Правительство Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего профессионального образования «Национальный исследовательский университет «Высшая школа экономики»

Московский институт электроники и математики Национального исследовательского университета «Высшая школа экономики»

Кафедра информационных технологий и автоматизированных систем

ИНФОРМАТИКА

Учебно-методическое пособие для практических занятий

Москва 2013

Составители: канд. техн. наук И.В. Назаров
канд. техн. наук Т.А. Потапова

УДК 519.6

Информатика: Учебно-метод. пособие для практ. занятий по курсам «Информатика» и «Информационные технологии» / Московский институт электроники и математики Национального исследовательского университета «Высшая школа экономики»; Сост: И.В. Назаров, Т.А. Потапова. М., 2013. — 24 с.
Библиогр.: 5 назв.

Содержит необходимые сведения по системам счисления, булевой алгебре и основам цифровой электроники для практических занятий и самостоятельного изучения некоторых общих вопросов курсов «Информатика» и «Информационные технологии», а также для подготовки к домашним и контрольным работам.

Для бакалавров направлений 210700.62 «Инфокоммуникационные технологии и системы связи» и 211000.62 «Конструирование и технология электронных средств» I курса Факультета электроники и телекоммуникаций, изучающих информатику и информационные технологии.

ISBN 978-5-94506-311-2

ПРЕДИСЛОВИЕ

В настоящее время бакалавр инженерного направления подготовки не может быть современным специалистом без знания программирования. Это предполагает не столько использование языков программирования, сколько умение логически и алгоритмически мыслить, формализовать постановку задачи, находить реальные пути ее решения.

Рассмотрены три основных раздела, являющиеся базовыми в информатике и информационных технологиях. Системы счисления и булева алгебра – азбука информатики. Преобразования из одной системы счисления в другую, двоичная арифметика, логические операции, представление о логических элементах цифровых схем – необходимые составляющие квалификации специалистов самых различных областей информационных технологий: от создателей систем на основе микроконтроллеров до программистов игр и разработчиков Web-сайтов.

1. СИСТЕМЫ СЧИСЛЕНИЯ

1.1. Основные определения

Система счисления – это набор знаков (цифр и букв) для представления чисел и арифметических действий с ними.

Классификация систем счисления.

1. Позиционные системы счисления.

В позиционных системах счисления значение (вес) каждой цифры в числе зависит от ее положения, т.е. каждая позиция в числе характеризуется весом. Например, 1, 10, 100, 1000 (10^0,10^1,10^2,10^3) называются единицами, десятками, сотнями, тысячами в зависимости от положения цифры 1 в десятичном числе.

2. Непозиционные системы счисления.

В непозиционных системах счисления значение цифры не зависит от ее позиции в числе. К непозиционным системам счисления относится, например, римская система счисления, в которой цифра образуется из черточек: III – 3, V – 5, X – 10, C – 100, L – 500, M – 1000. Арабское число 32 в римской системе счисления имеет вид XXXII. Десятки в числе отражает цифра X, значение которой не зависит от ее позиции в числе и всегда равно 10.

Рассмотрим представление числа в позиционной системе счисления:

$324_{10}=(3\times100)+(2\times10)+(4\times1)=3\times10^2+2\times10^1+4\times10^0$.

Для представления десятичного числа больше 9 требуются дополнительные цифры в позиции «десятков», «сотен» и т.д. То есть, каждая добавляемая слева цифра имеет вес в 10 раз больший, чем соседняя цифра справа.

Рассмотрим пример для двоичной системы счисления:

$101_2=1\times2^2+0\times2^1+1\times2^0$.

Чтобы определить значение любой заданной позиции двоичной цифры – бита (бит: binary digit (bit) – наименьший квант информации) при

конструировании двоичного числа, нужно удвоить вес предшествующей позиции, начиная с младших разрядов.

Рассмотрим представление двоичного числа:

Позиция цифры в числе	10	9	8	7	6	5	4	3	2	1	0
Биты	1	1	1	1	1	1	1	1	1	1	1
Степень числа 2	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Десятичное значение	1024	512	256	128	64	32	16	8	4	2	1

В примере рассмотрено двоичное число:
1111111111_2=1024+512+…+2+1=2047.

Основные понятия позиционной системы счисления.

1. Основание системы счисления – это цифра или число, на котором строится система счисления и которая отражает набор знаков (цифр, букв), существующих в системе счисления.

Если основание системы счисления больше 9 и цифр для представления числа не хватает, то используются латинские буквы: A, B и т.д., например, для образования числа в шестнадцатеричной системе счисления – буквы от A до F.

Рассмотрим представление шестнадцатеричного числа:

Десятичное число	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
16-ричное число	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Двоичное число	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111

2. Вес позиции цифры в числе – это соотношение между соседними позициями в числе. Вес позиции показывает, во сколько раз позиция слева в числе больше («тяжелее») позиции справа. Или, более строгое определение: степень основания системы счисления в каждой позиции на единицу больше, чем в предыдущей позиции.

Сравним веса систем счисления и десятичные веса. Каждая позиция цифры в числе имеет два веса: вес системы счисления и десятичный вес:

Позиция цифры в числе (разряд)	Десятичная система счисления		Двоичная система счисления		Шестнадцатеричная система счисления	
	Вес в системе счисления	Десятичный вес	Вес в системе счисления	Десятичный вес	Вес в системе счисления	Десятичный вес
0	10^0	1	2^0	1	16^0	1
1	10^1	10	2^1	2	16^1	16
2	10^2	100	2^2	4	16^2	256
3	10^3	1000	2^3	8	16^3	4096
4	10^4	10000	2^4	16	16^4	65536

1.2. Способы преобразования чисел из одной системы счисления в другую

Существуют несколько способов преобразования чисел из одной системы счисления в другую.

1. Деление на основание системы счисления. Остатки от деления являются цифрами числа в новой системе счисления, начиная с младших разрядов.

