истинен тогда и только тогда, когда оба высказывания одновременно либо ложны, либо истинны. На формальном языке алгебры логики операция эквивалентности обозначается как *«~»*, *«$\leftrightarrow$»* (например, $F = A \sim B$). Значение логической функции F можно определить из таблицы истинности этой функции (см. табл. 2.6).

Таблица 2.6

Таблица истинности операции эквивалентность

A	B	$F = A \sim B$
0	0	1
0	1	0
1	0	0
1	1	1

Эту операцию можно представить через базовые операции следующим образом: $A \leftrightarrow B = A \cdot B + \overline{A} \cdot \overline{B}$, $A \leftrightarrow B = (\overline{A} + B) \cdot (A + \overline{B})$.

4. *Штрих Шеффера (логическое «И-НЕ»).*

Результат операции ложен тогда и только тогда, когда оба высказывания одновременно истинны. На формальном языке алгебры логики операция обозначается как *«|»* (например, $F = A \mid B$). Значение логической функции F можно определить из таблицы истинности этой функции (см. табл. 2.7).

Таблица 2.7

Таблица истинности операции штрих Шеффера

A	B	$F = A \mid B$
0	0	0

A	*B*	*F = A | B*
0	*1*	*0*
1	*0*	*0*
1	*1*	*1*

Операцию штрих Шеффера («И–НЕ», англ. nand = «not and») можно записать как $F = A | B = \overline{A \cdot B}$.

5. *Стрелка Пирса (логическое «ИЛИ-НЕ»).*

Результат операции истинен тогда и только тогда, когда оба высказывания одновременно ложны. На формальном языке алгебры логики операция обозначается как *«↓»* (например, *F = A ↓ B)*. Значение логической функции *F* можно определить из таблицы истинности этой функции (см. табл. 2.8).

Таблица 2.8

Таблица истинности операции стрелка Пирса

A	*B*	*F = A ↓ B*
0	*0*	*1*
0	*1*	*0*
1	*0*	*0*
1	*1*	*0*

Операцию стрелка Пирса («ИЛИ–НЕ», англ. nor = «not or») можно записать как $F = A \downarrow B = \overline{A + B}$.

Логические переменные, объединенные знаками логических операций, составляют *логические выражения*. Пример логического выражения: $F = (a \wedge \bar{b} \vee c) \wedge (a \vee b)$. При определении значения логического выражения принято следующее *старшинство (приоритет) логических операций*:

1. отрицание;
2. логическое произведение;
3. логическое сложение, исключающее или;
4. импликация, эквиваленция.

Скобки меняют порядок выполнения операций.

В табл. 2.9 приведены основные законы алгебры логики, позволяющие производить тождественные преобразования логических выражений.

Таблица 2.9

Основные законы алгебры логики

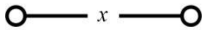
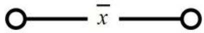
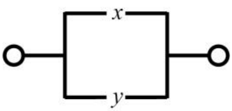
Закон	**Для дизъюнкции**	**Для конъюнкции**
Переместительный	$A+B=B+A$	$A\cdot B=B\cdot A$
Сочетательный	$(A+B)+C=A+(B+C)$	$(A\cdot B)\cdot C=A\cdot(B\cdot C)$
Распределительный	$A\cdot(B+C)=A\cdot B+A\cdot C$	$A+B\cdot C=(A+B)\cdot(A+C)$
Законы де Моргана	$\overline{A+B}=\overline{A}\cdot\overline{B}$ $\overline{\overline{A}+\overline{B}}=A\cdot B$	$\overline{A\cdot B}=\overline{A}+\overline{B}$ $\overline{\overline{A}\cdot\overline{B}}=A+B$
Идемпотенции	$A+A=A$	$A\cdot A=A$
Поглощения	$A+A\cdot B=A$	$A\cdot(A+B)=A$
Склеивания	$(A\cdot B)+(\overline{A}\cdot B)=B$	$(A+B)\cdot(\overline{A}+B)=B$
Переменная и ее инверсия	$A+\overline{A}=1$	$A\cdot\overline{A}=0$
Переменная и константа	$A+0=A$ $A+1=1$	$A\cdot 1=A$ $A\cdot 0=0$
Двойное отрицание	$\overline{\overline{A}}=A$	

Синтез переключательных схем

В компьютерах и других автоматических устройствах широко применяются электрические схемы, содержащие сотни и тысячи переключательных элементов: реле, выключателей и т.п. Разработка таких схем весьма трудоёмкое дело, для этого может быть использован аппарат алгебры логики. *Переключательная схема* — схематическое изображение некоторого устройства, состоящего из переключателей и соединяющих их проводников, а также из входов и выходов, на которые подаётся и с которых снимается электрический сигнал. Каждый переключатель имеет только два состояния: замкнутое и разомкнутое. Переключателю X поставим в соответствие логическую переменную x, которая принимает значение 1 в том и только в том случае, когда переключатель X замкнут и схема проводит ток; если же переключатель разомкнут, то x равен нулю. Будем считать, что два переключателя X и $\overline{X}$ связаны таким образом, что когда X замкнут, то $\overline{X}$ разомкнут, и наоборот. Следовательно, если переключателю X поставлена в соответствие логическая переменная x, то переключателю $\overline{X}$ должна соответствовать переменная $\overline{x}$. Всей переключательной схеме также можно поставить в соответствие логическую переменную, равную единице, если схема проводит ток, и равную нулю — если не проводит. Эта переменная является функцией от переменных, соответствующих всем переключателям схемы, и называется функцией проводимости. Ниже приведены функции проводимости *F* некоторых переключательных схем (табл. 2.10).

