Организация ЭВМ и вычислительных систем

ЛЕКЦИЯ 9

Тема 4. Центральный процессор

4.2. Арифметико-логическое устройство (продолжение)

Обработка информации (структурная схема АЛУ), поступающая от входов ***N1***, ***N2***, ..., ***NS*** в АЛУ имеет следующую структуру: входы чисел ***A0-A3*** и ***B0-B3***, входы управления ***S0-S3*** и ***M***, вход переноса ***C0***. Работа АЛУ поясняется таблицей функционирования, изображенной ниже.

№ операции	Состояние входов S				Состояние входа M	
	S3	S2	S1	S0	*M=1*	*M=0 (C=0)*
1	0	0	0	0	***A***	***A∨1***
2	0	0	0	1	***A+B***	***(A+B)∨1***
3	0	0	1	0	***A+B***	***(A+B)∨1***
4	0	0	1	1	***0***	***0***
5	0	1	0	0	***A+B***	***(A∨A)x(B∨1)***
6	0	1	0	1	***B***	***(A+B)∨A(B∨1)***
7	0	1	1	0	***AxB∨AxB***	***A-B***
8	0	1	1	1	***AxB***	***AxB***
9	1	0	0	0	***A+B***	***A∨AxB∨1***
10	1	0	0	1	***AxB+AxB***	***A∨B∨1***
11	1	0	1	0	***B***	***(A+B)∨AxB∨1***
12	1	0	1	1	***AxB***	***AxB***
13	1	1	0	0	***1***	***A∨A∨1***
14	1	1	0	1	***A+B***	***(A+B)∨A1∨1***
15	1	1	1	0	***A+B***	***(A+B)∨A∨1***
16	1	1	1	1	***A***	***A***

Вход ***M*** определяет вид выполняемых операций (при ***M = 1*** над ***A*** и ***B*** выполняется 16 логических операций, при ***M=0*** выполняются арифметические операции). В таблице знаком \/ обозначается логическое сложение, арифметическое сложение обозначается плюсом (+), логическое умножение – знаком "х", ***A1*** – это число ***A***, сдвинутое на один разряд вправо. АЛУ может выполнять следующие операции:

- арифметическое суммирование чисел (при ***M=0*** операция №10);
- арифметическое вычитание чисел (при ***M=0*** операция №7);
- сравнение чисел – операция №7 при ***C0=1***;
- формирование модуля числа ***A***. При ***M=1*** на входы ***S3-S0*** параллельно подается знак числа ***A***, равный 1 при положительном числе ***A***. Если ***A*** отрицательно, то знак числа равен 0. Для всех ***S=1*** выполняется 16-я операция, а если все ***S=0*** – 1-я;
- мультиплексирование чисел ***A*** и ***B***. При ***M=1*** и 16-й операции на выход поступает число ***A***, а в11-я операции – число ***B***. 1-я и 6-я операции выполняют мультиплексирование с инверсией.

4.3. Компьютерная арифметика

Каждый раз, когда наибольший цифровой символ, например цифра 9 в десятичной системе счисления, возрастает на единицу (1) в любом разряде числа, тогда в этом разряде образуется нуль (0) и происходит перенос единицы (1) в следующий, более высший разряд. Например,

$09 = 0 \times 10^1 + 9 \times 10^0$;

$01 = 0 \times 10_1 + 1 \times 10^0$;

$09 + 01 = 10 =$ "1" $\times 10^1 + 0 \times 10^0$.

Здесь символом **"1"** представлен перенос единицы в старший разряд. Этот прием сложения справедлив и для других систем счисления. Этот прием сложения справедлив и для других систем счисления.

Двоичная арифметика использует правила, заданные таблицами сложения, вычитания и умножения:

Сложение	*Вычитание*	*Умножение*
0 + 0 = 0 **0 + 1 = 1** **1 + 0 = 1** **1 + 1 = 10**	**0 – 0 = 0** **1 – 0 = 1** **1 – 1 = 0** **10 – 1 = 1**	**1 = 1**

Машинные коды чисел

Различают прямой код (ПК), обратный код (ОК) и дополнительный код (ДК) двоичных чисел.

