Лабораторная работа №2:

«Машинные коды»

Цель работы: Изучить основы машинной арифметики, представления чисел в прямом, обратном и дополнительном кодах и арифметических операций над ними.

Общие сведения из теории

Для представления информации в памяти ЭВМ (как числовой, так и нечисловой) используется двоичный способ кодирования.

Элементарная ячейка памяти ЭВМ имеет длину 8 бит (байт). Каждый байт имеет свой номер (его называют адресом). Наибольшую последовательность бит, которую ЭВМ может обрабатывать как единое целое, называют машинным словом. Длина машинного слова зависит от разрядности процессора и может быть равной 16, 32 битам и т.д.

Для кодирования символов достаточно одного байта. При этом можно предста-вить 256 символов (с десятичными кодами от 0 до 255). Набор символов персо-нальных ЭВМ IBM PC чаще всего является расширением кода ASCII (American Standart Code for Information Interchange — стандартный американский код для обмена информацией).

В некоторых случаях при представлении в памяти ЭВМ чисел используется смешанная двоично-десятичная «система счисления», где для хранения каждого десятичного знака нужен полубайт (4 бита) и десятичные цифры от 0 до 9 представляются соответствующими двоичными числами от 0000 до 1001. Например, упакованный десятичный формат, предназначенный для хранения целых чисел с 18 значащими цифрами и занимающий в памяти 10 байт (старший из которых знаковый), использует именно этот вариант.

В ЭВМ применяются две формы представления чисел:

- **естественная форма**, или **форма с фиксированной запятой (точкой) — ФЗ (ФТ)**;
- **нормальная форма**, или **форма с плавающей запятой (точкой) - ПЗ (ПТ).**

Для представления чисел в ЭВМ обычно используют битовые наборы — последовательности нулей и единиц фиксированной длины. Организовать обработку наборов фиксированной длины технически легче, чем наборов переменной длины. Позиция в битовом наборе называется ***разрядом*.**

1. Представление целых чисел

Целые числа в компьютере хранятся в формате с **фиксированной запятой.** Для записи числа с фиксированной точкой машинное слово (операнд) делится на два фиксированных поля (части) поле знака и поле цифр.

Эта форма наиболее проста, естественна, но имеет небольшой диапазон представления чисел и поэтому чаще всего неприемлема при вычислениях.

Необходимость кодирования чисел определяется тем, что в ЭВМ нельзя вводить числа в том виде, в котором они изображаются человеком на бумаге. Существует несколько соглашений о едином формате представления как положительных так и отрицательных чисел.

Для алгебраического представления чисел, т. е. для представления чисел с учетом их знака, в вычислительных машинах используются специальные коды:

- прямой код числа;
- обратный код;
- дополнительный код.

Все три способа используют самый левый (старший) разряд битового набора длины *k* для кодирования знака числа: знак “плюс” кодируется нулем, а “минус” — единицей. Остальные разряды (называемые цифровой частью) используются для представления абсолютной величины числа.

Прямой код числа $P_{пр}(X)$- это такое представление двоичного числа *x*, при котором знак «+» кодируется нулем в старшем разряде числа, знак «-»- единицей. При этом старший разряд называется **знаковым.**

Пример: Получить прямой код для числа 5 и -5

$+5 = +101 = 0'101_2$

$-5 = -101 = 1'101_2$

Здесь апострофом для удобства определения знака я отделяю знаковые разряды.

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

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

Обратный код для положительных чисел совпадает с прямым кодом. Чтобы представить отрицательное двоичное число в обратном коде, нужно оставить в знаковом разряде 1, во всех значащих разрядах заменить 1 на 0, а 0 на 1. Такая операция называется инвертированием и обозначается горизонтальной чертой над инвертируемым выражением.

Пример: Получить обратный код для числа -11

$P_{пр}(X) = 1'1011_2$

$P_{обр}(X)=1'0100_2$

Дополнительный код $P_{доп}(X)$ образуется следующим образом:

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

Дополнительный код для положительных чисел совпадает с прямым кодом.

