Двоичная арифметика и Арифметико-логические устройства (АЛУ)

7.1. Системы счисления и перевод чисел из одной системы счисления в другую

Над цифровыми сигналами могут выполняться не только логические, но и арифметические операции, такие как сложение +, вычитание -, умножение *, деление /, целочисленное деление **mod**, получение остатка **rem**. Информация, как уже было отмечено, обычно представляется в цифровых системах в виде двоичных кодов, соответствующих значениям сигналов. Отдельные элементы двоичного кода, имеющие значение 0 или 1, называют **разрядами или битами (bit)**. Обычно используется позиционная система представления чисел в двоичном коде, при которой вес каждого разряда равен степени двойки. Например, двоичный код 1011 соответствует десятичному числу 11 ($1* 2^3 +0 *2^2 +1* 2^1 + 1* 2^0 =8+0+2+1 =11$).

Двоичная система, удобная для ЭВМ, для человека неудобна из-за громоздкости и непривычной записи. Для этого и разработаны 8 система счисления (с/с) и 16 с/с.

Числа в этих с/с читаются почти так же легко, как и десятичные, требуют соответственно в три (8с/с) и в четыре (16с/с) раза меньше разрядов, чем в 2с/с (числа 8 и 16, соответственно 3-я и 4-я степени числа 2).

Перевод чисел из 8 с/с и 16 с/с в 2 с/с очень прост: достаточно каждую цифру заменить эквивалентной ей двоичной триадой (тройкой цифр) или тетрадой (четверкой цифр):

Например,

$537,1_8=101\ 011\ 111,001_2$

$1A3,F_{16}=1\ 1010\ 0011,1111_2$

Чтобы перевести число из 2с/с в 8с/с или 16с/с, нужно разбить влево и вправо от запятой на триады (для 8с/с) или тетрады (для 16с/с) и каждую такую группу заменить соответствующей 8-ричной или 16-ричной цифрой.

$10101001,10111_2=10\ 101\ 001,\ 101\ 110_2=251,56_8$

$10101001,10111_2=1010\ 1001,\ 1011\ 1000_2=A9,B8_{16}$

Перевод чисел из 10 с/с в 2с/с, 8 с/с, 16 с/с

Перевод целых чисел:

- Исходное число разделить на основание новой с/с (Р). Остаток от деления записать в новой с/с. Это младшая цифра искомого числа.
- Если частное от деления равно 0, то искомое число получено

- Если частное не равно 0, то разделить его на P. Остаток от деления записать в новой с/с. Это предыдущая цифра искомого числа.

Число с основанием P записывается как последовательность остатков от деления в обратном порядке, начиная с последнего.

Перевод правильных десятичных дробей:

- Правильную десятичную дробь при переводе необходимо последовательно умножать на основание той системы счисления, в которую она переводится, отделяя после каждого умножения целую часть произведения.
- Число в новой с/с записывается как последовательность полученных целых частей произведения.

Умножение производится до тех пор, пока дробная часть произведения не станет равна 0. Это значит, что сделан точный перевод. В противном случае, перевод осуществляется до заданной точности.

Перевод смешанных чисел осуществляется в 3 этапа:

- Осуществляется перевод целой части числа в новую с/с.
- Осуществляется перевод дробной части числа в новую с/с.
- Складываются результаты переводов первых 2-х пунктов.

Перевод чисел из 2 с/с, 8 с/с, 16 с/с и т.д. (из P-ой с/с) в 10 с/с.

- Записать переводимое число в виде полинома в старой с/с.
- В полученном полиноме заменить основание и все коэффициенты числами в новой с/с (10 с/с).
- Выполнить арифметические действия в новой (10-ой) с/с.

Например,

$9B3_{17}=9*17^2+B*17^1+3*17^0=2791_{10}$

$110101,01_2=1*2^5+1*2^4+0*2^3+1*2^2+0*2^1+1*2^0+0*2^{-1}+$
$+1*2^{-2}=32+16+4+1+0,25=53,25_{10}$

При выполнении арифметических операций над двоичными числами без знака следует учитывать необходимость выравнивания положений чисел по младшим разрядам и возможность изменения разрядности результата. Например, при сложении разрядность результата может превышать разрядность наиболее длинного операнда на 1, а при умножении- длина результата равна сумме разрядностей операндов. Например, сложении двух двоичных кодов 11 и 1 выполняется как 11+01 и дает результат 100 (три разряда), а умножении 11 на 11 дает 1001.

7.2. Двоичные коды чисел.

В ЭВМ при выполнении арифметических операций применяют **двоичный прямой, обратный и дополнительный коды чисел**. К кодам чисел выдвигаются следующие требования:

1) разряды числа в коде жестко связаны с определенной разрядной сеткой;
2) для записи кода знака в разрядной сетке отводится фиксированный разряд.

Такое представление чисел называется представлением с **фиксированной запятой**.

Например, если за основу представления двоичного кода числа взят один байт, то для представления значения числа будет отведено 7 разрядов, а для записи кода знака один разряд. В примерах мы знаковый разряд числа будем отделять символом запятой и располагать слева.

Прямой код (ПК) числа. Прямой код двоичного числа совпадает по изображению с записью самого числа. Значение знакового разряда для положительных чисел равно 0, а для отрицательных чисел 1. Например, в случае, когда для записи кода числа выделен один байт, для положительного десятичного числа + 9, в двоичном коде представленного как +1001, ПК равен 0,0001001. Для отрицательного числа 9 – двоичный ПК равен 1,0001001.

Обратный код (ОК). Обратный код для положительного числа совпадает с прямым кодом. Для отрицательного числа все цифры числа заменяются на противоположные (1 на 0, 0 на 1), а в знаковый разряд заносится единица. Например, для двоичного кода положительного числа +9, представляемого в 8-ми разрядной сетке, ОК равен 0,0001001. Для отрицательного числа -9, двоичный ОК равен 1,1110110.

Дополнительный код (ДК). Дополнительный код положительного числа совпадает с прямым кодом. Для отрицательного числа дополнительный код образуется путем получения обратного кода и добавлением к младшему разряду единицы.

Ниже приведен пример представления одного и того же отрицательного числа (-9 или в двоичной системе -1001) в восьмиразрядной сетке в различных кодах

Прямой код	Обратный код	Дополнительный код
1,0001001	1,1110110	1,1110111

Особенности сложения чисел в обратном и дополнительном кодах.

Операция вычитания обычно выполняется в ЭВМ как операция сложения отрицательного значения вычитаемого и поэтому ниже рассмотрена только операция сложения. При сложении чисел в дополнительном коде (ДК) возникающая единица переноса в знаковом разряде отбрасывается. При сложении чисел в обратном коде (ОК) возникающая единица переноса в знаковом разряде прибавляется к младшему разряду суммы. Результат операции обычно хранится в соответствующем коде, но иногда, например, при выводе на печать, необходимо преобразовать его в прямой код. При этом ОК преобразуется в ПК заменой цифр во всех разрядах, кроме знакового, на противоположные. ДК преобразуется в ПК также, как и обратный код, с последующим прибавлением единицы к младшему разряду (см. примеры 7.1 и 7.2).

Пример 7.1. Требуется сложить десятичные числа X=7 и Y= -3.

В двоичной системе это коды X=+111 и Y=-11 , они же, представленные в 8-ми разрядной сетке со знаковым разрядом слева, Аргументы обозначим как X= 0,0000111 и Y=1,0000011, результат S. Сложение этих чисел в обратном и дополнительном коде представлено в табл.7.1 Так как результат – положительное число, то результаты в ПК, ОК и ДК совпадают.

Таблица 7.1. Сложение чисел в прямом, дополнительном и обратном и коде, результат положительный

Аргументы и результат	ПК	ДК	ОК
X	0,0000111	0,0000111	0,0000111
Y	1,0000011	1,1111101	1,1111100
Промежуточный результат	0,0000100	1<- 0,0000100 1 переноса за знаковый разряд отбрасывается	1<- 0,0000011 1 переноса за знаковый разряд прибавляется к младшим разрядам
Результат S	0,0000100	0,0000100	0,0000100

Пример 7.2. Требуется сложить отрицательные двоичные числа X=-101 и Y=-11 представленные в 8-ми разрядной сетке в обратном и дополнительном кодах. Результат отрицательный (см. табл.7.2).

Таблица 7.2. Сложение чисел в прямом, дополнительном и обратном коде, результат отрицательный

Аргументы и результат	ПК	ДК	ОК
X	1,0000101	1,1111011	1,1111010
Y	1,0000011	1,1111101	1,1111100
Промежуточный результат	1,0001000	1<- 1,1111000 перенос за знаковый разряд отбрасывается	1<- 1,1110110 1 переноса за знаковый разряд прибавляется к младшим разрядам
Результат S операции в соответствующем коде	1,0001000	1,1111000	1,1110111

7.3. Модифицированные обратный (МОК) и дополнительный (МОД) коды чисел

При переполнении разрядной сетки, в которой представлены двоичные числа со знаком, происходит перенос единицы в знаковый разряд. Это приводит к неправильному результату, причем положительное число, получившееся в результате арифметической операции, может восприниматься как отрицательное (так как в знаковом разряде "1") и наоборот. Например, при сложении двоичных чисел X=0,1000110 и Y=0,1 000011 имеем результат = 1, 0001001. Здесь X и Y - коды положительных чисел, и результат положителен, но результат их сложения – это код отрицательного числа ("1" в знаковом разряде). Для обнаружения переполнения разрядной сетки используются **модифицированные коды (МК)**.