Прямой код (пк) двоичного числа образуется из абсолютного значения этого числа и кода знака (нуль или единица) перед его старшим числовым разрядом.

Пример

$(A)_{10} = +10$ $(A)_2 = +1010$ $[A_2]$пк = 0.1010;

$(B)_{10} = -13$ $(B)_2 = -1101$ $[B_2]$пк = 1.1101.

Точкой здесь отмечена условная граница, отделяющая знаковый разряд от значащих.

Обратный код (ок) двоичного числа образуется следу-ющим образом: ***обратный код положительных чисел совпадает с их прямым кодом; обратный код отрицательного числа содержит единицу в знаковом разряде числа, а значащие разряды числа заменяются на инверсные, то есть нули заменяются единицами, а единицы – нулями.***

Пример

$(A)_{10} = +10$; $(A)_2 = +1010$; $[A_2]$пк = 0.1010; $[A_2]$ок = 0.1010;

$(B)_{10} = -13$; $(B)_2 = -1101$; $[B_2]$пк = 1.1101; $[B_2]$ок = 1.0010;

Наиболее важные свойства обратного кода чисел:

- сложение положительного числа C с его отрицательным значением в обратном коде дает так называемую машинную единицу **МЕок** = 1.111... 11, состоящую из единиц в знаковом и значащих разрядах числа;
- нуль в обратном коде имеет двоякое значение. Он может быть положительным – 0.00...0 и отрицательным числом – 1.11... 11. Значение отрицательного нуля совпадает с **МЕок**. Двойственное представление нуля явилось причиной того, что в современных ЭВМ все числа пред ставляются не обратным, а дополнительным кодом.

Дополнительный код (дк) ***положительных чисел совпадает с их прямым кодом. Дополнительный код отрицательного числа представляет собой результат суммирования обратного кода числа с единицей***.

Пример

$(A)_{10}$ = +10; $(A)_2$ = +1010; $[A_2]$пк = 0.1010; $[A_2]$ок = +0.1010; $[A_2]$дк = 0.1010;

$(B)_{10}$ = -13; $(B)_2$ = - 1101; $[B_2]$пк = 1.1101; $[B_2]$ок = =1.0010; $[B_2]$дк = 1.0011.

Основные свойства дополнительного кода:

- сложение дополнительных кодов положительного числа ***A*** с его отрицательным значением дает так называемую машинную единицу дополнительного кода:
- **МЕдк = МЕок**+20 =10.00...00,то есть число 10 (два) в знаковых разрядах числа;
- дополнительный код получил такое свое название потому, что представление отрицательных чисел является дополнением прямого кода чисел до машинной единицы **МЕдк**.

Модифицированные обратные* и *дополнительные коды двоичных чисел отличаются от обратных и дополнительных кодов удвоением количества знаковых разрядов. Знак «+» в этих кодах кодируется двумя нулевыми знаковыми разряда-ми, а «-» – двумя единичными разрядами.

Пример

$(A)_{10} = 9$; $(A)_2 = +1001$; $[A_2]$пк = $[A_2]$ок = $[A_2]$дк = 0.1001;
$[A_2]$мок = $[A_2]$мдк = 00.1001;
$(B)_{10} = -9$; $(B)_2 = -1001$; $[B_2]$ок = 1.0110; $[B_2]$дк = 1.0111;
$[B_2]$мок = 11.0110 $[B_2]$мдк = 11.0111.