Дополнительный код отрицательных чисел представляет собой суммирование обратного кода числа с единицей младшего разряда.

Пример: Получить дополнительный код для числа -13

$P_{пр}(X)= 1'1101_2$

$P_{обр}(X)= 1'0010_2$

$P_{доп}(X)= 1'0011_2$

Иногда возникает такая ситуация, когда под поле цифр разрядов выделено больше, чем это необходимо для представления числа X.

Пусть сетка имеет в составе t двоичных разрядов. Один из разрядов (самый старший) резервируется для знака числа. Сам знак кодируется следующим образом: «-» соответствует 1, «+» соответствует 0. Само число представляется в двоичной системе счисления и заносится в разрядную сетку так, что его самый младший разряд помещается в самый младший разряд сетки, а последующие разряды – следом. Если число имеет меньшее количество разрядов, чем может вместить разрядная сетка, оставшиеся незаполненными разряды заполняются нулями.

Пример: Пусть разрядная сетка имеет 8 двоичных разрядов. Разместить в ней двоичное число -10111_2.

Для начала представим число в прямом коде. $P_{пр}(X)= 1'10111_2$

номера разрядов	7	6	5	4	3	2	1	0
содержимое разрядов	1	0	0	1	0	1	1	1

Здесь разряд 7 – знаковый, имеет значение 1, поскольку исходное число отрицательно. В разрядах 0 – 4 размещено само исходное число, разряды 5 и 6 заполнены дополнительными нулями.

Пример: Пусть разрядная сетка имеет 8 двоичных разрядов. Разместить в ней двоичное число 10111_2.

Для начала представим число в прямом коде. $P_{пр}(X)= 0'10111_2$

номера разрядов	7	6	5	4	3	2	1	0
содержимое разрядов	0	0	0	1	0	1	1	1

Но с отрицательными числами в дополнительном коде такая схема работает не правильно.

Преобразование дополнительного кода при расширении разрядной сетки выполняется следующим образом: Если исходное число было положительным, то все дополнительные биты заполняются нулями, а если отрицательным - единицами. Эта операция называется **расширением знака.**

Пример: Пусть разрядная сетка имеет 8 двоичных разрядов. Разместить в ней двоичное число -10010_2 в дополнительном коде.

Для начала преобразуем число в прямой, обратный и дополнительный коды, получим:

$P_{пр}(X)= 1'10010_2$

$P_{обр}(X)= 1'01101_2$

$P_{доп}(X)= 1'01110_2$

номера разрядов	7	6	5	4	3	2	1	0
содержимое разрядов	1	1	1	0	1	1	1	0

2. Представление вещественных чисел

Вещественными числами (в отличие от целых) в компьютерной технике называются числа, имеющие дробную часть. При их изображении вместо запятой принято ставить точку. Так, например, число 5 — целое, а числа 5.1 и 5.0 — вещественные.

Для представления вещественных чисел в современных компьютерах принят способ представления **с плавающей запятой.**

В форме представления с плавающей запятой (точкой) число изображается в виде двух групп цифр:

- мантисса;
- порядок.

Абсолютная величина мантиссы должна быть меньше 1, а порядок- целым числом.

Число в форме с плавающей точкой может быть представлено в следующем виде:

$\mathbf{N = \pm M \cdot P^{\pm r}}$,

где **M** — мантисса числа;

p — основание СС;

r — порядок числа- показывает, на сколько разрядов и в какую сторону сдвинута запятая при замене формы записи числа.

Число может иметь множество представлений в форме с плавающей запятой, например, число 12 может быть записано в видах: $0{,}12 \cdot 10^2$, $0{,}012 \cdot 10^3$, $0{,}0012 \cdot 10^4$ и так далее.

Из приведенного выше примера видно, что благодаря изменению порядка точка перемещается по мантиссе. При этом, если порядок положительный, точка смещается по мантиссек ВЛЕВО, а если отрицательный- ВПРАВО.

Все числа с плавающей запятой хранятся в машине в нормализованном виде.

