Есть вариант классификации людей, т.е. разделение всех людей на группы, в котором таких групп 10. Это те люди, кто знает про существование двоичной системы счисления, и те, кто о ней не знает.

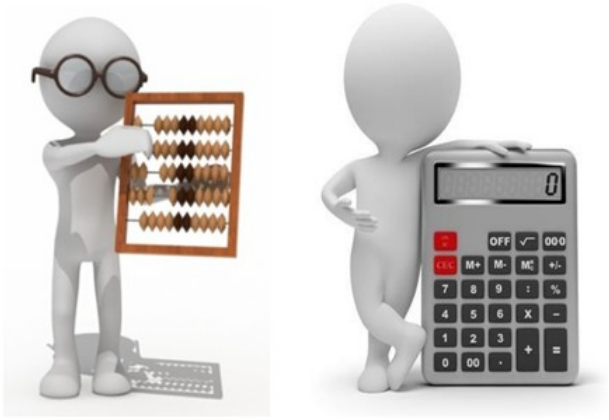
Ученая мудрость.

РАЗДЕЛ 1
СИСТЕМЫ СЧИСЛЕНИЯ И ДВОИЧНАЯ АРИФМЕТИКА

В цифровых электронных системах при обработке числовых данных приходится использовать систему представления числовой информации посредством двух символов, соответствующих двум уровням напряжения в цифровой электронике. Поэтому для понимания принципов построения электронных устройств, обрабатывающих числовые данные и осуществляющих вычисления, необходимо более подробно ознакомиться с двоичной системой счисления (и другими, тесно с ней связанными), а также рассмотреть основные способы выполнения арифметических операций при двоичном представлении чисел.

Понятие **системы счисления** связано с методом записи чисел с помощью набора знаков (символов). Существует множество систем счисления, которые по принципу организации записи числа разделяют на ***позиционные*** и ***непозиционные***, хотя существуют также ***смешанные*** системы, где сочетаются принципы записи позиционных систем и непозиционные принципы.

В настоящее время наибольшее распространение получили позиционные системы счисления. При этом для представления числовой информации и вычислений люди обычно используют т.н. "десятичную систему" с т.н. "арабскими" цифрами. Однако, как уже говорилось, в цифровой электронике развитие получили системы с преобразованием небольшого дискретного числа уровней, и, прежде всего, – двухуровневые. В связи с этим, для цифровой электроники огромное значение имеет представление числовых данных в двухсимвольных система, а именно в двоичной системе счисления, а также некоторыми другими близкими к ней, и алгоритмы вычислений при таких видах представления чисел.



1.1. Позиционные системы счисления

✓ В общем случае **позиционной системы счисления с основанием *b***:

• Положительное число *N* рассматривается как разложение по степеням *b* в виде

$$N = a_{n-1}\, b^{n-1} + a_{n-2}\, b^{n-2} + \ldots + a_0\, b^0 + \ldots + a_{-m}\, b^{-m} = \sum_{i=-m}^{n-1} a_i\, b^i \quad ,$$

где основание b – целое число больше 1, a_i – целое число в диапазоне от 0 до $b - 1$.

• Для обозначения целых чисел от 0 до $(b - 1)$ используют символы, называемые **цифрами** (самыми распространенными являются используемые здесь т.н. "арабские" цифры).

• Число записывается как последовательность цифр, обозначающих числа a_i.

• Последовательность цифр a_{n-1}, a_{n-2}, ..., a_0 образует запись **целой части** числа *N*, а последовательность a_{-1}, a_{-2} ,..., a_{-m} составляет запись **дробной части** *N*. При визуальном отображении записи числа целая и дробная части разделяются запятой или точкой.

• Позиция, в которой располагается цифра a_i, называется *i*-м **разрядом** числа (i – номер разряда, a_i – значение разряда, b^i – **вес разряда**). Разряды a_{n-1} и a_{-m} называются **старшим** и **младшим разрядом** соответственно.

Запись числа *N* будет иметь вид: $\boldsymbol{a_{n-1}a_{n-2} \ldots a_1a_0, \ldots a_{-1}a_{-2} \ldots a_{-m}}$, при этом величина записанного таким образом числа, определяется выражением:

$$N = \sum_{i=-m}^{n-1} a_i\, b^i$$

! Формально при абстрактных рассуждениях числа *n* и *m* (количество разрядов в целой и дробной частях числа соответственно) могут полагаться неограниченными. При практическом представлении они конечны. Это означает ограничение на максимальную целую часть числа, которое можно записать при данном *n* и ограниченную точность записи иррациональных чисел и произвольных рациональных чисел, зависящую от значения *m*. Возможность представления рационального числа позиционной системе счисления с конечным значением *m* (и требуемое *m*) определяется конкретными значениями числа и основания системы счисления.

! Очевидно, что следующие подряд нулевые разряды, включающие старший или младший, при визуальном представлении записи числа могут опускаться, например: 123,5 = 123,500 = 00123,50.

✓ При необходимости, для исключения неоднозначности понимания, при записи числа указывается основание системы счисления, например, в виде:

$$(a_{n-1}\, a_{n-2} \ldots a_1\, a_0,\ldots,\, a_{-1}\, a_{-2} \ldots a_{-m})_b$$

Пример: Для привычной нам "**десятичной** системы" b =10 и используются 10 цифр для обозначения чисел от 0 до 9 (до b –1). Значение числа $(234{,}54)_{10}$ есть результат суммирования:

$$234{,}54 = 2\cdot10^2 + 3\cdot10^1 + 4\cdot10^0 + 5\cdot10^{-1} + 4\cdot10^{-2}\ .$$

	__	2	3	4	,	5	4	__
	↓	↓	↓	↓		↓	↓	↓
…	$\times10^3$	$\times10^2$	$\times10^1$	$\times10^0$		$\times10^{-1}$	$\times10^{-2}$	$\times10^{-3}$ …
…	тысячи	сотни	десятки	единицы		десятые	сотые	тысячные …

✓ При b = 2 система счисления называется **двоичной**.

В записи числа используются две различные цифры: 0 и 1, обозначающие соответствующие целые числа.

Пример:
Величина двоичного числа 1010,01 определяется значением многочлена:

$$1010{,}01_2 = 1\cdot2^3 + 0\cdot2^2 + 1\cdot2^1 + 0\cdot2^0 + 0\cdot2^{-1} + 1\cdot2^{-2} = 2^3 + 2^1 + 2^{-2}$$

Это значение соответствует десятичному числу $10{,}25_{10}$

! Иногда разряды двоичного числа называют **битами** (сокращение от binary digits – двоичные цифры). Строго говоря, понятие "бит" введено как мера количества информации в теории информации, использующей аппарат теории вероятностей и математической статистики. В большинстве случаев (*но не всегда*) представления информации в виде последовательности двоичных разрядов (в частности в виде двоичных чисел) каждый разряд действительно отражает количество информации 1 бит, а общее количество информации соответствует числу разрядов.

✓ Для случаев b >10, необходимо введение дополнительных цифр (кроме привычных "арабских" цифр "0", "1", "2" …. "9" для обозначения чисел от 0 до 9). Общепринято обозначения целых чисел от 10 до 15 посредством латинских буквенных символов "A", "B", "C", "D", "E" и "F" соответственно. Это обеспечивает общепринятый вариант записи чисел в позиционных системах счисления с b = 11 – 16.

! Значение числа, записанного одной цифрой a (один разряд) не зависит от основания системы счисления (должно выполняться $b > a$). Если число записано двумя и более разрядами для определения значения обязательно надо знать основание системы счисления. 11_3 – четыре, 11_{16} – семнадцать…
! Очевидно, что с ростом основания системы счисления числа записываются более компактно (уменьшается необходимое количество разрядов), но требуется использование большего количества цифр.

