Задача 5

1	знак	порядок	знак	мантисса	Умножить (со сдвигом промеж. рез-та) два наибольших числа и поделить большее на меньшее (с восстановлением остатка) . Представить конечный результат в стандарте IEEE 754.
A	1	1001	1	110111	
B	1	1011	0	111001	

или

1	знак	порядок	знак	мантисса
A	1	1001	1	110111
	Ап		A_M	
B	1	1011	0	111001
	Вп		B_M	

Теория

Умножение чисел в формате с плавающей точкой происходит по следующему алгоритму: сложение порядков, умножение мантисс, нормализация результата.

Если есть два числа $A = A_M 2^{Ап}$; $B = B_M 2^{Вп}$, то результат их умножения можно получить как $A \times B = A_M \ B_M \ 2^{Ап+Вп}$. Формат с плавающей точкой накладывает ограничения, что количество бит мантиссы исходных чисел и результат должен сохраниться, и для хранения всех результатов перемножения отдельных весовых коэффициентов на другое число нет возможности. Поэтому при умножении, после получения двух результатом умножения весовых коэффициентов, происходит сложение и получение промежуточного результата, а дальше к полученному промежуточному результату прибавляются составляющие, до тех пор, пока не закончатся все биты. Для получения регулярности операции первым результатом является 0 в регистре промежуточного результата.

Для соблюдения длины хранения результата после каждого суммирования происходит сдвиг. Сдвигают или промежуточный результат или исходное число. Начинать можно как со старшего бита, так и с младшего. Эта вариация и приводит к четырем басовым методикам перемножения

Умножение будем производить без учета знака. Знак можно сразу получить путем применения операции XOR к битам знака. Но есть варианты умножения и отрицательных чисел.

В нашем примере мантиссы имеют разные знаки, поэтому результат перемножения отрицательный.

Будем сдвигать промежуточный результат и начинать с последнего бита.

В некоторых системах при сдвиге могут хранить бит, который отбрасывается при сдвиге чтобы по окончанию процесса перемножения произвести правильно округление.

	00 . 110111		½A_M½
	00 . 111001		½B_M½
	00 . 000000		Первичное заполнение регистра хранения промежуточного результата нулями
+	00 . 110111		Результат умножения последнего бита B_M на мантиссу A_M
	00 . 110111		Результат сложения
	00 . 011011	1	Результат сдвига
+	00 . 000000		Результат умножения 2 с конца бита B_M на мантиссу A_M
	00 . 011011		Результат сложения
	00 . 001101	1	Результат сдвига
+	00 . 000000		Результат умножения 3 с конца бита B_M на мантиссу A_M
	00 . 001101		Результат сложения
	00 . 000110	1	Результат сдвига
+	00 . 110111		Результат умножения 4 с конца бита B_M на мантиссу A_M
	00 . 111101		Результат сложения
	00 . 011110	1	Результат сдвига
+	00 . 110111		Результат умножения 5 с конца бита B_M на мантиссу A_M
	01 . 010101		Результат сложения
	00 . 101010	1	Результат сдвига
+	00 . 110111		Результат умножения 6 с конца бита B_M на мантиссу A_M
	01 . 100001		Результат сложения
	00 . 110000	1	Результат нормализации путем сдвига
	00 . 110001		Результат округления

Получен результат с переполнение. Его следует нормализовать путем сдвига. Это повлечет увеличение порядка на 1. В буфере на последнем этапе имеется единица, поэтому округляем путем прибавления к младшему биту.

Сложим порядки. Порядки отрицательные, поэтому переведем в дополнительный код. Поскольку дальше, для перевода в IEEE 754 потребуется прибавить 127. для весовых коэффициентов использует 9 бит, и еще два бита для знака.