Нормализованное число- такое число, старший разряд мантиссы которого больше нуля, то есть первый разряд мантиссы после запятой не может равняться нулю.

$-348 = -0{,}348 \cdot 10^5$ $+1427 = +0{,}1427 \cdot 10^4$

$+54{,}7 = +0{,}547 \cdot 10^2$ $+721{,}355 = +0{,}721355 \cdot 10^3$

$+0{,}00058 = +0{,}58 \cdot 10^{-3}$ $+0{,}00328 = +0{,}328 \cdot 10^{-2}$

$+300 = +0{,}3 \cdot 10^3$ $+30 = +0{,}3 \cdot 10^2$

$-10301{,}202 = -0{,}10301202 \cdot 10^5$ $+0{,}00056 = +0{,}56 \cdot 10^{-3}$

3. Арифметические операции над числами с фиксированной точкой

Сложение (вычитание). Операция вычитания приводится к операции сложения путем преобразования чисел в обратный или дополнительный код. Знаковые разряды участвуют в операции сложения наравне с цифровыми.

Требуемая операция	Необходимое преобразование
A+B	A+B
A-B	A+(-B)
-A+B	(-A)+B
-A-B	(-A)+(-B)

При выполнении сложения цифр необходимо соблюдать следующие правила:

1. Слагаемые размещаются в равных разрядных сетках в прямых кодах. Для выравнивания разрядной сетки слагаемых можно дописывать незначащие нули слева к целой части числа и незначащие нули справа к дробной части числа.

2. Слагаемые складываются по правилам сложения двоичных чисел. При этом знаковые разряды участвуют в вычислениях наряду с числовыми;.

3. Если выполняется операция сложения в обратных кодах, единица переноса из знакового разряда суммы прибавляется к ее младшему разряду (т. е. выполняется циклический перенос). При использовании ДК единица переноса теряется.

5. Если результат положителен – он представлен в прямом коде и не требует никаких преобразований. Если результат отрицателен, то он представлен в обратном или дополнительном коде в зависимости от того, в каком коде происходило сложение. Результат в таком случае преобразуется в прямой код.

Пример 1. Сложить в обратном коде числа –34 и +15.

Преобразуем слагаемые в прямые коды и разместим их в разрядных сетках:

$-34 = -100010_2$

1	0	1	0	0	0	1	0

$+15 = +1111_2$

0	0	0	0	1	1	1	1

Преобразуем отрицательное слагаемое -34 в обратный код:

1	1	0	1	1	1	0	1

Складываем слагаемые:

1	1	0	1	1	1	0	1
0	0	0	0	1	1	1	1

Получаем:

1	1	1	0	1	1	0	0

Единица переноса не образована; судя по знаку, результат отрицателен, значит, представлен в обратном коде (поскольку сложение выполнялось в этом коде) и требует перевода в прямой код:

1	0	0	1	0	0	1	1

Таким образом, получено число -10011_2. Для проверки правильности результата представим его в десятичной системе счисления. Имеем: $-10011_2 = -19$, что соответствует правильному результату.

Пример 2. Сложить в обратном коде числа –34 и -15. Разрядная сетка – 8 бит.

Преобразуем слагаемые в прямые коды и разместим их в разрядных сетках:

$-34 = -100010_2$

1	0	1	0	0	0	1	0

$-15 = -1111_2$

1	0	0	0	1	1	1	1

Преобразуем отрицательные слагаемые в обратный код:

для -34

1	1	0	1	1	1	0	1

для -15

1	1	1	1	0	0	0	0

Складываем слагаемые:

1	1	0	1	1	1	0	1
1	1	1	1	0	0	0	0

Получаем:

1	1	0	0	1	1	0	1

1

Образовалась единица переноса из знакового разряда. В соответствии с правилами сложения в обратном коде, она прибавляется к младшему числовому разряду, получаем:

11001110

Судя по знаку, результат отрицателен, значит, представлен в обратном коде (поскольку сложение выполнялось в этом коде) и требует перевода в прямой код:

10110001

Таким образом, получено число -110001_2. Для проверки правильности результата представим его в десятичной системе счисления. Имеем: $-110001_2 = -49$, что соответствует правильному результату.