Пример. Преобразовать десятичное число 247 в двоичное число:

247	2							
–246	123	2						
1	–122	61	2					
	1	–60	30	2				
		1	–30	15	2			
			0	–14	7	2		
				1	–6	3	2	
					1	–2	1	
						1		

Ответ: $11110111_2 = 247_{10}$.

Проверка: $11110111_2 = 1\times2^7+1\times2^6+1\times2^5+1\times2^4+1\times2^3+1\times2^2+1\times2^1+1\times2^0=$
$=128+64+32+16+4+2+1=247_{10}$.

Пример. Преобразовать десятичное число 2476 в шестнадцатеричное число:

2476	16		
–2464	154	16	
12	–144	9	
	10		

Ответ: $2476_{10} = 9AC_{16}$.

Проверка: $9AC_{16}=9\times16^2+A\times16^1+C\times16^0=9\times256+10\times16+12\times1=2304+172=2476_{10}$.

2. Вычитание наибольшего десятичного веса. В полученном двоичном числе в позициях, соответствующих вычитаемому наибольшему десятичному весу, находятся единицы, в остальных позициях – нули.

Пример. Преобразовать десятичное число 97 в двоичное число:

Преобразование числа	Двоичный вес	Номер позиции
97		
–64	$=2^6$	6
33		
–32	$=2^5$	5
1	$=2^0$	0

Ответ: $97_{10}=1100001_2$.

Проверка: $1\times2^6+1\times2^5+1\times2^0=64+32+1=97_{10}$.

3. По четности десятичного числа. Если число четное, то в соответствующей позиции двоичного числа будет единица, если нечетное – нуль. В ответе двоичное число записывается, начиная с младших разрядов.

Пример. Преобразовать десятичное число 275 в двоичное число:

275	1
137	1
68	0
34	0
17	1
8	0
4	0
2	0
1	1

Ответ: $275_{10}=100010011_2$.
Проверка:$100010011_2= 1\times2^8+1\times2^4+1\times2^1+1\times2^0=256+16+2+1=275_{10}$.

Легко заметить связь между представлением чисел в двоичной и шестнадцатеричной системах счисления: каждая шестнадцатеричная цифра эквивалентна группе из четырех двоичных цифр, т.к. число $16_{10}=10_{16}=10000_2$ (см. выше). Таким образом, преобразование из двоичной системы счисления в шестнадцатеричную систему можно осуществить разбиением двоичного числа на группы по четыре разряда. Если в группе старших разрядов (слева) не хватает цифр, то приписать слева необходимое количество нулей, затем представить каждую группу в виде шестнадцатеричной цифры.

Пример. Преобразовать двоичные числа в шестнадцатеричные:
$0001\ 1001\ 1100_2 = 19C_{16}$, $0100\ 0001\ 1100_2 = 41C_{16}$.

Пример. Преобразовать шестнадцатеричные числа в двоичные:
$A3F_{16} = 1010\ 0011\ 1111_2$, $123_{16} = 0001\ 0010\ 0011_2$.

Аналогичным образом можно преобразовывать двоичные числа в восьмеричные и обратно. При этом двоичное число разбивается на группы по три разряда.

Можно сформулировать общие правила преобразования чисел из одной системы счисления с другую систему.

1. Чтобы преобразовать десятичное число в соответствующее число в любой системе счисления, нужно десятичное число делить на основание системы счисления до тех пор, пока в остатке не окажется цифра, существующая в данной системе счисления. Все остатки от деления представляют собой цифры в позициях числа новой системы счисления, начиная с младших разрядов.
2. Чтобы преобразовать число из любой системы счисления в десятичное число, нужно его вначале представить в виде суммы весов данной системы счисления, затем в виде суммы десятичных весов и эти веса сложить.
3. Преобразование числа любой системы счисления в любую систему нужно проводить, используя десятичную систему счисления.

1.3. Двоичная арифметика

В двоичной арифметике рассмотрим три операции: сложение, вычитание и операцию сдвига.

1. Сложение двоичных чисел производится, как и десятичных чисел, путем переноса избытка в старший разряд.

Пример:

```
 1111      15
+0111     + 7
10110      22
```

Проверка: $10110_2=16+4+2=22_{10}$.

Правила двоичного сложения отражены в таблице истинности:

Операнд 1	Операнд 2	Сумма	Перенос в старший разряд
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

2. Вычитание двоичных чисел можно выполнить двумя способами.

Путем заема из старшего разряда, аналогично тому, как это происходит при десятичном вычитании. Правила вычитания отражены в таблице истинности:

Операнд 1	Операнд 2	Разность	Заем из старшего разряда
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Пример:

```
  0111
 _100001     _33
  11110      30
 ------     ---
 000011      3
```

Второй способ называется дополнением до единицы с циклическим переносом.

Прежде чем привести пример вычитания двоичных чисел вторым способом, дадим пояснения его названия. В этом способе используются две операции. Операция «дополнение до 1» представляет собой замену всех нулей в двоичном числе на единицы и наоборот. Операция «циклический перенос» заключается в переносе единицы из самого старшего разряда для сложения с младшим разрядом того же числа. Результат этих двух операций и будет результатом двоичного вычитания.

Рассмотрим тот же самый пример:

```
 _100001      100001
  11110     +  00001
 ------     --------
 000011      100010
           +       1
           --------
            000011
```

Обратим внимание на особенности второго способа вычитания двоичных чисел:

1) Вместо вычитания выполняются две операции сложения.
2) Операция «дополнение до единицы» выполняется над вычитаемым, при этом уменьшаемое остается без изменений.
3) При циклическом переносе единицы в старшем разряде промежуточной суммы (**1**00010) записывается **0** (**0**00011).

3. Операция сдвига.

Эту операцию можно наблюдать на индикаторе калькулятора, когда при наборе каждой новой цифры происходит сдвиг предыдущих цифр влево. Цифры можно сдвигать как влево, так и вправо. Соответственно, существуют две операции сдвига. Рассмотрим операции сдвига в пределах одного байта (байт: 8-битовая единица информации).

Сдвиг влево: каждый бит байта сдвигается на один разряд влево, при этом крайний левый исчезает, а в крайний правый записывается нуль.