В **модифицированном обратном (МОК**) и **модифицированном дополнительном (МОД)** кодах под знак числа отводится не один, а два разряда: "00" соответствует знаку "+", "11" - знаку "-". Любая другая комбинация ("01" или "10"), получившаяся в знаковых разрядах служит признаком переполнения разрядной сетки. В остальном, сложение чисел в модифицированных кодах ничем не отличается от сложения в обычных обратном и дополнительном кодах.

Итак, основные правила выполнения операций сложения и вычитания чисел.

1. *Числа в прямом коде со знаком*:

Сложение. При одинаковых знаках чисел сложить. Результату присвоить тот же знак. Переполнение возникает при возникновении переноса из старшего разряда.

При разных знаках чисел вычесть меньшее число из большего. Результату присвоить знак большего числа. Переполнение не возникает.

Вычитание. Изменить знаковый бит числа и провести сложение.

2. *Числа в дополнительном коде*:

Сложение. Сложить два числа, игнорируя перенос из старшего разряда. Переполнение возникает, если переносы в старший разряд и из старшего разряда различны.

Вычитание. Инвертировать все биты числа – вычитаемого и сложить с числом – уменьшаемым. К результату прибавить единицу в младший разряд.

7.4. Функциональная схема АЛУ для сложения и вычитания чисел с фиксированной запятой

Функциональная схема АЛУ для сложения и вычитания чисел с фиксированной запятой представлена на следующем рисунке

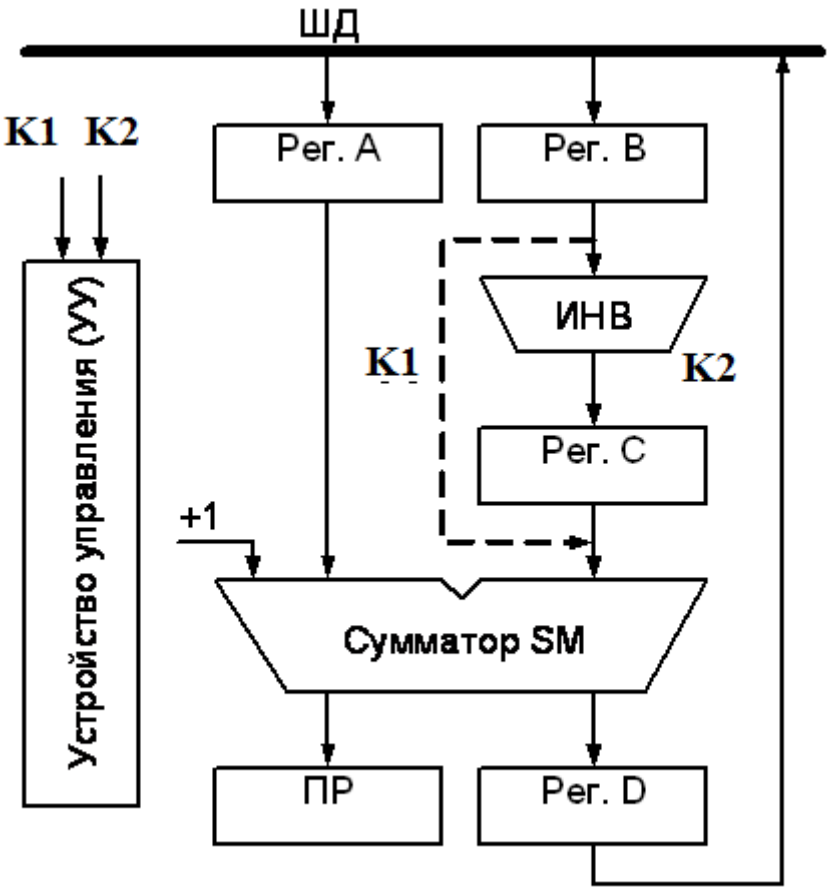


Рис. 7.1. Функциональная схема АЛУ для сложения и вычитания чисел с фиксированной запятой в дополнительном коде

В состав АЛУ входят регистры Рег. A, Рег. B, Рег. C и Рег. D, комбинационный сумматор SM, блок инвертирования ИНВ, блок выработки признаков результата (ПР) и Устройство управления (УУ).

Блок ПР вырабатывает признак результата вычисления φ. Это:

φ = (00) – результат положительный,

φ = (11) – результат отрицательный,

φ = (01) – результат нулевой,

φ = (10) – переполнение.

Регистры подсоединены к Шине данных (ШД) компьютера. На входы Устройства управления (УУ) подаются команды: K1 – "сложение", K2 – "вычитание".

Алгоритм работы АЛУ можно представить в форме псевдокода на VHDL (Русские имена регистров заменены на латинские).

```
If  K = K1 then  --сложение чисел
  REG_A<=SHD; -- запись первого числа в регистр A
   REG_B<=SHD;  --запись второго числа в регистр B
  SM <= A + B ; --сложение двух чисел
```

```
  if FI /= “10” then – нет переполнения
    REG_D <=  SM; --запись результата в регистр D
   SHD <= REG_D ;--передача результата на шину данных
  end if; --при переполнении в УУ результат в память
                --не передается
end if; --Таким образом, сложение двух чисел в АЛУ выполняется за 5
машинных тактов.
```

Else --*программа вычитания чисел имеет следующий вид*:

```
If K = K2 then
  REG_A<= SHD; -- запись первого числа в регистр A
  REG_B<= SHD;  --запись второго числа в регистр B
  REG_C<= not B;  --запись инвертированного числа B в регистр C
  SM <=REG_ A + REG_C +1; --вычитание в дополнительном коде
 if FI /= “10” then – нет переполнения
   REG_D <=  SM; --запись результата в регистр D
  SHD <= REG_D ;--передача результата на шину данных
end if;         --при переполнении в УУ результат в память
                --не передается

end if; --Таким образом, вычитание двух чисел в АЛУ выполняется за 7
          машинных тактов
```

.

Алгоритмы управления сигналами, поступающими на регистры для выполнения операций сложения и вычитания реализованы в Устройстве управления АЛУ, которое является конечным автоматом со схемной или программированной логикой.

7.5. АЛУ для умножения чисел с фиксированной запятой

7.5.1. Методы умножения чисел

1. *Умножение целых чисел, начиная с младших разрядов со сдвигом множимого влево при неподвижной сумме частичных произведений.*

Пример

$1011 \rightarrow 11_{10}$ – множимое

*

$1101 \rightarrow 13_{10}$ – множитель

1011 – умножение множимого на правую единицу

0000 – умножение на 0 во 2-м разряде

01011 – сумма частичных произведений

1011 – умножение на 1 в 3-м разряде

110111 – сумма частичных произведений

1011 – умножение на 1 в 4-м разряде

10001111 – сумма частичных произведений – результат = 143_{10}

Структура аппаратной реализации этого алгоритма умножения показана на следующем рисунке

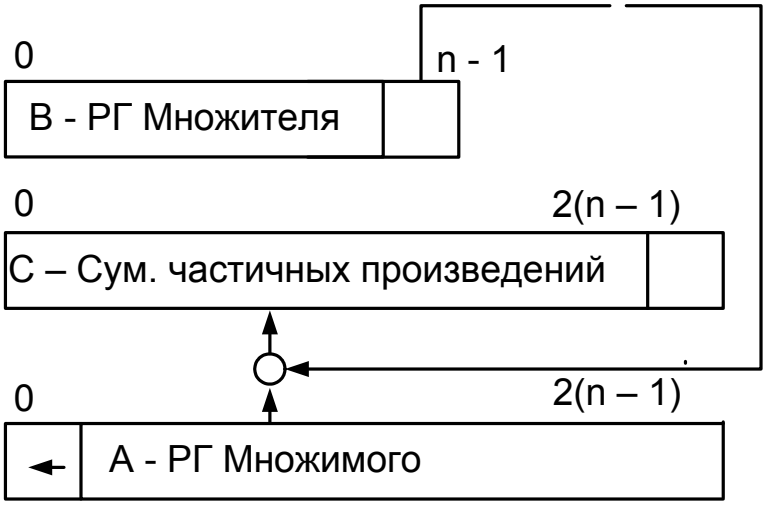


Рис. 7.2. Умножение со сдвигом множимого влево

АЛУ для умножения содержит три регистра: регистр А для множимого, регистр В для множителя и регистр С для сумм частичных произведений. Разрядность регистра В равна разрядности числа (n), разрядности регистров А и С – 2n. В состав регистра С входит комбинационный сумматор.

Умножение проводится за n тактов. На каждом такте число в регистре А сдвигается влево на один разряд. Если в соответствующем разряде числа множителя (регистр В) единица, то число из регистра А передается в регистр С и складывается с суммой частичных произведений. Иначе число из регистра А в регистр С не передается. Результат получается в регистре С.