Приведем представление чисел от 0 до 18 в различных позиционных системах счисления:

Основание *b*	**2**	**3**	**4**	**8**	**10**	**12**	**16**
	0	0	0	0	0	0	0
	1	1	1	1	1	1	1
	10	2	2	2	2	2	2
	11	10	3	3	3	3	3
	100	11	10	4	4	4	4
	101	12	11	5	5	5	5
	110	20	12	6	6	6	6
	111	21	13	7	7	7	7
	1000	22	20	10	8	8	8
Числа *N*	1001	100	21	11	9	9	9
	1010	101	22	12	10	A	A
	1011	102	23	13	11	B	B
	1100	110	30	14	12	10	C
	1101	111	31	15	13	11	D
	1110	112	32	16	14	12	E
	1111	120	33	17	15	13	F
	10000	121	100	20	16	14	10
	10001	122	101	21	17	15	11
	10010	200	110	22	18	16	12

✓ Важным понятиям для алгоритмов преобразования чисел, представленных в позиционной системе счисления с заданным основанием, является ***дополнение цифры***, которое равно разности между наибольшей цифрой по основанию b и цифрой a:

$$a' = (b - 1) - a.$$

В десятичной системе счисления $a' = 9 - a$ (например, для цифры 5 дополнением будет $9 - 5 = 4$).

✓ Представление числа последовательностью двоичных значений в непосредственном соответствии с правилами представления позиционной системе счисления с основанием 2 называют ***прямым двоичным кодом числа*** (ПДК).

! Такой термин связан с выделением данного представления среди многих других вариантов представления числа последовательностью двоичных разрядов, применяемых в цифровой электронике (некоторые рассмотрены далее).

! Рассмотрение прямого двоичного кода числа обычно подразумевает заданное фиксированное число двоичных разрядов, выделенных для представления чисел в данном коде (n-разрядный ПДК). При этом отдельно может оговариваться число разрядов в целой и дробной частях.

! Прямой двоичный код, как и некоторые другие коды с двоичными разрядами, имеет очевидные преимущества для применения в цифровой электронике, преобразующей двухуровневые сигналы. Однако в силу ряда причин, в алгоритмах преобразования числовых данных в цифровых электронных системах некоторое распространение получили также представления чисел в восьмеричной и шестнадцатиричной системах счисления.

1.2. Преобразование чисел из одной позиционной системы счисления в другую

При работе с цифровых устройств с числовыми данными необходимо применять алгоритмы преобразования числа из одной системы счисления в другую. Чаще всего необходим перевод из десятичной системы в двоичную и наоборот.

Кратко рассмотрим основные правила преобразований, которые иллюстрируют возможности построения алгоритмов перевода, и в том числе могут использоваться для "ручного" преобразования человеком.

✓ *Преобразование двоичных (и других) чисел в десятичные*

Данное преобразование легко получается простым вычислением в десятичной системе значения полинома, дающего величину числа по определению позиционной системы счисления. Поскольку вычисление суммы следует производить в "конечной" системе счисления, т.е. в десятичной, то такой подсчет удобно проводить обычному человеку.

Пример: Перевод двоичного числа 1101,01 в десятичную систему.

$1101{,}01_2 = ?$ (в десятичной системе)

$1\cdot 2^3 + 1\cdot 2^2 + 1\cdot 2^0 + 1\cdot 2^{-2} = 8 + 4 + 1 + 0{,}25 = 13{,}25_{10}$.

Пример: Перевод из шестнадцатиричной в десятичную систему.

$1F{,}8_{16} = ?$ (в десятичной системе)

$1\cdot 16^1 + 15\cdot 16^0 + 8\cdot 16^{-1} = 16 + 15 + 0{,}5 = 31{,}5_{10}$.

! Данный метод формально применим для перевода чисел из любой позиционной системы счисления в любую. Но для применения прямого расчета суммы коэффициентов $a_i b^i$ при переводе в систему счисления с $b \neq 10$ необходимо произвести расчет не в десятичной системе счисления. Как правило, это проблематично для большинства людей, а при применении цифровых электронных устройств необходимо реализовать алгоритмы вычислений для конечной системы счисления.

✓ *Перевод из десятичной системы в двоичную (и другие)*

Целесообразно в качестве альтернативы рассмотреть метод, в котором вычисления проводятся в исходной системе счисления. Таким методом удобно переводить числа из десятичной системе в другие, т.к. вычисления надо проводить в десятичной системе.

При таком переводе необходимо по отдельности обрабатывать целую и дробную части числа. Которые надо описать по отдельности.

1) Преобразование *целого десятичного числа* N_{10} (целой части числа) в систему счисления с основанием b.

Можно показать, что для нахождения коэффициентов $a_{n-1},\ldots a_0$ следует проводить деление на b остатком начиная с N и далее применяя деление к получаемым целым частным. Остатки будут задавать значения разрядов конечного

числа начиная с младшего. После *нулевого целого частного*, будут найдены все искомые коэффициенты a_i (формальное продолжение деления будет давать нулевые частные и остатки, не меняющие значение конечного числа).

Пример: Преобразование десятичного числа 57 в двоичное

Шаг	Деление	Целое частное	Остаток
1	57/2	28	1 (младший разряд)
2	28/2	14	0
3	14/2	7	0
4	7/2	3	1
5	3/2	1	1
6	1/2	0	1 (старший разряд)
…	0/2	0	0

Результат: $57_{10}=111001_2$

Пример: Преобразование десятичного числа 58506 в шестнадцатиричное

Шаг	Деление	Целое частное	Остаток
1	58506/16	3656	A (58506–3658·16=10_{10} = A_{16}, младший разряд)
2	3656/16	228	8
3	228/16	14	4
4	14/16	0	E (14_{10} = E_{16},старший разряд)
…	…	…	…
…	0/16	0	0

Результат: 58506_{10}= $E48A_{16}$

Очевидно, что для заданного конечного N процедура будет завершена за конечное число шагов n, такое, что $2^{n-1} < N < 2^n$.

2) Перевод правильной десятичной дроби, т.е. *дробной части* десятичного числа в систему счисления с основанием b.

Для нахождение значений разрядов $a_{-1},\ldots a_{-m}$ необходимо проводить умножение обе части равенства на b начиная с исходной десятичной дроби и затем последовательно умножая на b получающиеся дробные части. Целые части результатов будут равны значениям искомых разрядов начиная со старшего (нетрудно убедиться, что целые части произведений будут принимать значения от 0 до $b < 1$). Процесс прекращается, если получается нулевая дробная часть, однако процесс может быть бесконечным (на практике он обрывается при достижении требуемой точности).

Пример: Преобразование десятичного числа $0{,}34375_{10}$ в двоичное.

Шаг	Умножение	Целая часть произведения
1	$2 \cdot 0{,}34375 = 0{,}6875$	0 (старший разряд)
2	$2 \cdot 0{,}6875 = 1{,}375$	1
3	$2 \cdot 0{,}375 = 0{,}75$	0
4	$2 \cdot 0{,}75 = 1{,}5$	1
…	$2 \cdot 0{,}5 = 1{,}0$	1 (младший разряд)
…	$2 \cdot 0 = 0$	0

Результат: $0{,}34375_{10} = 0{,}01011_2$

! Нетрудно убедиться, что процедура преобразования носит характер бесконечного повторения, если, например, попытаться перевести $1/3 = 0{,}(3)_{10}$ в двоичную систему. По описанной процедуре, получим: $0{,}(3)_{10} = 0{,}010101\ldots_2 = 0{,}(01)_2$.

! Метод можно использовать для перевода записи чисел из произвольной позиционной системы в любую другую позиционную систему. Однако вычисления (деление, умножение) производятся в исходной системе счисления, поэтому для обычного человека метод удобен только при переводе из десятичной системы счисления в другие.

✓ *Переводы в системах счисления с основаниями кратными по степени*

Выше приведены примеры алгоритмов для преобразования записи числа из системы счисления с произвольным основанием b_1, в систему с другим основанием b_2. При этом при выполнении вычислений можно сначала преобразовать число из системы с основанием b_1 в десятичную систему, а затем, из десятичной в систему с основанием b_2.

Особым является случай, когда одно основание является степенью другого с целым показателем m. Если $b_1 > b_2$, указанное условие может быть записано в виде:

$$\boldsymbol{b_1 = (b_2)^m}$$

В такой ситуации возможны существенно более простые алгоритмы преобразований из системы с основанием b_1 в систему с основанием b_2 и наоборот.

Можно доказать, что при выполнении указанной кратности по степени, каждой цифре числа в системе счисления с основанием b_1 соответствует группа из m цифр в системе с основанием b_2.