Таблица 2.10

Функции проводимости некоторых переключательных схем

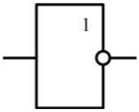
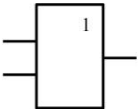
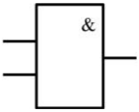
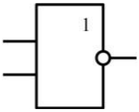
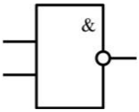
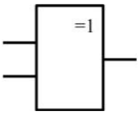
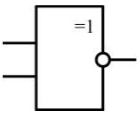
Переключательная схема	Функция проводимости	Примечание
	$F=1$	схема не содержит переключателей и проводит ток всегда
	$F=0$	схема содержит один постоянно разомкнутый контакт
	$F(x) = x$	схема проводит ток, когда переключатель *x* замкнут, и не проводит, когда *x* разомкнут
	$F(x) = \bar{x}$	схема проводит ток, когда переключатель *x* разомкнут, и не проводит, когда *x* замкнут
	$F(x) = x \cdot y$	схема проводит ток, когда оба переключателя замкнуты
	$F(x)=x \lor y$	схема проводит ток, когда хотя бы один из переключателей замкнут

Логические элементы (вентили)

Логический элемент (логический вентиль) — базовый элемент цифровой схемы, выполняющий элементарную логическую операцию (например: И, ИЛИ, НЕ, НЕ-ИЛИ, НЕ-И и другие). Логический вентиль на схемах изображают прямоугольником. Линии, которые находятся с левой стороны этого прямоугольника, называются входами, а с правой — выходами элемента. Внутри самого прямоугольника изображен указатель логической функции, которую выполняет данный элемент. Условно-графические обозначения (УГО) некоторых логических элементов представлены в табл. 2.11 в соответствии с ГОСТ 2.743-91 «Элементы цифровой техники».

Таблица 2.11

Логические вентили

Операция	**Логический вентиль**	
	УГО	**Название**
Инверсия		НЕ (инвертор) / NOT
Дизъюнкция		ИЛИ / OR
Конъюнкция		И / AND
Стрелка Пирса		ИЛИ-НЕ / NOR
Штрих Шеффера		И-НЕ / NAND
Сумма по модулю 2		Исключающее ИЛИ / XOR
Эквивалентность		Исключающее ИЛИ с инверсией / XNOR

Пример 2.1. Упростить логическое выражение: $F = A \cdot \overline{B} \oplus (B \vee A) \oplus 1$.

Решение.

При проведении преобразований нужно использовать соотношение: $x \oplus 1 = \overline{x}$. Этот результат получается из определения операции Исключающее ИЛИ: $x \oplus y = x \cdot \overline{y} \vee \overline{x} \cdot y$. При подстановке в это выражение *y=1* , получим $x \oplus 1 = \bar{x}$. Используя закон де Моргана, получим: $F = A \cdot \bar{B} \oplus (\overline{B \vee A}) = A \cdot \bar{B} \oplus (\bar{B} \wedge \bar{A})$. Выражая операцию неравнозначности через конъюнкцию, дизъюнкцию и инверсию, получим: $F = A \cdot \bar{B} \cdot (\overline{\bar{B} \cdot \bar{A}}) \vee (\overline{A \cdot \bar{B}}) \cdot \bar{B} \cdot \bar{A} =$

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

***Пример* 2.2.** Задана булева функция от трех переменных: $\mathrm{F(A,B,C)} = (\bar{A} \oplus B) \wedge (A \rightarrow \bar{C})$. Необходимо: 1) построить таблицу истинности в MS Excel без упрощения выражения, используя встроенные логические функции И, ИЛИ, НЕ, ЕСЛИ; 2) упростить логическое выражение или указать его результат (при его однозначности). Результат упрощения может содержать только операции инверсии, конъюнкции и дизъюнкции.

Решение.

1) Сначала необходимо записать все возможные комбинации переменных A, B, C. MS Excel не поддерживает такие логические операции, как «исключающее ИЛИ» и «следование», но их можно выразить через другие допустимые операции. Операцию Xor для данного примера можно заменить операцией =ЕСЛИ (A<>B;1;0), а операцию следования — =ЕСЛИ (И(A=1;C=0);0;1). Чтобы в таблице истинности представить значения «Истина» и «Ложь» как 1 и 0 соответственно, необходимо использовать функцию =Ч(аргумент).

Для отображения формул в ячейках вместо значений можно использовать режим отладки формул (CTRL+~).

A	B	C	F1=not A	F2 = F1 xor B	F3=not F2	F4=not C	F5=A →F4	F6=F3*F5
0	0	0	=Ч(НЕ(A2))	=ЕСЛИ(D2<>B2;1;0)	=Ч(НЕ(E2))	=Ч(НЕ(C2))	=ЕСЛИ(И(A2=1; G2=0);0;1)	=Ч(И(F2;H2))
0	0	1						
0	1	0						
0	1	1						
1	0	0						
1	0	1						
1	1	0						

Рис. 2.1. Таблица истинности для примера 2.2 в режиме отладки формул

После того как записаны формулы для первой комбинации переменных, необходимо выполнить автозаполнение ячеек для остальных комбинаций A, B, C.

На рис. 2.2 приведена таблица истинности для заданного выражения.

