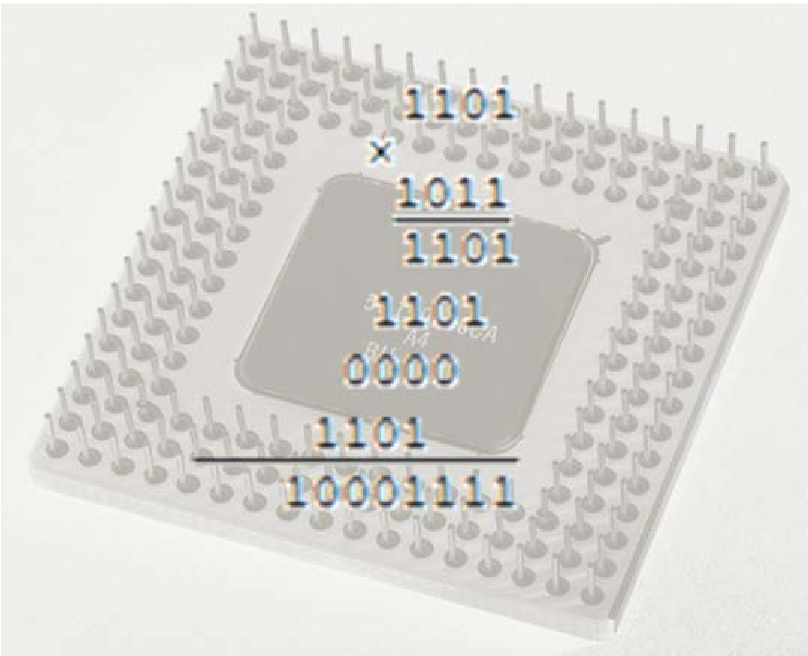
М.Б. Будько, В.А. Грозов, Д.И. Милосердов

РЕАЛИЗАЦИЯ ПРОЦЕССОРОМ АРИФМЕТИЧЕСКИХ ОПЕРАЦИЙ

Учебно-методическое пособие по выполнению домашних заданий по дисциплине "Дискретная математика"



Санкт-Петербург

2014

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

М.Б. Будько, В.А. Грозов, Д.И. Милосердов

РЕАЛИЗАЦИЯ ПРОЦЕССОРОМ АРИФМЕТИЧЕСКИХ ОПЕРАЦИЙ

Учебно-методическое пособие
по выполнению домашних заданий
по дисциплине
"Дискретная математика"

Санкт-Петербург

2014

Будько М.Б., Грозов В.А., Милосердов Д.И. «Реализация процессором арифметических операций» – СПб: НИУ ИТМО, 2014. – 68 с.

В учебном пособии рассмотрено представление чисел в ЭВМ и выполнение арифметических операций над ними. Для каждой арифметической операции предложен единый подход, позволяющий разработать универсальные алгоритмы для написания программ, имитирующих машинную арифметику.

Рекомендовано бакалаврам и магистрантам по направлению «Информационная безопасность».

Рекомендовано к печати Ученым советом Института комплексного военного образования протокол №4 от 28 апреля 2014 г. в качестве учебного пособия для обучающихся на кафедре мониторинга и прогнозирования информационных угроз.

В 2009 году Университет стал победителем многоэтапного конкурса, в результате которого определены 12 ведущих университетов России, которым присвоена категория «Национальный исследовательский университет». Министерством образования и науки Российской Федерации была утверждена программа его развития на 2009–2018 годы. В 2011 году Университет получил наименование «Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики»

Введение

В основе многих информационных дисциплин лежит глубокое знание принципов организации и функционирования устройств ЭВМ. Удобным объектом для их изучения является, например, реализация машинной арифметики.

В пособии рассмотрено выполнение арифметических операций над целыми числами. Для каждой арифметической операции предложен единый подход, позволяющий разработать универсальные алгоритмы для написания программ, имитирующих машинную арифметику.

Пособие состоит из четырех разделов, посвященных каждой из арифметических операций: сложению, вычитанию, умножению, делению. Для всех операций приводятся необходимые теоретические сведения, подробные примеры с пошаговым описанием и задания для самостоятельной работы. Предполагается, что задания состоят из двух частей: 1) выполнение каждой операции вручную и представление результатов в виде отчета; 2) составление программы, реализующей все арифметические операции.

Настоящее пособие можно рассматривать как продолжение и развитие работы [1] (Довгий П.С., Поляков В.И. Арифметические основы ЭВМ. Учебно-методическое пособие по выполнению домашних заданий по дисциплине «Дискретная математика». – СПб: СПбГУ ИТМО, 2010. – 56 с.), сделанное в поисках более удобного для алгоритмизации варианта выполнения на ЭВМ арифметических операций.

1 СЛОЖЕНИЕ ЦЕЛЫХ ЧИСЕЛ

1.1 Основные положения

Сложение целых чисел выполняется в байтном формате (размер разрядной сетки – 8 бит).

При сложении целых чисел с фиксированной запятой используются их дополнительные коды[1], при этом знаковый разряд участвует в операции точно так же, как и цифровые. В результате этого автоматически формируется знак результата, причем результат получается в дополнительном коде.

При сложении целых чисел с фиксированной запятой используются следующие арифметические флаги:

CF – Carry Flag (флаг переноса). В нем фиксируется перенос из старшего разряда при сложении.

PF – Parity Flag (флаг четности). Он устанавливается при наличии четного числа единиц в младшем байте результата, в противном случае – сбрасывается.

SF – Sign Flag (флаг знака). В него копируется старший разряд результата, интерпретируемый как знак[2].

Диапазон представления чисел в байтном формате выглядит следующим образом:

– целых знаковых: $-128 = -2^7 \leq A_{Ц}^{ЗН} \leq 2^7 - 1 = 127$

$\uparrow$ 1.0000000 (для -128) $\quad$ $\uparrow$ 0.11111111 (для 127)

– целых беззнаковых: $0 \leq \boldsymbol{A}_{Ц}^{БЗН} \leq 2^8 - 1 = 255$

AF – Auxiliary Flag (флаг вспомогательного переноса). В нем фиксируется межтетрадный (из старшей тетрады в младшую) перенос при сложении.

ZF – Zero Flag (флаг нуля). Он устанавливается при нулевом результате операции, в противном случае – сбрасывается.

OF – Overflow Flag (флаг переполнения). Он устанавливается в командах сложения в случае, если результат операции не помещается в формате операндов.

Корректность при беззнаковой интерпретации (БзИ) проверятся по флагу ***CF***:

1) ***CF*** = 0. Перенос из старшего разряда при сложении отсутствует, следовательно, результат БзИ корректен;

2) ***CF*** = 1. Перенос из старшего разряда при сложении имеет место, следовательно, результат БзИ некорректен.

---

[1] Напомним, что перевод числа в дополнительный код осуществляется следующим образом: если число положительно, т.е. старший разряд равен 0, цифровые разряды остаются неизменными. Если старший разряд равен 1, то есть число отрицательно, то инвертируются все разряды, кроме старшего, и к результату прибавляется единица. Обратное преобразование осуществляется аналогично.

[2] При знаковой интерпретации (ЗИ) старший разряд числа интерпретируется как знак (его значение равно нулю, если число положительно, единице – если отрицательно), при беззнаковой интерпретации (БзИ) – как обычный цифровой разряд. Например, 1.0101101: в ЗИ это число –45, в БзИ – число 173.

Корректность при ЗИ проверятся по флагу $\boldsymbol{OF}$:

1) $\boldsymbol{OF} = 0$. Переполнения формата разрядной сетки при сложении не произошло, результат ЗИ корректен;

2) $\boldsymbol{OF} = 1$. Результат ЗИ некорректен, т.к. при сложении произошло переполнение формата разрядной сетки, о котором можно судить по двум признакам:

а) знаки операндов одинаковы, а знак суммы отличается от них;

б) сравниваются переносы из старшего цифрового разряда в знаковый и из знакового за пределы формата, при этом один из переносов имеет место, а другой отсутствует.

Примечание 1. При $\boldsymbol{OF} = 1$ имеет место так называемый особый случай переполнения формата. Такое переполнение происходит при условии $\boldsymbol{A} + \boldsymbol{B} > 128$, откуда получаем $128 - \boldsymbol{A} < \boldsymbol{B} \leq 127$. В случае сложения переполнение имеет место при одинаковых знаках операндов.

Примечание 2. В пункте 3 задания (см. ниже) $\boldsymbol{A}$ подбирается из условия $\boldsymbol{A} + \boldsymbol{B} = 128$, при этом при сложении положительных чисел будет фиксироваться переполнение, а при сложении отрицательных – не будет.

1.2 Задание

1) Для заданных чисел $\boldsymbol{A}$ и $\boldsymbol{B}$ выполнить операцию знакового сложения со всеми комбинациями знаков операндов (4 случая).

Для каждого примера выполнить следующее:

а) Проставить межразрядные переносы, возникающие при сложении.

б) Дать ЗИ операндов и результата. При получении отрицательного результата предварительно преобразовать его из дополнительного кода в прямой.

в) Дать БзИ операндов и результата. При получении неверного результата пояснить причину его возникновения.

г) Показать значения арифметических флагов для каждого из примеров.

2) Сохранив значение первого операнда $\boldsymbol{A}$, выбрать такое значение операнда $\boldsymbol{B}$, чтобы в операции сложения с одинаковыми знаками имел место особый случай переполнения формата. Выполнить два примера, иллюстрирующие эти случаи (см. выше «Примечание 1»). Для каждого из них выполнить следующее:

а) Проставить межразрядные переносы, возникающие при сложении.

б) Дать ЗИ операндов и результата. При получении отрицательного результата предварительно преобразовать его из дополнительного кода в прямой.

в) Дать БзИ операндов и результата. При получении неверного результата пояснить причину его возникновения.

г) Показать значения арифметических флагов для каждого из примеров.

3) Сохранив операнд ***B***, подобрать такое значение операнда ***A***, чтобы при сложении положительных операндов имело место переполнение формата, а при сложении отрицательных операндов (таких же по модулю, как и положительных) результат операции был бы корректным. Выполнить два примера, для каждого из которых выполнить следующее:

а) Проставить межразрядные переносы, возникающие при сложении.

б) Дать ЗИ операндов и результата. При получении отрицательного результата предварительно преобразовать его из дополнительного кода в прямой.

в) Дать БзИ операндов и результата. При получении неверного результата пояснить причину его возникновения.

г) Показать значения арифметических флагов для каждого из примеров.

Варианты заданий приведены в табл. 1 Приложения.

1.3 ПРИМЕР ВЫПОЛНЕНИЯ ЗАДАНИЯ

Операция двоичного сложения реализуется поразрядно, начиная с младших разрядов, при этом учитываются возникающие при этом межразрядные переносы. В каждом разряде сложение реализуется в соответствии со следующей таблицей:

Таблица 1.1

a_i	0	0	0	0	1	1	1	1
b_i	0	0	1	1	0	0	1	1
p_{i-1}	0	1	0	1	0	1	0	1
s_i	0	1	1	0	1	0	0	1
p_i	0	0	0	1	0	1	1	1

Здесь a_i – значение i-го разряда 1-го слагаемого, b_i – значение i-го разряда 2-го слагаемого, p_{i-1} – значение (i – 1)-го разряда в i-й разряд, s_i – сумма i-го разряда, p_i – перенос из i-го разряда в (i + 1)-й разряд.

Задание 1.1) ***A*** = 57, ***B*** = 49.

1) ***A*** > 0, ***B*** > 0 ***A*** = 0.0111001 ***B*** = 0.0110001

Так как числа положительны, то оставляем их в неизменном виде.

		ЗИ	БзИ
A =	0.0 1 1 1 0 0 1	57	57
B =	+ 0.0 1 1 0 0 0 1	+ 49	+ 49
C =	0.1 1 0 1 0 1 0		106
$C_{испр}$ =	0.1 1 0 1 0 1 0	106	

Установка флагов:

CF = 0 – отсутствие переноса из старшего разряда;

PF = 1 – число единиц четно;

AF = 0 – отсутствие межтетрадного переноса;

ZF = 0 – отсутствие нулевого результата;

SF = 0 – положительный результат;

OF = 0 – переполнение для знаковых чисел отсутствует.

Результат БзИ корректен (флаг ***CF*** = 0).

Используемые функции (предлагается написать самостоятельно):

*binout (**n**)* \\ *вывод **n** в двоичном формате*

*bindopcod (**n**)* \\ *получение дополнительного кода для **n***

*binpramcod (**n**)* \\ *получение прямого кода для **n***

*plus (**а**, **в**)* \\ *поразрядное сложение **а** и **в***

Алгоритм выполнения:

начало

вывод ***A*** и ***B*** в двоичном виде

C = ***A*** + ***B***

перевод ***C*** в двоичную систему счисления

вывод результата

вывод значений флагов

конец

2) ***A*** < 0, ***B*** > 0 ***A*** = 1.0111001 ***B*** = 0.0110001

Т.к. ***A*** < 0, преобразовываем его в дополнительный код, ***B*** не преобразовываем.

		ЗИ	**БзИ**
A =	1.1 0 0 0 0 1 1 1	−57	199
B =	+ 0.0 1 1 0 0 0 1	+ 49	+ 49
C =	1.1 1 1 1 0 0 0		248
$C_{испр}$ =	1.0 0 0 1 0 0 0	−8	

CF = 0; ***PF*** = 0; ***AF*** = 0; ***ZF*** = 0; ***SF*** = 1; ***OF*** = 0.

Результат БзИ корректен (флаг ***CF*** = 0).

Алгоритм выполнения:

начало

перевод ***A*** в дополнительный код

вывод ***A*** и ***B*** в двоичном виде

C = ***A*** + ***B***

перевод ***C*** в двоичную систему счисления

вывод ***C***

если (***C*** < 0)

начало если

перевод ***C*** в прямой код

вывод результата

конец если

вывод значений флагов

конец

3) $\boldsymbol{A} > 0, \boldsymbol{B} < 0$ $\boldsymbol{A} = 0.0111001$ $\boldsymbol{B} = 1.0110001$

Т.к. $\boldsymbol{B} < 0$, преобразовываем его в дополнительный код, $\boldsymbol{A}$ не преобразовываем.

		ЗИ	**БзИ**
A =	0.0 1 1 1 0 0 1	57	57
B =	+ 1.1 0 0 1 1 1 1	+ −49	+ 207
C =	0.0 0 0 1 0 0 0		8 ?
Сиспр=	0.0 0 0 1 0 0 0	8	

$\boldsymbol{CF} = 1$; $\boldsymbol{PF} = 0$; $\boldsymbol{AF} = 1$; $\boldsymbol{ZF} = 0$; $\boldsymbol{SF} = 0$; $\boldsymbol{OF} = 0$.

Для БзИ результат некорректен вследствие возникающего переноса из старшего разряда (флаг $\boldsymbol{CF} = 1$).

Алгоритм выполнения:

начало

перевод $\boldsymbol{B}$ в дополнительный код

вывод $\boldsymbol{A}$ и $\boldsymbol{B}$ в двоичном виде

$\boldsymbol{C} = \boldsymbol{A} + B$

перевод $\boldsymbol{C}$ в двоичную систему счисления

вывод $\boldsymbol{C}$

если ($\boldsymbol{C} < 0$)

начало если

перевод $\boldsymbol{C}$ в прямой код

вывод результата

конец если

вывод значений флагов

конец

4) $\boldsymbol{A} < 0, \boldsymbol{B} < 0$ $\boldsymbol{A} = 1.0111001$ $\boldsymbol{B} = 1.0110001$

Т.к. $\boldsymbol{A}$ и $\boldsymbol{B}$ отрицательны, преобразовываем оба операнда в дополнительный код.

		ЗИ	**БзИ**
A =	1.1 0 0 0 1 1 1	−57	199
B =	+ 1.1 0 0 1 1 1 1	+ −49	+ 207
C =	1.0 0 1 0 1 1 0		150 ?
Сиспр =	1.1 1 0 1 0 1 0	−106	

$\boldsymbol{CF} = 1$; $\boldsymbol{PF} = 1$; $\boldsymbol{AF} = 1$; $\boldsymbol{ZF} = 0$; $\boldsymbol{SF} = 1$; $\boldsymbol{OF} = 0$.

Для БзИ результат некорректен вследствие возникающего переноса из старшего разряда (флаг $\boldsymbol{CF} = 1$).

Алгоритм выполнения:

начало

перевод $\boldsymbol{A}$ и $\boldsymbol{B}$ в дополнительный код

вывод $\boldsymbol{A}$ и $\boldsymbol{B}$ в двоичном виде

$\boldsymbol{C} = \boldsymbol{A} + \boldsymbol{B}$

перевод $\boldsymbol{C}$ в двоичную систему счисления

вывод ***C*** (дополнительный код)
перевод ***C*** в прямой код
вывод результата
вывод значений флагов
конец

Задание 1.2) ***B*** подбирается из условия ***A*** + ***B*** > 128, откуда получаем 128 – ***A*** < ***B*** ≤ 127.

A = 57 (неизменно), ***B*** = 96 (подобрано).

1) ***A*** > 0, ***B*** > 0 ***A*** = 0.0111001 ***B*** = 0.1100000

		ЗИ	БзИ
A =	0.0 1 1 1 0 0 1	57	57
B =	+ 0.1 1 0 0 0 0 0	+ 96	+ 96
C =	1.0 0 1 1 0 0 1		153
Сиспр =	1.1 1 0 0 1 1 1	-103 ?	

CF = 0; ***PF*** = 1; ***AF*** = 0; ***ZF*** = 0; ***SF*** = 1; ***OF*** = 1.

Для ЗИ результат некорректен из-за переполнения формата разрядной сетки (флаг ***OF*** = 1), для БзИ результат корректен (флаг ***CF*** = 0).

Алгоритм выполнения:

начало
вывод ***A*** и ***B*** в двоичном виде
C = ***A*** + ***B***
перевод ***C*** в двоичную систему счисления
вывод ***C*** (дополнительный код)
перевод ***C*** в прямой код
вывод результата
вывод значений флагов
конец

2) ***A*** < 0, ***B*** < 0 ***A*** = 1.1000111 ***B*** = 1.0100000

		ЗИ	БзИ
A =	1.1 0 0 0 1 1 1	-57	199
B =	+ 1.0 1 0 0 0 0 0	+ -96	+ 160
C =	0.1 1 0 0 1 1 1		103 ?
Сиспр =	0.1 1 0 0 1 1 1	103 ?	

CF = 1; ***PF*** = 0; ***AF*** = 0; ***ZF*** = 0; ***SF*** = 0; ***OF*** = 1.

Для ЗИ результат некорректен из-за переполнения формата разрядной сетки (флаг ***OF*** = 1), для БзИ результат некорректен вследствие возникающего переноса из старшего разряда (флаг ***CF*** = 1).