2. *Умножение целых чисел, начиная с младших разрядов со сдвигом вправо сумм частичных произведений при неподвижном множителе.*

Пример

$1011 \rightarrow 11_{10}$ – множимое

*

$1101 \rightarrow 13_{10}$ – множитель

1011 – умножение множимого на правую единицу

0000 – умножение на 0 во 2-м разряде

→ → 1011 – сумма частичных произведений

1011 – умножение на 1 в 3-м разряде

→110111 – сумма частичных произведений

1011 – умножение на 1 в 4-м разряде

10001111 – сумма частичных произведений = 143_{10}

Структура аппаратной реализации этого алгоритма умножения показана на следующем рисунке

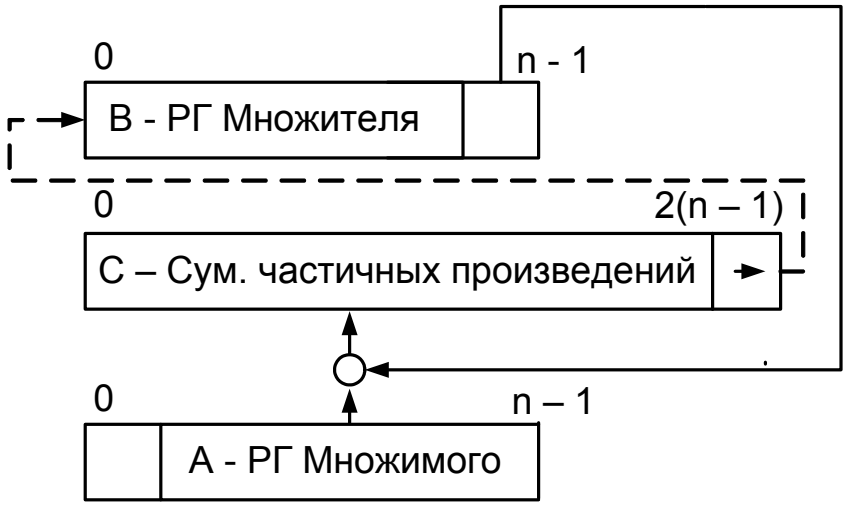


Рис. 7.3. Умножение со сдвигом частичного произведения вправо

Разрядности регистров A и B равна разрядности числа (n), разрядности регистра C – 2n. В состав регистра C входит комбинационный сумматор.

Умножение проводится за n тактов, начиная с младшего разряда множителя. На каждом такте число в регистре C сдвигается вправо на один разряд. Если в соответствующем разряде числа множителя (регистр B) единица, то число из регистра A передается в регистр C и складывается с частичным произведением. Иначе число из регистра A в регистр C не передается. Результат умножения получается в регистре C.

Экономная реализация этого метода умножения получается в случае, когда множитель на каждом такте сдвигается на разряд вправо ("отработанные" младшие разряды множителя пропадают). В освободившиеся разряды регистра B записываются младшие разряды суммы частных произведений. В этом случае все три регистра A, B и C имеют разрядность n.

3. *Умножение, начиная со старших разрядов множителя со сдвигом суммы частичных произведений влево при неподвижном множителе*

Пример

1011 → 11_{10} – множимое

*

1101 → 13_{10} – множитель

1011 – умножение на 1 в 4-м разряде – частичное произв.

← 1011 – сдвиг частичного произведения влево
1011 – множимое – умножение на 1 в 3-м разряде
100001 – сумма частичных произведений
← ← 100001 – сдвиг суммы частичных произведений влево
1011 – множимое – умножение на 1 в 1-м разряде
10001111 – частичное произведение – результат = 143_{10}

Структура Аппаратной реализации этого алгоритма умножения показана на следующем рисунке

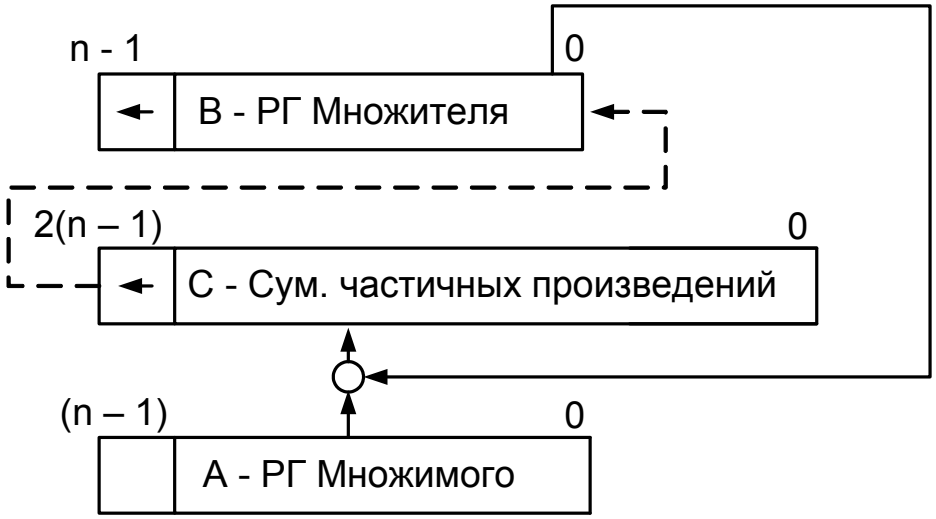


Рис. 7.4. Умножение со сдвигом частичного произведения влево

Разрядности регистров A и B равна разрядности числа (n), разрядности регистра C – 2n. В состав регистра C входит комбинационный сумматор.

Умножение проводится за n тактов, начиная со старшего разряда множителя. На каждом такте число в регистре C сдвигается влево на один разряд. Если в соответствующем разряде числа множителя (регистр B) единица, то число из регистра A передается в регистр C и складывается с суммой частичных произведений. Иначе число из регистра A в регистр C не передается. Результат умножения получается в регистре C.

Удачная реализация этого метода умножения получается в случае, когда множитель на каждом такте сдвигается на разряд влево ("отработанные" старшие разряды множителя пропадают). В освободившиеся разряды регистра B записываются старшие разряды суммы частных произведений. В этом случае все три регистра A, B и C имеют разрядность n.

4. *Умножение, начиная со старшего разряда множителя, со сдвигом вправо множимого и неподвижным частичным произведением*.

Пример

$1011 \rightarrow 11_{10}$ – множимое
*
$1101 \rightarrow 13_{10}$ – множитель
1011 – умножение на 1 в 4-м разряде – частичное произв.

→ 1011 – сдвиг множимого вправо
100001 – сумма частичных произведений в 3-м разряде
→ 1011 – сдвиг множимого вправо на 2 разряда
10001111 – сумма частичных произведений = 143_{10}

Структура Аппаратной реализации этого алгоритма умножения показана на следующем рисунке

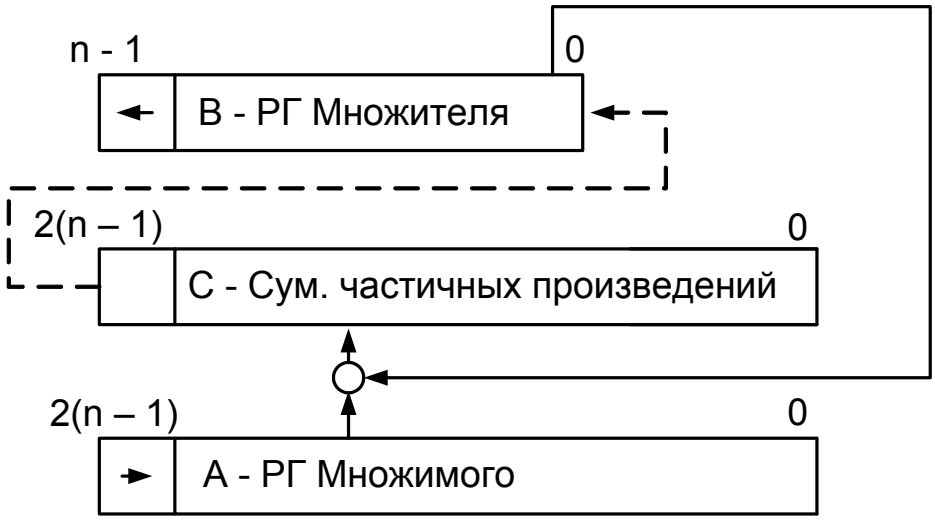


Рис. 7.5. Умножение со сдвигом множимого вправо

Разрядности регистра А равна разрядности числа (n), разрядности регистров А и С – 2n. В состав регистра С входит комбинационный сумматор.

Умножение проводится за **n** тактов, начиная со старшего разряда множителя. На каждом такте число в регистре А сдвигается вправо на один разряд. Если в соответствующем разряде числа множителя (регистр В) единица, то число из регистра А передается в регистр С и складывается с суммой частичных произведений. Иначе число из регистра А в регистр С не передается. Результат умножения получается в регистре С.

Удачная реализация этого метода умножения получается в случае, когда множитель на каждом такте сдвигается на разряд влево ("отработанные" старшие разряды множителя пропадают). В освободившиеся разряды регистра В записываются старшие разряды суммы частных произведений. В этом случае регистры В и С имеют разрядность **n**.

7.5.2. Функциональная схема АЛУ для умножения целых чисел.

Функциональная схема АЛУ для умножения чисел с фиксированной запятой, начиная с младших разрядов со сдвигом вправо сумм частичных

произведений при неподвижном множителе, представлена на следующем рисунке.