$N =_{b_1}$:	…	a_1		a_0	,	a_{-1}	a_{-2}	…
$N =_{b_2}$:	…	{… q_2 q_1 q_0}	{… q_2 q_1 q_0}		,	{… q_2 q_1 q_0}	{… q_2 q_1 q_0}	…
		m разрядов	m разрядов			m разрядов	m разрядов	

При этом m-разрядная группа {… q_2, q_1, q_0}, соответствующая i-му разряду в позиционной системе счисления с основанием b_1 равна значению этого разряда a_i, записанного в позиционной системе счисления с основанием b_2.

! Нетрудно видеть, что для записи *любой* одиночной цифры системы с b_1 в систему с b_2 достаточно m цифр, действительно:

• $(q_{m-1} \dots q_0)_{b2} < b_2^m$ или $\max\{(q_{m-1} \dots q_0)_{b2}\} = (1 \dots 1)_{b2} = b_2^m$, *пример*: $(111)_2 =$ (в десятичной системе счисления)$= 2^3 - 1 = 7$
• $\max\{(a)_{b1}\} = b_1 - 1 = b_2^m - 1$.

Если для записи a_i в системе счисления с основанием b_2 нужно менее, чем m разрядов, старшие разряды в m-разрядной группе заполняются нулями.

! Приведенное правило позволяет существенно упростить алгоритм преобразования, поскольку вместо проведения расчетов (деление с остатком и умножения дробных частей, вычисление полинома по степеням основания и т.п.) достаточно использовать таблицу записи целых чисел 0, 1 …$b_1 - 1$ в системе с основанием b_2.

• При переводе числа из системы счисления с основанием b_1 в систему с основанием b_2 *каждая* цифра заменяется m-разрядной группой, в соответствие с указанным выше правилом.
• При переводе числа из системы с b_2 в систему с b_1 *все* цифры в целой и дробной части объединяются в m-разрядные группы, начиная от запятой (если в старших разрядах целой части или в младших разрядах дробной части значащих разрядов недостаточно для образования m-разрядной группы справа и слева записываются нулевые разряды). Далее каждая группа заменяется цифрой a_i в соответствие с указанным выше правилом.

Примеры:

Пусть $b_1 = 8$, $b_2 = 2$, $m = 3$.
Фиксируем соответствие цифр восьмеричной системы и трехразрядных групп двоичной:

Восьмеричная, a_i	0	1	2	3	4	5	6	7
Двоичная, $\{q_2, q_1, q_0\}$	000	001	010	011	100	101	110	111

а) Записать число $331{,}54_8$ в двоичной системе.

Цифра "3" заменяется на группу 011, "1" на 001, "5" на 101 и "4" на 100.
Результат: $331{,}54_8 = 011011001{,}101100_2$ или, опуская нули слева и справа = $11011001{,}1011_2$

Альтернативный вариант:
• сначала перевод в десятичную систему ($3 \cdot 8^2 + 3 \cdot 8^1\ 1 \cdot 8^0 + 5/8 + 4/64 = 217{,}6875$);
• затем перевод целой части в двоичную систему (пошаговое деление 217 на 2 с остатком);
• затем перевод дробной части в двоичную систему (умножение 2 на 0,6875 затем 0,375 и т.д.).
Таким образом подтверждается полученный результат, но процедура значительно сложнее и продолжительнее.

б) Записать число $1101111{,}01001_2$ в восьмеричной системе.

Исходный код:	1 1 0 1 1 1 1 , 0 1 0 0 1
Разбиение на трехразрядные группы:	{0 0 1} {1 0 1} {1 1 1} , {0 1 0) {0 1 0}
Замена группы на восьмеричную цифру:	1 5 7 , 2 2

Результат: $1101111{,}01001_2 = 157{,}22_8$.

Альтернативный вариант:

• сначала перевод в десятичную систему (64+32+8+4+2+1+1/4 +1/32 = 111,281);

• затем перевод целой части в восьмеричную систему (пошаговое деление 111 на 8 с остатком);

• затем перевод дробной части в восьмеричную систему (умножение 8 на 0,281 затем 0,25).

! Правильность упрощенного алгоритма доказывается аналитически для любых систем счисления с основаниями кратными по степени.

! Непосредственно в электронных схемах для представления числовых данных необходимо использовать двоичную систему (либо коды с использованием 0 и 1). Однако простота рассмотренной упрощенной процедуры перевода привела к тому, что в применяемых в компьютерной технике алгоритмах, наряду с десятичной системой (нужна при представлении данных человеку или ввода данных человеком) для сокращения длины чисел получили некоторое распространение использование 8-ричной и 16-чной систем ($8=2^3$, $16=2^4$), т.е. именно системы с основанием, кратным по степени двойке.

1.3. Двоичная арифметика

Четыре основных арифметических операции: сложение, вычитание, умножение, и деление можно выполнять в позиционной системе счисления с любым основанием, например, используя такие "типовые" процедуры, как сложение (умножение) "в столбик". Поскольку цифровые электронные устройства оперируют, в основном с двоичными числами, то целесообразно более подробно рассматривать именно двоичную арифметику.

✓ Сложение двух (положительных) чисел.

• Сложение начинается с младшего разряда, заканчивается старшим.
• Если число, полученное в результате сложения в данном разряде, оказывается больше либо равно основанию системы, то осуществляется **перенос** в следующий разряд (к сумме значений в следующем разряде прибавляется 1), а в данный разряд записывается цифра, равная разности этого числа и основания системы счисления.
• Для каждого разряда складываются значения разряда одного слагаемого, второго слагаемого и значение переноса из предыдущего разряда (для младшего разряда перенос из предыдущего исключен).

Для двоичных чисел суммирование в разряде допускает следующие варианты:

Суммы значений разрядов:

$$0+0 = 0; \qquad 0+1 = 1+0 = 1; \qquad 1+1 = [1]0.$$

С учетом переноса из предыдущего разряда:

$$0+0+1 = 1; \qquad 0+1+1 = 1+0+1 = [1]0; \qquad 1+1+1 = [1]1.$$

Здесь скобками [ ] выделена единица, переносимая в следующий разряд, результатом суммы данного разряда будет число рядом со скобками.

Пример: 1111.01 + 111.1

		1←	1←	1←					← Переносы
+		**1**	**1**	**1**	**1**	**,**	**0**	**1**	$= 15,25_{10}$
		0	**1**	**1**	**1**	**,**	**1**	**0**	$= 7,5_{10}$
	1	**0**	**1**	**1**	**0**	**,**	**1**	**1**	$= 22,75_{10}$

! Процедура полностью аналогична слоению "в столбик" десятичных чисел.

Ситуация, когда рещультат суммы двух чисел требует для представления больше разрядов, чем предусматривает используемый код, в котором представлены слагаемые, называется ***переполнением***. Очевидно, что в таком случае невозможно обеспечить получение корректного результата. Контроль возникновения переполнения осуществляется дополнительными средствами и здесь не рассматривается.

✓ Вычитание одного числа из другого.

Данная процедура предполагает (как и в десятичной системе счисления), что из большего по абсолютному значению числа вычитается меньшее (знак результата определяется отдельно).

• Вычитание так же выполняется поразрядно, от младшего разряда к старшему.

• Если результат вычитания в разряде оказывается меньшим нуля, то осуществляется **заем** единицы из следующего более старшего разряда, и в данный разряд добавляется величина, равная основанию. Результат вычитания в разряде, из которого был сделан заем, уменьшается на 1.

В двоичной системе для текущего разряда возможны варианты:

Неотрицательный результат: 0 – 0 = 0; 1 – 0 = 1; 1 – 1=0. Если был заём из данного разряда: 1 – 0 –1 = 0.
Отрицательный результат: 0 – 1 = – 1. Если был заём из данного разряда: 0 –1 –1 = – 10 (-2_{10}); 0 – 0 –1 = –1; 1 – 1 –1 = –1.
С учетом заёма из более старшего $(+2_{10}) = 10_2$: – 1–1 +10 = 0; – 1 + 10 = 1.

Пример: 10010.11 – 1100.1

		1→		1→						← Заём из разряда
				+2	+2					← Заём в разряд
–		**1**	**0**	**0**	**1**	**0**	**,**	**1**	**1**	$= 18.75_{10}$ – уменьшаемое
		0	**1**	**1**	**0**	**0**	**,**	**1**	**0**	$= 12.5_{10}$ – вычитаемое
		0	**0**	**1**	**1**	**0**	**,**	**0**	**1**	$= 6.25_{10}$ – разность

! Процедура полностью аналогична слоению "в столбик" десятичных чисел.