Алгоритм выполнения:

начало
перевод ***A*** и ***B*** в дополнительный код
вывод ***A*** и ***B*** в двоичном виде

$C = A + B$
перевод C в двоичную систему счисления
вывод результата
вывод значений флагов
конец

Задание 1.3) Фиксируем B = 49, A подбирается из условия $A + B$ = 128, откуда получаем A = 79. При этих значениях при сложении положительных чисел будет фиксироваться переполнение, а при сложении отрицательных – не будет.

1) $A > 0$, $B > 0$ A = 0.1001111 B = 0.0110001

		ЗИ		**БзИ**
A =	0.1 0 0 1 1 1 1	79		79
B =	+ 0.0 1 1 0 0 0 1	+ 49		+ 49
C =	1.0 0 0 0 0 0 0			128
$C_{испр}$ =	1.0 0 0 0 0 0 0	-128	?	

CF = 0; PF = 0; AF = 1; ZF = 0; SF = 1; OF = 1.

Для ЗИ результат некорректен вследствие возникающего переполнения (флаг OF = 1), для БзИ результат корректен (флаг CF = 0).

Алгоритм выполнения:

начало
вывод A и B в двоичном виде
$C = A + B$
перевод C в двоичную систему счисления
вывод C (дополнительный код)
перевод C в прямой код
вывод результата
вывод значений флагов
конец

2) $A < 0$, $B < 0$ A = 1.0110001 B = 1.1001111

		ЗИ	**БзИ**	
A =	1.0 1 1 0 0 0 1	-79	177	
B =	+ 1.1 0 0 1 1 1 1	+ -49	+ 207	
C =	1.0 0 0 0 0 0 0		128	?
$C_{испр}$ =	1.0 0 0 0 0 0 0	-128		

CF = 1; PF = 0; AF = 1; ZF = 0; SF = 1; OF = 0.

Для ЗИ результат корректен (флаг OF = 0), для БзИ результат некорректен вследствие возникающего переполнения (флаг CF = 1).

Алгоритм выполнения:

начало
перевод A и B в дополнительный код
вывод A и B в двоичном виде

C = **A** + **B**
перевод **C** в двоичную систему счисления
вывод **C** (дополнительный код)
перевод **C** в прямой код
вывод результата
вывод значений флагов
конец

2 ВЫЧИТАНИЕ ЦЕЛЫХ ЧИСЕЛ

2.1 Основные положения

Вычитание целых чисел выполняется в байтном формате (размер разрядной сетки – 8 бит).

При вычитании целых чисел с фиксированной запятой используются их дополнительные коды, при этом знаковый разряд участвует в операции точно так же, как и цифровые. В результате этого автоматически формируется знак результата, причем результат получается в дополнительном коде.

При вычитании целых чисел с фиксированной запятой используются следующие арифметические флаги:

CF – Carry Flag (флаг переноса). В нем фиксируется заем в старший разряд при вычитании.

PF – Parity Flag (флаг четности). Он устанавливается при наличии четного числа единиц в младшем байте результата, в противном случае – сбрасывается.

AF – Auxiliary Flag (флаг вспомогательного переноса). В нем фиксируется межтетрадный (из младшей тетрады в старшую) заем при вычитании.

ZF – Zero Flag (флаг нуля). Он устанавливается при нулевом результате операции, в противном случае – сбрасывается.

SF – Sign Flag (флаг знака). В него копируется старший разряд результата, интерпретируемый как знак.

OF – Overflow Flag (флаг переполнения). Он устанавливается в командах вычитания в случае, если результат операции не помещается в формате операндов.

Корректность при БзИ проверятся по флагу ***CF***:

1) ***CF*** = 0. Заем в старший разряд при вычитании отсутствует, следовательно, результат БзИ корректен;

2) ***CF*** = 1. Заем в старший разряд при вычитании имеет место, следовательно, результат БзИ некорректен.

Корректность при ЗИ проверятся по флагу ***OF***:

1) ***OF*** = 0. Переполнения формата разрядной сетки при вычитании не произошло, результат ЗИ корректен;

2) $OF = 1$. Результат ЗИ некорректен, т.к. при вычитании произошло переполнение формата разрядной сетки, о котором можно судить по двум признакам:

а) знаки операндов разные, а знак суммы отличается от знака первого операнда;

б) не совпадают заемы в два старших разряда: один есть, а другого нет.

Примечание 1. При $OF = 1$ имеет место так называемый особый случай переполнения формата. Такое переполнение происходит при условии $A + B > 128$, откуда получаем $128 - A < B \le 127$. В случае вычитания переполнение имеет место при разных знаках операндов.

Примечание 2. В пункте 3 настоящего задания (см. ниже) A подбирается из условия $A + B = 128$, при этом при вычитании из положительного числа отрицательного будет фиксироваться переполнение, а при вычитании из отрицательного числа положительного переполнение фиксироваться не будет.

2.2 Задание

1) Для заданных чисел A и B выполнить операцию знакового вычитания со всеми комбинациями знаков операндов (4 случая).

Для каждого примера выполнить следующее:

а) Проставить межразрядные заемы, возникающие при вычитании.

б) Дать ЗИ операндов и результата. При получении отрицательного результата предварительно преобразовать его из дополнительного кода в прямой.

в) Дать БзИ операндов и результата. При получении неверного результата пояснить причину его возникновения.

г) Показать значения арифметических флагов для каждого из примеров.

2) Сохранив значение первого операнда A, выбрать такое значение операнда B, чтобы в операции вычитания с разными знаками операндов имел место особый случай переполнения формата. Выполнить два примера, иллюстрирующие эти случаи (см. выше «Примечание 1»). Для каждого из них выполнить следующее:

а) Проставить межразрядные заемы, возникающие при вычитании.

б) Дать ЗИ операндов и результата. При получении отрицательного результата предварительно преобразовать его из дополнительного кода в прямой.

в) Дать БзИ операндов и результата. При получении неверного результата пояснить причину его возникновения.

г) Показать значения арифметических флагов для каждого из примеров.

3) Сохранив операнд ***B***, подобрать такое значение операнда ***A***, чтобы при вычитании из положительного операнда отрицательного имело место переполнение формата, а при вычитании из отрицательного операнда положительного (таких же по модулю) результат операции был бы корректным. Выполнить два примера, для каждого из которых выполнить следующее:

а) Проставить межразрядные заемы, возникающие при вычитании.

б) Дать ЗИ операндов и результата. При получении отрицательного результата предварительно преобразовать его из дополнительного кода в прямой.

в) Дать БзИ операндов и результата. При получении неверного результата пояснить причину его возникновения.

г) Показать значения арифметических флагов для каждого из примеров.

Варианты заданий приведены в табл. 1 Приложения.

2.3 ПРИМЕР ВЫПОЛНЕНИЯ ЗАДАНИЯ

Операция двоичного вычитания реализуется поразрядно, начиная с младших разрядов, при этом учитываются возникающие при этом межразрядные заемы. В каждом разряде вычитание реализуется в соответствии со следующей таблицей:

Таблица 2.1

a_i	0	0	0	0	1	1	1	1
b_i	0	0	1	1	0	0	1	1
z_{i-1}	0	1	0	1	0	1	0	1
r_i	0	1	1	0	1	0	0	1
z_i	0	1	1	1	0	0	0	1

Здесь a_i – значение i-го разряда уменьшаемого, b_i – значение i-го разряда вычитаемого, z_{i-1} – значение заема из i-го разряда в (i – 1)-й предыдущий младший разряд, s_i –разность в i-м разряде, z_i – значение заема из (i + 1)-го старшего разряда в i-й разряд.

Задание 2.1) ***A*** = 67, ***B*** = 51.

1) ***A*** > 0, ***B*** > 0 ***A*** = 0.1000011 ***B*** = 0.0110011

		ЗИ	БзИ
A =	– 0.1 0 0 0 0 1 1	– 67	– 67
B =	0.0 1 1 0 0 1 1	51	51
C =	0.0 0 1 0 0 0 0		16
$C_{испр}$ =	0.0 0 1 0 0 0 0	16	

CF = 0; ***PF*** = 0; ***AF*** = 0; ***ZF*** = 0; ***SF*** = 0; ***OF*** = 0.

Результат БзИ корректен (флаг ***CF*** = 0).

Используемые функции (предлагается написать самостоятельно):

*binout (**n**)* \\ *вывод **n** в двоичном формате*

*bindopcod (**n**)* \\ *получение дополнительного кода для **n***

*binpramcod (**n**)* \\ *получение прямого кода для **n***

*minus (**a**, **в**)* \\ *поразрядное вычитание **a** и **в***

Алгоритм выполнения:

начало

вывод ***A*** и ***B*** в двоичном виде

C = ***A*** – ***B***

перевод ***C*** в двоичную систему счисления

вывод ***C***

если (***C*** < 0)

начало если

перевод ***C*** в прямой код

вывод результата

конец если

вывод значений флагов

конец

2) ***A*** < 0, ***B*** > 0   ***A*** = 1.0111101   ***B*** = 0.0110011

		ЗИ	БзИ
A =	– 1.0 1 1 1 1 1 0 1	– -67	– 189
B =	0.0 1 1 0 0 1 1	51	51
C =	1.0 0 0 0 1 0 1 0		138
$С_{испр}$ =	1.1 1 1 1 0 1 1 0	-118	

CF = 0; ***PF*** = 0; ***AF*** = 0; ***ZF*** = 0; ***SF*** = 1; ***OF*** = 0.

Результат БзИ корректен (флаг ***CF*** = 0).

Алгоритм выполнения:

начало

перевод ***A*** в дополнительный код

вывод ***A*** и ***B*** в двоичном виде

C = ***A*** – ***B***

перевод ***C*** в двоичную систему счисления

вывод ***C*** (дополнительный код)

перевод ***C*** в прямой код

вывод результата

вывод значений флагов

конец

3) ***A*** > 0, ***B*** < 0   ***A*** = 0.1000011   ***B*** = 1.1001101

		ЗИ	БзИ
A =	– 0.1 0 0 0 0 0 1 1	– 67	– 67
B =	1.1 0 0 1 1 0 1	-51	205
C =	0.1 1 1 1 0 1 1 0		118 ?

$C_{испр}$ = 0.1 1 1 0 1 1 0   118

CF = 1; ***PF*** = 0; ***AF*** = 1; ***ZF*** = 0; ***SF*** = 0; ***OF*** = 0.

Для БзИ результат некорректен вследствие возникающего заема из разряда за пределами формата (флаг ***CF*** = 1).

Алгоритм выполнения:

начало

перевод ***B*** в дополнительный код

вывод ***A*** и ***B*** в двоичном виде

C = ***A*** – ***B***

перевод ***C*** в двоичную систему счисления

вывод результата

вывод значений флагов

конец

4) ***A*** < 0, ***B*** < 0   ***A*** = 1.0111101   ***B*** = 1.1001101

		ЗИ	БзИ	
A =	1.0 1 1 1 1 0 1	–67	189	
B =	– 1.1 0 0 1 1 0 1	– –51	– 205	
C =	1.1 1 1 0 0 0 0		240	?
$C_{испр}$ =	1.0 0 1 0 0 0 0	–16		

CF = 1; ***PF*** = 1; ***AF*** = 0; ***ZF*** = 0; ***SF*** = 1; ***OF*** = 0.

Для БзИ результат неверен вследствие возникающего заема из разряда за пределами формата (флаг ***CF*** = 1).

Алгоритм выполнения:

начало

перевод ***A*** и ***B*** в дополнительный код

вывод ***A*** и ***B*** в двоичном виде

C = ***A*** – ***B***

перевод ***C*** в двоичную систему счисления

вывод ***C***

если (***C*** < 0)

начало если

перевод ***C*** в прямой код

вывод результата

конец если

вывод значений флагов

конец

Задание 2.2) ***B*** подбирается из условия ***A*** + ***B*** > 128, откуда 128 – ***A*** < ***B*** < 127.

A = 67 (неизменно), ***B*** = 64 (подобрано).

1) ***A*** < 0, ***B*** > 0   ***A*** = 1.0111101   ***B*** = 0.1000000

		ЗИ	БзИ
A =	− 1.0 1 1 1 1 0 1	− -67	− 189
B =	0.1 0 0 0 0 0 0	64	64
C =	0.1 1 1 1 1 0 1		125
$C_{испр}$ =	0.1 1 1 1 1 0 1	125 ?	

CF = 0; ***PF*** = 1; ***AF*** = 0; ***ZF*** = 0; ***SF*** = 0; ***OF*** = 1.

Для ЗИ результат некорректен из-за переполнения формата разрядной сетки (флаг ***OF*** = 1), для БзИ результат корректен (флаг ***CF*** = 0).

Алгоритм выполнения:

начало

вывод ***A*** и ***B*** в двоичном виде

C = ***A*** – ***B***

перевод ***C*** в двоичную систему счисления

вывод ***C*** (дополнительный код)

перевод ***C*** в прямой код

вывод результата

вывод значений флагов

конец

2) ***A*** > 0, ***B*** < 0 ***A*** = 0.1000011 ***B*** = 1.1000000

		ЗИ	БзИ
A =	− 0.1 0 0 0 0 1 1	− 67	− 67
B =	1.1 0 0 0 0 0 0	-64	192
C =	1.0 0 0 0 0 1 1		131 ?
$C_{испр}$ =	1.1 1 1 1 1 0 1	-125 ?	

CF = 1; ***PF*** = 0; ***AF*** = 0; ***ZF*** = 0; ***SF*** = 1; ***OF*** = 1.

Для ЗИ результат некорректен из-за возникающего переполнения (***OF*** = 1), для БзИ результат некорректен из-за возникающего заема в старший разряд (***CF*** = 1).

Алгоритм выполнения:

начало

перевод ***A*** и ***B*** в дополнительный код

вывод ***A*** и ***B*** в двоичном виде

C = ***A*** – ***B***

перевод ***C*** в двоичную систему счисления

вывод ***C*** (дополнительный код)

перевод ***C*** в прямой код

вывод результата

вывод значений флагов

конец

Задание 2.3) Фиксируем ***B*** = 51, ***A*** подбирается из условия ***A*** + ***B*** = 128, откуда получаем ***A*** = 77. При этих значениях при вычитании из положительного числа отрицательного будет фиксироваться переполнение, а при вычитании из отрицательного числа положительного переполнение фиксироваться не будет.

1) $A > 0, B < 0$ $A = 0.1001101$ $B = 1.1001101$

		ЗИ	БзИ
A =	− 0.1 0 0 1 1 0 1	− 77	− 77
B =	1.1 0 0 1 1 0 1	-51	205
C =	1.0 0 0 0 0 0 0		128 ?
$C_{испр}$ =	1.0 0 0 0 0 0 0 0	-128 ?	

CF = 1; ***PF*** = 0; ***AF*** = 0; ***ZF*** = 0; ***SF*** = 1; ***OF*** = 1.

Для ЗИ результат некорректен из-за возникающего переполнения (***OF*** = 1), для БзИ результат некорректен из-за возникающего заема в старший разряд (***CF*** = 1).

Алгоритм выполнения:

начало

вывод ***A*** и ***B*** в двоичном виде

C = ***A*** – ***B***

перевод ***C*** в двоичную систему счисления

вывод ***C*** (дополнительный код)

перевод ***C*** в прямой код

вывод результата

вывод значений флагов

конец

2) $A < 0, B > 0$ $A = 1.0110011$ $B = 0.0110011$

		ЗИ	БзИ
A =	− 1.0 1 1 0 0 1 1	− -77	− 179
B =	0.0 1 1 0 0 1 1	51	51
C =	1.0 0 0 0 0 0 0		128
$C_{испр}$ =	1.0 0 0 0 0 0 0 0	-128	

CF = 0; ***PF*** = 0; ***AF*** = 0; ***ZF*** = 0; ***SF*** = 1; ***OF*** = 0.

Для ЗИ и БзИ результаты корректны (флаги ***OF*** = 0; ***CF*** = 0).

Алгоритм выполнения:

начало

перевод ***A*** и ***B*** в дополнительный код

вывод ***A*** и ***B*** в двоичном виде

C = ***A*** – ***B***

перевод ***C*** в двоичную систему счисления

вывод ***C*** (дополнительный код)

перевод ***C*** в прямой код

вывод результата

вывод значений флагов

конец

3 УМНОЖЕНИЕ ЦЕЛЫХ ЧИСЕЛ

3.1 Задание

1). *В разрядной сетке длиной в один байт (один разряд знаковый и семь – цифровых) выполнить операцию умножения заданных целых чисел А и В со всеми комбинациями знаков, используя метод умножения с применением коррекции. При выполнении операции использовать способ умножения с поразрядным анализом множителя, начиная от его младших разрядов со сдвигом суммы частных произведений (СЧП) вправо. Для представления произведения использовать удвоенную разрядную сетку (16 двоичных разрядов: один – знаковый и 15 – цифровых). Результаты представить в десятичной системе счисления и проверить их правильность.*

2). *В той же разрядной сетке, что и в п.1, выполнить операцию умножения заданных целых чисел А и В со всеми комбинациями знаков, используя метод умножения без коррекции. Результаты представить в десятичной системе счисления и проверить их правильность.*

Варианты заданий приведены в табл. 2 Приложения.

3.2 Основные положения

3.2.1 Термины и обозначения

ЧП$_i$ – частное произведение множимого на i-й разряд множителя

СЧП$_i$ – сумма частных произведений на i-м шаге умножения

$\overrightarrow{\text{СЧП}_i}$ – сдвиг вправо СЧП$_i$ на 1 разряд

СЧП – сумма частных произведений (результат умножения)

Дополнение – представление числа в дополнительном коде, если исходное число представлено в прямом коде, и в прямом, если исходное записано в дополнительном коде

Простой сдвиг – сдвиг числа на 1 разряд вправо. Знак числа при этом не учитывается (т.е. старший разряд имеет значение 0)

Арифметический сдвиг – сдвиг числа на 1 разряд вправо. Знак числа сохраняется (т.е. старший разряд после сдвига сохраняет свое значение)

3.2.2 Принцип умножения

Для умножения двоичных чисел можно использовать способ умножения начиная с младших разрядов множителя. Пусть множимое A_2 = 1101, множитель $B_2 = b_3b_2b_1b_0$ = 1011, где b_i – значение i-го разряда множителя.

```
     1101
   × 1011
     ----
     1101
    1101
   0000
  1101
 --------
 10001111
```

ЧП$_0$ = А×b_0 = 1101×1 = 1101

ЧП$_1$ = А×b_1 = 1101×1 = 1101

ЧП$_2$ = А×b_2 = 1101×0 = 0000

ЧП$_3$ = А×b_3 = 1101×1 = 1101

СЧП = ЧП$_0$ + ЧП$_1$ + ЧП$_2$ + ЧП$_3$

Здесь $ЧП_i$ - частное произведение множимого на i-й разряд множителя, СЧП – сумма частных произведений. СЧП также является результатом операции умножения.