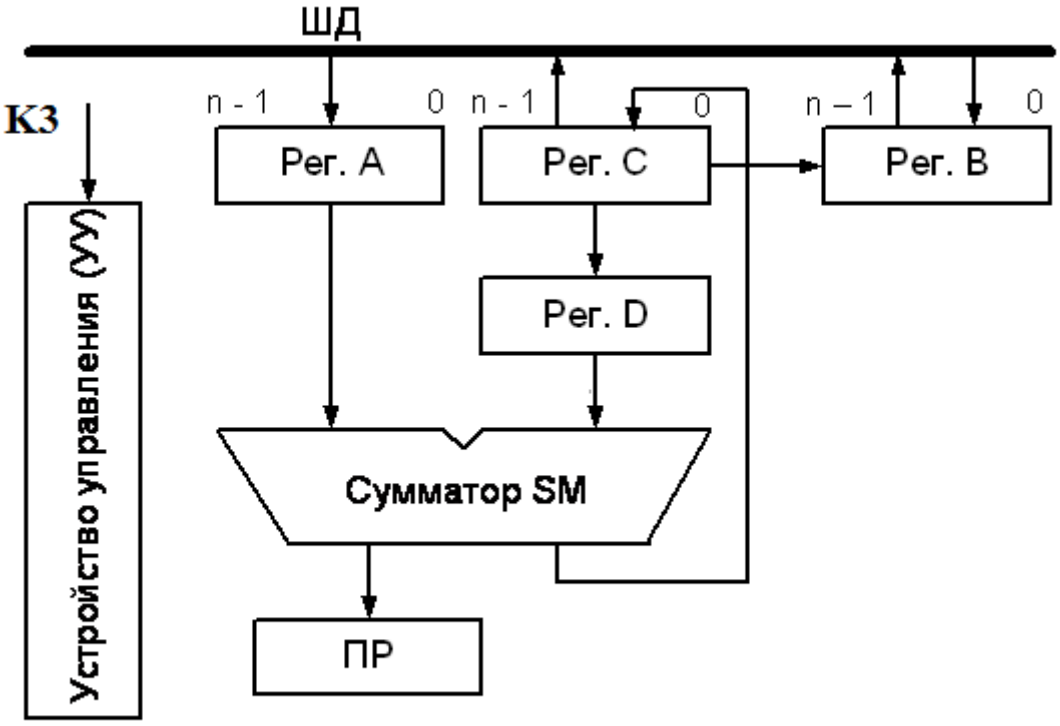


Рис. 7.6. АЛУ для умножения чисел с фиксированной запятой

Число – множимое с шины данных записывается в регистр A, а число – множитель – в регистр B. Сумма частичных произведений с выхода сумматора SM записывается в регистр C. Регистры C и B образуют единый регистр сдвига вправо, в котором младший разряд C [0] переписывается в старший разряд B[n]. Регистр D служит для промежуточного хранения содержимого регистра C, что предотвращает гонки при работе АЛУ.

- Множимое и множитель записываются в регистры A и B в прямых кодах со знаком. Регистры C и D обнуляются. В счетчик, находящийся в Устройстве управления, записывается число циклов, необходимых для умножения двух чисел.
- Если младший разряд множителя B [0] = 1, то к сумме частичных произведений в регистре C прибавляется модуль множимого из регистра A. Если B [0] = 0, то число в регистре C не изменяется.
- Число в регистрах C и B сдвигается вправо на один разряд. При этом в младший разряд B [0] записывается очередная цифра числа -

множителя, которая определяет процесс формирования суммы частичных произведений.

- После (n – 1) циклов процесс умножения чисел по модулю заканчивается. Производится дополнительный сдвиг суммы частичных произведений вправо, при этом в B [0] появляется старший (знаковый) разряд множителя. Знаковый разряд множимого A[n] и знаковый разряд множителя B [0] поступают на логический элемент "Исключающее ИЛИ", на выходе которого образуется число, определяющее знак произведения (если знаки множителя и множимого совпадают, то знак произведения "+", иначе знак "–").
- Результат умножения двух чисел образуется в регистрах C и B (B регистре C находятся старшие разряды произведения, в регистре B – младшие разряды). Произведение передается из этих регистров на шину данных.

Блок признаков ПР вырабатывает признаки результата операции.

Это:
φ = (00) – результат положительный,
φ = (11) – результат отрицательный,
φ = (01) – результат нулевой,

Алгоритм умножения двух целых чисел имеет следующий вид:
(К3 – команда умножения, СТ – счетчик циклов в УУ)

```
If K = K3 then
  REG_A<=SHD; -- запись первого числа в регистр A
   REG_B<=SHD; --запись второго числа в регистр B
 REG_C <=(others=> ‘0’); REG_D <=(others=> ‘0’);
      --запись нуля в регистры C и D
CT <= n -1; --запись числа циклов умножения
  While CT /= 0 loop
    If          B (0) /= ‘0’ then
       SM <= A +D; --сложение в сумматоре
      REG_C <= SM; --пересылка суммы в регистр C
    End if;
    REG_C<=shr (REG_C); REG_D<=shr (REG_D);
      --сдвиг вправо на один разряд
      CT <= CT – 1; --декремент числа в счетчике
  End loop;
     REG_C<=shr (REG_C); REG_D<=shr (REG_D);

      --определение знака произведения в УУ}
      REG_C (n – 1) <= --{знак произведения};
      SHD <= Рег. C;
      SHD <= Рег. B; --передача произведения на ШД
End if;
```

Выдача управляющих сигналов для выполнения умножения реализована в Устройстве управления АЛУ, которое является конечным автоматом со схемной или программированной логикой.

В результате умножения двух чисел, содержащих n – 1 цифровой разряд получаем произведение, содержащее 2(n – 1) цифровой разряд. Для правильного размещения знакового разряда это число необходимо сдвинуть на один разряд вправо. Таким образом произведение содержит 2(n – 1) + 1 разряд.

7.6. АЛУ для деления целых чисел

7.6.1. Алгоритм деления целых чисел

Операция деления целых чисел C = A/B проводится по методу последовательного вычитания делителя B из делимого A. На первом шаге алгоритма проводится вычитание делителя B из делимого A, что приводит к возникновению первого остатка R_1. На втором шаге из первого остатка R_1 вычитается делитель B, что приводит к возникновению остатка R_2 и т.д. Операция деления проводится над модулями чисел, а знак результата вычисляется отдельно.

Рассмотрим алгоритм деления без восстановления остатка. Алгоритм включает ряд вспомогательных операций:

- Анализ делимого и делителя на равенство нулю,
- Вычисление знака частного при помощи логической операции "Исключающее ИЛИ".
- Первый остаток R_1 находим по следующей формуле

$$R_1 = 2|A| + [-\overline{B}];$$

В этой формуле $|A|$ - модуль делимого. Умножение делимого на 2 обозначает сдвиг делимого на один разряд влево в регистре. Второе слагаемое $[-\overline{B}]$ является дополнительным кодом делителя B, у которого предварительно изменили знак. Таким образом, здесь представлена удобная для схемотехнической реализации операция вычитания

$$R_1 = 2|A| - B;$$

Проводим анализ знака первого остатка R_1. Если знак первого остатка отрицателен ($R_1 < 0$), то операция деления выполнима, иначе возникает переполнение. Если $R_1 < 0$, то старший разряд частного C_1 равен 0 ($C_1 = 0$).

- Следующие остатки R_i находятся следующим образом

$$R_{i+1} = \begin{cases} 2R_i + |B| & R_i < 0; \\ 2R_i + [\overline{B}] & R_i \geq 0. \end{cases}$$

- Знак остатка R_i определяет цифры в двоичных разрядах частного C_i, начиная со старшего разряда.

$$C_i = \begin{cases} 0 & R_i < 0; \\ 1 & R_1 \geq 0. \end{cases}$$

Процесс деления в АЛУ сопровождается сдвигом в регистре делимого A на один разряд влево на каждом шаге алгоритма. Поэтому регистр для делимого A имеет двойную длину.

Пример:

Провести деление числа $A = 49_{10}$. $[A] = 0{,}0110001_2$ на число $B = -7_{10}$.

$[B] = 1{,}111$. $[-\overline{B}] = 1{,}001$.

Вычисляем $2|A| = 0{,}1100010$. Это сдвиг делимого в регистре на один разряд влево. Далее вычисляем R_1

$$\begin{array}{r} 2|A| = 0{,}1100010 \\ +\ [\overline{B}] = 1{,}001 \\ \hline R_1 = 1{,}1110010 \end{array}$$

В старшем знаковом разряде R_1 находится 1. Это значит, что первый остаток отрицательный и деление возможно. Старший разряд частного C_1 равен 0 ($C_1 = 0$).

Далее, на следующем шаге алгоритма, сдвигаем первый остаток R_1 на один разряд влево (обозначаем $2R_1$) и складываем с модулем прямого кода делителя $|B|$.

$$\begin{array}{r} 2R_1 = 1{,}1100100 \\ +\ |B| = 0{,}111 \\ \hline R_2 = 0{,}1010100 \end{array}$$

Второй остаток R_2 положителен, поэтому следующий разряд частного $C_2 = 1$.

Далее, на следующем шаге алгоритма, сдвигаем второй остаток R_2 на один разряд влево (обозначаем $2R_2$) и складываем с числом –B, представленным в дополнительном коде $[-\overline{B}]$.

$$\begin{array}{r} +\ 2R_2 = 1{,}0101000 \\ [-\overline{B}] = 1{,}001 \\ \hline R_3 = 0{,}0111000 \end{array}$$

Третий остаток R_3 положителен, поэтому следующий разряд частного $C_3 = 1$.

Далее, на следующем шаге алгоритма, сдвигаем третий остаток R_3 на один разряд влево (обозначаем $2R_3$) и складываем с числом –B, представленным в дополнительном коде $[-\overline{B}]$.

$$\begin{array}{r} +\ 2R_3 = 0{,}1110000 \\ [-\overline{B}] = 1{,}001 \\ R_4 = 0{,}0000000 \end{array}$$

Четвертый остаток R_3 равен нулю, поэтому следующий разряд частного $C_4 = 1$. Алгоритм деления по методу без восстановления остатка закончен.

В результате получили модуль частного $|C| = 0{,}1110$.

Знак частного $A[7] \oplus B[7] = 0 \oplus 1 = 1$. Знак частного – отрицательный.

Поэтому частное от деления $C = 1{,}1110_2 = -7_{10}$.