Модифицированные коды нужны для фиксации и об-наружение случаев переполнения разрядной сетки, то есть, когда значение кода превышает максимально возможную длину в отведенной разрядной сетке ЭВМ. В этом случае перенос из значащего разряда может исказить значение младшего знакового разряда. ***Значение знаковых разрядов, равное «01» свидетельствует о положительном перепо-лнении разрядной сетки, а равное – «10» – об отрица-тельном переполнении.***

Операции над машинными кодами чисел

Все современные ЭВМ имеют достаточно развитую систему команд, включающую десятки и сотни машинных операций. Однако выполнение любой операции основано на использовании простейших ***микроопераций*** типа ***сложения*** и ***сдвига***. Это позволяет иметь единое арифметико-логическое устройство для выполнения любых операций, связанных с обработкой информации. Ниже, в таблице, показаны правила сложения двоичных цифр a_i, b_i одноименных разрядов с учетом возможных переносов из предыдущего разряда p_i-1.

Подобные таблицы можно было бы построить для любой другой арифметической и логической операции (вычитание, умножение и так далее), но именно данные этой таблицы положены в основу выполнения любой операции ЭВМ. Под знак чисел отводится специальный знаковый разряд. Знак «+» кодируется двоичным нулем, а знак «-» – единицей.

Значения разрядов двоичных чисел и переноса			Разряд суммы *Si*	Перенос в следующий разряд *Pi*
a_i	b_i	p_{i-1}		
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Действия над прямыми кодами двоичных чисел при выполнении операций создают большие трудности, связанные с необходимостью учета значений знаковых разрядов:

- во-первых, приходится отдельно обрабатывать значащие разряды чисел и разряды знака;
- во-вторых, значение разряда знака влияет на алгоритм выполнения операции (сложение может заменяться вычита-нием и наоборот).

Во всех ЭВМ без исключения все операции выполняются над числами, представленными специальными машинными кодами. Их использование позволяет обрабатывать знаковые разряды чисел так же, как и значащие разряды, а также заменять операцию вычитания операцией сложения.

Сложение (вычитание). Операция вычитания приводится к операции сложения путем преобразования чисел в обратный или дополнительный код. Пусть имеются числа $A \geq 0$ и $B \geq 0$, тогда операция алгебраического сложения выполняется в соответствии с таблицей.

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

Скобки в представленных выражениях указывают на замену операции вычитания операцией сложения с обратным или дополнительным кодом соответствующего числа. Сложение двоичных чисел осуществляется последовательно, поразрядно в соответствии с сложения, вычитания и умножения двоичных чисел

При выполнении сложения с использованием машинных кодов необходимо соблюдать следующие правила.

1. Слагаемые должны иметь одинаковое число разрядов. Для выравнивания разрядной сетки слагаемых можно дописывать нули слева к целой части числа и нули справа к дробной части числа.
2. Знаковые разряды чисел участвуют в сложении так же, как и значащие.
3. Необходимые преобразования кодов производятся с изменением знаков чисел. Приписанные незначащие нули изменяют свое значение при преобразованиях по общему правилу.
4. При образовании единицы переноса из старшего знакового разряда, в случае использования ОК, эта единица складывается с младшим числовым разрядом. При использовании ДК единица переноса отбрасывается. Знак результата формируется автоматически, результат представляется в том коде, в котором представлены исходные слагаемые.

Пример

Сложить два числа $(A)_{10}=7$ $(B)_{10}=16$

$$(A)_2 = +111;$$

$$(B)_2 = +10000.$$

Исходные числа имеют различную разрядность, необходимо провести выравнивание разрядной сетки:

$$[A_2]\text{пк} = [A_2]\text{ок} = [A_2]\text{дк} = 0.00111;$$

$$[B_2]\text{пк} = [B_2]\text{ок} = [B_2]\text{дк} = 0.10000.$$

Сложение в обратном или дополнительном коде дает один и тот же результат

$$\begin{array}{r} 0.00111 \\ + \\ 0.10000 \\ \hline (C)_2 = 0.10111 \end{array}$$

$$(C)_{10} = +23.$$

Следует обратить внимание на то, что при сложении цифр отсутствуют переносы в знаковый разряд и из знакового разряда, что свидетельствует о получении правильного результата.