7	6	5	4	3	2	1	0
0	**0**	**0**	**0**	**0**	**1**	**1**	**1**

$=1\times2^0+1\times2^1+1\times2^2=1+2+4=7_{10}$

7	6	5	4	3	2	1	0
0	**0**	**0**	**0**	**1**	**1**	**1**	**0**

$=1\times2^1+1\times2^2+1\times2^3=2+4+8=14_{10}$

7	6	5	4	3	2	1	0
0	**0**	**0**	**1**	**1**	**1**	**0**	**0**

$=1\times2^2+1\times2^3+1\times2^4=4+8+16=28_{10}$

Видно, что сдвиг влево на один разряд эквивалентен умножению исходного числа на 2 (на основание системы счисления).

Сдвиг вправо: каждый бит байта сдвигается на один разряд вправо, при этом крайний правый исчезает, а в крайний левый записывается нуль.

7	6	5	4	3	2	1	0
1	**1**	**1**	**0**	**0**	**0**	**0**	**0**

$=1\times2^7+1\times2^6+1\times2^5=128+64+32=224_{10}$

7	6	5	4	3	2	1	0
0	**1**	**1**	**1**	**0**	**0**	**0**	**0**

$=1\times2^6+1\times2^5+1\times2^4=64+32+16=112_{10}$

7	6	5	4	3	2	1	0
0	**0**	**1**	**1**	**1**	**0**	**0**	**0**

$=1\times2^5+1\times2^4+1\times2^3=32+16+8=56_{10}$

Сдвиг вправо на один разряд эквивалентен делению исходного числа на 2 (на основание системы счисления).

1.4. Числа со знаком

Как персональный компьютер работает с отрицательными числами?

Если байт или машинное слово (машинное слово: единица данных длиной несколько байтов в зависимости от разрядности процессора; 2, 4 и более байт) содержит число со знаком, то для старшего байта в слове (в случае нескольких байтов) его значение представляется только младшими битами (6-0), а старший бит (7) указывает знак числа и называется знаковым битом. Если число положительное, знаковый бит равен нулю, а если отрицательное, то знаковый бит равен единице.

Рассмотрим двоичное число размером один байт. В числе без знака все биты представляют значение числа. В числе со знаком старший бит вклад в значение числа не вносит, а содержит признак «отрицательности» числа:

7	6	5	4	3	2	1	0	
1	**1**	**1**	**0**	**0**	**0**	**1**	**0**	– число без знака (=226_{10})

7	6	5	4	3	2	1	0	
0	**1**	**1**	**0**	**0**	**0**	**1**	**0**	– положительное число со знаком (= 98_{10})

7	6	5	4	3	2	1	0	
1	**1**	**1**	**0**	**0**	**0**	**1**	**0**	– отрицательное число со знаком (= -30_{10})

Последнее полученное значение будет объяснено ниже.

Значение «минус единица» в двоичном виде представляется как 11111111, а не 10000001. Это произошло потому, что нуль не может иметь два различных представления:

+0 = 00000000,

–0 = 10000000.

Чтобы нуль имел только одно двоичное представление, для отрицательных чисел был разработан дополнительный код.

Для положительных чисел дополнительный код и обычное двоичное представление числа совпадают.

Чтобы найти дополнительный код для отрицательных чисел, нужно выполнить следующие действия.

1. Вычислить двоичный эквивалент десятичного числа.
2. Обратить каждый бит, т.е. заменить единицы нулями, а нули единицами (операция «дополнение до 1»).
3. К полученному результату прибавить единицу путем обычного двоичного сложения.

Рассмотрим пример для чисел размером один байт. Получим дополнительный код нескольких чисел со знаком (обратите внимание на знаковый (старший) бит байта для положительных и отрицательных чисел):

Десятичное число со знаком	Двоичное число в дополнительном коде
+7	00000111
+6	00000110
+5	00000101
+4	00000100
+3	00000011
+2	00000010
+1	00000001
0	00000000
–1	11111111
–2	11111110
–3	11111101
–4	11111100
–5	11111011
–6	11111010
–7	11111001

Сделаем обратное преобразование: получим десятичное число по его дополнительному коду.

Число в дополнительном коде:

11110000

00001111 (дополнение до 1)

\+ 1

00010000 $= 2^4 = 16_{10}$, но т.к. знаковый бит исходного числа равен 1, то ответ: $11110000 = -16_{10}$.

1.5. Применение двоичной и шестнадцатеричной систем счисления

Основное применение двоичной системы счисления – обработка, хранение и передача информации в компьютерных системах. Информацию можно разделить на два основных представления: числовую и символьную.

Объем информации измеряют в байтах (8 бит), килобайтах ($1К=1024=2^{10}$ байт), мегабайтах ($1М=1024К=1024\times1024=1048576=2^{20}$ байт), гигабайтах ($1Г=1024М=1024\times1048576=1073741824=2^{30}$ байт) и т.д.

В программах на языках программирования принято соглашение для идентификации информации использовать различные типы данных (целые, вещественные, символьные, логические). Каждый объект программы (переменная, константа, функция) должен принадлежать к одному из этих типов данных и быть явно описан в программе. Цель этого описания – зафиксировать диапазон значений, которые могут принимать объекты программы, и соответствующий объем выделяемой для них памяти.

Например, переменные целого типа, т.е. предназначенные для хранения целых чисел, обычно занимали в памяти два байта. Следовательно, диапазон представления (максимальное количество чисел, которые можно разместить в переменной этого объема памяти): от 0 до 65536 ($=2^{16}$). Однако, если имеется в виду целое число со знаком (см. выше), то максимальное десятичное значение, которое можно разместить в переменной целого типа, уменьшается в два раза:

от –32768 до 32767. Это происходит вследствие использования старшего бита старшего байта для хранения знака числа.

Переменные целого типа в языках программирования могут занимать разное количество байтов. Соответственно, диапазон десятичных значений числа, хранимого в переменной данного типа, будет увеличиваться (напр., при 4 байтах: от -2 147 483 648 до 2 147 483 647) или уменьшаться (напр., при 1 байте: от –128 до 127).