A	B	C	F1=not A	F2 = F1 xor B	F3=not F2	F4=not C	F5=A →F4	F6=F3*F5
0	0	0	1	1	0	1	1	0
0	0	1	1	1	0	0	1	0
0	1	0	1	0	1	1	1	1
0	1	1	1	0	1	0	1	1
1	0	0	0	0	1	1	1	1
1	0	1	0	0	1	0	0	0
1	1	0	0	1	0	1	1	0
1	1	1	0	1	0	0	0	0

Рис. 2.2. Таблица истинности для примера 2.2

2) Ниже представлены последовательные действия для упрощения исходного выражения, итоговый результат содержит только базовые логические операции:

$$\mathrm{F(A,B,C)} = \overline{(\overline{A}\oplus B)} \wedge (A \to \overline{C}) = (\overline{\overline{A}\cdot\overline{B} + A\cdot B})\cdot(\overline{A}+\overline{C}) =$$

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

$$= B\cdot\overline{A}+\overline{B}\cdot A\cdot\overline{C}+\overline{A}\cdot B\cdot\overline{C} = \overline{A}\cdot B(1+\overline{C})+\overline{B}\cdot A\cdot\overline{C} = \overline{A}\cdot B+\overline{B}\cdot A\cdot\overline{C}$$

Пример 2.3. Необходимо восстановить логическую функцию трех переменных: если $F(0,0,1) = F(0,1,0) = F(1,0,1) = True$, а для остальных наборов переменных значение функции равно *False.*

Решение.

Вначале нужно восстановить таблицу истинности по заданным условиям, далее — найти строки, в которых F=1 (табл. 2.12).

Таблица 2.12

Таблица истинности для примера 2.3

X_1	X_2	X_3	$F(X_1, X_2, X_3)$	
0	0	0	0	0
0	0	1	1	$\overline{X_1}\cdot\overline{X_2}\cdot X_3$
0	1	0	1	$\overline{X_1}\cdot X_2\cdot\overline{X_3}$
0	1	1	0	0
1	0	0	0	0
1	0	1	1	$X_1\cdot\overline{X_2}\cdot X_3$
1	1	0	0	0
1	1	1	0	0

Это вторая, третья и шестая строки. Для них необходимо записать логические произведения и объединить их операцией логического сложения. В результате искомая логическая функция имеет вид:

$$F(X_1, X_2, X_3) = \overline{X_1} \cdot \overline{X_2} \cdot X_3 + \overline{X_1} \cdot X_2 \cdot \overline{X_3} + X_1 \cdot \overline{X_2} \cdot X_3 .$$

В это выражение вошли произведения для строк, когда значение функции F=1, а вся сумма соответствует совокупности из трех строк. Вклад остальных пяти наборов значений равен нулю.

Пример 2.4. С помощью логических переменных и символов, обозначающих логические операции любое высказывание можно формализовать, т.е. заменить логической формулой. Составить логическое выражение, которое является истинным, если точка с координатами x, y принадлежит заштрихованной области на рис. 2.3 на языке алгебры логики, на языке Pascal и на C++.

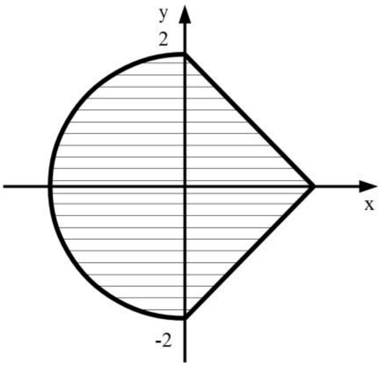


Рис. 2.3. График к примеру 2.4

Решение.

На языке алгебры логики логическое выражение можно записать как: $(x^2 + y^2 \le 4)\ and\ (|\,y\,| \le (2 - x))$, на языке Pascal: $(sqr(x,2) + sqr(y,2) <= 4)\ and (abs(y) <= (2 - x)),$ на C++: $(pow(x,2) + pow(y,2) <= 4)\ \&\&\ (abs(y) <= (2 - x))$.

Пример 2.5. Для заданной переключательной схемы (рис. 2.4) записать функцию проводимости. Построить аналог схемы, упростив функцию проводимости.

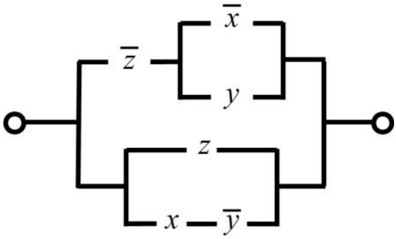


Рис. 2.4. Переключательная схема к примеру 2.5

Решение.

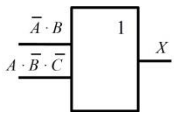
Функция проводимости: $F(x,y,z)=\overline{z}\cdot(\overline{x}\vee y)\vee(z\vee x\cdot\overline{y})=1$. Здесь первое логическое слагаемое $\overline{z}\cdot(\overline{x}\vee y)$ является отрицанием второго логического слагаемого $z\vee x\cdot\overline{y}$, а дизъюнкция переменной с ее инверсией равна 1.

Упрощенная схема: .

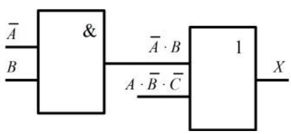
Пример 2.6. Составить логическую схему, соответствующую выражению $X=\overline{A}B+A\overline{B}\overline{C}$.

Решение.

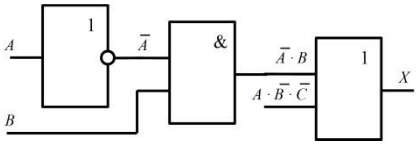
Последняя операция в соответствии с приоритетом логических операций — это логическое сложение, поэтому на выходе схемы будет стоять элемент «ИЛИ»:



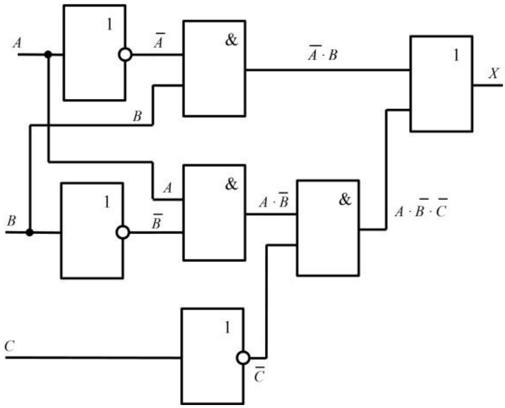
Для того чтобы получить на первом входе $\overline{A}\cdot B$, нужно умножить $\overline{A}$ на B, поэтому необходим элемент «И»:



Чтобы получить $\overline{A}$, необходим элемент «НЕ»:



Аналогично для второй ветви, которая поступает на второй вход элемента «ИЛИ»:



Пример 2.7. Для заданной логической схемы составить структурную формулу.