! Процедуры поразрядного двоичного сложения и вычитания подразумевают предварительное выравнивание числа разрядов операндов (добавление нулей слева от старшего разряда целой части и справа от младшего разряда дробной части), как это сделано в примерах.

✓ Перемножение двух двоичных чисел

Умножение двоичных чисел выполняется с помощью последовательного сложения. Как и в случае десятичных чисел умножение "в столбик" подразумевает умножения на значения разрядов второго операнда и сложения.

! В случае двоичной системы умножение числа на одноразрядный множитель тривиально (при умножении на 0 – результат ноль, при умножении на 1 число не меняется).

Пример: перемножение 11001,1 × 110,1

				×	1	1	0	0	1 ,	1	– множимое (= $25{,}5_{10}$)
							1	1	0 ,	1	– множитель (= $6{,}5_{10}$)
					1	1	0	0	1 ,	1	
+				0	0	0	0	0			– частичные произведения
			1	1	0	0	1	1			
		1	1	0	0	1	1				
1	0	1	0	0	1	0	1,	1	1		– произведение (сумма частных произведений) =$165{,}75_{10}$

✓ Двоичные числа со знаком

Запись числа в ПСС обычно предполагает представление в виде абсолютной величины, которой предшествует знак “+” или “–”, указывающий на то, является оно положительным или отрицательным (знак “+” часто опускается).

Для операций с положительными и отрицательными числами в ЦЭУ должна быть определены средства различия и определенные алгоритмы обработки чисел с разными знаками.

С этой целью предусмотрено следующее:

• Для представления чисел отводится фиксированное число разрядов (т. н. “***машинное слово***”).

• Старший разряд используется для указания знака. Его называют «***разряд знака***» или «***знаковый разряд***» и при записях часто дополнительно выделяют с помощью точки. “1” в знаковом разряде соответствует отрицательному числу, а “0” в разряде знака – положительному числу.

Эту форму представления обычно называют ***прямым двоичным кодом со знаком***.

! При вычислениях с положительными и отрицательными числами чаще всего используются варианты представления чисел, которые отличаются от прямого кода со знаком. Наиболее распространены *обратный* и *дополнительный* коды. Эти формы представления чисел позволяют выполнять операции сложения и вычитания на основе стандартного алгоритма поразрядного сложения. Далее рассмотрим эти коды на примере двоичных чисел.

! Преобразование прямого кода числа в обратный и дополнительный коды осуществляется по-разному для положительных и отрицательных чисел.

! При работе с числами в обратном и дополнительном кодах для представления чисел используются *фиксированное число разрядов* и преобразованиям подвергаются все разряды, включая последовательности идущих подряд нулей, охватывающих старший или младший разряды, хотя они не влияют на значение числа в прямом двоичном коде.

✓ Обратный код

Запись *положительного* числа в обратном коде *совпадает* с его представлением в прямом двоичном коде со знаком.

Для записи *отрицательного* двоичного числа в обратном коде необходимо заменить значения всех разрядов, кроме знакового на альтернативные значения (заменить нули на единицы и наоборот). Знаковый разряд преобразованию не подвергается.

Если число, представленное в обратном коде нужно преобразовать в прямой, то знаковый разряд указывает на положительность либо отрицательность числа, а остальные разряды не меняются в случае положительного числа и инвертируются в случае отрицательного числа.

! В системе счисления с $b \neq 2$ числа в разрядах заменяются на дополнения.
! В обратном коде имеет место т.н. двойственность представления нуля: число 0 может быть представлено в виде нулевых значений всех разрядов (включая знаковый) – положительный ноль, или в виде единичных значений всех разрядов (включая знаковый) – отрицательный ноль.

Примеры:

Двоичное число	5-значный двоичный код со знаком	
	прямой двоичный код со знаком	обратный код
11	0.0011	0.0011
100	0.0100	0.0100
–11	1.0011	1.1100
–100	1.0100	1.1011

✓ **Сложение положительных и отрицательных чисел в обратном коде**

Согласно общим принципам алгебры, использование положительных и отрицательных чисел и сложения таких чисел исключает отдельного определения такого арифметического действия, как вычитание.

Так же, как и при сложении положительных чисел, в случае сложения числе со знаком может оказаться, что абсолютная величина суммы двух чисел может требует для представления больше разрядов, чем предусматривает используемый код. Т.е. может возникнуть ***переполнение***, при котором невозможно обеспечить корректного представление результата и которое, вообще говоря, требует отдельного контроля и индикации.

Алгоритм сложения чисел в обратном коде можно представить в виде двух этапов:

1) Числа, представленные в обратном коде, подвергаются стандартной процедуре поразрядного двоичного сложения (как и в случае прямого двоичного кода), включая знаковый разряд.

2) В случае возникновения переноса из старшего, т.е. знакового разряда (более старший разряд, в котором мог быть учтен этот перенос отсутствует), нужно прибавить к результату первого этапа число имеющего единицу в младшем разряде (в остальных – нули). Такую операцию называют ***циклическим переносом*** – единица, переносимая из старшего разряда прибавляется к младшему.

В итоге будет получен корректный результат в обратном коде, если не возникает переполнения (это можно доказать теоретически).

! Если в старшем (знаковом) разряде переноса не возникает, то правильной суммой в обратном коде будет результат первого этапа.
! С технической точки зрения указанную процедуру можно трактовать как единый алгоритм, в котором значение переноса (1, если возник перенос или 0, если перенос отсутствует) из старшего разряда всегда добавляется к младшему.

Примеры: целесообразно привести ряд примеров с разным соотношением слагаемых по знаку и абсолютной величине.

а) $11_2 + 100_2 = 111_2$

Двоичное сложение.

```
  0 1 1
+ 1 0 0
-------
  1 1 1
```

Сложение в обратном (5-ти разрядном) коде.

```
  0 . 0 0 1 1     ← обр. код числа +11₂
+ 0 . 0 1 0 0     ← обр. код числа +100₂
---------------
  0 . 0 1 1 1     ← результат: +111₂
```

! В данном примере с положительными числами обратный код слагаемых и суммы совпадает с прямым двоичным кодом со знаком (5-значным).

б) $11_2 - 100_2 = -1_2$

Двоичное вычитание (из большего по модулю)

```
  1 0 0
– 1 1
-------
  0 0 1
```

Сложение в обратном (5-ти разрядном) коде.

```
  0 . 0 0 1 1     ← обр. код числа +11₂
+ 1 . 1 0 1 1     ← обр. код числа –100₂
---------------
  1 . 1 1 1 0     ← обр. код числа: –1₂
```

в) $-11_2 - 100_2 = -111_2$

Двоичное сложение (абсолютных величин)

```
    1 1
+ 1 0 0
-------
  1 1 1
```

Сложение в обратном (5-ти разрядном) коде.

```
  ←
  1 . 1 1 0 0     ← обр. код числа –11₂
+ 1 . 1 0 1 1     ← обр. код числа –100₂
---------------
  1 . 0 1 1 1
+                 ← циклический перенос
  0 . 0 0 0 0 1
---------------
  1 . 1 0 0 0     ← обр. код числа: –111₂
```

г) $100_2 - 11_2 = 1_2$

Двоичное вычитание

```
  1 0 0
+   1 1
-------
  0 0 1
```

Сложение в обратном (5-ти разрядном) коде.

```
  ←
  0 . 0 1 0 0     ← обр. код числа +100₂
+ 1 . 1 1 0 0     ← обр. код числа –11₂
---------------
  0 . 0 0 0 0 0
+                 ← циклический перенос
  0 . 0 0 0 0 1
---------------
  0 . 0 0 0 0 1   ← обр. код числа: +1₂
```

✓ Дополнительный код

Дополнительный код похож на обратный, но требует большего набора действий для преобразования.

• Запись *положительного* числа в дополнительном коде *совпадает* с его представлением в прямом двоичном коде со знаком (как и в случае обратного кода).

• Для записи *отрицательного* двоичного числа в дополнительном коде сначала преобразовать число в обратный код, а затем прибавить к нему число с единицей в младшем разряде (остальные заряды нулевые).

• Если положительное число, представленное в дополнительном коде (знаковый разряд имеет значение 0), нужно преобразовать в прямой двоичный код, то изменений разрядов не требуется (это очевидное следствие первого тезиса).