Таким образом, умножение сводится к последовательным сложениям частных произведений. При этом единица в разряде множителя означает, что к сумме частных произведений добавляется множимое с соответствующим сдвигом; если разряд множителя нулевой, то СЧП не изменяется. Поэтому кроме операции сложения чисел для получения произведения необходима операция сдвига чисел. При этом появляется возможность сдвигать множимое или сумму частных произведений, что дает основание для разных методов реализации операции умножения.

Будем использовать умножение, начиная с младших разрядов множителя со сдвигом СЧП вправо.

Представим множимое и множитель в следующем виде:

$$A = a_7a_6\ldots a_0, \quad B = b_7b_6\ldots b_0$$

Запишем множитель в виде суммы произведений значений разрядов и соответствующих степеней двойки:

$$B = b_7\times2^7 + b_6\times2^6 + \ldots +b_0\times2^0$$

Преобразуем множитель к виду:

$$B = ((((((((0 + b_0)\times2^{-1} + b_1)\times2^{-1} + b_2)\times2^{-1} + b_3)\times2^{-1} + b_4)\times2^{-1} + b_5)\times2^{-1} + b_6)\times2^{-1} + b_7)\times2^{-1})\times2^8$$

$$C = A\times B = ((((((((0 + A\times b_0)\times2^{-1} + A\times b_1)\times2^{-1} + A\times b_2)\times2^{-1} + A \times b_3)\times2^{-1} + A \times b_4)\times2^{-1} + A\times b_5)\times2^{-1} + A\times b_6)\times2^{-1} + A\times b_7)\times2^{-1})\times2^8 \quad (*)$$

Выделим в этой формуле следующие промежуточные значения: частные произведения и их суммы (см. табл. 3.1):

Таблица 3.1

№ шага	Частные произведения	Суммы частных произведений текущего шага	Суммы после сдвига
0	$ЧП_0 = A \times b_0$	$СЧП_0 = 0 + ЧП_0$	$\overrightarrow{СЧП}_0 = СЧП_0 \times 2^{-1}$
1	$ЧП_1 = A \times b_1$	$СЧП_1 = СЧП_0 + ЧП_1$	$\overrightarrow{СЧП}_1 = СЧП_1 \times 2^{-1}$
2	$ЧП_2 = A \times b_2$	$СЧП_2 = СЧП_1 + ЧП_2$	$\overrightarrow{СЧП}_2 = СЧП_2 \times 2^{-1}$
3	$ЧП_3 = A \times b_3$	$СЧП_3 = СЧП_2 + ЧП_3$	$\overrightarrow{СЧП}_3 = СЧП_3 \times 2^{-1}$
4	$ЧП_4 = A \times b_4$	$СЧП_4 = СЧП_3 + ЧП_4$	$\overrightarrow{СЧП}_4 = СЧП_4 \times 2^{-1}$
5	$ЧП_5 = A \times b_5$	$СЧП_5 = СЧП_4 + ЧП_5$	$\overrightarrow{СЧП}_5 = СЧП_5 \times 2^{-1}$
6	$ЧП_6 = A \times b_6$	$СЧП_6 = СЧП_5 + ЧП_6$	$\overrightarrow{СЧП}_6 = СЧП_6 \times 2^{-1}$
7	$ЧП_7 = A \times b_7$	$СЧП_7 = СЧП_6 + ЧП_7$	$\overrightarrow{СЧП}_7 = СЧП_7 \times 2^{-1}$

Приведем пример реализации формулы (*) в двоичных кодах. Рассмотрим вариант умножения начиная с младших разрядов множителя со сдвигом СЧП вправо:

	A	00001111		множимое
	B	00001101		множитель
№ шага	Разряд множителя	Промежуточные значения в двоичном коде		Операции
0	 b_0=1	00000000 00001111 00001111 00000111	 1	$СЧП_{нач}$ $ЧП_0$ $СЧП_0$ $\overrightarrow{СЧП_0}$
1	b_1=0	00000000 00000111 00000011	 1 11	$ЧП_1$ $СЧП_1$ $\overrightarrow{СЧП_1}$
2	b_2=1	00001111 00010010 00001001	 11 011	$ЧП_2$ $СЧП_2$ $\overrightarrow{СЧП_2}$
3	b_3=1	00001111 00011000 00001100	 011 0011	$ЧП_3$ $СЧП_3$ $\overrightarrow{СЧП_3}$
4	b_4=0	00000000 00001100 00000110	 0011 00011	$ЧП_4$ $СЧП_4$ $\overrightarrow{СЧП_4}$
5	b_5=0	00000000 00000110 00000011	 00011 000011	$ЧП_5$ $СЧП_5$ $\overrightarrow{СЧП_5}$
6	b_6=0	00000000 00000011 00000001	 000011 1000011	$ЧП_6$ $СЧП_6$ $\overrightarrow{СЧП_6}$
7	b_7=0	00000000 00000001 00000000	 1000011 11000011	$ЧП_7$ $СЧП_7$ $\overrightarrow{СЧП_7}$

$СЧП = СЧП_7 \times 2^8 = (00000000{,}11000011) \times 2^8 = 11000011$

Результат умножения получается простым перенесением запятой вправо на 8 разрядов.

Рассмотрим подробнее представленный в этой таблице алгоритм.

Сначала обнулим СЧП$_{нач}$. Если этого не сделать, то возможен случай, когда СЧП$_{нач}$ будет иметь ненулевое значение, и результаты операций с самого начала окажутся неверными. Обратимся к формуле (*).

$$B = ((((((((0 + b_0)\times 2^{-1} + b_1)\times 2^{-1} + b_2)\times 2^{-1} + b_3)\times 2^{-1} + b_4)\times 2^{-1} + b_5)\times 2^{-1} + b_6)\times 2^{-1} + b_7)\times 2^{-1})\times 2^8$$

Здесь $b_7, b_6, \ldots, b_0$ – разряды множителя, пронумерованные справа налево.

Для получения СЧП$_i$ сдвигаем СЧП$_{i-1}$ на 1 разряд вправо и прибавляем к нему результат перемножения множимого A и b_i. Продолжаем выполнять эти действия до тех пор, пока i не станет равным 7 (т.к. множимое и множитель – 8-разрядные).

Вычислив СЧП$_7$ = СЧП$_6\times 2^{-1} + A\times b_7$, можем получить из нее СЧП, сдвигая СЧП$_7$ вправо на единицу и домножая полученное значение на 2^8 (другими словами, сдвигая его на 8 разрядов влево) для получения целого числа.

3.2.3 Особенности используемого метода умножения

Используемый в данном задании метод умножения базируется на представлении положительных операндов в прямом, а отрицательных – в дополнительном кодах.

Достоинства этого метода:

а). Не требуется преобразовывать операнды и результат из дополнительного кода в прямой код и обратно.

$A>0, B>0 \quad C_{пр} = A_{пр}\times B_{пр}$
$A<0, B>0 \quad C_{доп} = A_{доп}\times B_{пр}$
$A>0, B<0 \quad C_{доп} = A_{пр}\times B_{доп}$
$A<0, B<0 \quad C_{пр} = A_{доп}\times B_{доп}$

б) Результат имеет не 14, а 15 числовых разрядов, что повышает точность.

в) Положительный результат операции представляется в прямом коде, а отрицательный – в дополнительном.

3.2.4 Замечания по реализации метода

3.2.4.1 Использование беззнаковых переменных

Особенностью использованного метода является то, что знаковые разряды используются в операции умножения наряду с цифровыми, т.е. фактически в умножении участвуют беззнаковые положительные операнды.

Например:

1). Один из операндов равен $\boxed{+15}$. Его двоичное представление в 8-битовом коде:

0	0	0	0	1	1	1	1

Участвующий в операции умножения операнд в десятичном беззнаковом представлении $\boxed{15}$.

2). Один из операндов равен $\boxed{-15}$. Его двоичное представление в 8-битовом оде:

$$\boxed{1}\boxed{1}\boxed{1}\boxed{1}\boxed{0}\boxed{0}\boxed{0}\boxed{1}$$

Участвующий в операции операнд в десятичном беззнаковом представлении $\boxed{241}$.

Таким образом, при программной реализации данного метода все участвующие в умножении операнды должны быть беззнаковыми и сформированными так, чтобы их битовый состав в точности повторял битовый состав соответствующих знаковых переменных.

3.2.4.2 Формирование результата операции

Поскольку операция умножения выполняется над беззнаковыми операндами, следует позаботиться об учете их реальных знаков. Для этого необходимо выполнить два вида коррекции, которые обеспечат получение положительного результата операции умножения в прямом коде, а отрицательного – в дополнительном.

3.2.4.2.1 Первый вид коррекции

С помощью коррекции первого вида выполняется учет реального знака множимого. На каждом шаге умножения формируется одно частное произведение и одна сумма частных произведений. Рассмотрим это для положительного и отрицательного множимых.

Для положительного множимого A=$\boxed{+15}$ битовое представление

$$\boxed{0}\boxed{0}\boxed{0}\boxed{0}\boxed{1}\boxed{1}\boxed{1}\boxed{1}$$

Соответствующий множимому беззнаковый операнд $\boxed{0}\boxed{0}\boxed{0}\boxed{0}\boxed{1}\boxed{1}\boxed{1}\boxed{1}$.

	00001100	0011	$СЧП_{i\text{-}1}$
	00000000		$ЧП_i$
b_i=0	$\boxed{0}$0001100	0011	$СЧП_i$
	0$\boxed{0}$000110	00011	$\overrightarrow{СЧП}_i$

Для отрицательного множимого A=$\boxed{-15}$ битовое представление

$$\boxed{1}\boxed{1}\boxed{1}\boxed{1}\boxed{0}\boxed{0}\boxed{0}\boxed{1}$$

Соответствующий множимому беззнаковый операнд $\boxed{1}\boxed{1}\boxed{1}\boxed{1}\boxed{0}\boxed{0}\boxed{0}\boxed{1}$.

	11110011	1101	$\overrightarrow{СЧП}_{i-1}$
	00000000		$ЧП_i$
b_i=0	$\boxed{1}$1110011	0011	$СЧП_i$
	0$\boxed{1}$111001	11101	$\overrightarrow{СЧП}_i$

Из приведенных примеров видно, что старший разряд $СЧП_i$, соответствующий знаковому разряду множимого, при сдвиге $\overrightarrow{СЧП}_i$ формирует очередной разряд промежуточного результата (СЧП) в соответствующем коде (для положительного множителя в прямом $\boxed{0}$, для отрицательного – в дополнительном $\boxed{1}$).

Результат умножения отрицательного множимого на любой бит положительного множителя (беззнаковый операнд считается положительным) должен быть отрицательным. При выполнении операции сдвига СЧП_i его старший бит, соответствующий знаковому разряду, автоматически заполняется нулем. Поэтому для сохранения знака результата старший бит СЧП_i должен быть скорректирован, т.е. должен совпадать со знаком множимого. Добиться этого можно с помощью сложения $\overrightarrow{\text{СЧП}_i}$ с беззнаковой переменной $k = \boxed{zn}\boxed{0}\boxed{0}\boxed{0}\boxed{0}\boxed{0}\boxed{0}\boxed{0}$, где zn - знаковый разряд множимого ($zn = 0$ при положительном множимом, $zn = 1$ при отрицательном). С учетом первого вида коррекции формулу (*) можно преобразовать к виду:

$$C = A\times B = (((((((((0 + A\times b_0)\times 2^{-1} + k + A\times b_1)\times 2^{-1} + k + A\times b_2)\times 2^{-1} + k + A\times b_3)\times 2^{-1} + k + A\times b_4)\times 2^{-1} + k + A\times b_5)\times 2^{-1} + k + A\times b_6)\times 2^{-1} + k + A\times b_7)\times 2^{-1} + k)\times 2^8 \qquad (*)$$

3.2.4.2.2 Второй вид коррекции

С помощью коррекции второго вида обеспечивается учет реального знака операнда, выполняющего функцию множителя.

Очевидно, что для положительного множителя его значение и значение соответствующей беззнаковой переменной совпадают, и, следовательно, результат умножения не требует коррекции.

Для отрицательного множителя (см. пример в п. 3.2.4.1) значение самого множителя $\boxed{-15}$, а беззнаковой переменной $\boxed{241}$, поэтому нужна коррекция результат умножения. Для ее выполнения необходимо определить связь между отрицательным множителем и его беззнаковым представлением. Эту связь можно записать в виде:

$$-15 = 241 - 256,$$

т.к. любое 8-разрядное отрицательное число можно представить как разность его беззнакового представления и $256 = 2^8$.

Таким образом, операция умножения с участием отрицательного множителя может быть представлена следующим образом:

$$\boldsymbol{C = A \times B = A \times (D - 2^8) = A \times D - A \times 2^8},$$

где $\boldsymbol{D}$ – это беззнаковое представление множителя $\boldsymbol{B}$.

Следовательно, формула умножения (*) с учетом двух коррекций может быть записана в общем виде:

$$C = A\times B = (((((((((0 + A\times b_0)\times 2^{-1} + k + A\times b_1)\times 2^{-1} + k + A\times b_2)\times 2^{-1} + k + A\times b_3)\times 2^{-1} + k + A\times b_4)\times 2^{-1} + k + A\times b_5)\times 2^{-1} + k + A\times b_6)\times 2^{-1} + k + A\times b_7)\times 2^{-1} + k - A\times b_7)\times 2^8 \qquad (*)$$

3.3 Особенности реализации алгоритма умножения

Нужно сформировать следующие 16-разрядные беззнаковые переменные (далее такие переменные будут заключаться в квадратные скобки) (см. табл. 3.3):

Таблица 3.3

Обозначение	Роль переменной
[A]	множимое
[B]	множитель
[ЧП]	частное произведение
[СЧП]	сумма частных произведений
[КОР1]	вспомогательная переменная для коррекции первого вида
[КОР2]	вспомогательная переменная для коррекции второго вида

Будем рассматривать эти беззнаковые переменные как дробные числа, у которых старшие байты являются целой частью, а младшие – дробной.

[*A*] В целую часть (старший байт) заносим значения 8-ми разрядов множимого ***A*** (см. п. 3.2.4.1), а дробную часть (младший байт) заполняем нулями.

[*B*] В младший байт заносим значения 8-ми разрядов множителя ***B*** (см. п. 3.2.4.1), а старший байт заполняем значениями знакового разряда множителя (если множитель положительный – нулями, если отрицательный – единицами).

[СЧП] Изначально в старший байт заносим значения 8-ми разрядов множимого ***A***, а в младший байт - значения 8-ми разрядов множителя ***B*** (см. п. 3.2.4.2). Это можно сделать, например, с помощью операции **[СЧП$_{\text{нач}}$]** = **[*B*]**.

[ЧП] Вычисляем по формуле **[ЧП$_i$] = [*A*]** × b_i. Так как разряд множителя b_i может принимать только два значения (0 или 1), **[ЧП$_i$] = [0]** или **[ЧП$_i$] = [*A*]**.

[КОР1] В старший разряд заносим значение старшего разряда **[*A*]** (0, если множимое положительное и 1, если отрицательное). Остальные 15 разрядов заполняем нулями.

[КОР2] Значение зависит от знака множителя. Если множитель отрицательный, в старший байт заносим дополнение ***A***, а младший байт заполняем нулями. Иначе все 16 разрядов заполняем нулями.

При умножении отрицательного множимого на младшие нули четного множителя сдвиг СЧП$_i$ осуществляется обычным образом (т.е. в освободившийся старший разряд СЧП$_i$ заносится 0) (см. пример 3.4.2).

> **На каждом *i*-м шаге операции (начиная с *i* = 1) в качестве СЧП$_{i-1}$ берется ее преобразованное (сдвинутое и скорректированное) на предыдущем шаге значение.**
>
> **Соответствующие значения и обозначения выделены в примерах полужирным шрифтом.**

3.4. Примеры, иллюстрирующие работу этого метода

3.4.1 *A*>0, *B*>0 (результат *C* получится в прямом коде). *A* = 15, *B* = 13.

Таблица 3.4. Начальные значения переменных

	Старший байт	Младший байт
[*A*]	00001111	00000000
B		00001101
[$СЧП_{нач}$]	00000000	00001101
[КОР1]	00000000	00000000
[КОР2]	00000000	00000000

Таблица 3.5. Пошаговое выполнение операции умножения

№ шага	Промежуточные значения в двоичном коде		Операции
	Старший байт	Младший байт	
	00000000	00001101	начальное значение [$СЧП_{нач}$]
0		1	выделить нулевой бит множителя b_0
	00001111	00000000	[$ЧП_0$]=[A]$\times b_0$
	00001111	00001101	[$СЧП_0$]= [$СЧП_{нач}$]+[$ЧП_0$]
	00000111	10000110	[$\overrightarrow{СЧП_0}$] на 1 бит
	00000000	00000000	[КОР1]
	00000111	**10000110**	занести в старший бит знак множимого
1		0	выделить первый бит множителя b_1
	00000000	00000000	[$ЧП_1$]=[A]$\times b_1$
	00000111	10000110	[$СЧП_1$]= [**$СЧП_0$**]+[$ЧП_1$]
	00000011	11000011	[$\overrightarrow{СЧП_1}$] на 1 бит
	00000000	00000000	[КОР1]
	00000011	**11000011**	Занести в старший бит знак множимого
2		1	выделить второй бит множителя b_2
	00001111	00000000	[$ЧП_2$]=[A]$\times b_2$
	00010010	11000011	[$СЧП_2$]= [**$СЧП_1$**]+[$ЧП_2$]
	00001001	01100001	[$\overrightarrow{СЧП_2}$] на 1 бит
	00000000	00000000	[КОР1]
	00001001	**01100001**	занести в старший бит знак множимого

3		$\boxed{1}$	выделить третий бит множителя b_3
	00001111	00000000	[$ЧП_3$]=[A]×b_3
	00011000	01100001	[$СЧП_3$]= [**$СЧП_2$**]+[$ЧП_3$]
	00001100	00110000	[$\overrightarrow{СЧП_3}$] на 1 бит
	$\boxed{0}$0000000	00000000	[КОР1]
	$\boxed{0}$0001100	**0011000$\boxed{0}$**	занести в старший бит знак множимого
4		$\boxed{0}$	выделить четвертый бит множителя b_4
	00000000	00000000	[$ЧП_4$]=[A]×b_4
	00001100	00110000	[$СЧП_4$]= [**$СЧП_3$**]+[$ЧП_4$]
	00000110	00011000	[$\overrightarrow{СЧП_4}$] на 1 бит
	$\boxed{0}$0000000	00000000	[КОР1]
	$\boxed{0}$0000110	**0001100$\boxed{0}$**	занести в старший бит знак множимого
5		$\boxed{0}$	выделить пятый бит множителя b_5
	00000000	00000000	[$ЧП_5$]=[A]×b_5
	00000110	00011000	[$СЧП_5$]= [**$СЧП_4$**]+[$ЧП_5$]
	00000011	00001100	[$\overrightarrow{СЧП_5}$] на 1 бит
	$\boxed{0}$0000000	00000000	[КОР1]
	$\boxed{0}$0000011	**0000110$\boxed{0}$**	занести в старший бит знак множимого
6		$\boxed{0}$	выделить шестой бит множителя b_6
	00000000	00000000	[$ЧП_6$]=[A]×b_6
	00000011	00001100	[$СЧП_6$]= [**$СЧП_5$**]+[$ЧП_6$]
	00000001	10000110	[$\overrightarrow{СЧП_6}$] на 1 бит
	$\boxed{0}$0000000	00000000	[КОР1]
	$\boxed{0}$0000001	**1000011$\boxed{0}$**	занести в старший бит знак множимого
7		$\boxed{0}$	выделить седьмой бит множителя b_7
	00000000	00000000	[$ЧП_7$]=[A]×b_7
	00000001	10000110	[$СЧП_7$]= [**$СЧП_6$**]+[$ЧП_7$]
	00000000	11000011	[$\overrightarrow{СЧП_7}$] на 1 бит
	$\boxed{0}$0000000	00000000	[КОР1]
	$\boxed{0}$0000000	**11000011**	занести в старший бит знак множимого
8	00000000	00000000	[КОР2]
	00000000	11000011	Сложить [**$СЧП_7$**] с [КОР2]

$C_2 = A_2 \times B_2 = (00000000.11000011)_2 \times 2^8 = 11000011$
$C_{10} = 195$

3.4.2 *A*<0, *B*>0 (результат *C* получится в дополнительном коде).
A = -15, *B* = 16.

