Лабораторная работа

МИКРОПРОГРАММИРОВАНИЕ АРИФМЕТИЧЕСКИХ ОПЕРАЦИЙ

Цель работы:

изучение способов представления чисел в микроЭВМ и алгоритмов арифметических операций;
микропрограммирование операций в системе микрофункций процессора К1804ВС1.

Основные положения.

Представление чисел в микроЭВМ.

Отрицательные числа обычно представляются в виде дополнений до основания системы счисления. При операциях над числами в микроЭВМ обычно полагают, что числа имеют следующий вид:

$$D = d_{n-1}d_{n-2}\dots d_1d_0$$

то есть точка находится справа и числа являются целыми.

В общем случае дополнение любого n – разрядного числа D до основания b системы счисления можно получить путем вычитания D из b^n. Если D находится в пределах от 1 до b^n –1, то при вычитании получается другое число в тех же пределах. Если D=0, то результат вычитания равен b^n и имеет вид 100..0 при общем числе разрядов, равном (n+1). Отбросив цифру старшего разряда, получим 0. Следовательно, в системе представления чисел дополнением до основания системы счисления существует только одно представление 0.

В десятичной системе счисления дополнение до основания есть дополнение до десяти, которое можно получить путем вычитания n – разрядного числа из 10^n. /*Пример*. Десятичное число A = 1849. Дополнение до десяти $[A]_{доп}$ = 10000-A= 8151/.

Для двоичных чисел дополнение до основания системы счисления называется дополнением до двух. В системе представления дополнением до двух, или в дополнительном коде, число является положительным, если значение старшего разряда d_{n-1} = 0, и отрицательным, если d_{n-1} = 1. Десятичный эквивалент двоичного числа, представленного дополнением до двух, вычисляется так же, как и для числа без знака, за исключением того, что вес старшего разряда равен – $2^{(n-1)}$, а не $+2^{(n-1)}$. Представляемые числа находятся в диапазоне от – $2^{(n-1)}$ до $+2^{(n-1)}$–1.

В системе представления чисел неполным дополнением до основания дополнение n – разрядного числа D получается путем его вычитания из b^n-1. Для двоичных чисел неполное дополнение называется дополнение до единицы или обратным кодом. При вычислении десятичного эквивалента числа, записанного как дополнение до единицы, старшему разряду приписывается вес – ($2^{(n-1)}$ – 1), а не –$2^{(n-1)}$. Представляемые числа находятся в диапазоне от – ($2^{(n-1)}$ – 1) до + ($2^{(n-1)}$ – 1). Нуль имеет два представления - положительный нуль (00..00) и отрицательный нуль (11..11).Представления положительных чисел в системах с дополнением до единицы и до двух совпадают, тогда как представления отрицательных чисел отличаются на 1.

Сложение и вычитание чисел в дополнительном коде.

Графическое представление 4–разрядных двоичных чисел в дополнительном коде приведено на рис.1. Сложение с положительными числами легко интерпретировать, перемещая указатель по часовой стрелке на + n позиций; вычитание (-***n***), перемещая указатель против часовой стрелки, или перемещая по часовой стрелке на (16–***n***) позиций, что равносильно замене вычитания сложением с дополнением числа до двух. Если при сложении получают результат, который выходит за пределы диапазона представляемых чисел, то имеет место переполнение.

Правило выявления переполнения. При сложении переполнение происходит только в том случае, если слагаемые имеют одинаковые знаки, а знак суммы отличается от знака слагаемых. Правило переполнения можно сформулировать иначе, используя понятие переносов, возникающих при сложении. Переполнение возникает, если значения переносов в знаковый разряд и из знакового разряда различны. Из анализа рис.1 следует, что переполнение возникает при сложении в случае, если указатель перейдет границу между позициями +7 и –8.

Числа в дополнительном коде складываются и вычитаются так же, как числа без знака той же длины. Поэтому для выполнения операций над числами обоих типов необходим всего один тип команды сложения или вычитания. Различие заключается лишь в том, что результаты операций интерпретируются по-разному в зависимости от того, какими числами оперирует ЭВМ: числами со знаком (то есть от –8 до +7) или без знака (от 0 до 15).

На рис.2 приведено графическое представление 4–разрядных двоичных чисел без знака. Из него видно, что двоичные кодовые комбинации занимают те же позиции, что и на рис.1, а сложение и вычитание можно осуществить, перемещая указатель на **n** позиций в том или ином направлении. При сложении чисел без знака результат выходит за пределы диапазона представления при переходе границы между 15 и 0. В этом случае говорят о возникновении переноса из старшего разряда. При вычитании чисел без знака результат выходит за пределы диапазона при переходе границы между 0 и 15. В этом случае возникает заем. Но так как вычитание ***n*** можно заменить сложением с дополнительным кодом числа ***n***, равным (16 – ***n***), то заем возникает при отсутствии переноса.