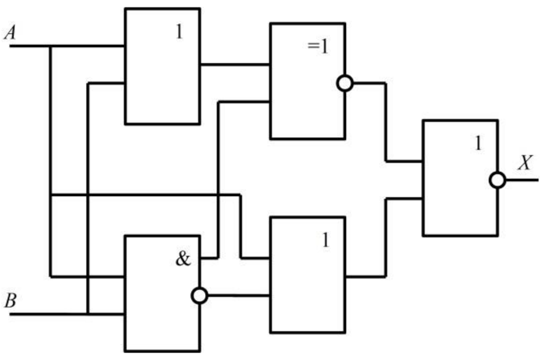


Рис. 2.5. Логическая схема к примеру 2.7

Решение.

Структурная формула для данной схемы имеет вид:

$$X=\overline{((A\vee B)\leftrightarrow(\overline{A\& B}))\vee((\overline{A\& B})\vee A)}\,.$$

Задания для самостоятельного решения

1. Задана булева функция от трех переменных (см. вариант задания в таблице). Построить таблицу истинности в MS Excel без упрощения выражения, используя встроенные логические функции И, ИЛИ, НЕ, ЕСЛИ. Упростить логическое выражение или указать его результат (при его однозначности). Результат упрощения может содержать только операции инверсии, конъюнкции и дизъюнкции.

Варианты заданий

Вариант	Логическое выражение
1.	$(A\wedge B\wedge C\leftrightarrow\overline{B\wedge C})\to AC$
2.	$(A\wedge C\oplus B\wedge C)\leftrightarrow A\wedge B\wedge C$
3.	$\overline{((A\vee B)\wedge(A\vee C))\oplus\overline{A}}$
4.	$(A\overline{B}\vee\overline{A}\vee B)\vee\overline{A\oplus B}\vee C$
5.	$(C\vee B\to A\wedge C)\to(C\wedge A\oplus A\wedge B)$
6.	$(A\to B\ \&\ C)\ \&\ (C\to B\ \&\ A)\ \&\ (B\to C\ \&\ A)$
7.	$(\overline{A}+BC)(\overline{A\to B\leftrightarrow\overline{C}}+AB\to\overline{B}\,\overline{C})$
8.	$\overline{AB\leftrightarrow A\overline{C}}\vee B(A\vee\overline{C}\to AB(\overline{A}\vee\overline{B}\vee AC))$
9.	$A\overline{C}\vee\overline{A}B(C\to AB\leftrightarrow A\vee BC(A\to C))$
10.	$A\vee\overline{BC}\to\overline{A}B\vee\overline{AB\overline{C}\leftrightarrow A\overline{B}\vee C}\to\overline{A}$
11.	$(A\vee\overline{C})(AB\to\overline{A}\vee\overline{B}(AB\overline{C}\leftrightarrow A\vee\overline{B}C))$
12.	$\overline{B}C\leftrightarrow\overline{A}\vee B\to A\overline{C}(\overline{A}\vee B\overline{C}))\cdot\overline{C\to A}$
13.	$\overline{AB\leftrightarrow A\vee\overline{C}}\to\overline{A}\leftrightarrow B\vee\overline{C}(\overline{A}\vee B)$
14.	$\overline{A}C\vee B(\overline{C}\to AB\vee C\leftrightarrow A\vee\overline{B})\to B$
15.	$A\vee\overline{B}\vee C(\overline{\overline{A}\leftrightarrow BC}\vee(B\to A\vee\overline{B\overline{C}}))$
16.	$(\overline{AC}\leftrightarrow\overline{A}\vee\overline{B}\to\overline{C}(A\vee B))\cdot A\vee\overline{C}$
17.	$A\&((\overline{B}\vee\overline{C})\vee\overline{B}\&C)\vee\overline{A}$
18.	$\overline{((A\vee B)\to(\overline{B\vee C}))}$

Вариант	Логическое выражение
19.	$C\&\overline{\overline{A\vee C}\vee\overline{B}}\vee((A\vee C)\&\overline{B})$
20.	$C\vee((A\&C)\vee B)\&\overline{\overline{(A\&C)}\&B}$
21.	$(A\vee B)\to(A\vee(A(B\overline{C}\vee\overline{C})))$
22.	$\overline{(\overline{A}\vee C)(A\vee\overline{B})(B\vee\overline{C})}$
23.	$\overline{\overline{(A\wedge B\oplus B\wedge C)}\vee B}$
24.	$\overline{A\wedge B\wedge C}\leftrightarrow C$
25.	$(\overline{\overline{A\wedge B}\vee C})\leftrightarrow(B\vee\overline{C})$

2. Восстановить логическое выражение по заданной таблице истинности. Упростить полученное выражение.