Таблица 3.6. Начальные значения переменных

	Старший байт	Младший байт
[*A*]	11110001	00000000
B		00010000
[$СЧП_{нач}$]	00000000	00010000
[КОР1]	10000000	00000000
[КОР2]	00000000	00000000

Таблица 3.7. Пошаговое выполнение операции умножения

№ шага	Промежуточные значения в двоичном коде		Операции
	Старший байт	Младший байт	
	00000000	00010000	начальное значение [$СЧП_{нач}$] (*)
0		0	выделить нулевой бит множителя b_0
	00000000	00000000	[$ЧП_0$]=[*A*]×b_0
	00000000	00010000	[$СЧП_0$]= [$СЧП_{нач}$]+[$ЧП_0$]
	00000000	00001000	[$\overrightarrow{СЧП_0}$] на 1 бит
	00000000	00000000	[КОР1]
	00000000	**00001000**	занести в старший бит знак множимого
1		0	выделить первый бит множителя b_1
	00000000	00000000	[$ЧП_1$]=[*A*]×b_1
	00000000	00001000	[$СЧП_1$]= [**$СЧП_0$**]+[$ЧП_1$]
	00000000	00000100	[$\overrightarrow{СЧП_1}$] на 1 бит
	00000000	00000000	[КОР1]
	00000000	**00000100**	занести в старший бит знак множимого
2		0	выделить второй бит множителя b_2
	00000000	00000000	[$ЧП_2$]=[*A*]×b_2
	00000000	00000100	[$СЧП_2$]= [**$СЧП_1$**]+[$ЧП_2$]
	00000000	00000010	[$\overrightarrow{СЧП_2}$] на 1 бит
	00000000	00000000	[КОР1]
	00000000	**00000010**	занести в старший бит знак множимого

		0	выделить третий бит множителя b_3
	00000000	00000000	$[\text{ЧП}_3]=[A]\times b_3$
3	00000000	00000010	$[\text{СЧП}_3]=$ $[\mathbf{СЧП_2}]$$+[\text{ЧП}_3]$
	00000000	00000001	$[\overrightarrow{\text{СЧП}_3}]$ на 1 бит
	00000000	00000000	[КОР1]
	00000000	**00000001**	занести в старший бит знак множимого
		1	выделить четвертый бит множителя b_4
	11110001	00000000	$[\text{ЧП}_4]=[A]\times b_4$
4	11110001	00000001	$[\text{СЧП}_1]=$ $[\mathbf{СЧП_3}]$$+[\text{ЧП}_4]$
	01111000	10000000	$[\overrightarrow{\text{СЧП}_4}]$ на 1 бит
	10000000	00000000	[КОР1]
	11111000	**10000000**	занести в старший бит знак множимого
		0	выделить пятый бит множителя b_5
	00000000	00000000	$[\text{ЧП}_5]=[A]\times b_5$
5	11111000	10000000	$[\text{СЧП}_5]=$ $[\mathbf{СЧП_4}]$$+[\text{ЧП}_5]$
	01111100	01000000	$[\overrightarrow{\text{СЧП}_5}]$ на 1 бит
	10000000	00000000	[КОР1]
	11111100	**01000000**	занести в старший бит знак множимого
		0	выделить шестой бит множителя b_6
	00000000	00000000	$[\text{ЧП}_6]=[A]\times b_6$
6	11111100	01000000	$[\text{СЧП}_6]=$ $[\mathbf{СЧП_5}]$$+[\text{ЧП}_6]$
	01111110	00100000	$[\overrightarrow{\text{СЧП}_6}]$ на 1 бит
	10000000	00000000	[КОР1]
	11111110	**00100000**	занести в старший бит знак множимого
		0	выделить седьмой бит множителя b_7
	00000000	00000000	$[\text{ЧП}_7]=[A]\times b_7$
7	11111110	00100000	$[\text{СЧП}_7]=$ $[\mathbf{СЧП_6}]$$+[\text{ЧП}_7]$
	01111111	00010000	$[\overrightarrow{\text{СЧП}_7}]$ на 1 бит
	10000000	00000000	[КОР1]
	11111111	**00010000**	занести в старший бит знак множимого
8	00000000	00000000	[КОР2]
	11111111	00010000	Сложить $[\mathbf{СЧП_7}]$ с [КОР2]

$C_2 = A_2 \times B_2 = (11111111.00010000)_2 \times 2^8 = 1111111100010000$
$C_{10} = -240$

3.4.3 *A*>0, *B*<0 (результат *C* получится в дополнительном коде).
A=15, *B* =-13.

Таблица 3.8. Начальные значения переменных

	Старший байт	Младший байт
[*A*]	00001111	00000000
B		11110011
[$СЧП_{нач}$]	00000000	00001101
[КОР1]	00000000	00000000
[КОР2]	11110001	00000000

Таблица 3.9. Пошаговое выполнение операции умножения

№ шага	Промежуточные значения в двоичном коде		Операции
	Старший байт	Младший байт	
	00000000	11110011	начальное значение [$СЧП_{нач}$] (*)
0		1	выделить нулевой бит множителя b_0
	00001111	00000000	[$ЧП_0$]=[*A*]×b_0
	00001111	11110011	[$СЧП_0$]= [$СЧП_{нач}$]+[$ЧП_0$]
	00000111	11111001	$[\overrightarrow{СЧП_0}]$ на 1 бит
	00000000	00000000	[КОР1]
	00000111	**11111001**	занести в старший бит знак множимого
1		1	выделить первый бит множителя b_1
	00001111	00000000	[$ЧП_1$]=[*A*]×b_1
	00010110	11111001	[$СЧП_1$]= [**$СЧП_0$**]+[$ЧП_1$]
	00001011	01111100	$[\overrightarrow{СЧП_1}]$ на 1 бит
	00000000		[КОР1]
	00001011	**01111100**	занести в старший бит знак множимого
2		0	выделить второй бит множителя b_2
	00000000	00000000	[$ЧП_2$]=[*A*]×b_2
	00001011	01111100	[$СЧП_2$]= [**$СЧП_1$**]+[$ЧП_2$]
	00000101	10111110	$[\overrightarrow{СЧП_2}]$ на 1 бит
	00000000	00000000	[КОР1]
	00000101	**10111110**	занести в старший бит знак множимого

3		$\boxed{0}$	выделить третий бит множителя b_3
	00000000	00000000	[ЧП$_3$]=[A]×b_3
	00000101	10111110	[СЧП$_3$]= [**СЧП$_2$**]+[ЧП$_3$]
	00000010	11011111	$[\overrightarrow{\text{СЧП}_3}]$ на 1 бит
	$\boxed{0}$0000000	00000000	[КОР1]
	$\boxed{0}$0000010	**1101111$\boxed{1}$**	занести в старший бит знак множимого
4		$\boxed{1}$	выделить четвертый бит множителя b_4
	00001111	00000000	[ЧП$_4$]=[A]×b_4
	00010001	11011111	[СЧП$_4$]= [**СЧП$_3$**]+[ЧП$_4$]
	00001000	11101111	$[\overrightarrow{\text{СЧП}_4}]$ на 1 бит
	$\boxed{0}$0000000	00000000	[КОР1]
	$\boxed{0}$0001000	**1110111$\boxed{1}$**	занести в старший бит знак множимого
5		$\boxed{1}$	выделить пятый бит множителя b_5
	00001111	00000000	[ЧП$_5$]=[A]×b_5
	00010111	11101111	[СЧП$_5$]= [**СЧП$_4$**]+[ЧП$_5$]
	00001011	11110111	$[\overrightarrow{\text{СЧП}_5}]$ на 1 бит
	$\boxed{0}$0000000	00000000	[КОР1]
	$\boxed{0}$0001011	**1111011$\boxed{1}$**	занести в старший бит знак множимого
6		$\boxed{1}$	выделить шестой бит множителя b_6
	00001111	00000000	[ЧП$_6$]=[A]×b_6
	00011010	11110111	[СЧП$_6$]= [**СЧП$_5$**]+[ЧП$_6$]
	00001101	01111011	$[\overrightarrow{\text{СЧП}_6}]$ на 1 бит
	$\boxed{0}$0000000	00000000	[КОР1]
	$\boxed{0}$0001101	**0111101$\boxed{1}$**	занести в старший бит знак множимого
7		$\boxed{1}$	выделить седьмой бит множителя b_7
	00001111	00000000	[ЧП$_7$]=[A]×b_7
	00011100	01111011	[СЧП$_7$]= [**СЧП$_6$**]+[ЧП$_7$]
	00001110	00111101	$[\overrightarrow{\text{СЧП}_7}]$ на 1 бит
	$\boxed{0}$0000000	00000000	[КОР1]
	$\boxed{0}$0001110	**00111101**	занести в старший бит знак множимого
8	11110001	00000000	[КОР2]
	11111111	00111101	Сложить [**СЧП$_7$**] с [КОР2]

$C_2 = A_2 \times B_2 = (11111111.00111101)_2 \times 2^8 = 1111111100111101$
$C_{10} = -195$

3.4.4 *A*<0, *B*<0 (результат *C* получится в прямом коде).

$A = -15$, $B = -13$.

Таблица 3.10. Начальные значения переменных

	Старший байт	Младший байт
[A]	11110001	00000000
B		11110011
[СЧП$_{\text{нач}}$]	00000000	11110011
[КОР1]	10000000	00000000
[КОР2]	11110001	00000000

Таблица 3.11. Пошаговое выполнение операции умножения

№ шага	Промежуточные значения в двоичном коде		Операции
	Старший байт	Младший байт	
	00000000	1111001$\boxed{1}$	начальное значение [СЧП$_{\text{нач}}$] (*)
0		$\boxed{1}$	выделить нулевой бит множителя b_0
	11110001	00000000	[ЧП$_0$]=[A]×b_0
	11110001	11110011	[СЧП$_0$]= [СЧП$_{\text{нач}}$]+[ЧП$_0$]
	01111000	11111001	[$\overrightarrow{\text{СЧП}_0}$] на 1 бит
	$\boxed{1}$0000000	00000000	[КОР1]
	$\boxed{1}$1111000	**1111100$\boxed{1}$**	занести в старший бит знак множимого
1		$\boxed{1}$	выделить первый бит множителя b_1
	11110001	00000000	[ЧП$_1$]=[A]×b_1
	11101001	11111001	[СЧП$_1$]= [**СЧП$_0$**]+[ЧП$_1$]
	01110100	11111100	[$\overrightarrow{\text{СЧП}_1}$] на 1 бит
	$\boxed{1}$0000000	00000000	[КОР1]
	$\boxed{1}$1110100	**1111110$\boxed{0}$**	занести в старший бит знак множимого
2		$\boxed{0}$	выделить второй бит множителя b_2
	00000000	00000000	[ЧП$_2$]=[A]×b_2
	11110100	11111100	[СЧП$_2$]= [**СЧП$_1$**]+[ЧП$_2$]
	01111010	01111110	[$\overrightarrow{\text{СЧП}_2}$] на 1 бит
	$\boxed{1}$0000000	00000000	[КОР1]
	$\boxed{1}$1111010	**0111111$\boxed{0}$**	занести в старший бит знак множимого

3		$\boxed{0}$	выделить третий бит множителя b_3
	00000000	00000000	[ЧП$_3$]=[A]$\times b_3$
	11111010	01111110	[СЧП$_3$]= [**СЧП$_2$**]+[ЧП$_3$]
	01111101	00111111	$[\overrightarrow{\text{СЧП}_3}]$ на 1 бит
	$\boxed{1}$0000000	00000000	[КОР1]
	$\boxed{1}$1111101	**0011111$\boxed{1}$**	занести в старший бит знак множимого
4		$\boxed{1}$	выделить четвертый бит множителя b_4
	11110001	00000000	[ЧП$_4$]=[A]$\times b_4$
	11101110	00111111	[СЧП$_4$]= [**СЧП$_3$**]+[ЧП$_4$]
	01110111	00011111	$[\overrightarrow{\text{СЧП}_4}]$ на 1 бит
	$\boxed{1}$0000000	00000000	[КОР1]
	$\boxed{1}$1110111	**0001111$\boxed{1}$**	занести в старший бит знак множимого
5		$\boxed{1}$	выделить пятый бит множителя b_5
	11110001	00000000	[ЧП$_5$]=[A]$\times b_5$
	11101000	00011111	[СЧП$_5$]= [**СЧП$_4$**]+[ЧП$_5$]
	01110100	00001111	$[\overrightarrow{\text{СЧП}_5}]$ на 1 бит
	$\boxed{1}$0000000	00000000	[КОР1]
	$\boxed{1}$1110100	**0000111$\boxed{1}$**	занести в старший бит знак множимого
6		$\boxed{1}$	выделить шестой бит множителя b_6
	11110001	00000000	[ЧП$_6$]=[A]$\times b_6$
	11100101	00001111	[СЧП$_6$]= [**СЧП$_5$**]+[ЧП$_6$]
	01110010	10000111	$[\overrightarrow{\text{СЧП}_6}]$ на 1 бит
	$\boxed{1}$0000000	00000000	[КОР1]
	$\boxed{1}$1110010	**1000011$\boxed{1}$**	занести в старший бит знак множимого
7		$\boxed{1}$	выделить седьмой бит множителя b_7
	11110001	00000000	[ЧП$_7$]=[A]$\times b_7$
	11100011	10000111	[СЧП$_7$]= [**СЧП$_6$**]+[ЧП$_7$]
	01110001	11000011	$[\overrightarrow{\text{СЧП}_7}]$ на 1 бит
	$\boxed{1}$0000000	00000000	[КОР1]
	$\boxed{1}$1110001	**11000011**	занести в старший бит знак множимого
8	00001111	00000000	[КОР2]
	00000000	11000011	Сложить [**СЧП$_7$**] с [КОР2]

$C_2 = A_2 \times B_2 = (0.11000011)_2 \times 2^8 = 11000011$
$C_{10} = 195$

3.5 Метод умножения без коррекции результата

3.5.1 Основные положения

Наряду с традиционными методами умножения, требующими коррекции результата, достаточно широкое применение в ЭВМ находит метод Бута, в котором не требуется выполнение коррекции результата.

Метод Бута относится к логическим методам ускорения умножения, позволяющим уменьшить количество сложений в ходе умножения. В основе алгоритма Бута лежит следующий прием, характерный для последовательности двоичных цифр:

Рассмотрим положительный множитель, состоящий из блока единиц, окруженных нулями, например 00111110. Произведение определяется по формуле:

$$M\times 00111110 = M\times(2^5 + 2^4 + 2^3 + 2^2 + 2^1) = M\times 62$$

где M – множимое. Количество операций может быть уменьшено вдвое, если представить произведение следующим образом, заменяя сумму степеней двойки ($2^5 + 2^4 + 2^3 + 2^2 + 2^1$) разностью ($2^6$ - 2^1):

$$M\times 00111110 = M\times 0(1)0000(-1)0 = M\times(2^6 - 2^1) = M\times 62.$$

Действительно, любая последовательность единиц в двоичном числе может быть разбита на разность двух двоичных чисел

$$(\ldots\overbrace{01\ldots\ldots\ldots10}^{n}\ldots)_2 \equiv (\ldots\overbrace{10\ldots\ldots\ldots00}^{n}\ldots)_2 - (\ldots\overbrace{00\ldots\ldots\ldots10}^{n}\ldots)_2$$

Таким образом, мы действительно можем заменить операцию умножения на последовательность единиц в множителе более простыми операциями, такими, как сложение с множителем, сдвиг частных произведений, вычитание множителя. Алгоритм использует тот факт, что нам не нужно делать ничего, кроме сдвига, когда очередной разряд в двоичном множителе равен нулю.

Эта схема может быть распространена на любое количество блоков единиц в множителе (включая случай одной единицы в блоке). Алгоритм Бута следует этой схеме путем выполнения сложения, когда встречается начало блока единиц (01) и вычитания, когда встречается конец блока единиц (10). Схема работает, в том числе, и для отрицательного множителя.

3.5.2 Особенности реализации

Сдвиг частных произведений реализуется как арифметический сдвиг (при сдвиге СЧП вправо значение старшего разряда сохраняется). **Операции**, выполняемые на каждом шаге умножения, зависят от комбинации значений текущего и предшествующего ему разрядов множителя (b_i, b_{i-1}) (см. табл. 3.12):

Таблица 3.12

Комбинация значений разрядов множителя (b_i, b_{i-1})	**Выполняемые операции**
$b_i = 0$, $b_{i-1} = 1$	СЧП = СЧП + A, $\overrightarrow{\text{СЧП}}$
$b_i = 1$, $b_{i-1} = 0$	СЧП = СЧП – A, $\overrightarrow{\text{СЧП}}$
$b_i = 0$, $b_{i-1} = 0$ или $b_i = 1$, $b_{i-1} = 1$	$\overrightarrow{\text{СЧП}}$

Для $i = 0$ считается, что $b_{i-1} = 0$.

Все операции вычитания множимого заменяются на операции сложения с его дополнением.