7.6.2. Функциональная схема АЛУ для деления целых чисел.

Функциональная схема АЛУ для деления чисел с фиксированной запятой, использующая метод деления без восстановления остатка, представлена на следующем рисунке.

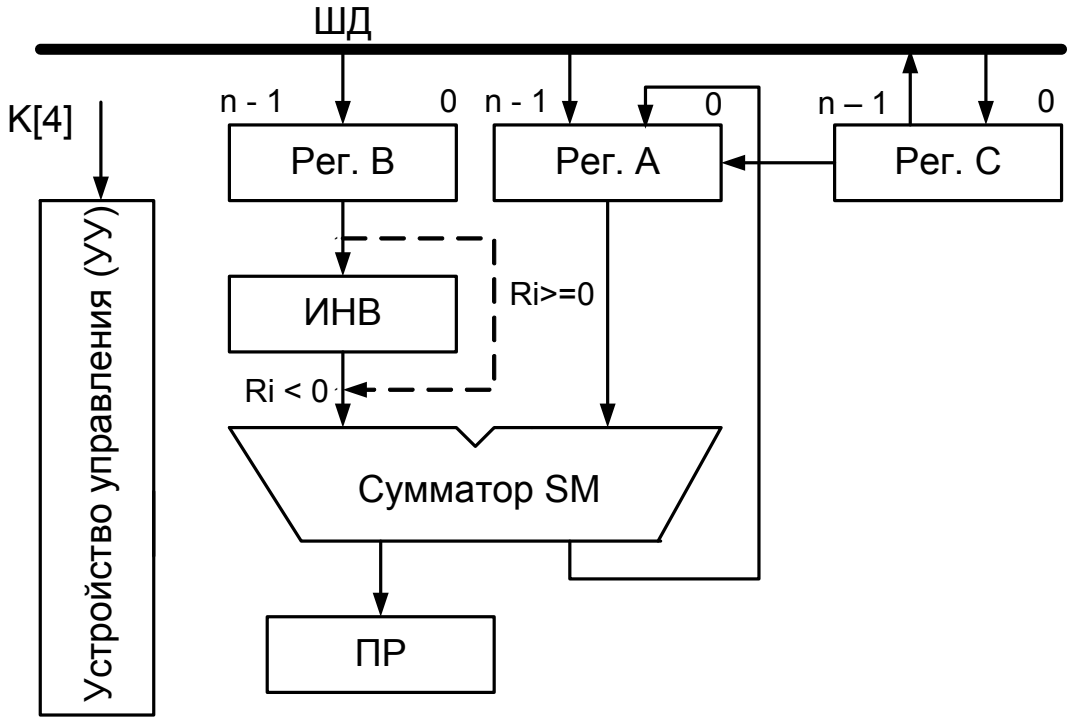


Рис. 7.7. АЛУ для деления чисел с фиксированной запятой

Делимое А записывается в прямом коде длиной 2n в регистры А и С, делитель В записывается в прямом коде в регистр В. В счетчик количества циклов СТ (в УУ) записывается число n – 1. Сумматор SM (n разрядов) проводит сложение 8-ми разрядных чисел и передает результат в регистр А. Блок инвертирования ИНВ передает прямой код делителя В, если остаток $R_i<0$, иначе преобразует прямой код делителя в дополнительный код. Признаки результата деления вырабатывает блок ПР.

После записи делимого и делителя в регистры проводится анализ на равенство чисел нулю, а затем вырабатываются соответствующие признаки результата.

Проверяется возможность деления чисел. Для этого содержимое регистров A и C сдвигается влево на один разряд, а затем в сумматоре SM к содержимому регистра A прибавляется прямой код делителя B и результат записывается в регистр A. Остаток находится в регистрах A и C. Если остаток имеет отрицательный знак, то деление продолжается.

В освободившийся после сдвига младший разряд регистра C записывается старший разряд частного C_1.

Таким образом, в регистрах A и C оказывается записанным первый остаток R_1 от деления числа A на число B.

Если деление возможно, то проводится следующий цикл операций:

- Содержимое регистров A и C сдвигается влево на один разряд, а затем к содержимому регистра A прибавляется прямой код делителя B ($R_i < 0$) или дополнительный код делителя $[\overline{B}]$ ($R_i \geq 0$). В освободившийся в результате сдвига младший разряд регистра C записывается очередной разряд частного.

Цикл заканчивается после того, как в счетчике количества циклов CT окажется ноль.

В результате деления частное C находится в регистре C и может быть передано на шину данных ШД.

7.7. Матричные умножители

Матричная схема умножения чисел соответствует математической структуре операции умножения столбиком. Такая схема позволяет уменьшить время, необходимое для выполнения операции умножения.

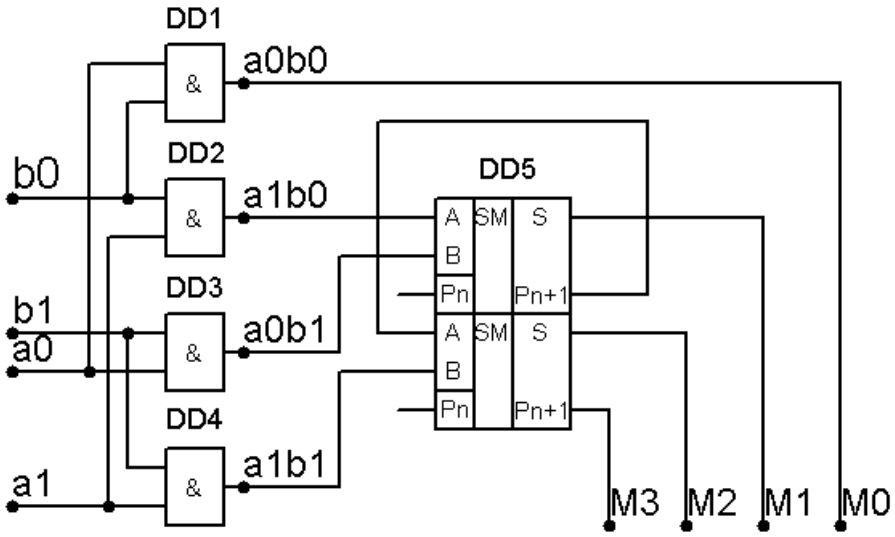
Перемножим два четырехразрядных числа A = a_3, a_2, a_1, a_0 и B = b_3, b_2, b_1, b_0 столбиком.

			x	a3	a2	a1	a0
				b3	b2	b1	b0
+				a3b0	a2b0	a1b0	a0b0
			a3b1	a2b1	a1b1	a0b1	
		a3b2	a2b2	a1b2	a0b2		
	a3b3	a2b3	a1b3	a0b3			
M7	M6	M5	M4	M3	M2	M1	M0

Произведение П = AB = M7, M6, M5, M4, M3, M2, M1, M0.

Процесс матричного умножения выполняется быстрее, чем умножение со сдвигом числа в регистрах, и схемотехнически может быть реализован на логических элементах без использования регистров сдвига. Двоичные произведения a_i b_j схемотехнически реализуется на двухвходовом логическом элементе "И". Суммирование по столбцам может быть реализовано при помощи обычных комбинационных сумматоров.

Схема двухразрядного матричного умножителя, выполненного на микросхемах средней степени интеграции, представлена на следующем рисунке



DD1 – DD4 – логические элементы КР****ЛИ1

DD5 – сумматор КР****ИМ5

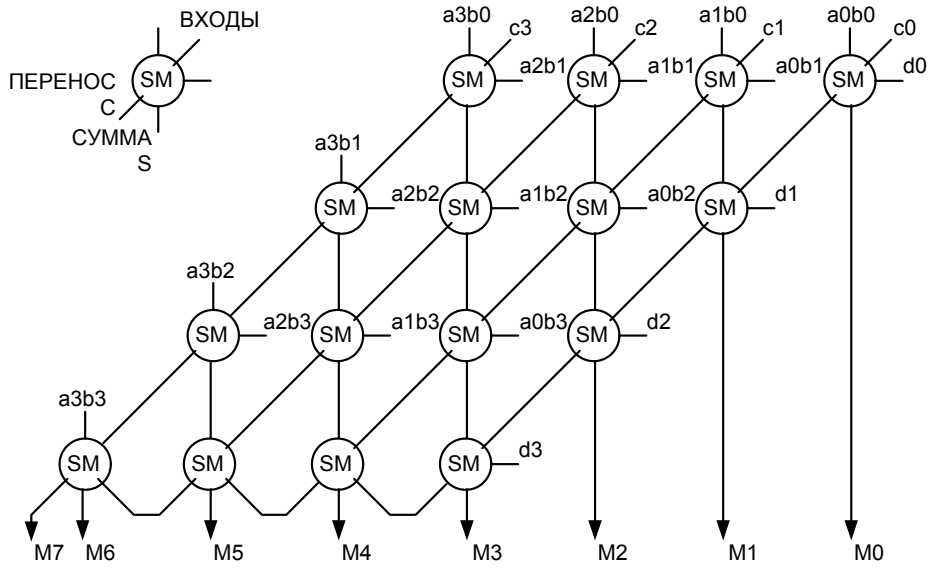
Рис. 7.8. Двухразрядный матричный умножитель

На вход умножителя подаются числа A = a_1, a_0 и B = b_1, b_0. На выходах логических элементов появляются парные произведения.

Суммирование проводится в двух одноразрядных сумматорах. Результат умножения появляются на выходах M3, M2, M1 и M0.