Варианты заданий

<table>
<tr><th>Вариант</th><th>Задание</th></tr>
<tr><td>1, 14</td><td>
<table>
<tr><td>A</td><td>B</td><td>C</td><td>F(A,B,C)</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>1</td></tr>
</table>
</td></tr>
<tr><td>2, 15</td><td>
<table>
<tr><td>A</td><td>B</td><td>C</td><td>F(A,B,C)</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>0</td></tr>
</table>
</td></tr>
</table>

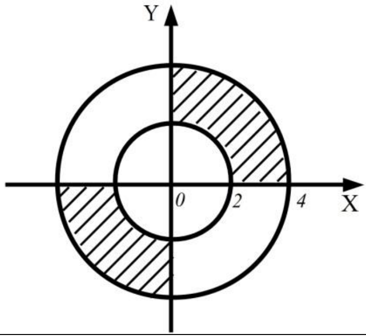
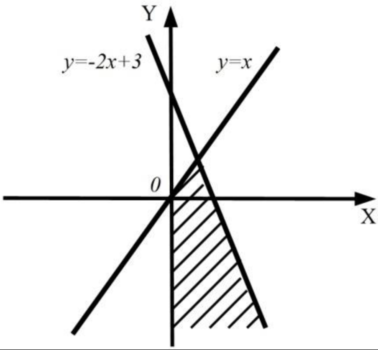
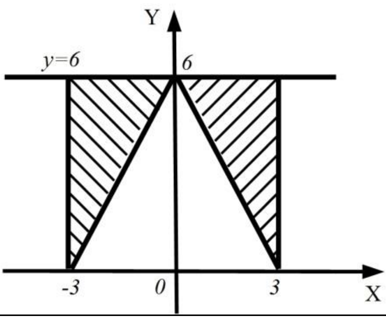
<table>
<tr><th>Вариант</th><th>Задание</th></tr>
<tr><td>3, 16</td><td>
<table>
<tr><td>A</td><td>B</td><td>C</td><td>F(A,B,C)</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>0</td></tr>
</table>
</td></tr>
<tr><td>4, 17</td><td>
<table>
<tr><td>A</td><td>B</td><td>C</td><td>F(A,B,C)</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>1</td></tr>
</table>
</td></tr>
<tr><td>5, 18</td><td>
<table>
<tr><td>A</td><td>B</td><td>C</td><td>F(A,B,C)</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>1</td></tr>
</table>
</td></tr>
<tr><td>6, 19</td><td>
<table>
<tr><td>A</td><td>B</td><td>C</td><td>F(A,B,C)</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>0</td></tr>
</table>
</td></tr>
</table>

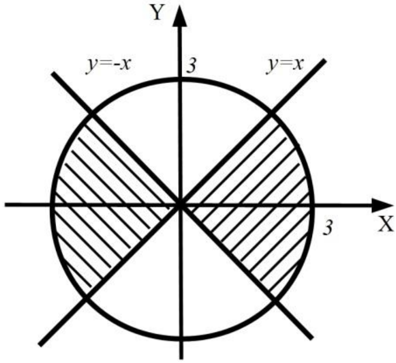
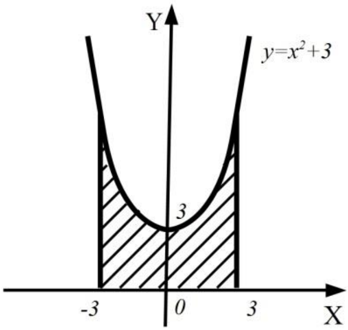
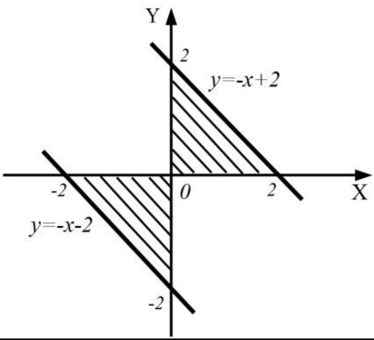
<table>
<tr><th>Вариант</th><th>Задание</th></tr>
<tr><td>7, 20</td><td>
<table>
<tr><td>A</td><td>B</td><td>C</td><td>$F(A,B,C)$</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>1</td></tr>
</table>
</td></tr>
<tr><td>8, 21</td><td>
<table>
<tr><td>A</td><td>B</td><td>C</td><td>$F(A,B,C)$</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>0</td></tr>
</table>
</td></tr>
<tr><td>9, 22</td><td>
<table>
<tr><td>A</td><td>B</td><td>C</td><td>$F(A,B,C)$</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>1</td></tr>
</table>
</td></tr>
<tr><td>10, 23</td><td>
<table>
<tr><td>A</td><td>B</td><td>C</td><td>$F(A,B,C)$</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>0</td></tr>
</table>
</td></tr>
</table>

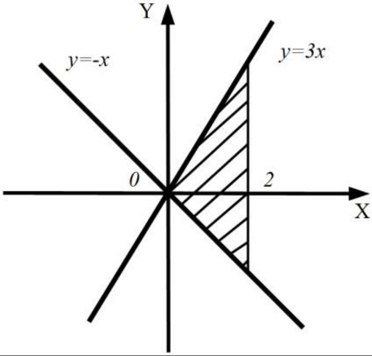
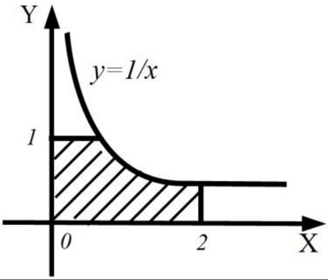
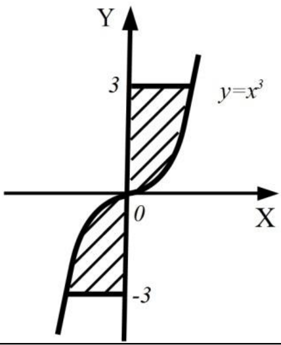
<table>
<tr><th>Вариант</th><th>Задание</th></tr>
<tr><td>11, 24</td><td>
<table>
<tr><td>A</td><td>B</td><td>C</td><td>F(A,B,C)</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>1</td></tr>
</table>
</td></tr>
<tr><td>12, 25</td><td>
<table>
<tr><td>A</td><td>B</td><td>C</td><td>F(A,B,C)</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>1</td></tr>
</table>
</td></tr>
<tr><td>13, 26</td><td>
<table>
<tr><td>A</td><td>B</td><td>C</td><td>F(A,B,C)</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>0</td></tr>
</table>
</td></tr>
</table>