• Если отрицательное число, представленное в дополнительном коде (знаковый разряд имеет значение 1), нужно преобразовать в прямой двоичный код, то следует сделать *те же преобразования*, которые выполняются при переводе числа из прямого кода в дополнительный:
– инвертирование разрядов (кроме знакового);
– прибавление числа с единицей в младшем разряде.

! Исходя из правил перевода числа в дополнительный код следовало ожидать, что обратный перевод потребует вычитания числа с единицей в младшем разряде, а затем инвертирование разрядов. Однако можно доказать, что результат будет эквивалентен результату повторения процедуры перевода в дополнительный код. Т.е. для перевода чисел из прямого кода в дополнительный и обратно можно использовать одинаковый алгоритм действий, что очень важно для упрощения реализации обработки числовых данных в электронных устройствах.

Примеры:

Двоичное число	прямой двоичный код со знаком	обратный код (инвертирование)	дополнительный код (прибавление 1)
11	0.0011	0.0011	0.0011
100	0.0100	0.0100	0.0100
–11	1.0011	1.1100	1.1101
–100	1.0100	1.1011	1.1100

Подробнее для числа –100:

	прямой ДК → доп. код.	доп. код. → прямой ДК
Исходный код:	1.0100 ← прямой (–100)	1.1100 ← дополнительный
Инвертирование:	1.1011	1.0011
Добавление 0.0001	1.1100 ← дополнительный	1.0100 ← прямой (–100)

✓ Сложение положительных и отрицательных чисел в обратном коде

Сложение чисел (положительных и отрицательных), представленных в *дополнительном коде* осуществляется процедурой поразрядного сложения, включая знаковый разряд (как для ПДК). Возникновение переноса из знакового разряда *игнорируется*. Результат поразрядного сложения представляет собой дополнительный код суммы (за исключением случая переполнения).

! Как и в случае других кодов в результате служения может возникнуть переполнение, когда сумма не может быть корректно отображена, и возникновение переполнения следует контролировать отдельно.

Примеры:

а) $11_2 + 100_2 = 111_2$

Двоичное сложение.

```
  0 1 1
+ 1 0 0
-------
  1 1 1
```

Сложение в обратном (5-ти разрядном) коде.

```
  0 . 0 0 1 1     ← доп. код числа +11₂
+ 0 . 0 1 0 0     ← доп. код числа +100₂
---------------
  0 . 0 1 1 1     ← результат: +111₂
```

! В данном примере с положительными числами обратный код слагаемых и суммы совпадает с прямым двоичным кодом со знаком (5-значным).

б) $11_2 - 100_2 = -1_2$

Двоичное вычитание (из большего по модулю)

```
  1 0 0
- 1 1
-------
  0 0 1
```

Сложение в обратном (5-ти разрядном) коде.

```
  0 . 0 0 1 1     ← доп. код числа +11₂
+ 1 . 1 1 0 0     ← доп. код числа −100₂
---------------
  1 . 1 1 1 1     ← доп. код числа: −1₂
```

Пояснение: доп.к. 1.1111 → (инверсия) 1.0000 → 1.0000+ 0.0001= 1.0001 → –1.

в) $-11_2 - 100_2 = -111_2$

Двоичное сложение (абсолютных величин)

```
    1 1
+ 1 0 0
-------
  1 1 1
```

Сложение в обратном (5-ти разрядном) коде.

```
  ←1 . 1 1 0 1     ← доп. код числа −11₂
+ 1 . 1 1 0 0     ← доп. код числа −100₂
---------------
[1] 1 . 1 0 0 1   ← доп. код числа −111₂
```

Пояснение: доп.к. 1.1001 → (инверсия) 1.0110 → 1.0110+ 0.0001= 1.0111 → -111_2.

г) $100_2 - 11_2 = 1_2$

Двоичное вычитание

```
  1 0 0
+   1 1
-------
  0 0 1
```

Сложение в обратном (5-ти разрядном) коде.

```
  ←0 . 0 1 0 0     ← доп. код числа +100₂
+ 1 . 1 1 0 1      ← доп. код числа −11₂
---------------
+ 0 . 0 0 0 1      ← доп. код числа +1₂
```

Можно констатировать следующие комментарии относительно применения обратного и дополнительного кодов:

• Обратный и дополнительный коды позволяют организовать хранение положительных и отрицательных чисел с возможностью их суммирования по относительно простому принципу, основанному на поразрядном суммировании с переносом. Это существенно проще, чем использовать алгоритм поразрядного вычитания с заёмами и с предварительным сравнением абсолютных величин или т.п. вариантами дополнительного нахождения уменьшаемого и вычитаемого, а также выяснения знака результата.

• Процедура перевода в дополнительный код и обратно в прямой немного сложнее, чем в случае обратного кода (отличие в операции прибавления единицы к младшему разряду).

• Процедура суммирования двух чисел в дополнительном коде заметно проще, чем в обратном (не требуется дополнительный этап с циклическим переносом).

! Операция перевода из прямого кода и обратно необходима относительно редко (при оперировании данных человеком), а суммирование как часть вычислений применяется значительно чаще. Это объясняет, что в электронных вычислениях достаточно широко распространено применение дополнительного кода.

1.4. Двоично-десятичные коды

При использовании двухсимвольных средств представления информации (т.е. двухуровневых сигналов) прямой двоичный код и тесно связанные с ним коды (обратный, дополнительный и т.п.) обладает многими практическими преимуществами и широко используется в при и обработке данных цифровыми электронными системами. Однако иногда удобно иметь дело с десятичной системой счисления, особенно при значительном объеме числовой информации, вводимой и выводимой человеком, поскольку большая часть числовых данных, которыми оперируют люди, представлены в десятичной форме. С целью упрощения проблемы связи человека с цифровыми электронными системами применяется ряд кодов, в которых десятичные цифры представляются последовательностями двоичных разрядов.

В двоично-десятичном коде (ДД коде) *каждая цифра в разряде десятичного числа* заменяется n-разрядной двоичной комбинацией. Наиболее распространены ДД коды с $n = 4$.

✓ Взвешенные коды

Для представления значений десятичных цифр 0, 1, 2, ... 9 в двоичной форме необходимо иметь, по крайней мере, четыре двоичных разряда, поскольку надо обеспечить возможность записи десяти неповторяющихся наборов последовательных комбинаций нулей и единиц). Однако поскольку для четырех разрядов возможны 16 комбинаций, из которых используется лишь 10, то можно построить большое число различных двоично-десятичных кодов.

Особый интерес представляет класс т.н. ***взвешенных кодов***, для которых:

• Каждому из четырех разрядов приписывается “вес” – w_1, w_2, w_3, w_4.

• Каждая четырехразрядная группа x_1, x_2, x_3, x_4 задает десятичное число Q, равное сумме “весов” тех двоичных разрядов, значения которых равны 1. $Q = w_4x_4 + w_3x_3 + w_2x_2 + w_1x_1$.

Последовательность значений двоичных разрядов $x_4x_3x_2x_1$, представляющую десятичную цифру, называют так же ***кодовым набором***.

Наиболее распространен код с весами [8, 4, 2, 1]. В результате, кодовый набор, соответствующий любому десятичному разряду, является прямым двоичным кодом данного десятичного разряда. Такой код называется ДД (двоично-десятичным) кодом ***8421 BCD*** (Binary-Coded Decimal) или просто ***ДД8421 кодом***.

Могут использоваться и другие варианты ДД кодов, в т.ч. варианты, в которых могут присутствовать отрицательные весовые коэффициенты. Например, коды с наборами весов [2;4;2;1] или [6;4;2;–3]. В этом случае некоторые десятичные цифры вообще говоря могут представляться не единственным образом, например, десятичную цифру 7 можно записать в ДД[2; 4; 2; 1] коде как в виде 1101, так и в виде 0111. Выбор представления чисел в этих кодах обусловлен требованием *самодополняемости* кодов. Код называется ***самодополняющимся***, если кодовый набор дополнения десятичной цифры Q, т.е. число $10 – Q – 1$, может получаться из

кодового набора Q путем инверсии, т.е. замены всех единиц на нули, а нулей на единицы. Например, в ДД[6; 4; 2; –3]-коде десятичное цифра 3 записывается как 1001, а ее дополнение – 6 = 9 – 3 – записывается как 0110.