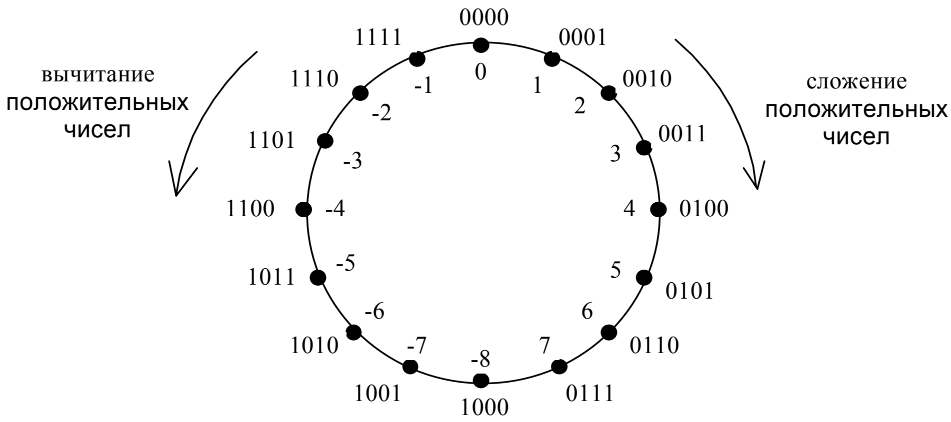
вычитание положительных чисел
сложение положительных чисел
0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111
0
1
2
3
4
5
6
7
-8
-7
-6
-5
-4
-3
-2
-1

Рис. 1

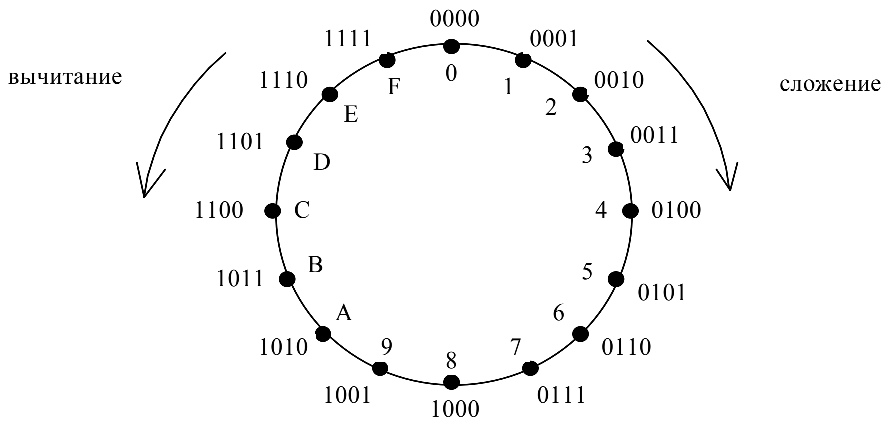
вычитание
сложение
0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111
0
1
2
3
4
5
6
7
8
9
A
B
C
D
E
F

Рис. 2

Двоично – десятичное сложение – вычитание.

При сложении двух двоично – десятичных чисел $A = a_{n-1}a_{n-2}..a_1a_0$ и $B = b_{n-1}b_{b-2}..b_1b_0$ поступают следующим образом. Если оба операнда имеют одинаковые знаки, то выполняют сложение модулей этих чисел (|A| + |B|), а знаковый разряд сумм определяют по знаку одного из слагаемых. Если операнды имеют разные знаки, то предварительно знак суммы устанавливают по знаку первого операнда A. Затем производят вычитание модулей чисел (|A| - |B|). Если полученная разность больше 0, знак суммы сохраняется без изменений. Если разность меньше 0, следует найти дополнительный код разности и изменить знак суммы на противоположный.

Операция сложения модулей (|A| + |B|) выполняется по алгоритму, схема которого приведена на рис. 3:

1) двоично – десятичный код первого операнда $a_{n-1}a_{n-2}…a_1a_0$ складывается с кодом 66..66, образуя первую промежуточную сумму $S'_{n-1}S'_{n-2}…S'_1S'_0$;
2) к полученной сумме прибавляется двоично – десятичный код 2-го операнда $b_{n-1}b_{n-2}…b_1b_0$, образуя вторую промежуточную сумму $S''_{n-1}S''_{n-2}…S''_1S''_0$;
3) выполняется потетрадно коррекция результата. Правило коррекции формулируется следующим образом: если в результате второго сложения перенос из i – ой тетрады отсутствует ($c_{i+1} = 0$), то к S''_I прибавляется код 1010_2 или A_{16}, что соответствует вычитанию 6. При возникновении переноса из i – ой тетрады коррекция не выполняется (или прибавляется код 0000), а полученный результат S''_I является истинным. При выполнении коррекции потетрадные переносы в двоичном сумматоре блокируются.