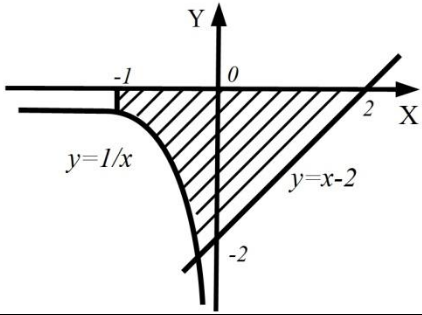
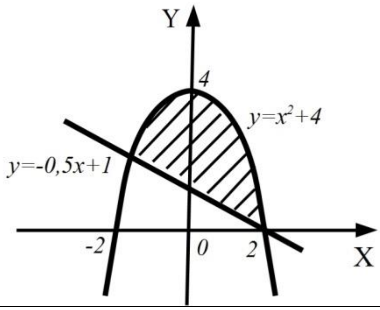
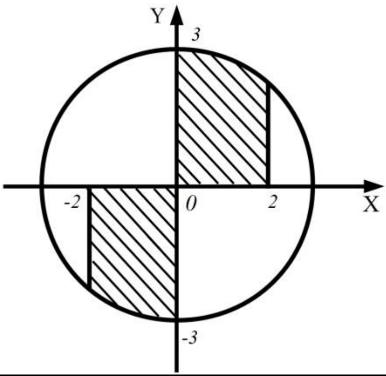
3. Составить логические выражения, которые являются истинными, если точка с координатами *x*, *y* принадлежит заштрихованным областям на языке алгебры логики, на языках Pascal и на C++.

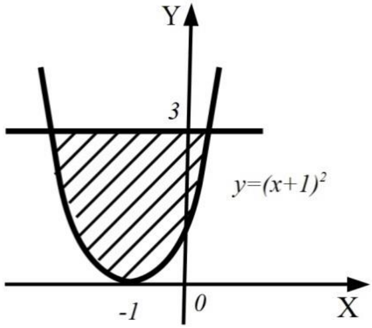
Варианты заданий

Вариант	Задание
1, 14	
2, 15	
3, 16	

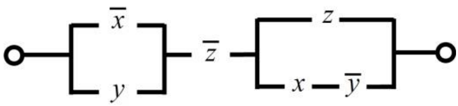
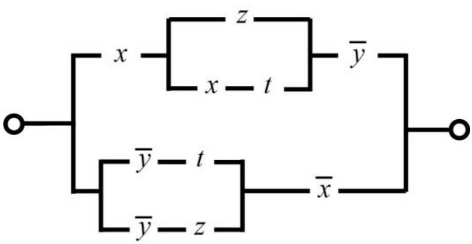
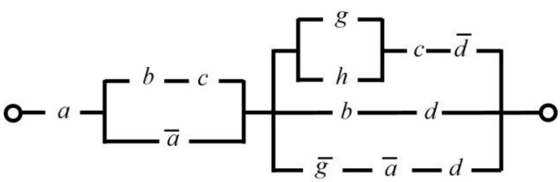
Вариант	Задание
4, 17	
5, 18	
6, 19	

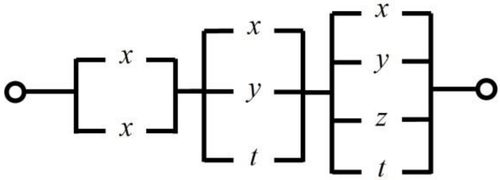
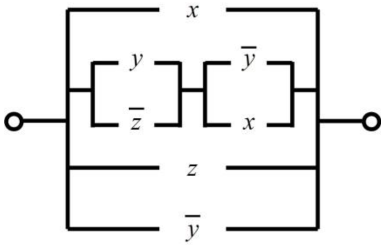
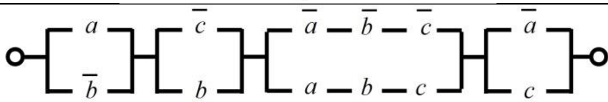
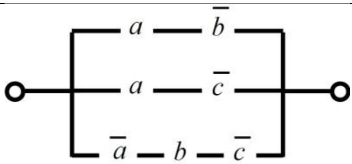
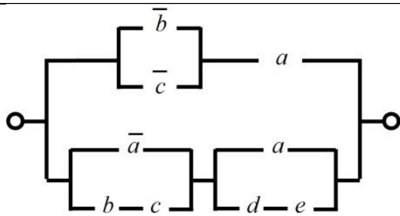
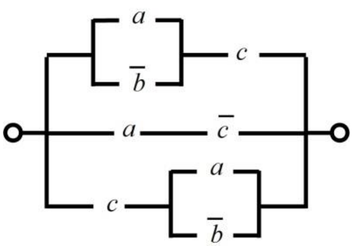
Вариант	Задание
7, 20	
8, 21	
9, 22	