! Для того чтобы ДД код мог быть самодополняющимся, необходимо чтобы сумма его весов равнялась 9.
! Код ДД8421 не является самодополняющимся (сумма весов 15, не 9).

Примеры взвешенных двоичных кодов

Десятичная цифра	веса двоичных разрядов											
	8	4	2	1	2	4	2	1	6	4	2	-3
0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1	0	1	0	1
2	0	0	1	0	0	0	1	0	0	0	1	0
3	0	0	1	1	0	0	1	1	1	0	0	1
4	0	1	0	0	0	1	0	0	0	1	0	0
5	0	1	0	1	1	0	1	1	1	0	1	1
6	0	1	1	0	1	1	0	0	0	1	1	0
7	0	1	1	1	1	1	0	1	1	1	0	1
8	1	0	0	0	1	1	1	0	1	0	1	0
9	1	0	0	1	1	1	1	1	1	1	1	1
	ДД[8;4;2;1]				ДД[2;4;2;1]				ДД[6;4;2;-3]			

✓ *Невзвешенные коды*

Существует много т.н. невзвешенных двоичных кодов, в которых значение десятичного числа отличается от суммирования значений разрядов с весом. Отметим некоторые часто упоминаемые случаи.

• ***Код с избытком три*** формируется путем сложения каждого кодового набора ДД8421 кода с 0011 (например, число 7 записывается в коде с избытком 3 в виде 0111+ 0011= 1010). Этот код является самодополняющимся.
• Во многих практических приложениях, например, при аналого-цифровом преобразовании желательно пользоваться кодами, в которых все последовательные кодовые наборы отличаются друг от друга лишь одним разрядом, эти коды называются ***циклическими***.
• ***Код Грея*** является особенно важным среди *циклических* кодов.
Если $b_n, \ldots, b_1, b_0$ соответствующее двоичное число, а $g_n, \ldots, g_1, g_0$ – кодовый набор в коде Грея, то разряд g_i можно выразить через соответствующее двоичное число следующим образом $g_n = b_n$, а при $i < n$ – $g_i = b_i \oplus b_i{+}1$. Здесь $\oplus$ – символ операции "сложение по модулю 2" (см. раздел 2), которая определяется соотношениями:

$$0 \oplus 0 = 0\ ;\ 1 \oplus 1 = 0;\ 0 \oplus 1 = 1;\ 1 \oplus 0 = 1.$$

Для кода Грея характерно то, что каждое число отличается от предыдущего измененным значением только одного разряда (и, соответственно от последующего).

! Код Грея не относится к ДД кодам и может использоваться для представления любых чисел.

Примеры невзвешенных двоичных кодов

Десятичное число	Прямой двоичный код	Код с избытком 3	Код Грея
0	0000	0011	0000
1	0001	0100	0001
2	0010	0101	0011
3	0011	0110	0010
4	0100	0111	0110
5	0101	1000	0111
6	0110	1001	0101
7	0111	1010	0100
8	1000	1011	1100
9	1001	1100	1101
10*	1010	–	1111
11*	1011	–	1110
12*	1100	–	1010
13*	1101	–	1011
14*	1110	–	1001
15*	1111	–	1001

* Эти числа превышают 9 и формально не относятся к ДД коду, хотя могут быть представлены в прямом двоичном коде и в коде Грея.

✓ *Коды с обнаружением ошибок*

Появление ошибки в одном из разрядов двоичного кода (одиночной ошибки) может привести к неправильному, но допустимому кодовому набору. Например, если в ДД8421-коде произойдет ошибка в младшем разряде набора 0110, то это приведет к кодовому набору 0111, который допустим, но ошибочен, и будет приводить к ошибкам и при дальнейшей обработке данных.

Кодом с обнаружением одиночной ошибки называют такой код, в котором есть допустимые наборы (используемые для передачи информации) и недопустимые (неиспользуемые) наборы, а появление любой одиночной ошибки преобразует допустимый кодовый набор в недопустимый.

На примере относительно простых вариантов ДД-кодов можно указать:

• ***ДД8421-код с проверкой на четность*** получается из ДД8421 кода присоединением к каждому кодовому набору дополнительного разряда *P* с тем, чтобы число единиц в любом кодовом наборе стало четным (возможно построение аналогичного кода формированием нечетного числа единиц в наборе). Дополнительный разряд *P* называется ***контрольным разрядом четности***.

• ***Код «2 из 5»***, состоящий из всех 10 возможных комбинаций пятиразрядных двоичных кодовых наборов с двумя единицами.

! За исключением представления нуля, код «2 из 5» является взвешенным, и может быть получен из [1; 2; 4; 7]-кода.

Для этих кодов число единиц в любом наборе четно. Если появляется одиночная ошибка, она преобразует допустимый кодовый набор в недопустимый (с нечетным количеством единиц). Вследствие этого сразу можно обнаружить наличие ошибки.

! Хотя проверка на четность предназначена для обнаружения лишь одиночных ошибок, на самом деле код с проверкой на четность обнаруживает любое нечетное число ошибок и некоторые варианты четного числа ошибок. Например, при получении кодового набора [10100] в ДД8421-коде с проверкой на четность, очевидно, что он ошибочен, хотя проверка именно на четность числа единиц этого не выявляет.

Примеры двоичных кодов с обнаружением ошибок

Десятичная цифра	ДД8421-код с проверкой на четность					Код “ 2 из 5”				
	8	4	2	1	P	0	1	2	4	7
0	0	0	0	0	0	0	0	0	1	1
1	0	0	0	1	1	1	1	0	0	0
2	0	0	1	0	1	1	0	1	0	0
3	0	0	1	1	0	0	1	1	0	0
4	0	1	0	0	1	1	0	0	1	0
5	0	1	0	1	0	0	1	0	1	0
6	0	1	1	0	0	0	0	1	1	0
7	0	1	1	1	1	1	0	0	0	1
8	1	0	0	0	1	0	1	0	0	1
9	1	0	0	1	0	0	0	1	0	1

В общем случае для *n*-разрядного кода с обнаружением ошибок (не обязательно ДД кодов) можно указать:

• В качестве допустимых комбинаций можно использовать не более половины из 2^n возможных комбинаций разрядов.
• Выбор допустимых кодовых наборов делается, чтобы для преобразования одного допустимого кодового набора в другой, по крайней мере, два разряда должны иметь противоположные значения.
• Для четырехразрядных кодовых наборов ($n = 4$) это условие означает, что из 16 возможных кодовых наборов можно выделить лишь 8 допустимых кодовых наборов. Поэтому, для построения кода с обнаружением ошибки для 10 десятичных цифр необходимо иметь, по крайней мере, 5 двоичных разрядов. В кодах "ДД8421 с проверкой на четность" и "2 из 5" использовано это минимальное количество разрядов.

Расстоянием между двумя кодовыми наборами в коде, называется число их несовпадающих разрядов (например, расстояние между наборами 1010 и 0100 равно трем). При этом ***минимальным кодовым расстоянием*** называется наименьшее число разрядов, которыми различаются любые два кодовых набора в коде. Например, минимальное кодовое расстояние для кодов "ДД8421" и "с

избытком три" равно 1, а для кодов "ДД8421 с проверкой на четность" и "2 из 5" оно равно двум. Так же можно показать, что код является кодом с обнаружением ошибок тогда, и только тогда, когда его минимальное расстояние не меньше двух.

✓ *Коды с исправлением ошибок*

Чтобы код позволял не только обнаружить, но и исправлять ошибки, необходимо еще увеличить его минимальное расстояние и, следовательно, количество разрядов.

Например, рассмотрим трехразрядный код ($n = 3$, $2^n = 8$), состоящий лишь из двух допустимых кодовых наборов 000 и 111, остальные 6 недопустимы. Минимальное кодовое расстояние равно трем. Если одиночная ошибка появляется в первом кодовом наборе, то он может преобразоваться в 100, 010 и 001, а если во втором, то могут получиться комбинации 011, 101 и 110. В данном случае можно не только зарегистрировать появление недопустимой комбинации, но и установить какому из исходных допустимых наборов она соответствует. Поэтому ошибка может быть не только обнаружена, но и исправлена.