Пример 3. Сложить в дополнительном коде числа -13 и +8

Преобразуем слагаемые в прямые коды и разместим их в разрядных сетках:

$-13 = -1101_2$

1	1	1	0	1

$+8 = +1000_2$

0	1	0	0	0

Преобразуем отрицательное слагаемое -13 в дополнительный код:

1	0	0	1	1

Складываем слагаемые:

1	0	0	1	1
0	1	0	0	0

Получаем:

1	1	0	1	1

В знаковом разряде стоит единица, значит, результат получен в дополнительном коде. Для перехода к прямому коду необходимо выполнить следующее преобразование: От младшего разряда полученного дополнительного кода отнять единицу, получим:

$P_{обр}(x) = P_{доп}(X) - 1 = 11011 - 1 = 11010$.

$P_{пр}(X) = P_{обр}(X) = 11010 = 10101$.

Пример 4. Сложить в дополнительном коде числа –34 и -15. Разрядная сетка – 8 бит.

Преобразуем слагаемые в прямые коды и разместим их в разрядных сетках:

$-34 = -100010_2$

1	0	1	0	0	0	1	0

$-15 = -1111_2$

1	0	0	0	1	1	1	1

Преобразуем отрицательные слагаемые в дополнительный код:

для -34

1	1	0	1	1	1	1	0

для -15

1	1	1	1	0	0	0	1

Складываем слагаемые:

1	1	0	1	1	1	1	0
1	1	1	1	0	0	0	1

Получаем:

1	1	0	0	1	1	1	1

1

Образовалась единица переноса из знакового разряда. Однако, поскольку сложение выполняется в дополнительном коде, единица переноса из знакового разряда теряется.

Таким образом, мы получили результат сложения в дополнительном коде. Поскольку он отрицателен, преобразуем его в прямой код. Тогда имеем:

$P_{обр}(x)= P_{доп}(X) -1=11001111-1=11001110.$

$P_{пр}(X)= P_{обр}(X)= 11001110=10110001.$

Результат совпадает с результатом примера 2.

Иногда при сложении в обратном или дополнительном коде возникают ситуации переполнения.

Пример 5. Сложить в обратном коде числа –64 и -67. Разрядная сетка – 8 бит.

Преобразуем слагаемые в прямые коды и разместим их в разрядных сетках

Для -64

1	1	0	0	0	0	0	0

Для -67

1	1	0	0	0	0	1	1

Преобразуем отрицательные слагаемые в обратный код:

для -64

1	0	1	1	1	1	1	1

Для -67

1	0	1	1	1	1	0	0

Складываем слагаемые:

1	0	1	1	1	1	1	1
1	0	1	1	1	1	0	0

Получаем:

0	1	1	1	1	0	1	1

1

Образовалась единица переноса из знакового разряда.

После сложения единицы переноса имеем окончательный результат:

01111100

Анализ показывает, что результат положительный, что противоречит исходным данным: складывались два отрицательных числа. Это свидетельствует о переполнении (overflow) разрядной сетки.

Таким образом, формальным признаком переполнения разрядной сетки при выполнении операции сложения является то, что знак результата отличается от знаков слагаемых. Такая ситуация может возникнуть только при сложении чисел с одинаковыми знаками. С подобными ситуациями при сложении целых чисел самостоятельно компьютер не справляется, требуется вмешательство программиста.

Умножение. Умножение двоичных чисел наиболее просто реализуется в прямом коде и приводится к операциям сложения и сдвигам.

Умножение производится путем образования частичных произведений и последующего их суммирования. Каждое частичное произведение равно нулю, если в соответствующем разряде множителя стоит 0, или равно множимому, сдвинутому на соответствующее число разрядов влево, если в разряде множителя стоит 1.

Пример 6. Умножить два числа 7 и 5.

Перемножим эти числа, представленные прямыми двоичными кодами, так же, как это делается в десятичной системе.

7=+111=0111

5=+101=0101

Результат: 0100011=+100011 что соответствует числу 35 в десятичной СС.