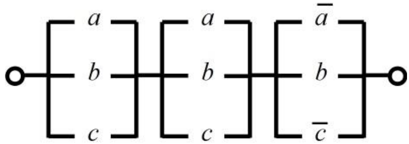
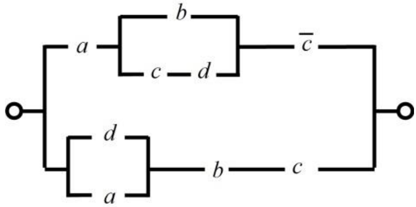
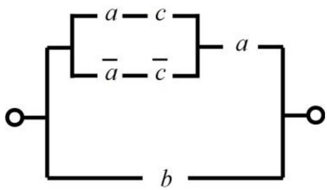
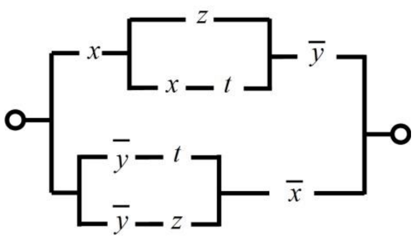
Вариант	Задание
10, 23	
11, 24	
12, 25	

Вариант	Задание
13, 26	

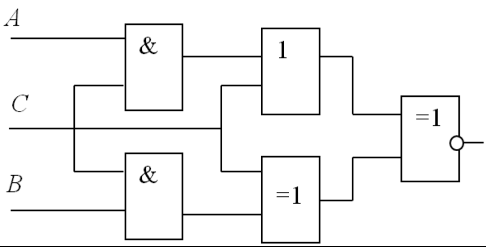
4. Для заданной переключательной схемы записать функцию проводимости. Построить аналог схемы, упростив функцию проводимости.

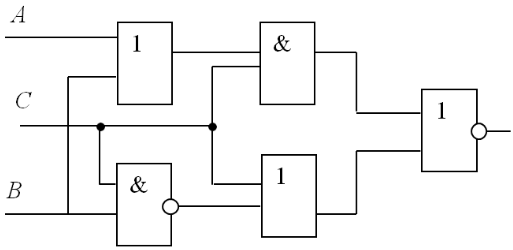
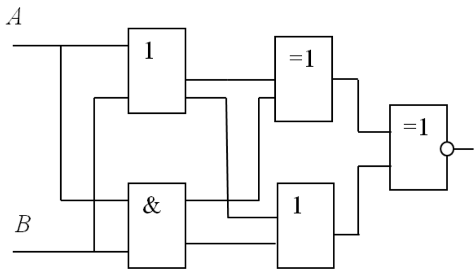
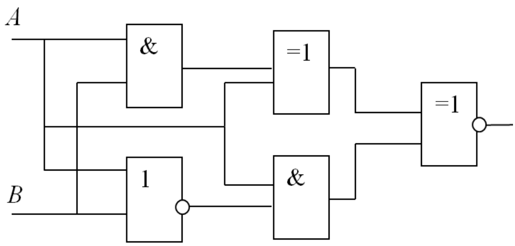
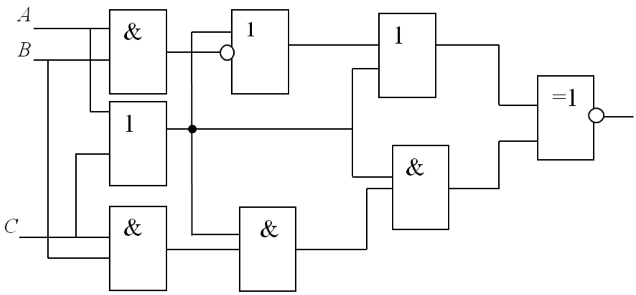
Вариант	Задание
1, 14	
2, 15	
3, 16	

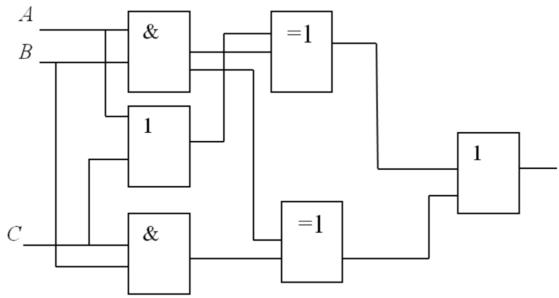
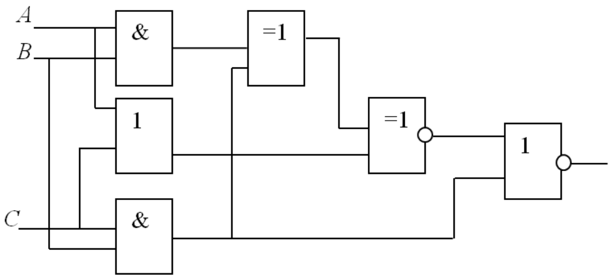
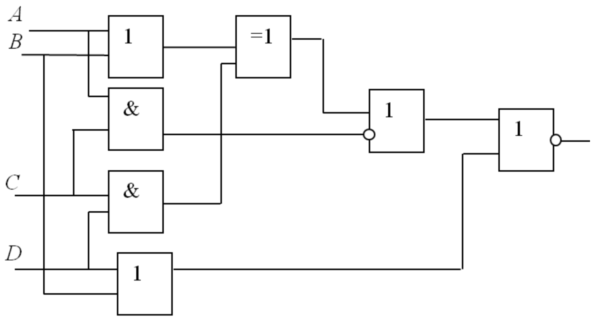
Вариант	Задание
4, 17	
5, 18	
6, 19	
7, 20	
8, 21	
9, 22	