3.6. Примеры, иллюстрирующие работу этого метода.

3.6.1 *A*>0, *B*>0 (результат *C* получится в прямом коде).

Пусть $A = 15$, $B = 13$.

Таблица 3.13

№ шага *i*	**Промежуточные значения в двоичном коде**		**b_i**	**b_{i-1}**	**Операции**
	00001111	0000000	0		множимое A
		0000110	1		множитель B
	00000000	0000110	1		СЧП$_{нач}$ (*)
0			1	0	$b_0 = 1$, $b_{-1} = 0$
	11110001				ЧП$_0$=A
	11110001	0000110	1		СЧП$_0$ = СЧП$_{нач}$ - ЧП$_0$
	11111000	1000011	0	1	$\overrightarrow{\text{СЧП}_0}$
1			0	1	$b_1 = 0$, $b_0 = 1$
	00001111				ЧП$_1$=A
	00000111	1000011	0	1	СЧП$_1$ = $\overrightarrow{\text{СЧП}_0}$ + ЧП$_1$
	00000011	1100001	1	0	$\overrightarrow{\text{СЧП}_1}$
2			1	0	$b_2 = 1$, $b_1 = 0$
	11110001				ЧП$_2$=A
	11110100	1100001	1		СЧП$_2$ = $\overrightarrow{\text{СЧП}_1}$ – ЧП$_1$
	11111010	0110000	1	1	$\overrightarrow{\text{СЧП}_2}$
3			1	1	$b_3 = 1$, $b_2 = 1$
	11111010	0110000	1		СЧП$_3$=$\overrightarrow{\text{СЧП}_2}$
	11111101	0011000	0	1	$\overrightarrow{\text{СЧП}_3}$
4			0	0	$b_4 = 0$, $b_3 = 1$
	00001111				ЧП$_4$=A

	00001100 00000110	0011000 0001100	0 0	 0	$СЧП_4 = \overrightarrow{СЧП_3} + A$ $\overrightarrow{СЧП_4}$
5	 00000110 00000011	 0001100 0000110	0 0 0	0 0	$b_5 = 0, b_4 = 0$ $СЧП_5=\overrightarrow{СЧП_4}$ $\overrightarrow{СЧП_5}$
6	 00000011 00000001	 0000110 1000011	0 0 0	0 0	$b_6 = 0, b_5 = 0$ $СЧП_6=\overrightarrow{СЧП_5}$ $\overrightarrow{СЧП_6}$
7	 00000001 00000000	 1000011 1100001	0 0 1	0	$b_7 = 0, b_6 = 0$ $СЧП_6=\overrightarrow{СЧП_5}$ $\overrightarrow{СЧП_7}$

$\boldsymbol{C_2 = A_2 \times B_2 = (0.11000011)_2 \times 2^8 = 11000011}$

$\boldsymbol{C_{10} = 195}$

3.6.2 *A*<0, *B*>0 (результат *C* получится в дополнительном коде).
$A = -15$, $B = 13$.

Таблица 3.14

№ шага i	Промежуточные значения в двоичном коде		b_i	b_{i-1}	Операции
	11110001	0000000	0		множимое A
		0000110	1		множитель B
	00000000	0000110	1		$СЧП_{нач}$ (*)
0	 00001111 00001111 00000111	 0000110 1000011	1 1 0	0 1	$b_0 = 1, b_{-1} = 0$ $ЧП_0=A$ $СЧП_0 = СЧП_{нач} - ЧП_0$ $\overrightarrow{СЧП_0}$
1	 11110001 11111000 11111100	 1000011 0100001	0 0 1	1 1 0	$b_1 = 0, b_0 = 1$ $ЧП_1=A$ $СЧП_1 = \overrightarrow{СЧП_0} + ЧП_1$ $\overrightarrow{СЧП_1}$
2	 00001111 00001011 00000101	 0100001 1010000	1 1 1	0 1	$b_2 = 1, b_1 = 0$ $ЧП_2=A$ $СЧП_2 = \overrightarrow{СЧП_1} – ЧП_1$ $\overrightarrow{СЧП_2}$

3	 00000101 00000010	 1010000 1101000	1 1 0	1 1	$b_3 = 1, b_2 = 1$ $СЧП_3=\overrightarrow{СЧП_2}$ $\overrightarrow{СЧП_3}$
4	 11110001 11110011 11111001	 1101000 1110100	0 0 0	0 0	$b_4 = 0, b_3 = 1$ $ЧП_4=A$ $СЧП_4 = \overrightarrow{СЧП_3} + A$ $\overrightarrow{СЧП_4}$
5	 11111001 11111100	 1110100 1111010	0 0 0	0 0	$b_5 = 0, b_4 = 0$ $СЧП_5=\overrightarrow{СЧП_4}$ $\overrightarrow{СЧП_5}$
6	 11111100 11111110	 1111010 0111101	0 0 0	0 0	$b_6 = 0, b_5 = 0$ $СЧП_6=\overrightarrow{СЧП_5}$ $\overrightarrow{СЧП_6}$
7	 11111110 11111111	 0111101 0011110	0 0 1	0 	$b_7 = 0, b_6 = 0$ $СЧП_6=\overrightarrow{СЧП_5}$ $\overrightarrow{СЧП_7}$

$C_2 = A_2 \times B_2 = (11111111.00111101)_2 \times 2^8 = 1111111100111101$
$C_{10} = -195$

3.6.3 $A>0$, $B<0$ (результат C получится в дополнительном коде).
$A = 15$, $B = -13$.

Таблица 3.15

№ шага *i*	Промежуточные значения в двоичном коде		b_i	b_{i-1}	Операции
	00001111	0000000	0		множимое A
		1111001	1		множитель B
	00000000	1111001	1		$СЧП_{нач}$ (*)
0	 11110001 11110001 11111000	 1111001 1111100	1 1 1	0 1	$b_0 = 1, b_{-1} = 0$ $ЧП_0=A$ $СЧП_0 = СЧП_{нач} - ЧП_0$ $\overrightarrow{СЧП_0}$
1	 11111000 11111100	 1111100 0111110	1 1 0	1 1	$b_1 = 1, b_0 = 1$ $СЧП_1 = \overrightarrow{СЧП_0}$ $\overrightarrow{СЧП_1}$

2	 00001111 00001011 00000101	 0111110 1011111	0 0 0	1 1	$b_2 = 0$, $b_1 = 1$ ЧП$_2$=A СЧП$_2$ = $\overrightarrow{\text{СЧП}_1}$ + ЧП$_1$ $\overrightarrow{\text{СЧП}_2}$
3	 00000101 00000010	 1011111 1101111	0 0 1	0 0	$b_3 = 0$, $b_2 = 0$ СЧП$_3$=$\overrightarrow{\text{СЧП}_2}$ $\overrightarrow{\text{СЧП}_3}$
4	 11110001 11110011 11111001	 1101111 1110111	1 1 1	0 1	$b_4 = 1$, $b_3 = 0$ ЧП$_4$=A СЧП$_4$ = $\overrightarrow{\text{СЧП}_3}$ - ЧП$_4$ $\overrightarrow{\text{СЧП}_4}$
5	 11111001 11111100	 1110111 1111011	1 1 1	1 1	$b_5 = 1$, $b_4 = 1$ СЧП$_5$=$\overrightarrow{\text{СЧП}_4}$ $\overrightarrow{\text{СЧП}_5}$
6	 11111100 11111110	 1111011 0111101	1 1 1	1 1	$b_6 = 1$, $b_5 = 1$ СЧП$_6$=$\overrightarrow{\text{СЧП}_5}$ $\overrightarrow{\text{СЧП}_6}$
7	 11111110 11111111	 0111101 0011110	1 1 1	1	$b_7 = 1$, $b_6 = 1$ СЧП$_6$=$\overrightarrow{\text{СЧП}_5}$ $\overrightarrow{\text{СЧП}_7}$

$C_2 = A_2 \times B_2 = (11111111.00111101)_2 \times 2^8 = 1111111100111101$
$C_{10} = -195$

3.6.4 A<0, B<0 (результат C получится в прямом коде).
$A = -15$, $B = -13$.

Таблица 3.16

№ шага *i*	**Промежуточные значения в двоичном коде**		**b_i**	**b_{i-1}**	**Операции**
	11110001	0000000	0		множимое A
		1111001	1		множитель B
	00000000	1111001	1		СЧП$_{нач}$ (*)
0	 00001111 00001111 00000111	 1111001 1111100	1 1 1	0 1	$b_0 = 1$, $b_{-1} = 0$ ЧП$_0$=A СЧП$_0$ = СЧП$_{нач}$ - ЧП$_0$ $\overrightarrow{\text{СЧП}_0}$

1	 00000111 00000011	 1111100 1111110	1 1 0	1 1	$b_1 = 1$, $b_0 = 1$ $СЧП_1 = \overrightarrow{СЧП_0}$ $\overrightarrow{СЧП_1}$
2	 11110001 11110100 11111010	 1111110 0111111	0 0 0	1 1	$b_2 = 0$, $b_1 = 1$ $ЧП_2=A$ $СЧП_2 = \overrightarrow{СЧП_1} + ЧП_1$ $\overrightarrow{СЧП_2}$
3	 11111010 11111101	 0111111 0011111	0 0 1	0 0	$b_3 = 0$, $b_2 = 0$ $СЧП_3=\overrightarrow{СЧП_2}$ $\overrightarrow{СЧП_3}$
4	 00001111 00001100 00000110	 0011111 0001111	1 1 1	0 1	$b_4 = 1$, $b_3 = 0$ $ЧП_4=A$ $СЧП_4 = \overrightarrow{СЧП_3} - ЧП_4$ $\overrightarrow{СЧП_4}$
5	 00000110 00000011	 0001111 0000111	1 1 1	1 1	$b_5 = 1$, $b_4 = 1$ $СЧП_5=\overrightarrow{СЧП_4}$ $\overrightarrow{СЧП_5}$
6	 00000011 00000001	 0000111 1000011	1 1 1	1 1	$b_6 = 1$, $b_5 = 1$ $СЧП_6=\overrightarrow{СЧП_5}$ $\overrightarrow{СЧП_6}$
7	 00000001 00000000	 1000011 1100001	1 1 1	1	$b_7 = 1$, $b_6 = 1$ $СЧП_6=\overrightarrow{СЧП_5}$ $\overrightarrow{СЧП_7}$

$\boldsymbol{C_2 = A_2 \times B_2 = (0.11000011)_2 \times 2^8 = 11000011}$
$\boldsymbol{C_{10} = 195}$

4 ДЕЛЕНИЕ ЦЕЛЫХ ЧИСЕЛ

4.1 Задание

1). *Выполнить операцию деления заданных целых чисел A и B со всеми комбинациями знаков, используя метод деления в дополнительных кодах. Для представления делимого (A) использовать 16 двоичных разрядов (один – знаковый и 15 – цифровых), для представления делителя (B) – 8 разрядов (один – знаковый и 7 – цифровых). Остаток от деления и частное представляются в той же разрядной сетке, что и делитель.*
2). *Результаты операции представить в десятичной системе счисления и проверить их правильность.*

Варианты заданий приведены в табл. 3 Приложения.

4.2 Основные положения

4.2.1 Термины и обозначения

R_i – остаток от деления на i-м шаге

$\overleftarrow{R}_i$ – сдвиг остатка от деления на i-м шаге на один разряд влево

РЕЗ_ОСТ – итоговый остаток от деления

Дополнение – представление числа в дополнительном коде, если исходное число представлено в прямом коде, и в прямом, если исходное число записано в дополнительном коде.

Простой сдвиг – сдвиг числа на 1 разряд влево. Младший разряд при этом имеет значение 0.

4.2.2 Принцип деления

Деление двоичных чисел похоже на деление десятичных чисел. Процесс деления заключается в последовательном поразрядном поиске цифры частного методом подбора. После ее нахождения цифра умножается на делитель, и получившееся произведение вычитается из делимого.

Существует множество разных методов деления. В универсальных вычислительных машинах обычно реализуется одна из разновидностей известного по школе алгоритма деления в столбик.

Этот алгоритм деления состоит в том, что, начиная со старших разрядов, на каждом шаге делитель вычитается из делимого столько раз, сколько это возможно для получения наименьшего положительного остатка. Цифра, равная числу делителей в делимом, на каждом шаге записывается в соответствующий разряд частного. Таким образом, процесс деления сводится к операциям вычитания и сдвига.

На примере двоичных чисел рассматриваемый алгоритм выглядит следующим образом:

```
Делимое        1100100 |1010           Делитель
              −        |---------------
               1010    |1010           Частное
Остаток        0101
              −
                1010
                1011
Восстановление +
остатка         1010
                01010
              −
                 1010
                00000
```

Цифры частного получаются последовательно, начиная со старшего разряда. На каждом шаге из полученного на предыдущем шаге остатка вычитается делитель. Цифра, которая заносится в текущий разряд, зависит от знака полученного на этом шаге остатка. В том случае, когда образуется положительный остаток, значение текущего разряда частного равно единице. Если получился отрицательный остаток, в текущий разряд частного записывается нуль, при этом следует восстановить текущий положительный остаток. Если остаток положителен, то для получения следующей цифры частного последний остаток сдвигается на 1 разряд влево и из него вычитается делитель и т.д. Если остаток отрицательный, то путем прибавления к нему делителя восстанавливается предыдущий положительный остаток. Восстановленный остаток необходимо сдвинуть влево на 1 разряд и вычесть из него делитель.

В зависимости от значения разряда результата C, полученного на предыдущем шаге, происходит либо сложение полученного на текущем шаге остатка с делителем (если значение разряда C, полученного на предыдущем шаге, равно нулю), либо вычитание делителя из текущего остатка (если значение разряда C, полученного на предыдущем шаге, равно единице).

Расписанный выше алгоритм деления получил название алгоритма деления с восстановлением остатка.

Пусть A – делимое, B – делитель, C – частное, D – остаток от деления.

На 1 шаге: $A_1 = A - B \times 2^3$. Пусть $A_1 > 0$, тогда $C_1 = 1$. Процесс деления продолжается дальше: $A_2 = A_1 - B \times 2^2$. Пусть $A_2 < 0$, тогда $C_2 = 0$ и производится восстановление остатка A_1: $A_2 = A_2 + B \times 2^2$.

Этот остаток принимается за A_2' и деление продолжается дальше следующим образом: $A_3 = A_2' - B \times 2^1$. Т.е. алгоритм деления можно описать в общем виде на i-м шаге:

$$A_i = A_{i-1} - B \times 2^{4-i}$$

Если $A >= 0$, то $C_i = 1$ и переход к следующему шагу. Если $A < 0$, то $C_i = 0$ и восстановление остатка: $A_{i-1} = A_i + B \times 2^{4-i}$.

К недостаткам этого метода можно отнести необходимость коррекции остатка на каждом шаге деления, где вычисляемый разряд частного равен нулю. Этого можно избежать, если математическую формулу этого метода несколько модифицировать:

$$D = A - B \times 2^3 - B \times 2^2 + B \times 2^2 - B \times 2^1 - B \times 2^0 = A - B \times 2^3 - B \times 2^2 + B \times (2^2 - 2^1) - B \times 2^0 = A - B\times2^3 - B \times 2^2 + B\times2^1 - B \times 2^0$$

Такой вариант называется делением без коррекции остатка.

4.2.3 Особенности используемого метода деления

Используемый в данном задании метод деления базируется на представлении положительных операндов в прямом, а отрицательных – в дополнительном кодах.

Достоинства этого метода:

а). Не требуется преобразовывать операнды и результат из дополнительного кода в прямой код и обратно.

$$A>0,\ B>0 \quad C_{\text{пр}} = A_{\text{пр}} : B_{\text{пр}}$$

$$A<0,\ B>0 \quad C_{\text{доп}} = A_{\text{доп}} : B_{\text{пр}}$$

$$A>0,\ B<0 \quad C_{\text{доп}} = A_{\text{пр}} : B_{\text{доп}}$$

$$A<0,\ B<0 \quad C_{\text{пр}} = A_{\text{доп}} : B_{\text{доп}}$$

б). Положительный результат операции представляется в прямом коде, а отрицательный – в дополнительном.

4.2.4 Замечания по реализации метода

4.2.4.1 Использование беззнаковых переменных

Особенностью использованного метода является то, что знаковые разряды используются в операции деления наряду с цифровыми, т.е. фактически в делении участвуют беззнаковые положительные операнды.

Например:

1). Один из операндов равен $\boxed{+15}$. Его двоичное представление в 8-битовом коде:

0	0	0	0	1	1	1	1

Участвующий в делении операнд в десятичном беззнаковом представлении $\boxed{15}$.

2). Один из операндов равен $\boxed{-15}$. Его двоичное представление в 8-битовом коде:

1	1	1	1	0	0	0	1

Участвующий в делении операнд в десятичном беззнаковом представлении $\boxed{241}$.

Таким образом, при программной реализации данного метода все участвующие в делении операнды должны быть беззнаковыми и сформированными так, чтобы их битовый состав в точности повторял битовый состав соответствующих знаковых переменных.

4.2.4.2 Формирование частного

На каждом шаге операции деления формируется один разряд частного, начиная с самого старшего (т.е. знакового) разряда и заканчивая самым младшим. Если знаки делителя и полученного на данном шаге остатка совпадают, то формируемый разряд частного равен единице, если не совпадают – нулю. Полученный на первом шаге разряд частного заносится в нулевой разряд остатка. На следующих шагах остаток сдвигается влево на единицу, и на место освободившегося в результате сдвига разряда заносится очередной разряд частного.

4.2.4.3 Проверка корректности деления и определение знакового разряда частного

Возможен случай, когда после деления получается частное, превышающее по размеру отведенные для него 8 разрядов. Такой результат считается некорректным. Это требует заранее проверять корректность результата деления.

Условие корректности деления можно записать в виде неравенств:

а) $|A|/|B| < 2^n$ – если знаки делимого A и делителя B совпадают;

б) $|A|/|B| < 2^n + 1$ – если знаки делимого A и делителя B различны.

Здесь n – число цифровых разрядов делителя (частного). Значение справа – наибольшее по модулю целое число, состоящее из n цифровых разрядов (или из $n + 1$, если учитывать знаковый разряд частного). Считаем, что положительное частное записывается в прямом коде (а), а отрицательное – в дополнительном (б).

Таким образом, для проверки корректности деления нужно вычислить разности:

а) $|A| - |B| \times 2^n$,

б) $|A| - |B| - |B| \times 2^n$.

При отрицательном значении разности операция деления корректна, при положительном – нет.

Очевидно, что проверка корректности деления для случая, когда знаки делимого и делителя совпадают, и когда они различны, несколько отличаются. Если модифицировать условие для проверки корректности деления при совпадении знаков делимого и делителя, то оба условия можно привести к общему виду:

$$|A| - |D| - |B| \times 2^n < 0,$$

где $|D| = 0$ при совпадении знаков делимого и делителя, $|D| = |B|$ при различных знаках делимого и делителя.