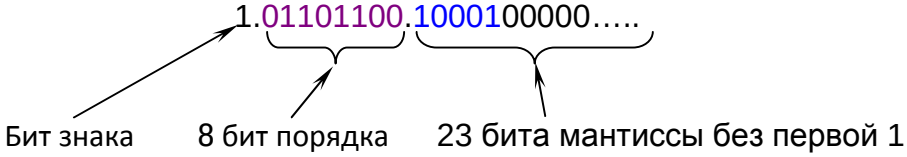
Ап = 11. 00001001пк = 11.11110110ок = 11.11110111дк.

Вп = 11. 00001011пк = 11.11110100ок = 11.11110101дк.

+	11 . 11110111		Ап
	11 . 11110101		Вп
+	~~1~~11 . 11101100		Результат сложения
	00 . 01111111		127
+	~~1~~00 . 01101011		Результат добавления 127
	00 . 00000001		Прибавления 1 вследствие нормализации мантиссы
	00 . 01101100		Окончательный результат

Тогда в стандарте окончательно результат получим

1.01101100.10001000000…..



Базовых алгоритмов деления два. Один с восстановлением остатка другой без. При делении чисел с плавающей точкой, алгоритм в целом похож на процесс умножения: Вычитание порядков, деление мантисс, нормализация результата.

Если есть два числа $A = A_M 2^{Aп}$; $B = B_M 2^{Bп}$, то результат их деления можно получить как $A/B = A_M / B_M \, 2^{Aп–Bп}$.

Для процесса необходимо получить отрицательное значение модуля делителя. Затем провести сложение (фактически вычитание). Если в результате получается отрицательное число, то в данном разряде результата деления пишем ноль, если положительное, то 1. Различие алгоритмов, которые указаны выше только в том, что в первом случае мы прибавляем модуль делителя (восстанавливаем остаток), а потом делаем сдвиг, то во втором случае, сразу делаем сдвиг. При получении положительного результат, операции одинаковые

Используем те же два числа, что и для умножения. И опять будем делить модули мантисс. Знак результата получим также, как и для умножения. И данном примере он также будет отрицательным

1	знак	порядок	знак	мантисса	Умножить (со сдвигом промеж. рез-та) два наибольших числа и поделить большее на меньшее (с восстановлением остатка) . Представить конечный результат в стандарте IEEE 754.
A	1	1001	1	110111	
B	1	1011	0	111001	

$½B_M½$= 00.111001; $–½B_M½$= 11.111001пк = 11.000110ок = 11.000111дк

Деление с восстановлением остатка

			Результат деления в ПК и комментарий
	00 . 110111	$½A_M½$	
+	11 . 000111	$–½B_M½$	
	11 . 111110		11.0 Результат отрицательный, поэтому записали 0
+	00 . 111001	$½B_M½$	
	100 . 110111		Восстановленный остаток
	01 . 101110		Результат сдвига
+	11 . 000111	$–½B_M½$	
	100 . 110101		11.01 Результат положительный, поэтому записали 1
	01 . 101010		Результат сдвига
+	11 . 000111	$–½B_M½$	
	100 . 110001		11.011 Результат положительный, поэтому записали 1
	01 . 100010		Результат сдвига
+	11 . 000111	$–½B_M½$	
	100 . 101001		11.0111 Результат положительный, поэтому записали 1
	01 010010		Результат сдвига
+	11 . 000111	$–½B_M½$	
	100 . 011001		11.01111 Результат положительный, записали 1
	00 . 110010		Результат сдвига
+	11 . 000111	$–½B_M½$	
	11 . 111001		11.011110 Результат отрицательный, записали 0
+	00 . 111001	$½B_M½$	
	100 . 110010		Восстановленный остаток
	01 . 100100		Результат сдвига
+	11 . 000111	$–½B_M½$	
	100 . 101101		11.0111101 Результат положительный, записали 1
	01 011010		Результат сдвига
+	11 . 000111	$–½B_M½$	
	100 . 100001		11.01111011 Результат положительный, записали 1

Полученных бит достаточно для формирования результата с учетом разрядной сетки и округления.