Переменные вещественного типа предназначены для хранения вещественных чисел. В общем случае, вещественные числа в памяти компьютера не могут быть представлены точно, а только с некоторой степенью точности. Это связано с существованием иррациональных чисел в математике, с одной стороны, и ограниченным объемом памяти компьютера – с другой.

Для хранения вещественных чисел принят специальный формат. Любое вещественное число можно представить и хранить как два целых числа со знаком, т.е. в виде: $\pm M \times 10^{\pm P}$, где M – мантисса (первые значащие цифры числа), P – порядок.

Например, в первой версии языка Си переменные вещественного типа занимали 4 байта. Часть из этих 32 битов отводились для хранения числа со знаком, являющегося мантиссой, а часть – для хранения числа со знаком, являющегося порядком. Количество битов, отведенных под мантиссу, определяет точность представления вещественного числа в памяти компьютера (количество значащих цифр), а количество битов, отведенных под порядок, – диапазон степени числа.

Если вещественные числа занимают 4 байта, то точность числа составляет 6–7 значащих цифр, а порядок числа: от –38 до 38. При увеличении объема памяти, отводимого для вещественных чисел (напр., до 8 байтов), точность числа увеличивается до 12–16 значащих цифр, а порядок – до ±302.

Символьная информация хранится в переменных «символьного» типа в закодированном виде. Т.е. каждому символу (букве, цифре, пробелу, знаку препинания, спецсимволу и т.д.) поставлено в соответствие целое число без знака. Соответствие определено в таблице, называемой кодовой. Одной из первых стандартных кодовых таблиц была таблица ASCII (American Standard Code for Information Interchange), в которой каждому символу присвоен код длиной один байт. Следовательно, максимальное количество кодируемых символов составляет 256, т.е. коды – это целые числа в диапазоне от 0 до 255_{10}. Так для прописных букв латинского алфавита выделен диапазон кодов от 65 до 90, для строчных букв – от 97 до 122, для цифр (символы 0…9) – от 48 до 57. Для букв национальных алфавитов (напр., кириллицы) отведен диапазон кодов от 128 до 255. Существуют и другие таблицы кодирования, использующие для одного символа большее количество байтов, напр., ANSI, Unicode, применяющиеся в Windows.

Таким образом, наблюдается тенденция увеличения объема памяти для хранения переменной любого типа с увеличением разрядности процессора и общего увеличения объема оперативной памяти компьютеров.

1.6. Вопросы для самопроверки

1. Что такое система счисления? Перечислить известные.
2. Каково представление числа в позиционной системе счисления?
3. Что такое основание и вес системы счисления?
4. Какие существуют способы преобразования числа из одной системы счисления в другую?
5. Правила сложения двоичных чисел.
6. Правила вычитания двоичных чисел.
7. Что такое операция сдвига?
8. Какое представление имеют отрицательные числа в памяти компьютера?
9. Как найти дополнительный код для двоичного числа?

2. ОСНОВЫ БУЛЕВОЙ АГЕБРЫ

2.1. Основные определения

В 1854 г. Дж. Булем был разработан формальный математический аппарат алгебры логики, получивший в дальнейшем широкое применение.

Рассмотрим вначале общее определение алгебраической системы. Она может состоять из трех множеств: множества элементов, множества операций и множества постулатов (законов).

Понятие «множество» в математике одно из фундаментальных, дадим нестрогое определение множества. Множество – это совокупность элементов, отвечающих условию принадлежности.

Например, (Z – множество целых чисел, R – множество вещественных чисел, C – множество комплексных чисел). Множество Z – это бесконечное множество. Элементы такого множества можно задать с помощью бесконечной последовательности: $Z=\{0,\pm1,\pm2,\pm3\ldots\}$. Для конечных счетных множеств элементы задаются в виде списка. Например: пусть S – это множество всех простых чисел между 20 и 40. Тогда $S=\{23, 29, 31, 37\}$. Как видно из примера, список имеет конечное число элементов.

Подмножеством S множества T называется любое множество, все элементы которого принадлежат T (если $a\in S$, то $a\in T$, или $S\subset T$).

Множество целых чисел Z состоит из подмножеств положительных чисел P, неотрицательных чисел N, отрицательных чисел E, поэтому можно записать: $P=\{1,2,3\ldots\}$; $N=\{0,1,2,3\ldots\}$, т.е., если $a\in P$ и $a\in N$, то $P\subset N\subset Z$, и говорят, что P содержится (включается) в N и N включается в Z. Таким образом, множество Z содержит следующие подмножества: P, N, E, Ø, Z, или $Z:\Re(Z)$, где $\Re(Z)$ – есть множество подмножеств множества Z.

Множество подмножеств (частей) данного множества U обозначают через $\Re(U)$. Оно содержит само множество U, пустое множество Ø и некоторое количество (2^n-2) собственных подмножеств S, удовлетворяющих условию $Ø\subset S\subset U$, $S\neq Ø, S\neq U$.

Рассмотрим теперь множество операций в алгебраической системе. Для множества целых чисел Z определены операции сложения и умножения, и правила их выполнения. Поэтому алгебраическую систему, образованную множеством целых чисел, можно записать в виде [Z,+,*]. Школьная элементарная алгебра, оперирующая вещественными числами, может быть определена как алгебраическая система вида [R,+,*]. Это один класс алгебраических систем.

К булевым алгебрам относится класс алгебраических систем, которые имеют вид $[\Re(U),\cap,\cup,']$. В данном классе алгебраических систем определены три теоретико-множественные или булевы операции: пересечение, объединение и дополнение. Все подмножества множества U, для которых определены булевы операции, называются булевыми алгебрами.

Рассматривается двоичная булева алгебра, которая является формальным математическим аппаратом построения аппаратной логики компьютеров и используется в программировании.

2.2. Двоичная булева алгебра

Рассмотрим последовательно множества, характеризующие двоичную булеву алгебру как алгебраическую систему: множество элементов, множество операций и множество постулатов.