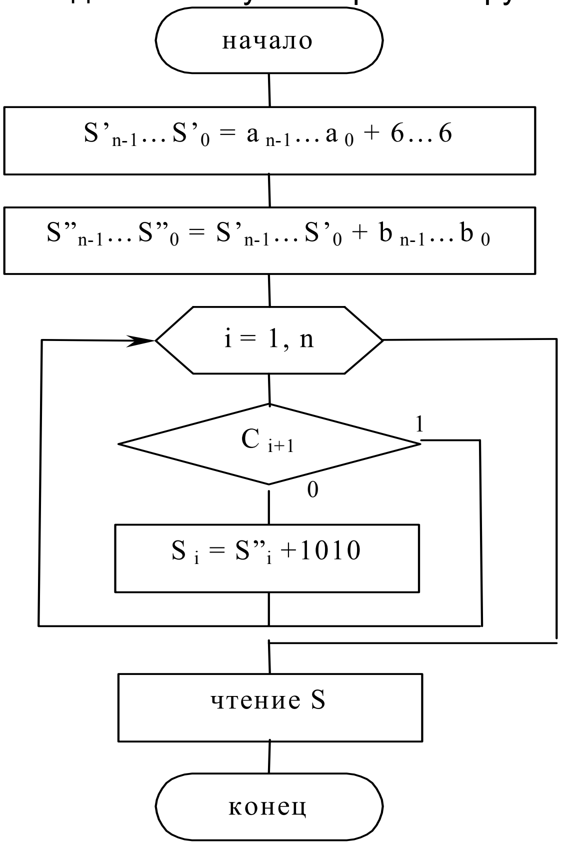

Рис. 3

Вычитание модулей (|A| - |B|) выполняют по алгоритму, схема которого представлена на рис. 4.

1) операнд B представляют двоично - десятичным дополнением до десяти;
2) двоично – десятичный код A складывают с дополнительным кодом B. Если в результате сложения образуется перенос из старшей тетрады (c_n = 1), результат является положительным. При отсутствии переноса результат является отрицательным и его следует перевести в дополнительный код.
3) Коррекция положительного результата осуществляется по правилу, сформулированному для сложения модулей чисел.

Коррекция отрицательного результата выполняется иначе:

Если имел место перенос из i – ой тетрады при сложении A + $[B]_{доп}$, то к i – ой тетраде прибавляется код 1010; если перенос отсутствует – прибавляется 0000 (см. приложение).

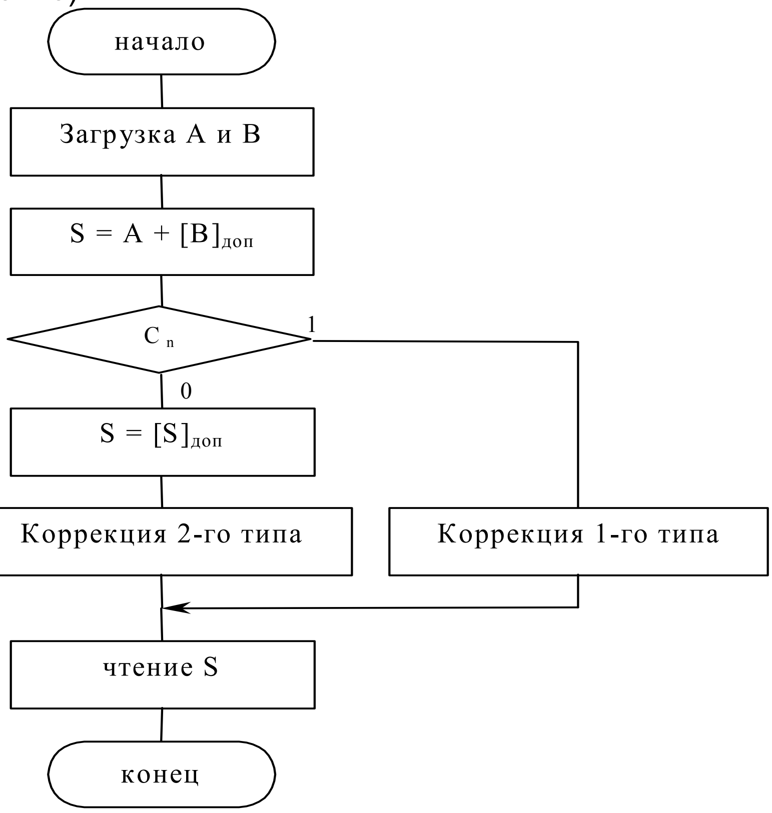

Рис. 4

Пример.