Организация четырёхразрядного матричного умножителя представлена на следующем рисунке

а)



б)

Рис. 7.9. Четырехразрядный матричный умножитель *а*) – одноразрядный сумматор, *б*) – схема множительно-суммирующего блока

Двухвходовые логические элементы "И" на этой схеме не показаны. Их выходы подключены к входам сумматоров. Частичные суммы произведений суммируются в вертикальных столбцах, по наклонным линиям передаются переносы. Входы d_i справа служат для подключения аналогичной схемы при расширении разрядности.

Время умножения определяется задержкой сигнала в двухвходовых логических элементах "И" и задержкой сигнала в наиболее длинной цепочке одноразрядных сумматоров.

7.8. Двоично – десятичные сумматоры

Двоично-десятичные сумматоры позволяют проводить операции непосредственно с кодами десятичных чисел, что в некоторых случаях позволяет сократить затраты и упростить схему вычислительного устройства.

Каждое десятичное число кодируется четырьмя разрядами двоичного кода. Поэтому за основу двоично-десятичных сумматоров принимаются обычные четырехразрядные двоичные сумматоры, по одному на каждый десятичный разряд.

При сложении четырехразрядных кодов десятичных цифр возможна ситуация, когда сумма двух цифр больше девяти. В этом случае необходимо провести коррекцию и правильно сформировать перенос в старший разряд.

В следующей таблице представлены двоичные и двоично-десятичные коды чисел. Тетрада 1 соответствует старшему десятичному разряду числа, тетрада 2 – младшему.

Таблица 7.3 – Двоично – десятичная арифметика

N дес	Двоичный код					Тетрада 1				Тетрада 2					
	a_4	a_3	a_2	a_1	a_0	b_3	b_2	b_1	b_0	P_2	c_3	c_2	c_1	c_0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	$0 \le N \le 9$ Коррекция не требуется
1	0	0	0	0	1	0	0	0	0	0	0	0	0	1	
2	0	0	0	1	0	0	0	0	0	0	0	0	1	0	
…		…	…	…	…	…	…	…	…	…	…	…	…	…	
9	0	1	0	0	1	0	0	0	0	0	1	0	0	1	
10	0	1	0	1	0	0	0	0	1	0	0	0	0	0	$10 \le N \le 15$ Коррекция: вычесть 10 и перенос
11	0	1	0	1	1	0	0	0	1	0	0	0	0	1	
…		…	…	…	…	…	…	…	…	0	…	…	…	…	
14	0	1	1	1	0	0	0	0	1	0	0	1	0	0	
15	0	1	1	1	1	0	0	0	1	0	0	1	0	1	
16	1	0	0	0	0	0	0	0	1	1	0	1	1	0	$16 \le N \le 19$ Коррекция: прибавить 6 и перенос
17	1	0	0	1	1	0	0	0	1	1	0	1	1	1	
18	1	0	1	0	0	0	0	0	1	1	1	0	0	0	
19	1	0	1	1	1	0	0	0	1	1	1	0	0	1	

Для первых девяти чисел двоичный код числа соответствует коду в тетраде 2. Поэтому коррекция не требуется.

Для группы чисел в диапазоне $10 \le N \le 15$ из двоичного кода числа необходимо вычесть 10 и сформировать перенос в старший десятичный разряд. Операция вычитания заменяется операцией прибавления числа 10 в дополнительном коде. Поэтому к числу, лежащему в этом диапазоне, необходимо прибавить число $6_{10} = 0110_2$ и сформировать перенос.

Для группы чисел в диапазоне $16 \leq N \leq 19$ к двоичному коду числа необходимо прибавить число $6_{10} = 0110_2$ и сформировать перенос.

При реализации двоично-десятичной арифметики на стандартных сумматорах (рис. 7.10) следует учесть, что, начиная с числа N = 16, происходит перенос единицы в 5-й разряд (формируется P = 1 на выходе переноса сумматора). Поэтому логическую функцию коррекции необходимо формировать только для чисел, лежащих в диапазоне $10 \leq N \leq 15$. Эта функция имеет следующий вид

$$F = a_3 \cdot (a_1 + a_2) + P;$$

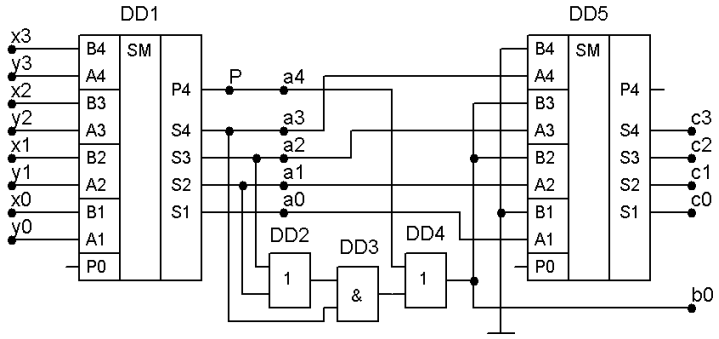


Рис. 7.10. Одноразрядный двоично-десятичный сумматор

Одноразрядный двоично-десятичный сумматор выполнен на паре двоичных сумматоров типа КР***ИМ6. Первый сумматор DD1 формирует сумму кодов десятичных чисел X = x_3, x_2, x_1, x_0 и Y = y_3, y_2, y_1, y_0. В результате получается двоичный код суммы A = a_4, a_3, a_2, a_1, a_0 (разряд a_4 получается на выходе переноса P).

Логическая функция коррекции F формируется логическими элементами DD2, DD3 и DD4. Если на выходе DD4 возникает F = 1, то вторые входы второго сумматора DD5 подается двоичное число $6_{10} = 0110_2$. Сигнал на выходе b_0 является переносом в старший десятичный разряд. В результате на выходах DD5 появляется скорректированный код младшего разряда десятичного числа C = c_3, c_2, c_1, c_0.

Если F = 0, то на выход DD5 передается код с выходов DD1 без коррекции.

7.9. АЛУ для выполнения логических операций.

АЛУ для выполнения логических операций. Каждая логическая операция имеет свой код, после получения которого АЛУ настраивается на выполнение этой операции.

Запишем список основных логических операций:

$y_0 := F_0(X_i \vee Y_i)$	**ИЛИ(OR)**
$y_1 := F_1(\overline{X_i \vee Y_i})$	**ИЛИ-НЕ(NOR)**
$y_2 := F_2(X_i \wedge Y_i)$	**И(AND)**
$y_3 := F_3(\overline{X_i \wedge Y_i})$	**И-НЕ(NAND)**
$y_4 := F_4(X_i \oplus Y_i)$	**(XOR)**
$y_5 := F_5(\overline{X_i \oplus Y_i})$	**XNOR**

Функциональная схема АЛУ представлена на следующем рисунке

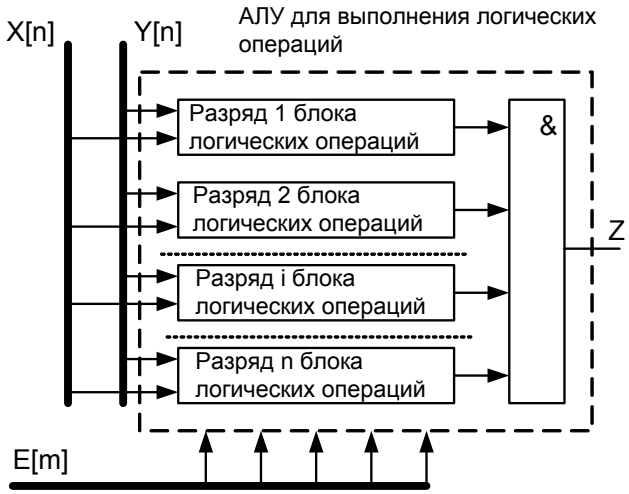


Рис. 7.11. Функциональная схема АЛУ для выполнения логических операций

АЛУ содержит n блоков поразрядных логических операций. На вход каждого блока подается соответствующие разряды двух входных операндов X и Y. На управляющие входы АЛУ подается вектор E, выбирающий одну из логических операций, выполняемых над операндами X и Y. Результат вычислений поступает на выход АЛУ Z.

Схема поразрядного блока логических операций, выполняющего перечисленные логические операции, показана на следующем рисунке

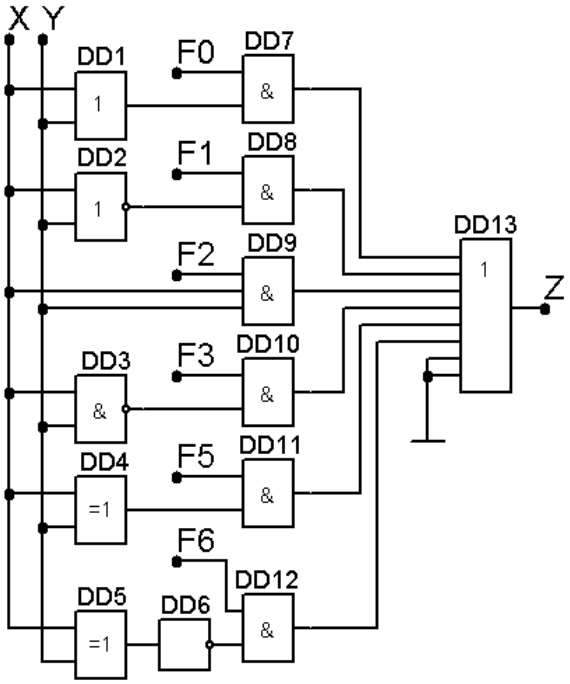


Рис. 7.12. Схема поразрядного блока логических операций

На входы X и Y подаются одинаковые разряды входных операндов. Вид выполняемой над ними логической операции выбирается путем подачи лог. 1 на соответствующий вход F_i. Результат логической операции появляется на выходе Z.