Множество элементов.

Пусть U = {a} – одноэлементное множество, тогда булева алгебра имеет вид: $[\Re(\{a\}),\cap,\cup,']$. Формально множество U имеет один элемент {a} и два подмножества (U и Ø). Поэтому можно ввести новое обозначение: U=1, Ø=0. Таким образом, в двоичной булевой алгебре будем говорить о двух элементах «1» и «0», которые еще называются логической единицей и логическим нулем (или логическими константами). В аппаратной логике эти константы соответствуют двоичным цифрам 1 и 0, а в программировании логическая единица эквивалентна выполнению некоторого условия, а логический нуль – его не выполнению.

Множество операций.

1. Операция «Пересечение». Обозначение: $\cap$ ($\wedge$).

Пусть A и B – подмножества множества U. Их пересечением ($A\cap B$) называется множество всех элементов, принадлежащих как A, так и B:

$$A\cap B=\{x \mid x\in A,\ x\in B\},\ A\cap\varnothing=\varnothing$$

В двоичной булевой алгебре эта операция носит название логического умножения (конъюнкции) или логической операции **И**. Путь x и y – логические переменные. Операции над переменными задаются с помощью таблиц, которые называются таблицами истинности.

Для операции **И** имеем:

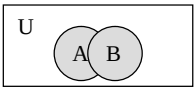
X	Y	XY
0	0	0
0	1	0
1	0	0
1	1	1

Результатом логического умножения является значение логической единицы тогда и только тогда, когда значения всех логических переменных равны логическим единицам.

2. Операция «Объединение». Обозначение: $\cup$ ($\vee$).

Пусть A и B – подмножества множества U. Объединением множеств A и B ($A \cup B$) называется множество всех элементов, принадлежащих либо A, либо B:

$$A \cup B = \{x \mid x \in A \text{ или } x \in B\},\ A \cup \varnothing = A$$



В двоичной булевой алгебре эта операция называется логическим сложением (дизъюнкцией) или логической операцией **ИЛИ**. Таблица истинности имеет вид:

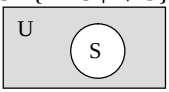
X	Y	X+Y
0	0	0
0	1	1
1	0	1
1	1	1

Результатом логического сложения является значение логического нуля тогда и только тогда, когда значения всех логических переменных равны логическим нулям.

3. Операция «Дополнение». Обозначение ' ($\neg$).

Пусть S – любое подмножество U. Множество тех элементов $x \in U$, которые не принадлежат S, называются дополнением S в U.

$$S' = \{x \in U \mid x \notin S\},\ S \cap S' = \varnothing$$



В двоичной булевой алгебре операция называется отрицанием (инверсией) или логической операцией **НЕ**.

Таблица истинности имеет вид:

X	$\bar{X}$
0	1
1	0

Постулаты булевой алгебры

Номер	Наименование законов	Обозначение в булевой алгебре	Обозначение в теории множеств
1	идемпотентности	x*x=x, x+x=x	S∪S=S, S∩S=S
2	коммутативности	xy=yx, x+y=y+x	
3	ассоциативности	(xy)z=x(yz)=xyz, (x+y)+z=x+(y+z)=x+y+z	
4	поглощения	x(x+y)=x, x+xy=x	S∩(S∪T)=S∪(S∩T)=S
5	дистрибутивности	x(y+z)=xy+xz, x+yz=(x+y)(x+z)	R∩(S∪T)=(R∩S)∪(R∩T) R∪(S∩T)=(R∪S)∩(R∪T)
6	дополняемость	$x * \bar{x} = 0$, $x + \bar{x} = 1$	
7	закон двойного отрицания	$\overline{(\bar{x})} = x$	
8	верхняя и нижняя границы	x*0=0, x+0=x x*1=x, x+1=1	R∩Ø=Ø, R∪Ø=R R∩U=R, R∪U=U
9	де Моргана	$\overline{(xy)} = \bar{x} + \bar{y}$ $\overline{(x+y)} = \bar{x} \cdot \bar{y}$	(S∩T)'=S'∪T' (S∪T)'=S'∩T'

Рассмотрим пояснение постулатов.

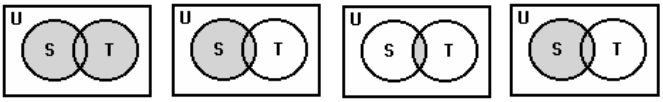
1. Постулаты идемпотентности устанавливают, что повторяющиеся переменные в выражении излишни, и их можно опустить. Таким образом, понятия возведение в степень и умножение на коэффициент, отличные от логических констант, т.е. на числа, в булевой алгебре смысла не имеют.

S∪S=S S∩S=S .

2. Постулаты коммутативности устанавливают, что порядок переменных при выполнении операции на ее результат не влияет.

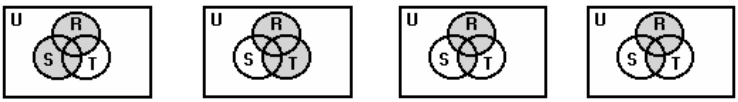
3. Постулаты ассоциативности определяют, что переменные можно группировать в любом порядке.

4. Постулат поглощения описывает эффект исчезновения (поглощения) одной из булевых переменных, связанных операциями **И** и **ИЛИ**: S∩(S∪T)=S∪(S∩T)=S



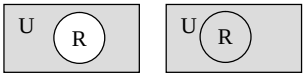
S∪T S∩(S∪T)=S S∩T S∪(S∩T)=S.

5. Постулаты дистрибутивности устанавливают, что в булевой алгебре возможно вынесение общего множителя за скобки: R∪(S∩T) =(R∪S)∩(R∪T)



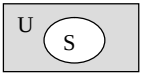
R∪(S∩T) R∪S R∪T (R∪S)∩(R∪T).

6. Дополняемость. Постулат подчеркивает взаимное дополнение булевых переменных.