Код называется ***кодом с исправлением ошибок*** (*m*-кратных ошибок), если всегда из кодового набора, образованного в результате *m*-кратной ошибки, можно восстановить исходный (правильный) кодовый набор.

Если минимальное кодовое расстояние равно 3, то любая одиночная ошибка переводит допустимый кодовый набор в недопустимый, находящийся на расстоянии, равном двум от любого другого допустимого кодового набора. Поэтому в коде с минимальным расстоянием 3, можно исправить любую одиночную ошибку. Аналогично, если код имеет минимальное расстояние, равное четырем, то он позволяет исправить одиночную и обнаружить двойную ошибку, либо же обнаружить тройную ошибку.

! Для исправления ошибок необходимо иметь возможность обнаруживать местоположение ошибочных разрядов. Если местоположение ошибки определено, то исправление ее производится путем инвертирования ошибочного разряда.

В теории кодирования разработан ряд эффективных и удобных для практического применения и технической реализации типов кодов с исправлением ошибок, например такие как **коды Рида-Соломона, Хэмминга** и другие (их рассмотрение выходит за рамки настоящего курса).

✓ Сложение в коде ДД8421

Поскольку многие из блоков цифровых электронных устройств предназначенные для выполнения операций над числами, работают с двоично-десятичным форматом представления чисел, целесообразно отметить алгоритмы сложения и вычитания в коде ДД8421.

При сложении двух чисел в ДД8421-коде, как и в случае обратных и дополнительных кодов, основной операцией является последовательное поразрядное сложение по правилам двоичной системы счисления (в части результатов суммирования в текущем разряде и переносов единицы в следующий), которые были рассмотрены в п. 1.3. Но поскольку запись чисел в ДД коде отличается от прямого двоичного кода, то после операции поразрядного

суммирования, полученный результат может не соответствовать истинному. Для получения правильного результата в ДД коде процедура суммирования должна быть дополнена операциями коррекции.

Учитывая структуру ДД коде можно определить конкретные условия возникновения некорректных комбинаций после поразрядного сложения и определить их признаки и требуемую коррекцию. Признаком необходимости коррекции в 4-х разрядной группе ДД8421-кода являются любое из двух условий:

• В группе сформировалась недопустимая комбинация (1010, 1100, 1101, 1110 или 1111). Такая ситуация возникает, когда сумма в текущей группе превышает 9, но не превышает 15. При поразрядном сложении в четырехразрядной группе возникают недопустимые комбинации, соответствующая десятичным числам 10, 11, 12, 13, 14 или 15, которые не могут присутствовать в ДД8421-коде и должны корректироваться.

• Происходит перенос единицы из данной группы в следующую. Такая ситуация возникает, когда сумма текущей группе превышает 15, в результате чего допустимая, но неверная комбинация. С точки зрения сложения в двоичной системе в следующую группу переносится единица 5-го разряда, т.е. 16, а т.к. следующая группа ДД кода соответствует одной десятичной цифре, то в нее "поступает" только 10, что приводит к некорректному результату.

С учетом анализа механизмов возникновения некорректного результата после поразрядного сложения, нетрудно указать необходимые процедуры коррекции. Для коррекции результата нужно прибавить 0110 одновременно ко всем группам, в которых возник любой из двух признаков необходимости коррекции ($0110_2 = 6_{10}$ разница между максимальным числом четырехразрядной двоичной комбинации – 15 и максимальным числом в десятичном разряде – 9). Если после корректирующей процедуры сложения в каких-либо группах вновь возникают недопустимые комбинации, коррекцию нужно провести снова. В итоге будет получен правильный результат суммирования *в ДД8421-коде*.

Примеры:

а) Сложить в ДД8421-коде десятичные числа 27 и 36.

Прежде всего можно указать числа в ДД коде: $27_{10} = 00100111_{ДД8421}$ и $36_{10} = 00110110_{ДД8421}$.

		+1←			+1←	+1←		← переносы;
	0	0	1	0	0	1	1	1	← 27;
+	0	0	1	1	0	1	1	0	← 36;
	0	1	0	1	1	1	0	1	← запрещенная комбинация 1101;
+									
	0	0	0	0	0	1	1	0	← коррекция (+0110);
	1	1	1	0	0	0	1	1	← результат в ДД8421-коде

Результат: $01100011_{ДД8421} = 63_{10}$

! Переносы возникающие в ходе коррекции здесь не показаны.

б) Сложить в ДД8421-коде десятичные числа 28 и 59.

ДД код: $28_{10} = 00101000_{ДД8421}$ и $59_{10} = 01011001_{ДД8421}$.

		+1←	+1←	+1←					← переносы;
+		0	0	1	0	1	0	0	0	← 28;
		0	1	0	1	1	0	0	1	← 59;
		1	0	0	0	0	0	0	1	← перенос из первой группы во вторую;
+										
		0	0	0	0	0	1	0	0	← коррекция (+0110 к первой группе);
		1	0	0	0	0	1	1	1	← результат в ДД8421-коде.

Результат: $10000111_{ДД8421} = 87_{10}$.

! Для некоторых вариантов суммируемых чисел, в результате поразрядного суммирования не возникает признаков, требующих коррекции. Это значит, что правильный результат в ДД коде формируется сразу после поразрядного суммирования. Так происходит, если суммы во всех *десятичных* разрядах исходных десятичных числе имеют значения менее 10.

✓ Вычитание в коде ДД8421

Алгоритм вычитания чисел в ДД8421-коде (здесь подразумевается, что из большего по модулю числа вычитается меньшее по модулю число) как и алгоритм сложения предполагает сначала проведение последовательного поразрядного вычитания, а затем коррекцию.

Признаком необходимости выполнения коррекции в случае вычитания является только наличие заема из одной группы в другую (т.е. заема из младшего разряда одной группы в старший разряд предыдущей группы. Если произошел такой заем, результат будет неправильным и требуется коррекция в виде вычитания комбинации 0110 из группы, которая получила заем.

Причина искажения результата поразрядного вычитания и метода коррекции понятна. Получение четырехразрядной двоичной группой единицы из более старшего разряда означает поступление числа 16_{10}, однако отбор единицы младшего разряда более старшей группы означает уменьшение более старшего десятичного разряда ДД кода на единицу и получение более младшим десятичным разрядом 10-ти. В результате значение в группе, получившей заем оказывается на 6 больше правильного, при этом комбинация может оказаться как допустимой, так и недопустимой. Кроме того, нетрудно понять, что результат поразрядного вычитания в группе, получившей заем, всегда будет более 6 (в двоичном коде более, чем 0110).

! Если после поразрядного вычитания не возникает заема между группами, то правильный результат в ДД коде формируется сразу после поразрядного вычитания.

Пример:

а) Из 61 вычесть 38 в ДД8421-коде.
Прежде всего запишем числа в ДД коде:
$61_{10} = 01100001_{ДД8421}$ и $38_{10} = 00111000_{ДД8421}$.

		→1	→1	→1				← переносы;
–	0	1	1	0	0	0	0	1	← 27;
	0	0	1	1	1	0	0	0	← 36;
	0	0	1	0	1	0	0	1	← есть заем в первую группу из второй;
–					0	1	1	0	← коррекция (– 0110);
	0	0	1	0	0	0	1	1	← результат в ДД8421-коде.

Результат: $01100011_{ДД8421} = 23_{10}$

Заемы, возникающие в ходе коррекции, здесь не показаны.

1.5. Алфавитно-цифровые коды

Выше рассматривалось кодирование двоичными разрядами непосредственно числовой информации, но в цифровых электронных системах необходимо вводить и обрабатывать нечисловую (символьную) информацию. Для этого разрабатывают специальные коды.

Наибольшее распространение получили коды ASCII (American Standart Code for Information Interchange – Американский стандартный код для обмена информацией), ANSI, UNICODE и некоторые другие.

С помощью ***семибитного ASCII*** кода (символам ставится в соответствие порядковый номер и семиразрядный двоичный код) закодированы 128 символов. Это буквы алфавита, десятичные цифры, специальные символы, такие как знаки препинания, "%", "+", "=", а также символы управляющих операций, таких как “конец передачи” (EOT – End of Transission), “конец текста” (EXT – End of Text) и т.д. Иногда этот код дополняют восьмым битом для обеспечения контроля четности.