Итоговая мантисса после сдвига 11.1111011пк, после округления 11.111110пк. Вследствие нормализации необходимо будет понизить порядок на единицу.

Вычитание порядков

Ап = 11. 00001001пк = 11.11110110ок = 11.11110111дк.

–Вп = 00. 00001011пк.

	11 . 11110111	Ап
+	00 . 00001011	Вп
	~~1~~00 . 00000010	Результат сложения
+	00 . 01111111	127
	00 . 10000001	Результат добавления 127
+	11 . 11111111	Прибавления (–1) вследствие нормализации мантиссы
	~~1~~00 . 10000000	Окончательный результат

Тогда в стандарте окончательно результат получим

1.10000000.11110000000…..

Бит знака — 8 бит порядка — 23 бита мантиссы без первой 1

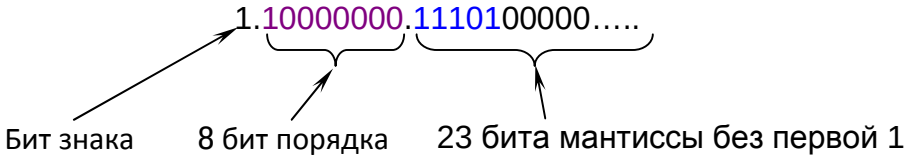
Деление без восстановления остатка.

Отличие от предыдущего алгоритма, что при получение отрицательного результата осуществляется сразу сдвиг, но к отрицательному остатку уже прибавляется ½B_M½ Формирование результата идентично.

½B_M½= 00.111001; –½B_M½= 11.111001пк = 11.000110ок = 11.000111дк

			Результат деления в ПК и комментарий
	00 . 110111	½A_M½	
+	11 . 000111	–½B_M½	
	11 . 111110		11.0 Результат отрицательный, поэтому записали 0
	11 . 111100		Результат сдвига
+	00 . 111001	½B_M½	
	~~1~~00 . 110101		
+	11 . 000111	–½B_M½	
	~~1~~00 . 110100		11.01 Результат положительный, поэтому записали 1
	01 . 101000		Результат сдвига
+	11 . 000111	–½B_M½	
	~~1~~00 . 101111		11.011 Результат положительный, поэтому записали 1
	01 . 011110		Результат сдвига
+	11 . 000111	–½B_M½	
	~~1~~00 . 100101		11.0111 Результат положительный, поэтому записали 1
	01 001010		Результат сдвига
+	11 . 000111	–½B_M½	
	~~1~~00 . 010001		11.01111 Результат положительный, записали 1
	00 . 100010		
+	11 . 000111	–½B_M½	
	11 . 101001		11.011110 Результат отрицательный, записали 0
	11 010010		Результат сдвига
+	00 . 111001	½B_M½	
	~~1~~00 . 001011		11.0111101 Результат положительный, записали 1
	00 . 010110		Результат сдвига
+	11 . 000111	–½B_M½	
	11 . 010101		11.01111010 Результат отрицательный, записали 0

Результат отличается в последних битах, что является особенностями и неэквивалентности операций сдвига отрицательных чисел в дополнительном коде и положительных чисел.



A	1	1001	1	110111	
B	1	1011	0	111001	

Число A= –1 (1 + 0,5 + 0,125 + 0,0625 + 0,03125) 2^{-9}

Число B = +1 (1 + 0,5 + 0,25 + 0,03125) 2^{-11}

Деление чисел в калькуляторе –3,859649122807018

Преобразование результата деления методом с восстановлением остатка – 3,8125

Преобразование результата деления методом без восстановления остатка – 3,875.

Первый метод результат деления смещает в сторону 0, второй в сторону плюс бесконечности. Если производить вычисления для последнего метода в обратном коде, то будет результат аналогичный методу с восстановлением остатка.