7.10. Микросхемы АЛУ

Промышленность выпускает несколько типов АЛУ в различных сериях микросхем средней степени интеграции. Рассмотрим микросхему АЛУ типа КР****ИП3(зарубежный аналог LS181).

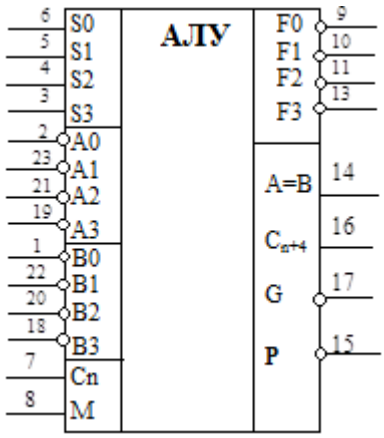


Рис. 7.13. Условное графическое обозначение АЛУ ИП3

Микросхема ИП3 предназначена для действий с двумя четырехразрядными словами *A=A3A2A1A0* и *B=B3B2B1B0*.

Конкретный вид операции, выполняемой микросхемой, задается *5*-разрядным кодом на входах *MS3S2S1S0*. Всего это АЛУ способно выполнить 32 операции (2^5=32):

- *16* логических (*И, И-НЕ, ИЛИ, ИЛИ-НЕ*, *исключающее ИЛИ* и др.) при M=1;
- *16* арифметико-логических (сложение, вычитание, удвоение, сравнение чисел и ряд иных) при M=0.

При выполнении логических операций внутренние переносы запрещаются.

Операции сложения и вычитания проводятся с ускоренным переносом из разряда в разряд. Кроме того, имеется вход приема сигнала переноса *Cn*.

На выходах *F3,F2,F2* и *F0* формируются результаты логических преобразований и арифметических действий. На выходе переноса *Cn+4* образуется сигнал старшего (пятого) разряда при выполнении арифметических операций.

Дополнительные выходы — образование ускоренного переноса G и распространение ускоренного переноса *P* — используются только при организации многоразрядных *АЛУ* в случае их сочетания с блоком ускоренного переноса *К155ИП4* (или *564ИП4* для микросхем *КМОП*), о чем будет сказано ниже.

Все виды операций и результаты вычислений применительно к положительной логике сведены в таблицу 7.4.

Таблица 7.4 - Функциональная зависимость выходов микросхемы ИПЗ от состояния Входов (положительная логика)

Выбор функции S3 S2 S1 S0	ВЫХОД-Логические операции (на входе M=1)	ВЫХОД-Логико-Арифметические операции (на входе M=0)
0 0 0 1	$\overline{A \vee B}$	$A \vee B$
0 0 1 0	$\overline{A}B$	$A \vee \overline{B}$
0 0 1 1	Логический 0	Минус 1
0 1 0 0	$\overline{AB}$	A плюс $A\overline{B}$
0 1 0 1	$\overline{B}$	$(A \vee B)$ плюс $A\overline{B}$
0 1 1 0	$A \oplus B$	A минус B минус 1
0 1 1 1	$A\overline{B}$	$A\overline{B}$ минус 1
1 0 0 0	$\overline{A} \vee B$	A плюс AB
1 0 0 1	$\overline{A \oplus B}$	A плюс B
1 0 1 0	B	$(A \vee \overline{B})$плюс AB
1 0 1 1	AB	AB минус 1
1 1 0 0	Логическая 1	A + A
1 1 0 1	$A \vee \overline{B}$	$(A \vee B)$плюс A
1 1 1 0	$A \vee B$	$(A \vee \overline{B})$плюс A
1 1 1 1	A	A минус 1

В таблице 7.4 предполагается, что арифметические операции выполняются в дополнительном коде. Как отмечалось, числа в

дополнительном и в обратном коде связаны простым соотношением $N_{доп} = N_{обр}+1$ или $N_{обр}=N_{доп} - 1$. Поэтому в тех строках таблицы 7.4, где указана операция «минус 1», результат арифметических действий представлен в обратном коде.

Старший разряд кода выбора операций (вход *M*) определяет характер действий, выполняемых *АЛУ*. Когда на этом входе сигнал высокого уровня(M=1), *АЛУ* производит логические операции поразрядно над каждой парой бит слов *A* и *B*. Внутренний перенос в этом режиме бездействует.

Если *АЛУ* выполняет логико-арифметическую операцию(M=0), логическая функция реализуется поразрядно, а арифметическая с переносом.

Например, входному коду $MS3S2S1S0=01101_2$ отвечает операция $(A \vee B)$ плюс *A* (третья снизу строка таблицы 7.4). Первой выполняется операция в скобках - $(A \vee B)$ - логическое сложение двух слов. Если $A=1010_2$ $B=0111_2$, то первая операция дает $(A \vee B)=1111_2$.Второй выполняется операция арифметического сложения числа A с результатом логического сложения. Следовательно, 1111_2 плюс $1010_2=11111_2$.

При использовании АЛУ в качестве компаратора сигнал снимают с входа *A=B* (вывод 14). Этот выход — с открытым коллектором, и к источнику питания его следует подключать через внешний резистор *1 кОм*.

Режим компаратора обеспечивается при *M=1* и $S3S2S1S0=0110_2$. Когда числа *A* и *B* равны, на входе *A=B* формируется сигнал высокого уровня.

Одновременно на выходе *Cn+4* (вывод 16) характеризует соотношение между числами *A* и *B* и в случае их неравенства, согласно таблицы 7.5.

Для арифметических действий над словами большей длины *АЛУ* включают последовательно. В этом случае время суммирования определяется задержкой распространения сигнала переноса со входа младшего разряда до выхода с последнего АЛУ и составляет $t_{зд.р}=4\tau_{зд.р}$, где τ — задержка распространения сигнала переноса в одной АЛУ.

Таблица 7.5 - Таблица истинности микросхемы К155ИП3 в режиме четырёх разрядного компаратора (S3=0, S2=1, S1=1, S0=0)

Вид логики	Состояние входов	Состояние выхода Cn+4
Cn	A и B	
Положительная логика		A≤B
	A<B	
	A>B	
	A≥B	

Уменьшить время суммирования можно применением микросхем К155ИП4 *(564ИП4, LS 182*), специально разработанных для организации ускоренного переноса между отдельными *АЛУ*, а также между группами АЛУ. Со схемой ускоренного переноса время суммирования сокращается примерно до $\tau_{зд.р}$

Если при выполнении арифметических операций к быстродействию не предъявляется высоких требований, то при каскадировании АЛУ схемы ускоренного переноса не используют.

7.11. АЛУ для сложения и вычитания чисел с плавающей запятой

Представление числа Z в компьютере в формате с плавающей запятой имеет следующий вид

$$Z = q^N \cdot M$$

В этой формуле: q – основание системы счисления (обычно 2),

N – порядок (целое число длиной k + 1 со знаком), M – мантисса числа (правильная дробь со знаком длиной r + 1).

Мантисса числа называется нормализованной, если в старшем разряде находится единица.

Формат числа с плавающей запятой имеет следующий вид

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
±	n4	n3	n2	n1	n0	±	m8	m7	m6	m5	m4	m3	m2	m1	m0
Порядок						Мантисса									

Мантисса имеет 9 двоичных разрядов, порядок – 5 двоичных разрядов. В 15 и 9 разрядах записываются знак порядка и мантиссы. Запятая неявно расположена перед 8-м разрядом.

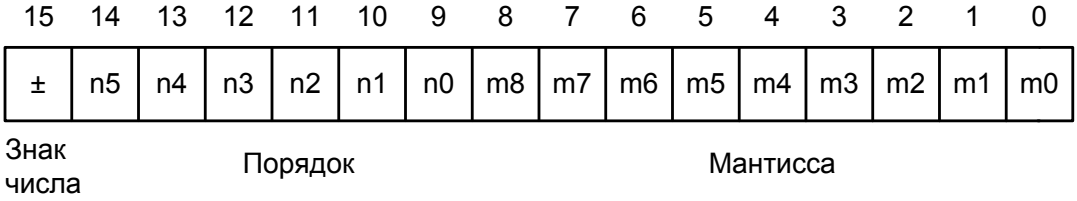
В современных компьютерах используются представления чисел в формате с плавающей запятой, содержащим 32, 64 или 80 двоичных разрядов. Увеличение числа двоичных разрядов в формате числа с плавающей запятой повышает точность вычислений.

Поскольку арифметические действия над числами с плавающей запятой требуют отдельных операций над мантиссами и порядками, то для управления операцией над порядками, операции проводятся над целыми положительными числами, применяя представление чисел со смещенными порядками. Для этого при записи числа в память к его порядку N прибавляется целое число – смещение $S = 2^k$, где k – число двоичных разрядов, используемый для расположения модуля порядка в разрядной сетке, тогда смещенный порядок $N_c = N + S$ и, поэтому всегда будет положительным. Тогда для его представления необходимо такое же количество разрядов, как и для модуля порядка $|N|$.

Свойства смещенного порядка. Если $N' > N''$, то и $N'_c > N''_c$. Следовательно, смещение порядков не влияет на операции с числами.