Пример

Сложить два числа *(A)*10 =+ 16 *(B)*10= -7 в ОК и ДК.

В с правилом ***A+(-B)***, второе слагаемое преобразуется с учетом знака

$[A_2]$пк = 0.10000; $[A_2]$ок = 0.10000; $[A_2]$дк = 0.10000;

$[B_2]$пк = 1.00111; $[B_2]$ок = 1.11000; $[B_2]$дк = 1.11001.

Сложение в ОК

$[A_2]$ок = 0.10000

+$[B_2]$ок = 1.11000

10.01000

\+ 1

0.01001

C_2 = 0.01001

C_{10} = +9

Сложение в ДК

$[A_2]$ = 0.10000

+$[B_2]$ = 1.11001

10.01001

C_2 = 0.01001

C_{10} = +9

При сложении чисел в ОК и ДК получены переносы в знаковый разряд и из знакового разряда. В случае ОК перенос требует дополнительного прибавления едини-цы младшего. В случае ДК этот перенос игнорируется.

Пример

Сложить два числа $(A)_{10}$ = - 16 $(B)_{10}$= +7 в ОК и ДК.

В соответствии с правилами искомая сумма должна быть реализована как зависимость ***(-A)+B***, в которой первое слагаемое преобразуется с учетом знака

$[A_2]$пк = -10000 = 1.10000; $[A_2]$ок = 1.01111; $[A_2]$дк = 1.10000;

$[B_2]$пк = + 00111= 0.00111; [B2]ок = 0.00111; [B2]дк = 0.00111.

Сложение в ОК	Сложение в ДК
[A2]ок = 1.01111	[A2] = 1.10000
+[B2]ок = 0.00111	+[B2] = 0.00111
1.10110	1.10111

При сложении чисел в ОК и ДК были получены отрицательные результаты («1» в знаковом разряде). Для перевода обратного кода отрицательного числа в прямой необходимо инвертировать значащие разряды, а знаковый разряд оставить без изменения. А для перевода дополнительного кода отрицательного числа в прямой код необходимо инвертировать значащие разряды и прибавить единицу к младшему разряда.

Таким образом, в ПК из ОК получено число 1.01001, а в ПК из ДК получено число 1.01000, то есть $[C_2]$пк = 1.01001 (с учетом прибавления 1 в младший разряд):

$(C)_{10}$ = -9 $(C)_{10}$ = -9

Пример

Сложить 2 числа: ***A***=+5 и ***B***=+6 в четырехразрядной сетке (с учетом знакового разряда).

Сложение в ОК

$[A_2]$ок = 0.101

$+[B_2]$ок = 0.110

1.011

$(C)_2$ = 1.100

$(C)_{10}$ = -4 (вместо +11)!

Сложение в ДК

$[A_2]$ = 0.101

$+[B_2]$ = 0.110

1.011

$(C)_2$ = 0.101

$(C)_{10}$ = -5 (вместо +11)!

Пример

Сложить 2 числа: A=-5 и ***B***=-6 в четырехразрядной сетке (с учетом знакового разряда).

Сложение в ОК

$[A_2]$ок = 1.010

$+[B_2]$ок = 1.011

10.101

1

0.110

$(C)_2$ = 0.110

$(C)_{10}$ = +6 (вместо -11)!

Сложение в ДК

$[A_2]$ = 1.011

$+[B_2]$ = 1.100

10.111

$(C)_2$ = 0.111

$(C)_{10}$ = +7 (вместо -11)!

Вывод. Из примеров видно, что при сложении положительных чисел получается отрицательный результат и наоборот. Это объясняется тем, что в трех значащих разрядах максимальное двоичное число быть равно семи, а результат в примерах равен, соответственно, +11 и -11.

Умножение. Умножение выполняется суммированием сдвинутых на один или несколько разрядов частичных произведений, каждое из которых является результатом умножения множимого на соответствующий разряд множителя.