Кроме того, поскольку проверка корректности деления требует пробного вычитания величины $|B| \times 2^n$, она может выполняться одновременно с первым шагом деления (формированием знакового разряда частного).

Опишем этот алгоритм:

- Выполнение операции $|A| - |D|$. При разных знаках операндов она заменяется операцией сложения делимого и делителя, выровненных по младшим разрядам. При одинаковых знаках операндов заменяется операцией сложения делителя с нулем.
- Сдвиг полученного результата на 1 разряд влево.
- Сложение результата с делителем, выровненным по старшим разрядам, при разных знаках операндов или вычитание делителя (сложением с его дополнением) при одинаковых знаках операндов.

- Сравнение знаков делимого и остатка, полученного на предыдущем шаге. Если знаки совпадают, то процесс деления завершается из-за некорректности результата.
- Определение значения знакового разряда частного по знакам делителя и остатка. Формируемый разряд частного равен единице при совпадении этих знаков и нулю, если знаки разные.
- Занесение знакового разряда частного на место освободившегося при сдвиге влево младшего разряда остатка.

Особенности проверки корректности деления:

а) Нулевой остаток считается положительным, т.к. имеет ноль в знаковом разряде.

б) При сложении делимого и делителя, выровненных по младшим разрядам, старшие разряды делителя дополняются его знаковым разрядом.

4.2.4.4 Формирование цифровых разрядов частного

Формирование цифрового разряда частного производится по определенному алгоритму:

- Остаток, полученный на предыдущем шаге, сдвигается на один разряд влево (освободившийся младший разряд заполняется нулем).
- Формирование нового остатка путем сложения старших разрядов остатка, полученного после предыдущего действия, с делителем или вычитания делителя из старших разрядов остатка (операция вычитания делителя заменяется сложением с его дополнением). Выполняемая далее (до его сдвига на 1 разряд влево) арифметическая операция определяется знаками остатка, полученного на предыдущем шаге, и делителя. При совпадении знаков – вычитание делителя из остатка, при разных знаках – сложение делителя с остатком. Нулевой остаток содержит в знаковом разряде нуль и поэтому рассматривается как положительный.
- Значение цифрового разряда частного определяется знаками делителя и остатка, полученного на данном шаге. Формируемый разряд частного равен единице при совпадении этих знаков и нулю, если знаки разные.
- Цифровой разряд частного заносится на место освободившегося при сдвиге влево младшего разряда остатка.

4.2.4.5 Коррекция остатка

Существуют два вида коррекции остатка:

а). Коррекция остатка после формирования всех разрядов частного

Эту коррекцию следует выполнять в случае, когда знак итогового остатка (т.е. остатка, получившегося после формирования частного) не совпадает со знаком делимого. Если знаки делителя и остатка совпадают, делитель вычитается из остатка; если не совпадают – делитель прибавляется к остатку. Иначе говоря, при коррекции остатка выполняется такое же действие, как при формировании частного (т.е. в основном цикле деления).

б). Промежуточная коррекция остатка

Такую коррекцию нужно выполнять тогда, когда на текущем шаге деления получается нулевой остаток.

Особый случай коррекции остатка имеет место при выполнении трех условий:

1) наличие нулевого промежуточного остатка на каком-либо шаге деления,
2) отрицательное делимое,
3) отрицательный окончательный остаток.

При выполнении этих условий делается попытка скорректировать полученный остаток. Для этого сравниваются модули окончательного остатка и делителя. При их совпадении производится коррекция остатка по общему правилу. Если в результате коррекции остаток получается нулевым, он является правильным. При получении ненулевого остатка он восстанавливается путем сложения с делителем (при положительном делителе) или с его дополнением (при отрицательном делителе).

4.2.4.6 Коррекция частного

Эта коррекция выполняется, если получен нулевой остаток при отрицательном делимом. Выполняя коррекцию частного, следует:

а). Если делитель положительный, вычесть единицу из отрицательного частного.

б). Если делитель отрицательный, прибавить к положительному частному.

4.2.4.7 Результат деления

В качестве результата при делении формируются два значения: частное и остаток. Частное составляют 8 младших разрядов результата коррекции частного, а остаток берется из 8 старших разрядов предыдущего шага - коррекции остатка.

4.3 Особенности реализации алгоритма деления

Нужно сформировать следующие 16-разрядные беззнаковые переменные (далее такие переменные будут заключаться в квадратные скобки) (см. табл. 4.1):

Таблица 4.1

Обозначение	**Роль переменной**
[A]	делимое
[B]	вспомогательная переменная для проверки корректности деления
[B1]	$B \times 2^n$
[B2]	(дополнение B) $\times 2^n$
`[KOP_R_1]`, `[KOP_R_2]`	вспомогательные переменные для коррекции частного

[*A*]	В эту переменную заносим значения 16-ми разрядов делимого ***A*** (см. п. 4.2.4.1).
[*B*]	Заполняется нулями при совпадении знаков делимого и делителя; при различных знаках делимого и делителя в младший байт заносится значение делителя, а старший байт заполняется знаковым разрядом делителя.
[*B*1]	В старший байт заносится значение 8-ми разрядов делителя, а в младший - нули.
[*B*2]	В старший байт заносится дополнение делителя, а в младший байт - нули.
[КОР_*R*_1]	Заполняется нулями.
[КОР_*R*_2]	При положительном делимом вся переменная заполняется нулями. При отрицательном в младший байт заносится 1 при отрицательном делителе или (-1) при положительном делителе. Старший байт заполняется нулями.

4.4. Примеры, иллюстрирующие работу этого метода

4.4.1 Пример деления: *A* > 0, *B* > 0 (*A* = 139, *B* = 13)

Таблица 4.2

	Старший байт	Младший байт
[*A*]	00000000	10001011
[*B*]	00000000	00000000
[*B*1]	00001101	00000000
[*B*2]	11110011	00000000
[КОР_*R*_1]	00000000	00000000
[КОР_*R*_2]	00000000	00000000

Таблица 4.3

		Знак делимого: 0 Знак делителя: 0		Знак остатка	Знак частного
1 Проверка корректности деления и определение знакового разряда	[*A*]	0 0 0 0 0 0 0 0	1 0 0 0 1 0 1 1		
	[*B*]	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0		
	[$R_{нач}$]	0 0 0 0 0 0 0 0	1 0 0 0 1 0 1 1	0	
	[$\overleftarrow{R}_{нач}$]	0 0 0 0 0 0 0 1	0 0 0 1 0 1 1 0		
	[*B*2]	1 1 1 1 0 0 1 1	0 0 0 0 0 0 0 0	0	
	сумма	1 1 1 1 0 1 0 0	0 0 0 1 0 1 1 0		
	[R_1]	1 1 1 1 0 1 0 0	0 0 0 1 0 1 1 0	1	0

2	[$\overleftarrow{R}_1$]	1 1 1 0 1 0 0 0	0 0 1 0 1 1 0 0		
	[$B1$]	0 0 0 0 1 1 0 1	0 0 0 0 0 0 0 0	1	
	сумма	1 1 1 1 0 1 0 1	0 0 1 0 1 1 0 0		
	[R_2]	1 1 1 1 0 1 0 1	0 0 1 0 1 1 0 0	1	0
3	[$\overleftarrow{R}_2$]	1 1 1 0 1 0 1 0	0 1 0 1 1 0 0 0		
	[$B1$]	0 0 0 0 1 1 0 1	0 0 0 0 0 0 0 0	1	
	сумма	1 1 1 1 0 1 1 1	0 1 0 1 1 0 0 0		
	[R_3]	1 1 1 1 0 1 1 1	0 1 0 1 1 0 0 0	1	0
4	[$\overleftarrow{R}_3$]	1 1 1 0 1 1 1 0	1 0 1 1 0 0 0 0		
	[$B1$]	0 0 0 0 1 1 0 1	0 0 0 0 0 0 0 0	1	
	сумма	1 1 1 1 1 0 1 1	1 0 1 1 0 0 0 0		
	[R_4]	1 1 1 1 1 0 1 1	1 0 1 1 0 0 0 0	1	0
5	[$\overleftarrow{R}_4$]	1 1 1 1 0 1 1 1	0 1 1 0 0 0 0 0		
	[B1]	0 0 0 0 1 1 0 1	0 0 0 0 0 0 0 0	1	
	сумма	0 0 0 0 0 1 0 0	0 1 1 0 0 0 0 0		
	[R_5]	0 0 0 0 0 1 0 0	0 1 1 0 0 0 0 1	0	1
6	[$\overleftarrow{R}_5$]	0 0 0 0 1 0 0 0	1 1 0 0 0 0 1 0		
	[B2]	1 1 1 1 0 0 1 1	0 0 0 0 0 0 0 0	0	
	сумма	1 1 1 1 1 0 1 1	1 1 0 0 0 0 1 0		
	[R_6]	1 1 1 1 1 0 1 1	1 1 0 0 0 0 1 0	0	0
7	[$\overleftarrow{R}_6$]	1 1 1 1 0 1 1 1	1 0 0 0 0 1 0 0		
	[B1]	0 0 0 0 1 1 0 1	0 0 0 0 0 0 0 0	1	
	сумма	0 0 0 0 0 1 0 0	1 0 0 0 0 1 0 0		
	[R_7]	0 0 0 0 0 1 0 0	1 0 0 0 0 1 0 1	0	1
8	[$\overleftarrow{R}_7$]	0 0 0 0 1 0 0 1	0 0 0 0 1 0 1 0		
	[B2]	1 1 1 1 0 0 1 1	0 0 0 0 0 0 0 0	0	
	сумма	1 1 1 1 1 1 0 0	0 0 0 0 1 0 1 0		
	[R_8]	1 1 1 1 1 1 0 0	0 0 0 0 1 0 1 0	1	0
Коррек-ция остатка	[B1]	0 0 0 0 1 1 0 1	0 0 0 0 0 0 0 0		
	сумма	0 0 0 0 1 0 0 1	0 0 0 0 1 0 1 0		
Коррек-ция частного	[КОР_R_1]	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0		
	Сумма	0 0 0 0 1 0 0 1	0 0 0 0 1 0 1 0		

$C_2 = A_2/B_2 = 00001010_2$, $C_{10} = 10_{10}$; $РЕЗ\_ОСТ_2 = 00001001_2$, $РЕЗ\_ОСТ_{10} = 9_{10}$

4.4.2 Пример деления 1: A < 0, B < 0 (A = -139, B = -13)

Таблица 4.4

	Старший байт	Младший байт
[A]	11111111	01110101
[B]	00000000	00000000
[B1]	11110011	00000000
[B2]	00001101	00000000
[КОР_R_1]	00000000	00000000
[КОР_R_2]	00000000	00000001

Таблица 4.5

		Знак делимого: 1 Знак делителя: 1		Знак остатка	Знак частного
1 Проверка корректности деления и определение знакового разряда частного	[A]	1 1 1 1 1 1 1 1	0 1 1 1 0 1 0 1		
	[B]	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0		
	[$R_{нач}$]	1 1 1 1 1 1 1 1	0 1 1 1 0 1 0 1	1	
	[$\overleftarrow{R}_{нач}$]	1 1 1 1 1 1 1 0	1 1 1 0 1 0 1 0		
	[B2]	0 0 0 0 1 1 0 1	0 0 0 0 0 0 0 0		
	сумма	0 0 0 0 1 0 1 1	1 1 1 0 1 0 1 0		
	R_1	0 0 0 0 1 0 1 1	1 1 1 0 1 0 1 0	0	0
2	$\overleftarrow{R}_1$	0 0 0 1 0 1 1 1	1 1 0 1 0 1 0 0		
	[B1]	1 1 1 1 0 0 1 1	0 0 0 0 0 0 0 0	0	
	сумма	0 0 0 0 1 0 1 0	1 1 0 1 0 1 0 0		
	R_2	0 0 0 0 1 0 1 0	1 1 0 1 0 1 0 0	0	0
3	$\overleftarrow{R}_2$	0 0 0 1 0 1 0 1	1 0 1 0 1 0 0 0		
	[B1]	1 1 1 1 0 0 1 1	0 0 0 0 0 0 0 0	0	
	сумма	0 0 0 0 1 0 0 0	1 0 1 0 1 0 0 0		
	R_3	0 0 0 0 1 0 0 0	1 0 1 0 1 0 0 0	0	0
4	$\overleftarrow{R}_3$	0 0 0 1 0 0 0 1	0 1 0 1 0 0 0 0		
	[B1]	1 1 1 1 0 0 1 1	0 0 0 0 0 0 0 0	0	
	сумма	0 0 0 0 0 1 0 0	0 1 0 1 0 0 0 0		
	R_4	0 0 0 0 0 1 0 0	0 1 0 1 0 0 0 0	0	0
5	$\overleftarrow{R}_4$	0 0 0 0 1 0 0 0	1 0 1 0 0 0 0 0		
	[B1]	1 1 1 1 0 0 1 1	0 0 0 0 0 0 0 0	0	
	сумма	1 1 1 1 1 0 1 1	1 0 1 0 0 0 0 0		

	R_5	1 1 1 1 1 0 1 1	1 0 1 0 0 0 0 1	1	1
6	$\overleftarrow{R}_5$	1 1 1 1 0 1 1 1	0 1 0 0 0 0 1 0		
	[B2]	0 0 0 0 1 1 0 1	0 0 0 0 0 0 0 0	1	
	сумма	0 0 0 0 0 1 0 0	0 1 0 0 0 0 1 0		
	R_6	0 0 0 0 0 1 0 0	0 1 0 0 0 0 1 0	0	0
7	$\overleftarrow{R}_6$	0 0 0 0 1 0 0 0	1 0 0 0 0 1 0 0		
	[B1]	1 1 1 1 0 0 1 1	0 0 0 0 0 0 0 0	0	
	сумма	1 1 1 1 1 0 1 1	1 0 0 0 0 1 0 0		
	R_7	1 1 1 1 1 0 1 1	1 0 0 0 0 1 0 1	1	1
8	$\overleftarrow{R}_7$	1 1 1 1 0 1 1 1	0 0 0 0 1 0 1 0		
	R_8	0 0 0 0 0 1 0 0	0 0 0 0 1 0 1 0	1	0
Коррек-ция остатка	[B2]	1 1 1 1 0 0 1 1	0 0 0 0 0 0 0 0	1	
	сумма	1 1 1 1 0 1 1 1	0 0 0 0 1 0 1 0		
Коррек-ция частного	[КОР_R_1]	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0		
	сумма	1 1 1 1 0 1 1 1	0 0 0 0 1 0 1 0		

$C_2 = A_2/B_2 = 00001010_2$, $C_{10} = 10_{10}$; $РЕЗ\_ОСТ_2 = 11110111_2$, $РЕЗ\_ОСТ_{10} = -9_{10}$

4.4.3 Пример деления 2: A < 0, B > 0 (A = -139, B = 13)

Таблица 4.6

	Старший байт	Младший байт
[A]	11111111	01110101
[B]	00000000	00001101
[B1]	00001101	00000000
[B2]	11110011	00000000
[КОР_R_1]	00000000	00000000
[КОР_R_2]	00000000	11111111

Таблица 4.7

		Знак делимого: 1 нак делителя: 0		Знак остатка	Знак част-ного
1 Провер-ка кор-ректности деления и определе-ние зна-	[A]	1 1 1 1 1 1 1 1	0 1 1 1 0 1 0 1		
	[B]	0 0 0 0 0 0 0 0	0 0 0 0 1 1 0 1		
	[$R_{нач}$]	1 1 1 1 1 1 1 1	1 0 0 0 0 0 1 0	1	
	[$\overleftarrow{R}_{нач}$]	1 1 1 1 1 1 1 1	0 0 0 0 0 1 0 0		
	[B1]	0 0 0 0 1 1 0 1	0 0 0 0 0 0 0 0		

кового	сумма	0 0 0 0 1 1 0 0	0 0 0 0 0 1 0 0		
разряда частного	R_1	0 0 0 0 1 1 0 0	0 0 0 0 0 1 0 1	0	1
2	$\overleftarrow{R}_1$	0 0 0 1 1 0 0 0	0 0 0 0 1 0 1 0		
	[B2]	1 1 1 1 0 0 1 1	0 0 0 0 0 0 0 0	0	
	сумма	0 0 0 0 1 0 1 1	0 0 0 0 1 0 1 0		
	R_2	0 0 0 0 1 0 1 1	0 0 0 0 1 0 1 1	0	1
3	$\overleftarrow{R}_2$	0 0 0 1 0 1 1 0	0 0 0 1 0 1 1 0		
	[B2]	1 1 1 1 0 0 1 1	0 0 0 0 0 0 0 0	0	
	сумма	0 0 0 0 1 0 0 1	0 0 0 1 0 1 1 0		
	R_3	0 0 0 0 1 0 0 1	0 0 0 1 0 1 1 1	0	1
4	$\overleftarrow{R}_3$	0 0 0 1 0 0 1 0	0 0 1 0 1 1 1 0		
	[B2]	1 1 1 1 0 0 1 1	0 0 0 0 0 0 0 0	0	
	сумма	0 0 0 0 0 1 0 1	0 0 1 0 1 1 1 0		
	R_4	0 0 0 0 0 1 0 1	0 0 1 0 1 1 1 1	0	1
5	$\overleftarrow{R}_4$	0 0 0 0 1 0 1 0	0 1 0 1 1 1 1 0		
	[B2]	1 1 1 1 0 0 1 1	0 0 0 0 0 0 0 0	0	
	сумма	1 1 1 1 1 1 0 1	0 1 0 1 1 1 1 0		
	R_5	1 1 1 1 1 1 0 1	0 1 0 1 1 1 1 0	1	0
6	$\overleftarrow{R}_5$	1 1 1 1 1 0 1 0	1 0 1 1 1 1 0 0		
	[B1]	0 0 0 0 1 1 0 1	0 0 0 0 0 0 0 0	1	
	сумма	0 0 0 0 0 1 1 1	1 0 1 1 1 1 0 0		
	R_6	0 0 0 0 0 1 1 1	1 0 1 1 1 1 0 1	0	1
7	$\overleftarrow{R}_6$	0 0 0 0 1 1 1 1	0 1 1 1 1 0 1 0		
	[B2]	1 1 1 1 0 0 1 1	0 0 0 0 0 0 0 0	0	
	сумма	0 0 0 0 0 0 1 0	0 1 1 1 1 0 1 0		
	R_7	0 0 0 0 0 0 1 0	0 1 1 1 1 0 1 1	0	1
8	$\overleftarrow{R}_7$	0 0 0 0 0 1 0 0	1 1 1 1 0 1 1 0		
	[B2]	1 1 1 1 0 0 1 1	0 0 0 0 0 0 0 0	0	
	сумма	1 1 1 1 0 1 1 1	1 1 1 1 0 1 1 0		
	R_8	1 1 1 1 0 1 1 1	1 1 1 1 0 1 1 0	1	0
Коррекция остатка	[B]	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0		
	Сумма	1 1 1 1 0 1 1 1	1 1 1 1 0 1 1 0		
Коррекция частного	[KOP_R_1]	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0		
	Сумма	1 1 1 1 0 1 1 1	1 1 1 1 0 1 1 0		