A = 237, B = 623. Найти разность A - B.

Двоично – десятичное представление чисел A и B:

A = 0010 0011 0111 B = 0110 0010 0011

Дополнение числа B до десяти: $[B]_{доп}$ = 1001 1101 1101

Складывая числа A + $[B]_{доп.}$, получаем:

$$\begin{array}{r} + \begin{array}{r} 0010\ 0011\ 0111 \\ \underline{0001\ 1101\ 1101} \end{array} \\ 1100\ 0001\ 0100 \end{array}$$

Дополнение суммы складываем с кодами коррекции:

\+ 0011 1110 1100

0000 1010 1010
0011 1000 0110 = 386

Умножение чисел без знака.

Наиболее просто умножение можно выполнить по итерационной схеме алгоритма, изображенной на рис. 5. После загрузки множимого А и множителя В в регистры общего назначения и обнуления регистра произведения П производится анализ содержимого регистра множителя. Если В $\neq$ 0, то к сумме частичных произведений П прибавляется множимое А. Затем содержимое регистра множителя уменьшается на 1 и цикл умножения повторяется до тех пор, пока содержимое регистра множителя не окажется равным 0. При умножении n – разрядных сомножителей 2n – разрядное произведение размещают в двух регистрах. Данный метод умножения находит ограниченное применение в сравнительно несложных микропроцессорных системах.

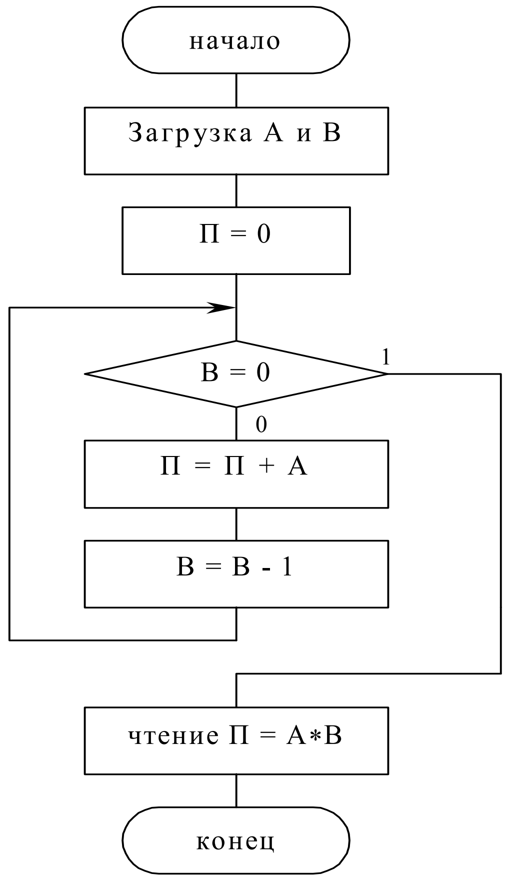

Рис. 5

На практике большое распространение имеют методы умножения путем сложения ряда сдвинутых относительно друг друга множимых, с учетом цифр множителя. Один из алгоритмов умножения, начиная с младших разрядов множителя, приведен на рис. 6. Этот алгоритм может быть использован для получения произведения двух двоичных чисел без знака. Количество итераций умножения N определяется числом разрядов множителя. Поскольку в процессе умножения на каждой итерации осуществляется сдвиг множителя В на 1 разряд вправо, на место освобождаемых разрядов можно записать выталкиваемые при сдвиге вправо разряды произведения П. При использовании ***n*** – разрядного сумматора или АЛУ исходные двоичные числа без знака не должны выходить за пределы диапазона от 1 до $2^{(n-1)}$ -1.