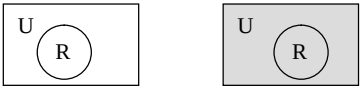
$R \cap R' = \varnothing$ $R \cup R' = U$

7. Инволютивный постулат устанавливает, что двойное отрицание эквивалентно пустой операции.



$(S')' = S$

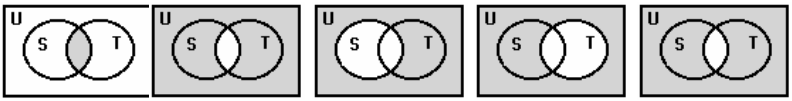
8. Аксиома устанавливает универсальные границы булевой алгебры: нижнюю и верхнюю: нижняя граница Ø(0), верхняя граница U(1).



$R \cap \varnothing = \varnothing$ $R \cap U = R$

$R \cup \varnothing = R$ $R \cup U = U$

9. Постулаты де Моргана описывают эффект отрицания (дополнения) булевых переменных, связанных операциями **И** и **ИЛИ**:



$S \cap T$ $(S \cap T)'$ S' T' $S' \cup T'$

$S \cup T$ $(S \cup T)'$ S' T' $S' \cap T'$

Все аксиомы представлены парой соотношений, которые обладают свойством симметрии. В каждой паре одно соотношение может быть получено из другого заменой:

– всех операций **И** на **ИЛИ** или всех операций **ИЛИ** на **И**;

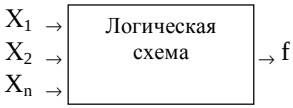
– всех логических 0 на логические 1 или наоборот.

Это свойство симметрии известно как принцип двойственности.

2.3. Применения булевой алгебры

Постулаты булевой алгебры находят применение в аппаратной логике и в программировании.

Назовем техническое устройство логической схемой и для описания его поведения применим метод моделирования. Логическую схему можно представить в виде модели черного ящика, т.е. устройства, внутренняя структура которого не известна.



Входные и выходные сигналы устройства являются булевыми переменными. Чтобы описать поведение модели черного ящика, нужно представить выходную переменную f как функцию входных переменных X_1, X_2… Xn: $f=f(X_1,X_2\ldots X_n)$.

Для описания работы логической схемы используются два метода: булево выражение и таблица истинности.

1. Булево выражение – это формула из булевых констант и переменных, связанных булевыми операциями. Например:
$$f(X_1, X_2,X_3)=(X_1+\overline{X_2})(\overline{X_1}+X_3)+X_2X_3.$$
2. Таблица истинности – это таблица, содержащая все возможные комбинации значений входных переменных и соответствующие им значения выходных переменных.

Составим таблицу истинности по булевому выражению, приведенному в примере.

X_1	X_2	X_3	$\overline{X_1}$	$\overline{X_2}$	$X_1+\overline{X_2}$	$\overline{X_1}+X_3$	$(X_1+\overline{X_2})(\overline{X_1}+X_3)$	X_2X_3	f
0	0	0	1	1	1	1	1	0	1
0	0	1	1	1	1	1	1	0	1
0	1	0	1	0	0	1	0	0	0
0	1	1	1	0	0	1	0	1	1
.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.
1	1	1	0	0	0	1	1	1	1

Существует и другая форма записи строк в таблице истинности:

$f(0,1,1)=(0+\bar{1})(\bar{0}+1)+1*1=(0+0)(1+1)+1=0*1+1=0+1=1;$

$f(1,0,0)=(1+\bar{0})(\bar{1}+0)+0*0=(1+1)(0+0)+0=1*0+0=0+0=0.$

Как можно применить постулаты булевой алгебры для описания работы логических схем? Они позволяют преобразовывать булевы выражения в эквивалентные выражения.

1. Упрощение булевых выражений

$f(X_1, X_2, X_3)=(X_1+X_3)(X_1+\overline{X_3})(\overline{X_2}+X_3)=(X_1+X_3*\overline{X_3})(\overline{X_2}+X_3)=(X_1+0)(\overline{X_2}+X_3)=$
$= X_1(\overline{X_2}+X_3).$

Для упрощения выражения последовательно применены постулаты: дистрибутивность, дополняемость, граница множества, дистрибутивность.

2. Отрицание булевых выражений. Операция отрицания выполняется с помощью постулатов де Моргана.

$$f(X_1,X_2,X_3)=\overline{((\overline{X}_1\overline{X}_2+X_3)\overline{X}_1)}=\overline{(\overline{X}_1\overline{X}_2+X_3)}+\overline{(\overline{X}_1)}=\overline{\overline{(\overline{X}_1\overline{X}_2)}\cdot\overline{X}_3}+X_1=\overline{(\overline{(\overline{X}_1)}+\overline{(\overline{X}_2)})}\cdot\overline{X}_3+X_1$$

$$=(X_1+X_2)\overline{X}_3+X_1=X_1\overline{X}_3+X_2\overline{X}_3+X_1=$$

$$=(X_1\overline{X}_3+X_1)+X_2\overline{X}_3=X_1+X_2\overline{X}_3.$$

Для упрощения выражения последовательно применены постулаты: дистрибутивность, де Моргана.

2.4. Вопросы для самопроверки

1. Что такое множество, подмножество, множество подмножеств?
2. Какая алгебраическая система называется булевой алгеброй?
3. Чему соответствуют логические элементы двоичной булевой алгебры в программировании и аппаратной логике?
4. Дать определения основным логическим операциям.
5. Пояснить каждый постулат двоичной булевой алгебры.
6. Какие существуют методы описания работы логических схем?

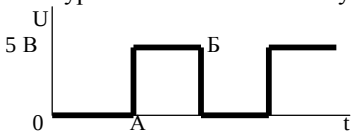
3. ОСНОВЫ ЦИФРОВОЙ ЭЛЕКТРОНИКИ

Если базовым логическим операциям булевой алгебры поставить в соответствие структурные схемы, которые называются двоичными логическими элементами, то можно описать любую логическую схему. Эти логические элементы составляют основу цифровых схем, образующих цифровые электронные устройства. Работа цифровых схем подчинена логическим законам, которые рассматривались в разделе булевой алгебры и которые описываются булевыми выражениями.