Знак произведения формируется по известному правилу: (+)•(+) = (+); (+)•(-) = (-); (-)•(+) = (-);(—)•(—) = (+). В ЭВМ, как было указано в , знак « —» числа кодируется единицей, а «+» — нулем, но правило формирования знака сохраняется. Операция, которую реализует ЭВМ для определения знака произведения, называется суммой по модулю два и обозначается $\oplus$: $0 \oplus 0 = 0$; $0 \oplus 1 = 1$; $1 \oplus 0 = 1$;$1 \oplus 1 = 0$.

Деление. Выполняется путем многократных сдвигов и вычитаний

$[A_2]_п = 101101$
$[B_2]_п = 101$

Делимое	Делитель
101101	| 101
-101	1001 – частное
0101	
- 101	
0	

$[C_2]_п = 1001$
$C_{10} = 9$

Деление произведено так же, как это делается обычно в десятичной системе. Сначала проверяется, можно ли вычесть значение делителя из старших разрядов делимого. Если возможно, то в разряде частного записывается единица и определяется частная разница. В противном случае в частное записывается нуль и разряды делителя сдвигаются вправо на один разряд по отношению к разрядам делимого. К полученной предыдущей разнице сносится очередная цифра делимого, и данный процесс повторяется, пока не будет получена необходимая точность.

Делимое перед операцией деления должно быть приведено к 2n-разрядной сетке. Только в этом случае при делении на n-разрядный делитель получается n-разрядное частное.

4. Арифметические операции над двоичными числами с плавающей точкой

Представление чисел	Арифметические операции
$X = X_M \cdot p^{Xr}$ $Y = Y_M \cdot p^{Yr}$	$X+Y = (X_M \cdot p^{Xr-Yr} + Y_M) \cdot p^{Yr}$ $X+Y = (X_M \cdot p^{Xr-Yr} - Y_M) \cdot p^{Yr}$ $X \cdot Y = (X_M \cdot Y_M) \cdot p^{Xr+Yr}$ $\frac{X}{Y} = (\frac{X_M}{Y_M}) \cdot p^{X_r - Y_r}$

Примеры:
$X = 0.3 \cdot 10^2 = 30$;
$Y = 0.2 \cdot 10^3 = 200$;

$X + Y = (0.3 \cdot 10^{2-3} + 0.2) \cdot 10^3 = 0.23 \cdot 10^3 = 230;$
$X - Y = (0.3 \cdot 10^{2-3} - 0.2) \cdot 10^3 = (-0.17) \cdot 10^3 = -170;$
$X \cdot Y = (0.3 \cdot 0.2) \cdot 10^{2+3} = 0.06 \cdot 10^5 = 6000;$
$X / Y = (0.3 / 0.2) \cdot 10^{2-3} = 1.5 \cdot 10^{-1} = 0.15$

Задания к лабораторной работе

В соответствии с выданным вариантом задания преподавателем выполнить:

1. Переведите данное число из десятичной системы счисления в двоично-десятичную.
2. Переведите данное число из двоично-десятичной системы счисления в десятичную.
3. Зашифруйте данный текст, используя таблицу ASCII-кодов
4. Дешифруйте данный текст, используя таблицу ASCII-кодов
5. Представьте числа в прямом, обратном, дополнительном кодах
6. Выполните сложение чисел в обратом и дополнительном кодах
7. Представьте числа в нормализованном виде
8. Выполните четыре арифметических действия над числами в формате с плавающей точкой

Контрольные вопросы:

1. Что такое кодирование информации в общем смысле?
2. Каково место кодирования среди процессов обработки информации?
3. Что такое код? Приведите примеры кодирования и декодирования.
4. Как получить прямой и дополнительный коды целого числа?
5. Как представляются действительные числа в памяти ЭВМ?
6. Какие коды называются двоичными? Приведите примеры.
7. Какой код используется для кодирования букв латинского алфавита буквами персонального компьютера?
8. Какие коды используются в вычислительной технике для кодирования букв русского алфавита?