$C_2 = A_2/B_2 = 11110110_2$, $C_{10} = -10_{10}$; РЕЗ_ОСТ$_2 = 11110111_2$, РЕЗ_ОСТ$_{10} = -9_{10}$

4.4.4 Пример деления 3: A > 0, B < 0 (A = 139, B = -13)

Таблица 4.8

	Старший байт	Младший байт
$\|A\|$	00000000	10001011
|B|	11111111	11110011
|B1|	11110011	00000000
|B2|	00001101	00000000
[КОР R 1]	00000000	00000000
[КОР R 2]	00000000	00000000

Таблица 4.9

		Знак делимого: 0 Знак делителя: 1		Знак остатка	Знак част-ного
1 Проверка корректности деления и определение знакового разряда частного	[A]	0 0 0 0 0 0 0 0	1 0 0 0 1 0 1 1		
	[B]	1 1 1 1 1 1 1 1	1 1 1 1 0 0 1 1		
	[$R_{нач}$]	0 0 0 0 0 0 0 0	0 1 1 1 1 1 1 0	0	
	[$\overleftarrow{R}_{нач}$]	0 0 0 0 0 0 0 0	1 1 1 1 1 1 0 0		
	[B1]	1 1 1 1 0 0 1 1	0 0 0 0 0 0 0 0		
	сумма	1 1 1 1 0 0 1 1	1 1 1 1 1 1 0 0		
	R_1	1 1 1 1 0 0 1 1	1 1 1 1 1 1 0 1	1	1
2	$\overleftarrow{R}_1$	1 1 1 0 0 1 1 1	1 1 1 1 1 0 1 0		
	[B2]	0 0 0 0 1 1 0 1	0 0 0 0 0 0 0 0	1	
	сумма	1 1 1 1 0 1 0 0	1 1 1 1 1 0 1 0		
	R_2	1 1 1 1 0 1 0 0	1 1 1 1 1 0 1 1	1	1
3	$\overleftarrow{R}_2$	1 1 1 0 1 0 0 1	1 1 1 1 0 1 1 0		
	[B2]	0 0 0 0 1 1 0 1	0 0 0 0 0 0 0 0	1	
	сумма	1 1 1 1 0 1 1 0	1 1 1 1 0 1 1 0		
	R_3	1 1 1 1 0 1 1 0	1 1 1 1 0 1 1 1	1	1
4	$\overleftarrow{R}_3$	1 1 1 0 1 1 0 1	1 1 1 0 1 1 1 0		
	[B2]	0 0 0 0 1 1 0 1	0 0 0 0 0 0 0 0	1	
	сумма	1 1 1 1 1 0 1 0	1 1 1 0 1 1 1 0		
	R_4	1 1 1 1 1 0 1 0	1 1 1 0 1 1 1 1	1	1
5	$\overleftarrow{R}_4$	1 1 1 1 0 1 0 1	1 1 0 1 1 1 1 0		
	[B2]	0 0 0 0 1 1 0 1	0 0 0 0 0 0 0 0	1	
	сумма	0 0 0 0 0 0 1 0	1 1 0 1 1 1 1 0		
	R_5	0 0 0 0 0 0 1 0	1 1 0 1 1 1 1 0	0	0
6	$\overleftarrow{R}_5$	0 0 0 0 0 1 0 1	1 0 1 1 1 1 0 0		

	[B1]	1 1 1 1 0 0 1 1	0 0 0 0 0 0 0 0	0	
	сумма	1 1 1 1 1 0 0 0	1 0 1 1 1 1 0 0		
	R_6	1 1 1 1 1 0 0 0	1 0 1 1 1 1 0 1	1	1
7	$\overleftarrow{R}_6$	1 1 1 1 0 0 0 1	0 1 1 1 1 0 1 0		
	[B2]	0 0 0 0 1 1 0 1	0 0 0 0 0 0 0 0	1	
	сумма	1 1 1 1 1 1 1 0	0 1 1 1 1 0 1 0		
	R_7	1 1 1 1 1 1 1 0	0 1 1 1 1 0 1 1	1	1
8	$\overleftarrow{R}_7$	1 1 1 1 1 1 0 0	1 1 1 1 0 1 1 0		
	[B2]	0 0 0 0 1 1 0 1	0 0 0 0 0 0 0 0	0	
	сумма	0 0 0 0 1 0 0 1	1 1 1 1 0 1 1 0		
	R_8	0 0 0 0 1 0 0 1	1 1 1 1 0 1 1 0	0	0
Коррекция остатка	[B]	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0		
	Сумма	0 0 0 0 1 0 0 1	1 1 1 1 0 1 1 0		
Коррекция частного	[KOP_R_1]	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0		
	Сумма	0 0 0 0 1 0 0 1	1 1 1 1 0 1 1 0		

$C_2 = A_2/B_2 = 11110110_2$, $C_{10} = -10_{10}$; РЕЗ_ОСТ$_2$ = 00001001_2, РЕЗ_ОСТ$_{10}$=9_{10}

4.4.5 Пример деления с нулевым остатком: A > 0, B > 0 (A = 72, B = 6)

Таблица 4.10

	Старший байт	Младший байт
$\lvert A\rvert$	00000000	01001000
$\lvert B\rvert$	00000000	00000000
$\lvert B1\rvert$	00000110	00000000
$\lvert B2\rvert$	11111010	00000000
[KOP R 1]	00000000	00000000
[KOP R 2]	00000000	00000000

Таблица 4.11

		Знак делимого: 0 Знак делителя: 0		Знак остатка	Знак частного
1 Проверка корректности деления и определение знакового разряда частного	[A]	0 0 0 0 0 0 0 0	0 1 0 0 1 0 0 0		
	[B]	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0		
	[$R_{нач}$]	0 0 0 0 0 0 0 0	0 1 0 0 1 0 0 0		
	[$\overleftarrow{R}_{нач}$]	0 0 0 0 0 0 0 0	1 0 0 1 0 0 0 0		
	[B2]	1 1 1 1 1 0 1 0	0 0 0 0 0 0 0 0	0	
	сумма	1 1 1 1 1 0 1 0	1 0 0 1 0 0 0 0		
	R_1	1 1 1 1 1 0 1 0	1 0 0 1 0 0 0 0	1	0
2	$\overleftarrow{R}_1$	1 1 1 1 0 1 0 1	0 0 1 0 0 0 0 0		

	[B1]	0 0 0 0 0 1 1 0	0 0 0 0 0 0 0 0	1	
	сумма	1 1 1 1 1 0 1 1	0 0 1 0 0 0 0 0		
	R_2	1 1 1 1 1 0 1 1	0 0 1 0 0 0 0 0	1	0
3	$\overleftarrow{R}_2$	1 1 1 1 0 1 1 0	0 1 0 0 0 0 0 0		
	[B1]	0 0 0 0 0 1 1 0	0 0 0 0 0 0 0 0	1	
	сумма	1 1 1 1 1 1 0 0	0 1 0 0 0 0 0 0		
	R_3	1 1 1 1 1 1 0 0	0 1 0 0 0 0 0 0	1	0
4	$\overleftarrow{R}_3$	1 1 1 1 1 0 0 0	1 0 0 0 0 0 0 0		
	[B1]	0 0 0 0 0 1 1 0	0 0 0 0 0 0 0 0	1	
	сумма	1 1 1 1 1 1 1 0	1 0 0 0 0 0 0 0		
	R_4	1 1 1 1 1 1 1 0	1 0 0 0 0 0 0 0	1	0
5	$\overleftarrow{R}_4$	1 1 1 1 1 1 0 1	0 0 0 0 0 0 0 0		
	[B1]	0 0 0 0 0 1 1 0	0 0 0 0 0 0 0 0	1	
	сумма	0 0 0 0 0 0 1 1	0 0 0 0 0 0 0 0		
	R_5	0 0 0 0 0 0 1 1	0 0 0 0 0 0 0 1	0	1
6	$\overleftarrow{R}_5$	0 0 0 0 0 1 1 0	0 0 0 0 0 0 1 0		
	[B2]	1 1 1 1 1 0 1 0	0 0 0 0 0 0 0 0	0	
	сумма	0 0 0 0 0 0 0 0	0 0 0 0 0 0 1 0		
	R_6	0 0 0 0 0 0 0 0	0 0 0 0 0 0 1 1	0	1
7	$\overleftarrow{R}_6$	0 0 0 0 0 0 0 0	0 0 0 0 0 1 1 0		
	[B2]	1 1 1 1 1 0 1 0	0 0 0 0 0 0 0 0	1	
	сумма	1 1 1 1 1 0 1 0	0 0 0 0 0 1 1 0		
	R_7	1 1 1 1 1 0 1 0	0 0 0 0 0 1 1 0	1	0
8	$\overleftarrow{R}_7$	1 1 1 1 0 1 0 0	0 0 0 0 1 1 0 0		
	[B1]	0 0 0 0 0 1 1 0	0 0 0 0 0 0 0 0	1	
	сумма	1 1 1 1 1 0 1 0	0 0 0 0 1 1 0 0		
	R_8	1 1 1 1 1 0 1 0	0 0 0 0 1 1 0 0	1	0
Коррекция остатка	[B1]	0 0 0 0 0 1 1 0	0 0 0 0 0 0 0 0		
	сумма	0 0 0 0 0 0 0 0	0 0 0 0 1 1 0 0		
Коррекция частного	[KOP_R_1]	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0		
	сумма	0 0 0 0 0 0 0 0	0 0 0 0 1 1 0 0		

$C_2 = A_2/B_2 = 00001100_2$, $C_{10} = 12_{10}$; РЕЗ_ОСТ$_2$ $= 00000000_2$, РЕЗ_ОСТ$_{10}$ $= 0_{10}$

4.4.6 Пример деления с нулевым остатком 1: A < 0, B < 0 (A = -72, B=-6)

Таблица 4.12

	Старший байт	Младший байт
[A]	11111111	10111000
[B]	00000000	00000000
[B1]	11111010	00000000

[B2]	00000110	00000000
[КОР R 1]	00000000	00000000
[КОР R 2]	00000000	00000001

Таблица 4.13

		Знак делимого: 1 Знак делителя: 1		Знак остатка	Знак частного
1 Проверка корректности деления и определение знакового разряда частного	[A]	1 1 1 1 1 1 1 1	1 0 1 1 1 0 0 0		
	[B]	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0		
	[Rнач]	1 1 1 1 1 1 1 1	1 0 1 1 1 0 0 0		
	$[\overleftarrow{R}_{нач}]$	1 1 1 1 1 1 1 1	0 1 1 1 0 0 0 0		
	[B2]	0 0 0 0 0 1 1 0	0 0 0 0 0 0 0 0	1	
	сумма	0 0 0 0 0 1 0 1	0 1 1 1 0 0 0 0		
	R_1	0 0 0 0 0 1 0 1	0 1 1 1 0 0 0 0	0	0
2	$\overleftarrow{R}_1$	0 0 0 0 1 0 1 0	1 1 1 0 0 0 0 0		
	[B1]	1 1 1 1 1 0 1 0	0 0 0 0 0 0 0 0	0	
	сумма	0 0 0 0 0 1 0 0	1 1 1 0 0 0 0 0		
	R_2	0 0 0 0 0 1 0 0	1 1 1 0 0 0 0 0	0	0
3	$\overleftarrow{R}_2$	0 0 0 0 1 0 0 1	1 1 0 0 0 0 0 0		
	[B1]	1 1 1 1 1 0 1 0	0 0 0 0 0 0 0 0	0	
	сумма	0 0 0 0 0 0 1 1	1 1 0 0 0 0 0 0		
	R_3	0 0 0 0 0 0 1 1	1 1 0 0 0 0 0 0	0	0
4	$\overleftarrow{R}_3$	0 0 0 0 0 1 1 1	1 0 0 0 0 0 0 0		
	[B1]	1 1 1 1 1 0 1 0	0 0 0 0 0 0 0 0	0	
	сумма	0 0 0 0 0 0 0 1	1 0 0 0 0 0 0 0		
	R_4	0 0 0 0 0 0 0 1	1 0 0 0 0 0 0 0	0	0
5	$\overleftarrow{R}_4$	0 0 0 0 0 0 1 1	0 0 0 0 0 0 0 0		
	[B1]	1 1 1 1 1 0 1 0	0 0 0 0 0 0 0 0	0	
	сумма	1 1 1 1 1 1 0 1	0 0 0 0 0 0 0 0		
	R_5	1 1 1 1 1 1 0 1	0 0 0 0 0 0 0 1	1	1
6	$\overleftarrow{R}_5$	1 1 1 1 1 0 1 0	0 0 0 0 0 0 1 0		
	[B2]	0 0 0 0 0 1 1 0	0 0 0 0 0 0 0 0	0	
	сумма	0 0 0 0 0 0 0 0	0 0 0 0 0 0 1 0		
	R_6	0 0 0 0 0 0 0 0	0 0 0 0 0 0 1 0	0	0
7	$\overleftarrow{R}_6$	0 0 0 0 0 0 0 0	0 0 0 0 0 1 0 0		
	[B1]	1 1 1 1 1 0 1 0	0 0 0 0 0 0 0 0	0	
	сумма	1 1 1 1 1 0 1 0	0 0 0 0 0 1 0 0		
	R_7	1 1 1 1 1 0 1 0	0 0 0 0 0 1 0 1	1	1
8	$\overleftarrow{R}_7$	1 1 1 1 0 1 0 0	0 0 0 0 1 0 1 0		
	[B1]	0 0 0 0 0 1 1 0	0 0 0 0 0 0 0 0	1	
	сумма	1 1 1 1 1 0 1 0	0 0 0 0 1 0 1 0		

	R_8	1 1 1 1 1 0 1 0	0 0 0 0 1 0 1 1	1	1
Коррек-ция остатка	[B1]	0 0 0 0 0 1 1 0	0 0 0 0 0 0 0 0		
	сумма	0 0 0 0 0 0 0 0	0 0 0 0 1 0 1 1		
Коррек-ция частного	[KOP_R_2]	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 1		
	сумма	0 0 0 0 0 0 0 0	0 0 0 0 1 1 0 0		

$C_2 = A_2/B_2 = 00001100_2$, $C_{10} = 12_{10}$; $РЕЗ\_ОСТ_2 = 00000000_2$, $РЕЗ\_ОСТ_{10} = 0_{10}$

4.4.7 Пример деления с нулевым остатком 2: A > 0, B < 0 (A = 72, B= -6)

Таблица 4.14

	Старший байт	Младший байт
[A]	00000000	01001000
[B]	11111111	11111010
[B1]	11111010	00000000
[B2]	00000110	00000000
[KOP R 1]	00000000	00000000
[KOP R 2]	00000000	00000000

Таблица 4.15

		Знак делимого: 0 Знак делителя: 1		Знак остатка	Знак част-ного
1 Про-верка коррект-ности деления и опре-деление знаково-го раз-ряда частного	[A]	0 0 0 0 0 0 0 0	0 1 0 0 1 0 0 0		
	[B]	1 1 1 1 1 1 1 1	1 1 1 1 1 0 1 0		
	$[R_{нач}]$	0 0 0 0 0 0 0 0	0 1 0 0 0 0 1 0		
	$[\overleftarrow{R}_{нач}]$	0 0 0 0 0 0 0 0	1 0 0 0 0 1 0 0		
	[B1]	1 1 1 1 1 0 1 0	0 0 0 0 0 0 0 0	0	
	сумма	1 1 1 1 1 0 1 0	1 0 0 0 0 1 0 0		
	R_1	1 1 1 1 1 0 1 0	1 0 0 0 0 1 0 1	1	1
2	$\overleftarrow{R}_1$	1 1 1 1 0 1 0 1	0 0 0 0 1 0 1 0		
	[B2]	0 0 0 0 0 1 1 0	0 0 0 0 0 0 0 0	1	
	сумма	1 1 1 1 1 0 1 1	0 0 0 0 1 0 1 0		
	R_2	1 1 1 1 1 0 1 1	0 0 0 0 1 0 1 1	1	1
3	$\overleftarrow{R}_2$	1 1 1 1 0 1 1 0	0 0 0 1 0 1 1 0		
	[B2]	0 0 0 0 0 1 1 0	0 0 0 0 0 0 0 0	1	
	сумма	1 1 1 1 1 1 0 0	0 0 0 1 0 1 1 0		
	R_3	1 1 1 1 1 1 0 0	0 0 0 1 0 1 1 1	1	1
4	$\overleftarrow{R}_3$	1 1 1 1 1 0 0 0	0 0 1 0 1 1 1 0		
	[B2]	0 0 0 0 0 1 1 0	0 0 0 0 0 0 0 0	1	
	сумма	1 1 1 1 1 1 1 0	0 0 1 0 1 1 1 0		

	R_4	1 1 1 1 1 1 1 0	0 0 1 0 1 1 1 1	1	1
5	$\overleftarrow{R}_4$	1 1 1 1 1 1 0 0	0 1 0 1 1 1 1 0		
	[B2]	0 0 0 0 0 1 1 0	0 0 0 0 0 0 0 0	1	
	сумма	0 0 0 0 0 0 1 0	0 1 0 1 1 1 1 0		
	R5	0 0 0 0 0 0 1 0	0 1 0 1 1 1 1 0	0	0
6	$\overleftarrow{R}_5$	0 0 0 0 0 1 0 0	1 0 1 1 1 1 0 0		
	[B1]	1 1 1 1 1 0 1 0	0 0 0 0 0 0 0 0	1	
	сумма	1 1 1 1 1 1 1 0	1 0 1 1 1 1 0 0		
	R_6	1 1 1 1 1 1 1 0	1 0 1 1 1 1 0 1	1	1
7	$\overleftarrow{R}_6$	1 1 1 1 1 1 0 1	0 1 1 1 1 0 1 0		
	[B2]	0 0 0 0 0 1 1 0	0 0 0 0 0 0 0 0	1	
	сумма	0 0 0 0 0 0 1 1	0 1 1 1 1 0 1 0		
	R_7	0 0 0 0 0 0 1 1	0 1 1 1 1 0 1 0	0	0
8	$\overleftarrow{R}_7$	0 0 0 0 0 1 1 0	1 1 1 1 0 1 0 0		
	[B1]	1 1 1 1 1 0 1 0	0 0 0 0 0 0 0 0	1	
	сумма	0 0 0 0 0 0 0 0	1 1 1 1 0 1 0 0		
	R_8	0 0 0 0 0 0 0 0	1 1 1 1 0 1 0 0	0	0
Коррек-ция остатка	[B]	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0		
	сумма	0 0 0 0 0 0 0 0	1 1 1 1 0 1 0 0		
Коррек-ция частного	[КОР_R_1]	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0		
	сумма	0 0 0 0 0 0 0 0	1 1 1 1 0 1 0 0		