При точном умножении двух чисел количество значащих цифр произве-дения может достичь двойного количества значащих цифр сомножителей. Еще сложнее возникает ситуация при умножении нескольких чисел.

Наиболее просто операция умножения выполняется в прямом коде. При этом на первом этапе определяется знак произведения путем сложения знаковых разрядов по модулю 2, затем производится перемножение модулей сомножителей. Результату присваивается полученный знак.

Произведение можно получить двумя путями:

1***) сдвигом множимого на требуемое количество разрядов и прибавлением полученного очередного частичного произведения к ранее накопленной сумме частичных произведений***;

2) ***сдвигом суммы ранее полученных частичных произведений на каждом шаге на 1 разряд и последующим прибавлением к сдвинутой сумме неподвижного множимого либо 0***.

Каждый из этих методов может различаться еще и тем, что умножение может начинаться с младших разрядов, а может и со старших.

Пример

Умножить два числа $[A_2]_{ПК} = 0,1101$ и $[B_2]_{ПК} = 0,1011$.

Первый способ

$[A_2]_{ПК} = 0,1101$

$[B_2]^{ПК} = 0,1011$

1101

1101

0000

1101

0,10001111

Второй способ

$[A_2]_{ПК} = 0,1101$

$[B_2]_{ПК} = 0,1011$

1011

0000

1011

1011

0,10001111

Проверка

$[A_2]_{ПК} = 0,1101$; $(A)_{10} = 1x2^{-1}+1x2^{-2}+0x2^{-3}+1x2^{-4} = 0+1/2+1/4+1/16 =13/16$;

$[B_2]_{ПК} = 0,1011$; $(B)_{10} = 11/16$;

$(A)_{10} \xrightarrow{x [B_2]_{ОК}} = (13/16) x (11/16) = 143/256.$

$[A_2]_{ПК} x [B_2]_{ПК} = 0,1000111 = 1x2^{-1}+0x2^{-2}+0x2^{-3}+0x2^{-4}+1x2^{-5}+1x2^{-6}+$

$+1x2^{-7}+1x2^{-8} = 1/2+1/32+1/64+1/128+1/256 = 143/256.$

Результаты умножения десятичных чисел и их двоичного выражения в прямом коде совпали. Значит умножение проведено правильно.

Исходя из примеров, можно создать 4 основных метода машинного умножения в прямом коде:

1) ***умножение младшими разрядами множителя со сдвигом накапливаемой суммы частных произведений вправо;***
2) ***умножение младшими разрядами множителя со сдвигом множимого влево;***
3) ***умножение старшими разрядами множителя со сдвигом накапливаемой суммы частных произведений влево;***
4) ***умножение старшими разрядами множителя со сдвигом множимого вправо.***

Алгоритм схемы умножения в первом случае будет следующим:

- ***содержимое сумматора обнуляется;***
- ***множимое умножается на очередной разряд множителя;***
- ***результат суммируется с содержимым сумматора;***
- ***содержимое сумматора сдвигается на 1 разряд вправо;***
- ***пункты 2, 3, 4 повторяются n-1 раз.***

<u>Пример</u>

Умножить два числа $(A)_2 = 0{,}0101$ и $(B)_2 = 0{,}1011$.

№ п/п	Разрядность множителя	Наименование операции	Сумматор S 1 2 3 4 5 6 7 8
1		Обнуление	0 0 0 0 0 0 0 0
	$B_1 = 1$	$Ax B_1$	0 1 0 1 0 0 0 0
		Сложение с содержимым S	0 1 0 1 0 0 0 0
		Сдвиг на 1 разряд вправо *R*	0 0 1 0 1 0 0 0
2	$B_2 = 1$	$Ax\ B_2$	0 1 0 1 0 0 0 0
		Сложение с содержимым S	0 1 1 1 1 0 0 0
		Сдвиг на 1 разряд вправо *R*	0 0 1 1 1 1 0 0
3	$B_3 = 0$	$Ax\ B_3$	0 0 0 0 0 0 0 0
		Сложение с содержимым S	0 0 1 1 1 1 0 0
		Сдвиг на 1 разряд вправо *R*	0 0 0 1 1 1 1 0
4	$B_4 = 1$	$Ax\ B_4$	0 1 0 1 0 0 0 0
		Сложение с содержимым S	0 1 1 0 1 1 1 0
		Сдвиг на 1 разряд вправо *R*	0 0 1 1 0 1 1 1