Чаще для кодирования символов используют восемь бит (т.е. один байт), что позволяет закодировать 256 различных символов. В таких кодах обычно первые 128 символов (с номерами от 0 до 127), как правило, совпадают с ASCII кодом. Отличие имеется в символах со 128 по 255. Эти символы используются для кодирования букв национальных алфавитов, а также некоторых специальных символов, в частности – псевдографики.

В литературе и в сети INTERNET нетрудно найти таблицы ASCII и ANSI кодов, где указаны предусмотрены этими кодами символы и присваиваемые им номера. Так же нетрудно найти описания кода KOI-8R, UNICODE и других.

1.6. Арифметика повышенной точности и арифметика с плавающей запятой

Сложные цифровые электронные устройства (микропроцессорные системы, компьютеры) оперируют данными, представляемыми машинными словами фиксированной длины. Длина машинного слова связана с техническими особенностями построения элементов цифрового электронного устройства, реализующего вычисления. В настоящее время типичными являются слова длиной 8, 12, 16, 32 и 64 бит. Восьмибитовое слово используется столь широко, что даже получило специальное название "**байт**". Длина слова, с которыми оперирует система, является мерой "мощности" преобразования данных.

Другой мерой "мощности" преобразования данных является количество слов или байтов памяти, к которым может адресоваться система. Например, для 16-разрядной цифровой системы число слов памяти в простейшем варианте будет 2^{16-1}= 32768. Часто применяются сокращение обозначений этих чисел, например, 32768 слов обозначаются 32К. Буква К или, например, М – происходят от десятичных приставок "кило" и "мега" и в данном контексте означают, соответственно $\cdot 2^{10}$=1024 и $\cdot 2^{20}$=1048576.

При обработке числовых данных длина слова задает диапазон возможных для представления чисел и, как результат, ограничивает точность вычислений. Иногда длина слова недостаточна для достижения требуемой точности вычисления. Например, 8-разрядные устройства позволяют использовать целые числа в диапазоне от –127 до +127. Очевидно, что для большинства задач такой небольшой диапазон неприемлем (ошибка может иметь порядок $1/254 \approx 0{,}4\%$). Но используя два 8-битовых слова, получается уже диапазон от –32768 до +32767 и ошибки могут сводиться к уровню одной шестидесятитысячной или $\pm 0.0015\%$. Организацию вычислений с представлением чисел посредством двух машинных слов называют ***двойной точностью***. В некоторых случаях используют тройную точность. Поскольку длина машинного слова связана техническими параметрами элементов электронного вычислителя, то реализация вычислений с двойной (и выше) точностью требует усложнения алгоритмов обработки числовых данных в электронном устройстве, поэтому такие вычисления могут быть требовать больших временных ресурсов, задействовать больший объем памяти и т.п.

Если необходимо иметь дело с числами, изменяющимися в намного более широком диапазоне значений, используется система представления числа ***с плавающей запятой***. В такой системе числа представляются в виде $A \cdot r^C$, где A – ***мантисса***, r – основание системы счисления ($r = 10$ – для десятичной системы, $r = 2$ – для двоичной), C – ***порядок***.

Пример: 4900000 можно записать как $0{,}49 \cdot 10^7$, где 0,49 – значение мантиссы, 7 – порядок, а число $0{,}00023 = 0{,}23 \cdot 10^{-3}$

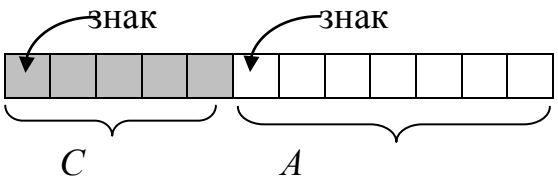
Существуют разные правила определения пределов значений мантиссы. В приведенных выше примерах подразумевалось, что мантисса может принимать значения в интервале $[1/r; 1[$. В соответствие с этим, "нормальной" формой числа считают представление с мантиссой, изменяющейся в диапазоне от 0,1 до 0,999…

для десятичного основания и от 0,5 до 0,999… для двоичного основания (в двоичном коде от 0,1 до 0,111….).

Другой распространенный вариант подразумевает изменение мантиссы в диапазоне [1; *r*[. В десятичной системе это значения от 1 до 9,999…, в двоичной от 1 до 1,999… (в двоичном коде от 1 до 1,111…).

! Обычно часть машинного слова, соответствующую порядку, называют **характеристикой**, а мантиссе – **дробью**.

В качестве примера рассмотрим использование плавающей запятой при 12-ти разрядном машинном слове. Пусть для характеристики отведено 5, а для дроби 7 разрядов (в обеих частях слова старший разряд – для знака):



Здесь значение мантиссы *A* представлено 7-ю разрядами: знак и шестиразрядная дробная часть двоичного числа. Значение порядка *C* представлено в 5-ти разрядах: знак и целое четырехразрядное двоичное число.

Примеры:

0 01110 1010 11 *C* =+7; *A* =11	$2^7 \cdot (2^{-1}+2^{-3}+2^{-5}+2^{-6}) = 2^7 \cdot (0{,}671875) = 86$
0 00111 1001 11 *C* =+3; *A* = –7	$-2^3 \cdot (2^{-1}+2^{-4}+2^{-5}+2^{-6}) = -2^3 \cdot (0{,}609375) = -4{,}875$
1 01010 1110 0 *C* =–5; *A* =+5	$2^{-5} \cdot (2^{-1}+2^{-2}+2^{-3}) = 2^{-5} \cdot (0{,}875) = 0{,}0274375$
1 01101 1010 01 *C* =–6; *A* = –9	$-2^{-6} \cdot (2^{-1}+2^{-3}+2^{-6}) = -2^{-6} \cdot (0{,}640625) = -0{,}0100097656$

Для 12-разрядного слова дроби должны иметь значения в пределах от $-0{,}111111_2$ до $0{,}111111_2$, а порядок от -1111_2 до 1111_2. Таким образом, можно представить числа от примерно от $-0{,}984 \cdot 2^{15}$ до $+0{,}984 \cdot 2^{15}$, т.е. примерно от $-3{,}2 \cdot 10^4$ до $+3{,}2 \cdot 10^4$. Наименьшим значением дробной части будет $0{,}1000000_2$, равное 2^{-1}, а наименьшей характеристикой –15, поэтому наименьшее представляемое число равно $2^{-1} \cdot 2^{-15}$ или 2^{-16}.

В другом варианте представления чисел с плавающей запятой мантисса *A* является целым числом, а часть слова, соответствующую мантиссе, называют **целой частью**. Целая часть числа с плавающей запятой представляет некоторое значение в виде его величины со знаком. Значение рассматриваемого числа по-прежнему равно $A \cdot 2^C$.

Пусть, как и в предыдущих примерах, для характеристики отведено 5, а для целой части 7 разрядов. С учетом двух разрядов, отведенных для знака мантиссы и порядка, в этом случае представление чисел с плавающей запятой допускает хранение в двоичном слове действительных чисел в диапазоне от $-63 \cdot 2^{15}$ до $63 \cdot 2^{15}$. Наименьшее по абсолютной величине число составляет 2^{-15}.

Примеры:

001110001011	$2^7 \cdot 11 = 1408$
$C = +7 \quad A = 11$	
000111000111	$2^3 \cdot (-7) = -56$
$C = +3 \quad A = -7$	
101010000101	$2^{-5} \cdot 5 = 5/32$
$C = -5 \quad A = +5$	
101101001001	$2^{-6} \cdot (-9) = -9/64$
$C = -6 \quad A = -9$	

В современной цифровой электронике используются и другие форматы представления чисел с плавающей запятой. При этом используемая система представления чисел с плавающей запятой учитывается при организации вычислений и применении вычислительных алгоритмов. В частности умножение $a \cdot 10^m$ на $b \cdot 10^n$ дает произведение $(a \cdot b) \cdot 10^{m+n}$. Чтобы разделить $a \cdot 10^m$ на $b \cdot 10^n$, частное надо записать как $a/в \cdot 10^{m-n}$. Для сложения чисел $a \cdot 10^m$ и $b \cdot 10^n$ необходимо сначала добиться равенства m и n (если $m = n$, то $a \cdot 10^m + b \cdot 10^n = (a+b) \cdot 10^m$). Процесс, приводящий к равенству m и n, называется ***выравниванием порядков*** (выравнивание порядков реализуется с применением специальных соответствующих алгоритмов).