Сложение и вычитание чисел с плавающей запятой производится по следующей формуле

$$x + y = q^{Nx} \cdot M_x + q^{Ny} \cdot M_y = q^{Nx}(M_x + M_y) \cdot q^{-(Nx - Ny)};$$



Алгоритм сложения и вычитания чисел с плавающей запятой

- Производится выравнивание порядков чисел. Порядок меньшего по модулю числа принимается равным порядку большего числа, а мантисса меньшего числа сдвигается вправо на число разрядов, равное разности порядков чисел.
- Производится сложение или вычитание мантисс;
- Порядок суммы или разности принимается равным порядку большего числа;
- Полученная сумма/разность нормализуется. В зависимости от типа ЭВМ арифметических операций над мантиссами и порядками выполняются либо отдельными узлами АЛУ, либо последовательно одним узлом.

Операция сложения и вычитание состоит из следующих этапов.

- Прием операндов;
- Выравнивание порядков со сдвигом мантисс;
- Сложение мантисс (аналогично сложению целых чисел);
- Нормализация результата.

Функциональная схема АЛУ для арифметических операций с плавающей запятой приведена на следующем рисунке

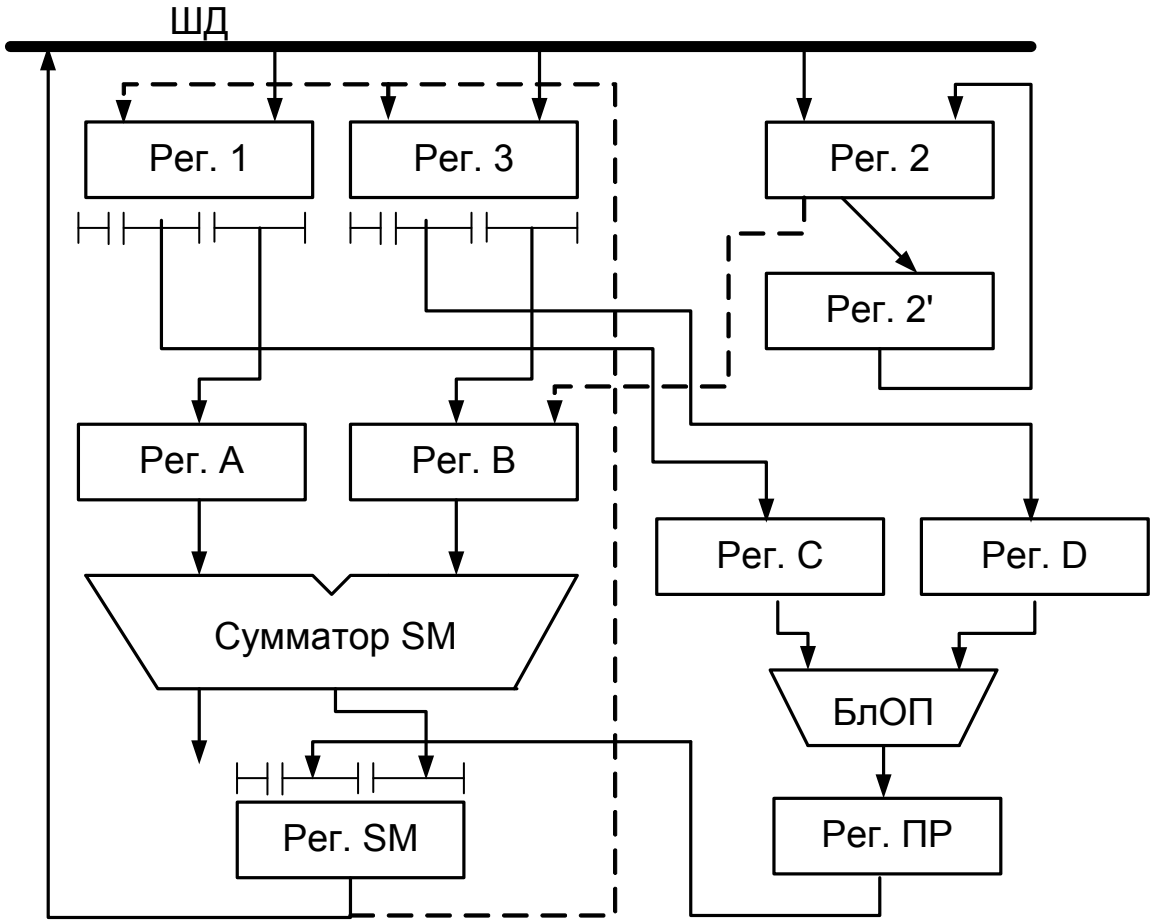


Рис. 7.14. Функциональная схема АЛУ для выполнения арифметических операций с плавающей запятой

АЛУ в режиме сложения работает следующим образом:

1. Запись исходных данных. В Рег. 1 и Рег. 3 с шины записываются слагаемые A и B. Знаки слагаемых запоминаются в специальных триггерах, находящихся в УУ.

2. Выравнивание порядков чисел. Порядки чисел из Рег. 1 и Рег. 3 записываются в регистры Рег. C и Рег. D блока обработки порядков БлОП. В этом блоке проводится сравнение этих чисел, затем мантисса числа с меньшим порядком сдвигается вправо в регистре Рег. 1 или Рег. 3 на величину разности разрядов, при этом порядки чисел выравниваются, а сумме присваивается порядок большего числа.

В зависимости от разности порядков возможны следующие ситуации:

1. $N_x - N_y > m$ (m – число разрядов мантиссы).

В этом случае в регистре мантиссы числа M_y после сдвига окажутся нули. Сумме в этом случае присваивается значения мантиссы числа M_x.

2. $N_y - N_x > m$.

В этом случае в регистре мантиссы числа M_x после сдвига окажутся нули. Сумме в этом случае присваивается значения мантиссы числа M_y.

3. $N_y = N_x$ (порядки двух чисел одинаковы).

Мантиссы складываются без сдвига.

4. $N_x - N_y = p < m$.

Мантисса числа M_y сдвигается на p разрядов вправо.

5. $N_y - N_x = p < m$.

Мантисса числа M_x сдвигается на p разрядов вправо.

Сдвиг мантисс проводится по следующему алгоритму. Число разрядов сдвига p записываются в счетчик. Сдвиг мантисс проводится в регистрах Рег. 1 или Рег. 3 до тех пор, пока в счетчике не окажется ноль. После этого в этих регистра хранятся сдвинутые мантиссы, а общий порядок – в регистре Рег. ПР.

Сложение мантисс имеют следующие особенности.

- При одинаковых знаках слагаемых модули мантисс складываются в сумматоре SM. Если окажется, что в старшем разряде суммы находится единица, то возникает переполнение, сумма сдвигается вправо, а порядок увеличивается на 1.

Если в старшем разряде порядка появляется единица, то возникает переполнение порядка. В этом случае выдается сигнал прерывания для аварийной остановки программы.

- При разных знаках слагаемых отрицательное слагаемое передается в регистры Рег. А или Рег. В в обратном коде. После суммирования к сумме добавляется единица.

Нормализация результата проводится аналогично. Особенностью этого случая является возможность исчезновения смещенного порядка. В этом случае результату присваивается ноль.

Замечание. Операции с плавающей запятой являются приближенными.

7.12. АЛУ для умножения чисел с плавающей запятой

Умножение двух чисел с плавающей запятой проводится по следующей формуле

$$x \cdot y = M_x \cdot M_y q^{N_x + N_y};$$

Алгоритм умножения чисел с плавающей запятой

- Производится сложение порядков чисел;
- Производится умножение мантисс;
- Полученное произведение нормализуется.
- Проверяются условия переполнения или исчезновения порядков. Если возникло переполнение порядка, то оно может исчезнуть после нормализации мантиссы.

Блоки, необходимые для умножения чисел с плавающей запятой, включены в состав схемы АЛУ, представленной на рис. 7.14.

1. Запись исходных данных. В Рег. 1 с шины записывается множимое, в Рег. 2 записывается множимое. Знаки слагаемых запоминаются в специальных триггерах, находящихся в УУ.

2. Пара регистров Рег. 2 и Рег. 2' служат для организации сдвига разрядов множителя в процессе умножения.

3. Мантисса множимого передается в сумматор SM из регистра Рег. А. Регистр Рег. В используется для хранения сумм частичных произведений в процессе последовательного сложения с множимым.

4. Порядки чисел складывается в блоке БлОП, знак числа вычисляется отдельно. Проводится анализ на переполнение.

5. Мантисса произведения нормализуется с коррекцией порядка.

7.13. АЛУ для деления чисел с плавающей запятой

Деление двух чисел с плавающей запятой проводится по следующей формуле

$$x / y = \frac{M_x}{M_y} q^{N_x - N_y};$$

Алгоритм деления чисел с плавающей запятой

- Производится вычитание порядков чисел;
- Производится деление мантисс;
- Полученное частное нормализуется.
- Проверяются условия переполнения или исчезновения порядков. Если возникло переполнение порядка, то оно может исчезнуть после нормализации мантиссы. Знак частного вычисляется отдельно. При вычитании порядков смещение пропадает, поэтому порядок необходимо скорректировать.

Блоки, необходимые для деления чисел с плавающей запятой, включены в состав схемы АЛУ, представленной на рис. 7.14.

1. Запись исходных данных. В Рег. 1 с шины записывается делимое, в Рег. 2 записывается делитель. Знаки слагаемых запоминаются в специальных триггерах, находящихся в УУ.

2. Пара регистров Рег. 2 и Рег. 2' служат для организации хранения и сдвига разрядов частного в процессе деления.

3. Мантисса делителя передается в сумматор SM из регистра Рег. А. Регистр Рег. В используется для хранения делителя.

4. Порядки чисел вычитаются и корректируются в блоке БлОП, знак числа вычисляется отдельно. Проводится анализ на переполнение.

5. Мантисса частного нормализуется