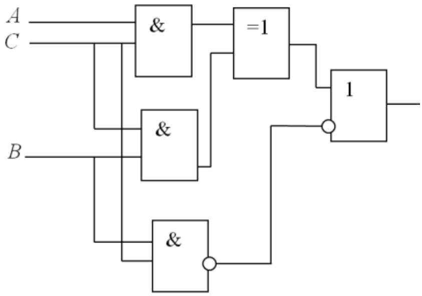
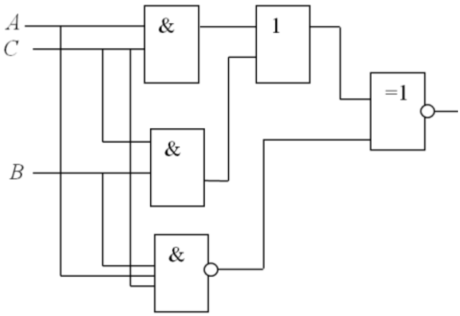
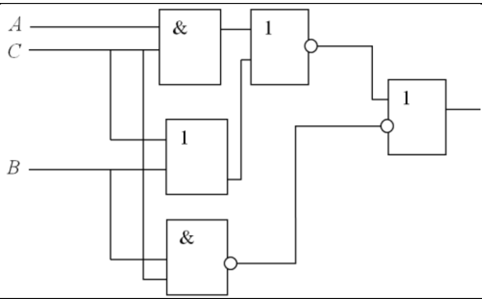
Вариант	Задание
10, 23	
11, 24	
12, 25	
13, 26	

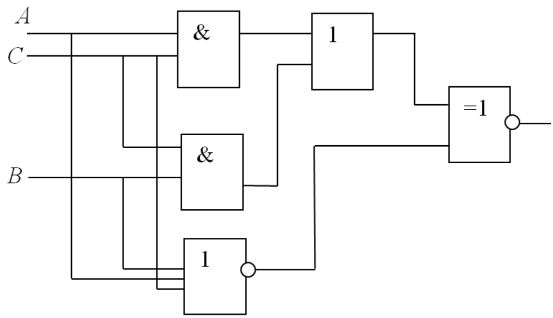
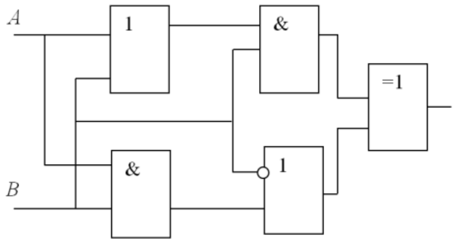
5. Определить структурную формулу для данных логических схем.

Вариант	Задание
1, 14	

Вариант	Задание
2, 15	
3, 16	
4, 17	
5, 18	

Вариант	Задание
6, 19	
7, 20	
8, 21	

Вариант	Задание
9, 22	
10, 23	
11, 24	

Вариант	Задание
12, 25	
13, 26	

Приложение

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«Белгородский Государственный Технологический Университет им. В.Г. Шухова»
(БГТУ им. В.Г. Шухова)

Институт энергетики информационных технологий и управляющих систем

Кафедра: «Электроэнергетика и автоматика»

Расчетно-графическое задание
по дисциплине: «Информатика»

Выполнил:
студент группы Э-11
Иванов И.И.

Принял:
Преподаватель:

Белгород, 2018 г.

Список литературы

1. *Акулов, О.А.* Информатика: базовый курс. Учебник / О.А. Акулов, Н.В. Медведев. — 2-е изд., испр. и доп. — М.: Омега, 2005. — 550 с.
2. Сборник задач по дисциплине «Информатика» для ВУЗов. Методические указания к проведению практических занятий по дисциплине «Информатика», для студентов первого курса специальностей 10.03.01 и 10.05.02. — М: СОЛОН-Пресс, 2016. — 104 с. — ISBN 978-5-91359-170-8.
3. *Алексеев, А.П.* Информатика 2015 [Электронный ресурс]: учебное пособие/ А.П. Алексеев — Электрон. текстовые данные.— М.: СОЛОН-ПРЕСС, 2015. — 400 с. — Режим доступа: http://www.iprbookshop.ru/53821.html.— ЭБС «IPRbooks».
4. Практикум по информатике [Электронный ресурс]: учебное пособие/ О.Г. Иванова [и др.].— Электрон. текстовые данные.— Тамбов: Тамбовский государственный технический университет, ЭБС АСВ, 2014. — 112 с. — Режим доступа: http://www.iprbookshop.ru/63891.html. — ЭБС «IPRbooks».
5. *Стативко, Р.У.* Информатика: учеб. пособие/ Р.У. Стативко. – Белгород: изд-во БГТУ, 2017. — 216 с.
6. *Сальникова, Н.А.* Информатика. Основы информатики. Представление и кодирование информации. Часть 1 [Электронный ресурс]: учебное пособие/ Н.А. Сальникова. — Электрон. текстовые данные.— Волгоград: Волгоградский институт бизнеса, Вузовское образование, 2009. — 94 с. — Режим доступа: http://www.iprbookshop.ru/11321.html. — ЭБС «IPRbooks».
7. *Чернова, С.Б.* Информатика. Программирование в среде PascalABC.NET. Лабораторный практикум // Чернова С.Б., Старченко Д.Н. — Белгород: Изд-во БГТУ, 2015. — 90 с.

Учебное издание

Методические указания

к выполнению расчетно-графического задания
по дисциплине «Информатика»
для студентов направления бакалавриата
130302 – Электроэнергетика и электротехника

Составители: Рыбин Илья Александрович
Рыбина Анна Васильевна
Анисимова Зинаида Григорьевна

Подписано в печать Формат . Усл. печ. л . Уч.-изд. л .
Тираж экз. Заказ Цена
Отпечатано в Белгородском государственном технологическом университете
им. В.Г. Шухова
308012, г. Белгород, Костюкова, 46