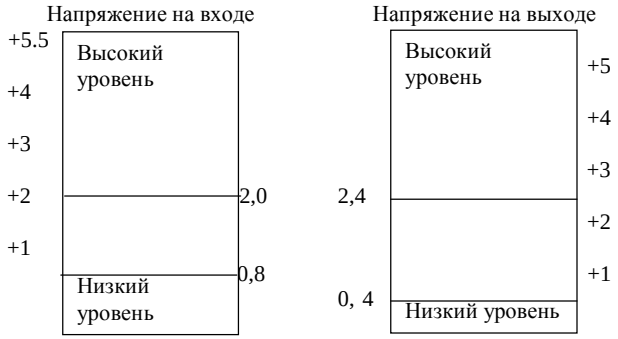
3.1. Цифровой сигнал

Цифровые схемы (устройства) называются цифровыми, потому что они оперируют с цифровыми сигналами.

Генератор прямоугольного импульса выдает сигнал прямоугольной формы, который можно увидеть на экране осциллографа. В точке А напряжение возрастает от 0 до 5 В, некоторое время остается равным 5 В, в точке Б падает до 0 и некоторое время равняется 0 В. Эти значения напряжения называются напряжением высокого и низкого уровня. Обычно считается, что высокий уровень напряжения эквивалентен значению логической единицы, низкий уровень – логического нуля.



Рассмотрим пример инвертора. Диапазон напряжения питания для схем транзисторно-транзисторной логики (ТТЛ) составляет 0…+5 В.



Область сигналов неопределенного уровня находится в диапазоне 0,8…2,0 В для входных сигналов и в диапазоне 0,4…2,4 В для выходных сигналов. Стандартным уровнем напряжения на выходе является 3,5 В для высокого уровня и 0,1 В для напряжения низкого уровня.

Величина напряжения на выходе зависит от величины сопротивления нагрузки на выходе логического элемента. Если на выходе сигнал неопределенного логического уровня, то можно предположить, что в схеме есть неисправность. Причина различия в уровне логических сигналов на входе и выходе заключается в стремлении обеспечить помехоустойчивость цифровых схем, т.е. нечувствительность к посторонним электрическим сигналам. Различия уровней напряжения гарантируют то, что цифровая схема не будет реагировать на нежелательные помехи.

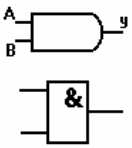
Описанные диапазоны характерны для семейства ТТЛ-схем, для других схем эти цифры будут другие.

3.2. Логические элементы (вентили)

Рассмотрим реализацию основных логических операций, описанных в 2.2.

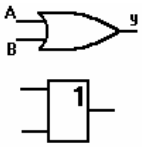
1. Логическая функция **И** (логическое умножение): A*B=y.

A	B	y
0	0	0
0	1	0
1	0	0
1	1	1
входы		выход



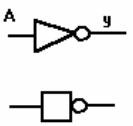
2. Логическая функция **ИЛИ** (логическое сложение): A+B=y.

A	B	y
0	0	0
0	1	1
1	0	1
1	1	1
входы		выход



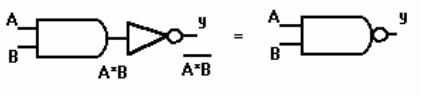
3. Логическая функция **НЕ** (инверсия): $y=\overline{A}$.

A	y
0	1
1	0
вход	выход



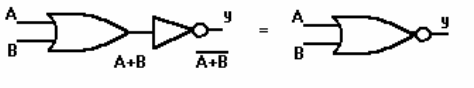
4. Логическая функция **И-НЕ**: $\overline{A*B}=Y$. Вентиль получается путем инвертирования выходов элемента **И**.

A	B	**И**	**И-НЕ**
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0
входы		выходы	



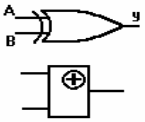
5. Логическая функция **ИЛИ-НЕ**: $\overline{A+B}=Y$. Вентиль получается путем инвертирования выходов элемента **ИЛИ**.

A	B	**ИЛИ**	**ИЛИ-НЕ**
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0
входы		выходы	



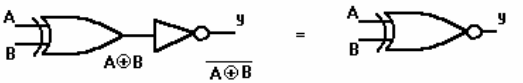
6. Логическая функция **Исключающее ИЛИ**: $A\oplus B=Y$. Символ $\oplus$ называется сложением по модулю 2.

A	B	**ИЛИ**	**Искл. ИЛИ**
0	0	0	0
0	1	1	1
1	0	1	1
1	1	1	0
входы		выходы	



7. Логическая функция **Исключающее ИЛИ-НЕ.** Вентиль получается путем инвертирования выходов элементов **Исключающего ИЛИ**.

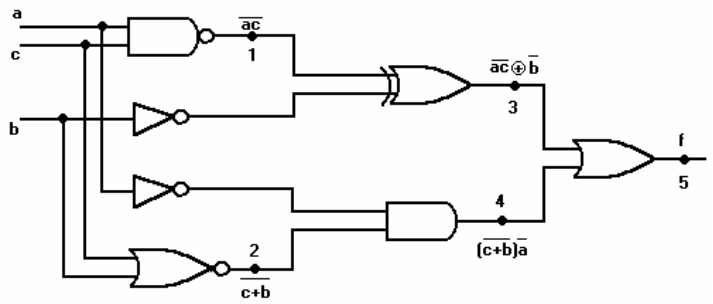
A	B	**Искл. ИЛИ**	**Искл. ИЛИ-НЕ**
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1
входы		выходы	



3.3. Конструирование логических схем на основе булевых выражений

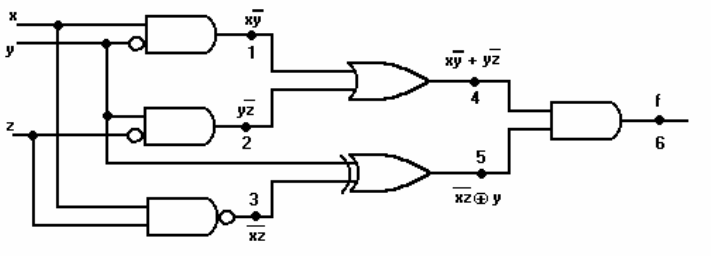
Ниже представлены примеры реализации булевых выражений на логических вентилях и приведены таблицы истинности в отмеченных точках схемы.