Алгоритм схемы умножения для второго случая аналогичен за исключением направления сдвига:

- содержимое сумматора обнуляется;

- множимое умножается на очередной разряд множи-теля;

- результат суммируется с содержимым сумматора;

- содержимое сумматора сдвигается на 1 разряд влево;

- пункты 2, 3, 4 повторяются n-1 раз.

Выполнение умножения по 3-му и 4-му способам умно-жения можно рассмотреть по аналогии к выше рассмот-ренным способам. Наиболее удобными для применения в ЭВМ являются 1 и 4 схемы умножения.

Умножение чисел, представленных в форме с плавающей запятой. Если операнды заданы в форме с плавающей запятой: $A = ax2^{ma}$ **и** $B = =bx2^{mb}$, то их произведение $C = AxB$ или $C = axbx2^{m(a+b)}$.

Алгоритм умножения нормализованных чисел состоит из следующих этапов:

1) определение знака произведения путем сложения знаковых разрядов мантисс операндов по модулю 2;

2) алгебраическое сложение порядков сомножителей с целью определения порядка произведения;

3) умножение модулей мантисс сомножителей по правилам умножения чисел с фиксированной запятой;

4) нормализация и округление мантиссы результата. При этом следует учитывать, что мантиссы сомножителей являются нор-мализованными числами. Поэтому денормализация мантиссы произ-ведения возможна только на один разряд вправо. Она устраняется путем сдвига мантиссы на один разряд влево и вычитания 1 из поряд-ка результата;

5) присвоение знака результату.

Первые три операции могут выполняться одновременно, так как они независимы. Умножение мантисс выполняется в прямом коде.

При выполнении операции умножения с плавающей запятой может получиться переполнение отрицательного порядка, кото-рое будет интерпретировано как машинный нуль, ес-ли прог-раммой игнорируется признак исчезновения порядка. Может также возникнуть положительное переполнение порядка. В этом случае в первую очередь необходимо нормализовать мантиссу результата. Если и после этого переполнение порядка не устра-няется, то формируется признак переполнения порядка.

Кроме приведенных алгоритмов имеют место еще ***ускоренные методики умножения*** и ***матричный метод умножения***.

Суть первых методов заключается в совмещение во времени отдельных составных частей процесса умножения, а также в пропуске тактов суммирования в тех случаях, когда очередная цифра множителя равна 0.

Во втором случае одновременно вычисляются частные произведения составляющих матрицы, а затем производится суммирование группы. А в предельном случае – всех, частных произведений.

Деление. Операция деления встречается сравнительно редко (вероятность деления среди других арифметических операций равна 0,02), однако, реализация ее в подпрограмме занимает достаточно большое время. Поэтому в большинстве современных ЭВМ деление реализуется специальными операционными блоками.

Деление, также как и умножение, проще всего выполняется в прямом коде. Но в отличие от умножения дробных чисел, где не может возникнуть переполнение разрядной сетки, при делении правильных дробей такое переполнение возможно в случае, когда делимое больше делителя. Признаком переполнения является появление целой части в частном, что грубо искажает результат.

Частное определяется путем деления модулей исходных чисел. При этом во избежание переполнения разрядной сетки должно соблюдаться условие: $|A|<|B|$, где A – делимое, B – делитель.

Известны два основных алгоритма выполнения операции деления:

- деление с восстановлением остатков;

- деление без восстановления остатков.