$C_2 = A_2/B_2 = 11110100_2$, $C_{10} = -12_{10}$; РЕЗ_ОСТ$_2$ = 00000000_2, РЕЗ_ОСТ$_{10}$=0_{10}

4.4.8 Пример деления с нулевым остатком 3: A < 0, B > 0 (A = -72, B= 6)

Таблица 4.16

	Старший байт	Младший байт
[A]	11111111	10111000
[B]	00000000	00000110
[B1]	00000110	00000000
[B2]	11111010	00000000
[КОР R 1]	00000000	00000000
[КОР R 2]	00000000	11111111

Таблица 4.17

		Знак делимого: 1 Знак делителя: 0		Знак остатка	Знак част-ного
1 Провер-ка кор-	[A]	1 1 1 1 1 1 1 1	1 0 1 1 1 0 0 0		
	[B]	0 0 0 0 0 0 0 0	0 0 0 0 0 1 1 0		

ректности деления и определение знакового разряда частного	[$R_{нач}$]	1 1 1 1 1 1 1 1	1 0 1 1 1 1 1 0		
	[$\overleftarrow{R}_{нач}$]	1 1 1 1 1 1 1 1	0 1 1 1 1 1 0 0		
	[B1]	0 0 0 0 0 1 1 0	0 0 0 0 0 0 0 0	1	
	сумма	0 0 0 0 0 1 0 1	0 1 1 1 1 1 0 0		
	R_1	0 0 0 0 0 1 0 1	0 1 1 1 1 1 0 1	0	1
2	$\overleftarrow{R}_1$	0 0 0 0 1 0 1 0	1 1 1 1 1 0 1 0		
	[B2]	1 1 1 1 1 0 1 0	0 0 0 0 0 0 0 0	0	
	сумма	0 0 0 0 0 1 0 0	1 1 1 1 1 0 1 0		
	R_2	0 0 0 0 0 1 0 0	1 1 1 1 1 0 1 1	0	1
3	$\overleftarrow{R}_2$	0 0 0 0 1 0 0 1	1 1 1 1 0 1 1 0		
	[B2]	1 1 1 1 1 0 1 0	0 0 0 0 0 0 0 0	1	
	сумма	0 0 0 0 0 0 1 1	1 1 1 1 0 1 1 0		
	R_3	0 0 0 0 0 0 1 1	1 1 1 1 0 1 1 1	0	1
4	$\overleftarrow{R}_3$	0 0 0 0 0 1 1 1	1 1 1 0 1 1 1 0		
	[B2]	1 1 1 1 1 0 1 0	0 0 0 0 0 0 0 0	1	
	сумма	0 0 0 0 0 0 0 1	1 1 1 0 1 1 1 0		
	R_4	0 0 0 0 0 0 0 1	1 1 1 0 1 1 1 1	0	1
5	$\overleftarrow{R}_4$	0 0 0 0 0 0 1 1	1 1 0 1 1 1 1 0		
	[B2]	1 1 1 1 1 0 1 0	0 0 0 0 0 0 0 0	1	
	сумма	1 1 1 1 1 1 0 1	1 1 0 1 1 1 1 0		
	R_5	1 1 1 1 1 1 0 1	1 1 0 1 1 1 1 0	1	0
6	$\overleftarrow{R}_5$	1 1 1 1 1 0 1 1	1 0 1 1 1 1 0 0		
	[B1]	0 0 0 0 0 1 1 0	0 0 0 0 0 0 0 0	1	
	сумма	0 0 0 0 0 0 0 1	1 0 1 1 1 1 0 0		
	R_6	0 0 0 0 0 0 0 1	1 0 1 1 1 1 0 1	0	1
7	$\overleftarrow{R}_6$	0 0 0 0 0 0 1 1	0 1 1 1 1 0 1 0		
	[B2]	1 1 1 1 1 0 1 0	0 0 0 0 0 0 0 0	0	
	сумма	1 1 1 1 1 1 0 1	0 1 1 1 1 0 1 0		
	R_7	1 1 1 1 1 1 0 1	0 1 1 1 1 0 1 0	1	0
8	$\overleftarrow{R}_7$	1 1 1 1 1 0 1 0	1 1 1 1 0 1 0 0		
	[B1]	0 0 0 0 0 1 1 0	0 0 0 0 0 0 0 0	1	
	сумма	0 0 0 0 0 0 0 0	1 1 1 1 0 1 0 0		
	R_8	0 0 0 0 0 0 0 0	1 1 1 1 0 1 0 1	0	1
Коррекция остатка	[B]	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0		
	сумма	0 0 0 0 0 0 0 0	1 1 1 1 0 1 0 1		
Коррекция частного	[КОР_R_2]	0 0 0 0 0 0 0 0	1 1 1 1 1 1 1 1		
	сумма	0 0 0 0 0 0 0 0	1 1 1 1 0 1 0 0		

$C_2 = A_2/B_2 = 11110100_2$, $C_{10} = -12_{10}$; $РЕЗ\_ОСТ_2 = 00000000_2$, $РЕЗ\_ОСТ_{10} = 0_{10}$

4.4.9 Пример некорректного деления: A > 0, B > 0 (A = 2560, B = 10)

Таблица 4.18

	Старший байт	Младший байт
$\|A\|$	00001010	00000000
$\|B\|$	00000000	00000000
$\|B1\|$	00001010	00000000
$\|B2\|$	11110110	00000000
[КОР R 1]	00000000	00000000
[КОР R 2]	00000000	00000000

Таблица 4.19

	Знак делимого: 0. Знак делителя: 0			Знак остатка	Знак частного
1 (Проверка корректности деления и определение знакового разряда частного)	[A]	0 0 0 0 1 0 1 0	0 0 0 0 0 0 0 0		
	[B]	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0		
	$[R_{нач}]$	0 0 0 0 1 0 1 0	0 0 0 0 0 0 0 0	0	
	$[\overleftarrow{R}_{нач}]$	0 0 0 1 0 1 0 0	0 0 0 0 0 0 0 0		
	[B2]	1 1 1 1 0 1 1 0	0 0 0 0 0 0 0 0	0	
	сумма	0 0 0 0 1 0 1 0	0 0 0 0 0 0 0 0		
	$[R_1]$	0 0 0 0 1 0 1 0	0 0 0 0 0 0 0 0	1	0
2					
3					
4					
5					
6					
7					

8					
Коррек-					
Коррек-					

Знак полученного остатка совпадает со знаком делимого, поэтому операция деления некорректна.
Окончательный результат получается уже после выполнения первого шага алгоритма.

4.4.10 Пример некорректного деления 1: `A > 0, B < 0 (A = 2560, B= -10)`

Таблица 4.20

	Старший байт	Младший байт
[*A*]	00001010	00000000
[B]	11111111	11110110
[B1]	11110110	00000000
[B2]	00001010	00000000
[КОР_R_1]	00000000	00000000
[КОР_R_2]	00000000	00000000

Таблица 4.21

		Знак делимого: 0. Знак делителя: 1		Знак остатка	Знак частного
1 (Проверка корректности деления и определение знакового разряда частного)	[A]	0 0 0 0 1 0 1 0	0 0 0 0 0 0 0 0		
	[B]	1 1 1 1 1 1 1 1	1 1 1 1 0 1 1 0		
	$[R_{нач}]$	0 0 0 0 1 0 0 1	1 1 1 1 0 1 1 0	0	
	$[\overleftarrow{R}_{нач}]$	0 0 0 1 0 0 1 1	1 1 1 0 1 1 0 0		
	[B1]	1 1 1 1 0 1 1 0	0 0 0 0 0 0 0 0		
	сумма	0 0 0 0 1 0 0 1	1 1 1 0 1 1 0 0		
	R_1	0 0 0 0 1 0 0 1	1 1 1 0 1 1 0 0	1	1
2					
3					
4					

5					
6					
7					
8					
Коррек-					
Коррек-					

Знак полученного остатка совпадает со знаком делимого, поэтому операция деления некорректна.
Окончательный результат получается уже после выполнения первого шага алгоритма.

ПРИЛОЖЕНИЕ

Таблица 1

Номер варианта	*A*	*B*	Номер варианта	*A*	*B*	Номер варианта	*A*	*B*
1.	78	47	**41.**	45	81	**81.**	55	70
2.	82	46	**42.**	70	43	**82.**	63	84
3.	100	26	**43.**	63	32	**83.**	92	25
4.	73	49	**44.**	84	44	**84.**	103	27
5.	51	62	**45.**	95	33	**85.**	72	54
6.	65	53	**46.**	53	68	**86.**	90	37
7.	107	15	**47.**	45	81	**87.**	60	57
8.	67	61	**48.**	25	102	**88.**	80	23
9.	61	24	**49.**	104	21	**89.**	35	77
10.	42	80	**50.**	51	30	**90.**	83	37
11.	112	19	**51.**	72	52	**91.**	62	59
12.	38	88	**52.**	77	43	**92.**	49	63
13.	103	13	**53.**	83	19	**93.**	53	61
14.	68	55	**54.**	41	57	**94.**	37	69
15.	46	61	**55.**	101	13	**95.**	49	38
16.	81	17	**56.**	25	83	**96.**	51	29
17.	69	37	**57.**	63	33	**97.**	82	11
18.	36	79	**58.**	74	27	**98.**	56	43
19.	31	91	**59.**	81	36	**99.**	53	55
20.	109	17	**60.**	32	84	**100.**	93	32
21.	54	66	**61.**	44	83	**101.**	37	70
22.	54	67	**62.**	113	21	**102.**	64	52
23.	36	89	**63.**	38	78	**103.**	65	45
24.	51	69	**64.**	33	82	**104.**	49	58
25.	41	86	**65.**	46	68	**105.**	68	50
26.	27	93	**66.**	65	39	**106.**	73	55
27.	70	47	**67.**	67	46	**107.**	57	62
28.	85	34	**68.**	85	37	**108.**	111	18
29.	92	28	**69.**	74	48	**109.**	49	66
30.	56	61	**70.**	110	15	**110.**	92	28
31.	34	83	**71.**	11	114	**111.**	76	38
32.	67	60	**72.**	43	81	**112.**	54	72
33.	68	54	**73.**	102	25	**113.**	47	81
34.	32	95	**74.**	53	62	**114.**	35	64
35.	99	23	**75.**	79	25	**115.**	61	30
36.	47	63	**76.**	38	67	**116.**	44	39
37.	78	28	**77.**	101	17	**117.**	83	41
38.	33	85	**78.**	60	43	**118.**	49	56
39.	75	36	**79.**	39	86	**119.**	77	21
40.	48	48	**80.**	69	56	**120.**	18	70

Таблица 2

Номер варианта	***A***	***B***	**Номер варианта**	***A***	***B***	**Номер варианта**	***A***	***B***
1.	61	47	**41.**	117	14	**81.**	16	118
2.	22	81	**42.**	19	101	**82.**	38	62
3.	82	21	**43.**	119	20	**83.**	120	28
4.	20	83	**44.**	46	63	**84.**	18	121
5.	23	84	**45.**	102	17	**85.**	27	122
6.	64	37	**46.**	123	26	**86.**	24	85
7.	65	45	**47.**	67	50	**87.**	46	49
8.	66	47	**48.**	36	86	**88.**	53	15
9.	44	67	**49.**	29	124	**89.**	103	68
10.	25	87	**50.**	51	45	**90.**	35	126
11.	88	36	**51.**	104	11	**91.**	13	125
12.	89	26	**52.**	43	69	**92.**	12	22
13.	90	35	**53.**	106	21	**93.**	105	91
14.	70	34	**54.**	12	107	**94.**	33	48
15.	41	71	**55.**	52	63	**95.**	55	57
16.	72	33	**56.**	92	27	**96.**	42	109
17.	73	28	**57.**	13	108	**97.**	23	74
18.	93	25	**58.**	44	56	**98.**	41	40
19.	37	94	**59.**	14	170	**99.**	59	29
20.	95	24	**60.**	75	42	**100.**	58	111
21.	15	96	**61.**	112	11	**101.**	16	97
22.	76	32	**62.**	113	17	**102.**	32	50
23.	38	77	**63.**	62	54	**103.**	48	43
24.	78	40	**64.**	98	30	**104.**	60	115
25.	35	81	**65.**	83	27	**105.**	53	85
26.	63	77	**66.**	72	89	**106.**	38	94
27.	33	115	**67.**	76	23	**107.**	103	38
28.	62	78	**68.**	83	36	**108.**	79	49
29.	91	52	**69.**	94	18	**109.**	57	41
30.	59	68	**70.**	59	61	**110.**	48	58
31.	31	79	**71.**	114	34	**111.**	31	80
32.	18	99	**72.**	53	39	**112.**	39	41
33.	19	100	**73.**	116	30	**113.**	125	94
34.	52	61	**74.**	57	63	**114.**	24	37
35.	57	84	**75.**	35	71	**115.**	43	79
36.	70	39	**76.**	61	42	**116.**	92	38
37.	33	78	**77.**	81	59	**117.**	26	84
38.	56	66	**78.**	69	42	**118.**	58	74
39.	34	85	**79.**	76	45	**119.**	91	76
40.	20	98	**80.**	67	49	**120.**	51	63

Таблица 3

Номер варианта	*A*	*B*	Номер варианта	*A*	*B*	Номер варианта	*A*	*B*
1.	3038	31	**41.**	1571	23	**81.**	1716	26
2.	1682	24	**42.**	1536	22	**82.**	1410	15
3.	1318	19	**43.**	1303	20	**83.**	1449	21
4.	1382	18	**44.**	986	12	**84.**	1020	15
5.	1344	20	**45.**	1654	18	**85.**	1272	12
6.	1422	21	**46.**	2076	22	**86.**	1248	13
7.	1305	14	**47.**	964	12	**87.**	1152	14
8.	1630	17	**48.**	2328	27	**88.**	1904	28
9.	1834	26	**49.**	1182	12	**89.**	965	10
10.	2072	22	**50.**	1145	17	**90.**	1924	30
11.	2566	29	**51.**	1816	21	**91.**	946	10
12.	954	10	**52.**	1833	27	**92.**	1088	26
13.	982	15	**53.**	1436	19	**93.**	1380	15
14.	1916	26	**54.**	1644	23	**94.**	932	10
15.	1804	25	**55.**	981	15	**95.**	960	12
16.	1534	15	**56.**	1058	18	**96.**	1460	20
17.	1643	24	**57.**	974	11	**97.**	1080	12
18.	944	12	**58.**	1211	17	**98.**	1924	26
19.	1684	25	**59.**	1911	27	**99.**	1056	18
20.	1645	35	**60.**	1933	30	**100.**	938	14
21.	2461	31	**61.**	2164	29	**101.**	1093	27
22.	2182	27	**62.**	1375	21	**102.**	2137	22
23.	1589	24	**63.**	2194	19	**103.**	1074	11
24.	1426	19	**64.**	1054	13	**104.**	2468	26
25.	1748	18	**65.**	2389	31	**105.**	2391	28
26.	2374	27	**66.**	1987	26	**106.**	1076	13
27.	1146	17	**67.**	1654	18	**107.**	1540	22
28.	1271	18	**68.**	1022	15	**108.**	1634	19
29.	1228	17	**69.**	2076	23	**109.**	994	14
30.	1522	23	**70.**	1354	19	**110.**	924	11
31.	1435	21	**71.**	1217	18	**111.**	1440	20
32.	1036	15	**72.**	1800	23	**112.**	1056	12
33.	1302	17	**73.**	1017	13	**113.**	1462	17
34.	3184	35	**74.**	2171	23	**114.**	1584	24
35.	2781	46	**75.**	1789	26	**115.**	3012	29
36.	1734	22	**76.**	1925	28	**116.**	2345	28
37.	3267	41	**77.**	1123	14	**117.**	1894	21
38.	974	33	**78.**	968	41	**118.**	1473	15
39.	2367	25	**79.**	1885	17	**119.**	1844	18
40.	3106	39	**80.**	2043	19	**120.**	1242	27

ЛИТЕРАТУРА

1. П.С. Довгий, В.И. Поляков. Арифметические основы ЭВМ. - Санкт-Петербург, 2010. – 57 с.
2. Савельев А.Я. Прикладная теория цифровых автоматов. – М: Высш. шк., 1987. - 272 с.
3. Савельев А.Я. Основы информатики. - М.: Изд-во МГТУ им Н.Э. Баумана, 2001. - 328 с.
4. Поспелов Д. А. Арифметические основы вычислительных машин дискретного действия. – М: Высш. шк., 1970. - 308 с.
5. Ковригин Б. Н. Алгоритмы умножения. – М.: Изд-во МИФИ, 2007. - 40 с.
6. Википедия. Алгоритм Бута. http://ru.wikipedia.org/wiki/%C0%EB%E3%EE%F0%E8%F2%EC_%C1%F3%F2%E0

Содержание

В 2009 году Университет стал победителем многоэтапного конкурса, в результате которого определены 12 ведущих университетов России, которым присвоена категория «Национальный исследовательский университет». Министерством образования и науки Российской Федерации была утверждена программа его развития на 2009–2018 годы. В 2011 году Университет получил наименование «Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики»

КАФЕДРА МОНИТОРИНГА И ПРОГНОЗИРОВАНИЯ ИНФОРМАЦИОННЫХ УГРОЗ

Кафедра организована в 2002 году. Первоначальное название кафедры «Мониторинга и прогнозирования чрезвычайных ситуаций». Кафедра готовила специалистов по направлениям подготовки «прикладная математика» и «организации и технологии защиты информации». С 2011 года кафедра перешла на двухуровневую систему образования, началась подготовка бакалавров и магистров по направлению «информационная безопасность».

Марина Борисовна Будько
Владимир Андреевич Грозов
Дмитрий Игоревич Милосердов

Реализация процессором арифметических операций

Учебно-методическое пособие по выполнению домашних заданий по дисциплине "Дискретная математика"

В авторской редакции
Редакционно-издательский отдел НИУ ИТМО
Зав. РИО Н.Ф. Гусарова
Лицензия ИД № 00408 от 05.11.99
Подписано к печати
Заказ №
Тираж
Отпечатано на ризографе

Редакционно-издательский отдел
Санкт-Петербургского национального исследовательского университета информационных технологий, механики
и оптики
197101, Санкт-Петербург, Кронверкский пр., 49