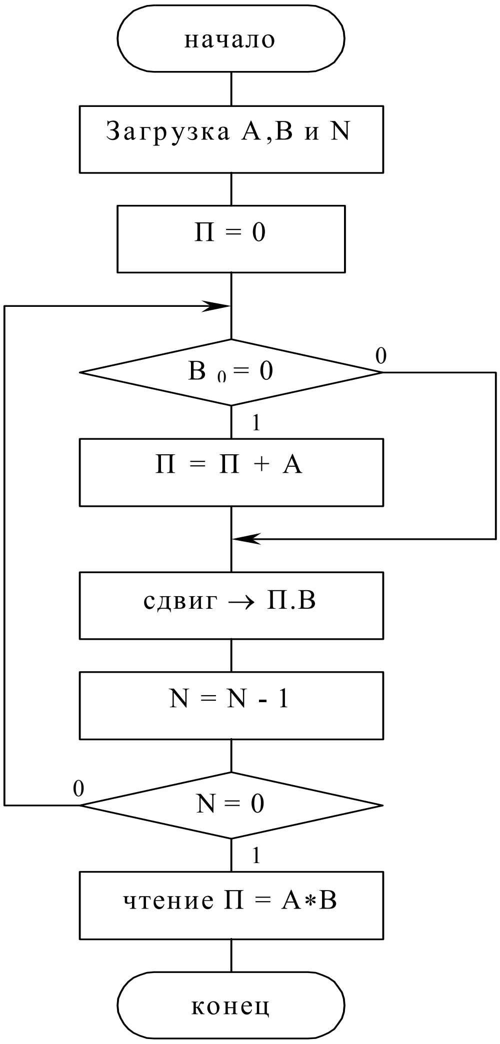

Рис. 6

Деление чисел без знака.

Для типичного алгоритма деления делимым является двойное слово, а делителем – одинарное; частное и остаток получаются в виде одинарных слов. Если при выполнении такого деления окажется, что делитель равен 0 , или для представления частного потребуется более одного слова, то происходит переполнение. Последнее имеет место в том случае, если делитель больше или равен старшего слова делимого.

В качестве примера рассмотрим метод деления А/В без восстановления остатка. В этом случае алгоритм деления представляет итерационную процедуру, на каждой итерации которой производится либо вычитание делителя В, представленного в дополнительном коде, либо прибавление В, в зависимости от знака остатка, полученного на предыдущей итерации деления. Если полученный остаток был больше 0, при очередной итерации деления производится вычитание В; если остаток был меньше 0, производится прибавление В. Перед каждым вычитанием (или сложением) производят удвоение остатка путем сдвига влево. На начальной итерации деления делимое сдвигается на 1 разряд влево.

Ниже приведен пример деления 8 – разрядного числа A на 4–разрядное число B   ($C = c_4c_3c_2c_1$ – частное, P – перенос).

$A = 19 = 0001\ 0011_2$

$B = 4 = 0100_2$ $[B]_{доп} = 1100_2$

```
 'P'   0001  0011        ; делимое A
       0010  0110        ; сдвиг A влево
     + 1100              ; вычитание B
 0     1110              ; s1 – 1-ый остаток
                  0      ; c4 =  0 Определение разряда частного
       1100  1100        ; s1*c
     + 0100              ; прибавление B
 1     0000              ; s2 – 2-ой остаток
                  1      ; c3 = 1
       0001 1001         ; s2*c
     + 1100              ; вычитание B
 0     1101              ; s3 – 3-й остаток
                  0      ; c2 = 0
       1011  0010        ; s3*c
     + 0100              ; прибавление B
 0     1111              ; s4 – 4-й остаток
                  0      ; c1 = 0
             0100        ;частное
     + 0100              ; прибавление B для получения остатка
       0011              ; остаток
```

Задание для самостоятельной подготовки

Составить схемы алгоритмов и подготовить микропрограммы по всем пунктам работы. Написать примеры для проверки работы микропрограмм.

Порядок выполнения работы

1. Выполнить операции сложения и вычитания двух 4 – разрядных чисел со знаком и без знака. Привести примеры образования признаков переноса, заема, переполнения и нуля
2. Разработать и выполнить микропрограммы сложения и вычитания 8 – разрядных чисел без знака.
3. Разработать и выполнить микропрограммы сложения и вычитания 8 – разрядных чисел со знаком. Отрицательные числа представлены в дополнительном коде.
4. Разработать и выполнить микропрограмму сложения модулей 2–разрядных двоично – десятичных чисел.
5. Разработать и выполнить в автоматическом режиме микропрограмму умножения 4-разрядных сомножителей без знака по схеме алгоритма на рис. 6.
6. Разработать и выполнить микропрограмму деления чисел без знака, полагая известным, что делитель всегда больше 0 и переполнение невозможно для заданных операндов.
7. Составить отчет. Отчет должен содержать: схемы алгоритмов, микропрограммы, примеры арифметических операций, таблицы с результатами наблюдений.