Пример 1. $y=\overline{(c+b)}\,\bar{a}+(\overline{ac}\oplus\bar{b})$.



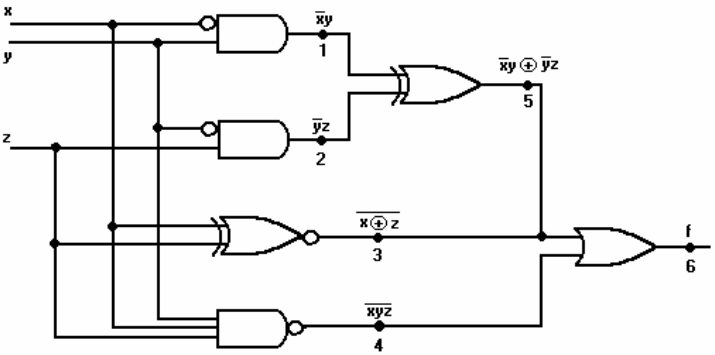
a	b	c	$\bar{a}$	$\bar{b}$	1	2	3	4	5
0	0	0	1	1	1	1	0	1	1
0	0	1	1	1	1	0	0	0	0
0	1	0	1	0	1	0	1	0	1
0	1	1	1	0	1	0	1	0	1
1	0	0	0	1	1	1	0	0	0
1	0	1	0	1	0	0	1	0	1
1	1	0	0	0	1	0	1	0	1
1	1	1	0	0	0	0	0	0	0

Пример 2. $f=(x\bar{y}+yz)\cdot(\overline{xz}\oplus y)$.



x	y	z	$\bar{y}$	1	2	3	4	5	6
0	0	0	1	0	0	1	0	1	0
0	0	1	1	0	0	1	0	1	0
0	1	0	0	0	0	1	0	0	0
0	1	1	0	0	1	1	1	0	0
1	0	0	1	1	0	1	1	1	1
1	0	1	1	1	0	0	1	1	1
1	1	0	0	0	0	1	0	0	0
1	1	1	0	0	1	0	1	1	1

Пример 3. $f=(\bar{x}y \oplus \bar{y}z)+\overline{(x \oplus z)}+\overline{xyz}$.



x	y	z	$\bar{x}$	$\bar{y}$	1	2	3	4	5	6
0	0	0	1	1	0	0	1	1	0	1
0	0	1	1	1	0	1	0	1	1	1
0	1	0	1	0	1	0	0	1	1	1
0	1	1	1	0	1	0	0	1	1	1
1	0	0	0	1	0	0	0	1	0	1
1	0	1	0	1	0	1	0	1	1	1
1	1	0	0	0	0	0	0	1	0	1
1	1	1	0	0	0	0	1	0	0	0

3.4. Вопросы для самопроверки

1. Что такое цифровой сигнал?
2. Какие напряжения соответствуют переключению уровней сигнала?
3. Что такое область сигнала неопределенного уровня?
4. Каковы стандартные уровни напряжения на входе и выходе ТТЛ-схем?
5. Какие Вам известны вентили, реализующие основные логические операции?

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. В.А. Каймин. Информатика. Учебник. – М.: Проспект. 2009. – 272 с.
2. П. Хоровиц, У. Хилл. Искусство схемотехники. – М.: Мир, Бином. 2009. – 704 с.
3. И.Г. Гинденко, С.А. Семеновская. Информатика. – СПб.: Нева. 2003. – 320 с.
4. А.Н. Степанов. Информатика: Учебник для вузов. – СПб.: Питер, 2006 – 684 с.
5. Р.Токхейм. Основы цифровой электроники. – М.: Мир. 1988. – 392 с.

ОГЛАВЛЕНИЕ

ПРЕДИСЛОВИЕ 3

1. СИСТЕМЫ СЧИСЛЕНИЯ 3
1.1. Основные определения 3
1.2. Способы преобразования чисел из одной системы счисления в другую 5
1.3. Двоичная арифметика 7
1.4. Числа со знаком 9
1.5. Применение двоичной и шестнадцатеричной систем счисления 10
1.6. Вопросы для самопроверки 12

2. ОСНОВЫ БУЛЕВОЙ АЛГЕБРЫ 12
2.1. Основные определения 12
2.2. Двоичная булева алгебра 13
2.3. Применения булевой алгебры 16
2.4. Вопросы для самопроверки 18

3. ОСНОВЫ ЦИФРОВОЙ ЭЛЕКТРОНИКИ 18
3.1. Цифровой сигнал 18
3.2. Логические элементы (вентили) 19
3.3. Конструирование логических схем на основе булевых выражений . . 21
3.4. Вопросы для самопроверки 22

БИБЛИОГРАФИЧЕСКИЙ СПИСОК. 22

Учебное издание

ИНФОРМАТИКА

Составители: НАЗАРОВ Игорь Васильевич
ПОТАПОВА Татьяна Александровна

Редактор С.П. Клышинская
Технический редактор О.Г. Завьялова

Подписано в печать 6.12.12. Форма 60x84/16. Бумага офсетная.
Печать – ризография. Усл.-печ.л. 1,5. Уч.-изд.л. 1,35. Изд. № 76. Тираж 50 экз.
Заказ . Бесплатно.
Московский институт электроники и математики
Национального исследовательского университета «Высшая школа экономики».
109028, Москва, Б. Трехсвятительский пер. 3/12.
Редакционно-издательский отдел Московского института электроники и математики Национального исследовательского университета «Высшая школа экономики».113054, Москва, ул. М. Пионерская, 12.