МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное
учреждения высшего образования
«Юго-Западный государственный университет»
(ЮЗГУ)

Кафедра информационной безопасности

УТВЕРЖДАЮ
Проректор по учебной работе
_____ О.Г. Локтионова
«_1_» _02_ 2018 г.

МОДЕЛИРОВАНИЕ РАБОТЫ ОПЕРАЦИОННОГО АВТОМАТА, ВЫПОЛНЯЮЩЕГО ОПЕРАЦИЮ СУММИРОВАНИЯ В ДОПОЛНИТЕЛЬНО И ОБРАТНОМ КОДЕ

Методические рекомендации для лабораторной работы №1
для студентов укрупненной группы специальностей и
направлений подготовки 10.00.00 «Информационная безопасность»

Курск 2018

УДК 004
Составитель: С.С. Шевелев

Рецензент

Кандидат технических наук, доцент кафедры
«Информационная безопасность» А.Л. Марухленко

Моделирование работы операционного автомата, выполняющего операцию суммирования в дополнительном и обратном коде [Текст] : методические рекомендации для лабораторной работы №1 по дисциплине «Организация ЭВМ и вычислительных систем» / Юго-Зап. гос. ун-т; сост.: С.С. Шевелев – Курск, 2018. – 28 с.: ил.10. – Библиогр.: с. 9.

Содержат сведения по вопросам моделирования работы операционного автомата, выполняющего операцию суммирования в дополнительном и обратном коде. Указывается порядок выполнения лабораторной работы, правила оформления отчета.

Методические рекомендации соответствуют требованиям программы, утвержденной учебно-методическим объединением по специальности.

Предназначены для студентов укрупненной группы специальностей и направлений подготовки 10.00.00 «Информационная безопасность».

Текст печатается в авторской редакции

Подписано в печать 1.02.18. Формат 60х84 1/16.
Усл.печ. л. 1,63. Уч.-изд. л. 1,47. Тираж 100 экз. Заказ. Бесплатно. 229
Юго-Западный государственный университет.
305040, г. Курск, ул. 50 лет Октября, 94.

Цель работы: изучить структуру операционного автомата, на основе данной структуры создать модель операционного автомата, выполняющего операции суммирование чисел в прямом, дополнительном и обратном коде.

Задача: По представленным блок-схемам алгоритмов: сложение чисел с фиксированной запятой в дополнительном и обратном коде, протестировать программу на языке высокого уровня.

1. Теоретическая часть.

1.1 Форматы представления данных и расположение информации в оперативной памяти ЭВМ.

Наименьшая единица информации – двоичный разряд (0 или 1), получивший название бит. Группа двоичных разрядов, изображающих символ в ЭВМ, называется *слогом*. Для представления символов в ЭВМ используется 8-разрядный слог, называемый *байтом* (рис.1).

В современных ЭВМ наименьшей адресуемой структурной единицей информации принят байт и байтовая организация информации в оперативной памяти (ОП).

Для представления алфавитно-цифровой информации в ЭВМ обычно используется *машинное слово* – совокупность символов, которая считывается из ОП или записывается в нее за одно обращение. Обычно машинное слово содержит целое число байтов. Байты адресуются последовательно, начиная с нуля.

Нумерация битов операнда производится слева направо, начиная с нулевого разряда (рис.2). Каждому байту обычно придается 9-й контрольный разряд К, называемый разрядом четности. В 9-м разряде ставится 1, дополняющая количество единиц в байте до нечетного.

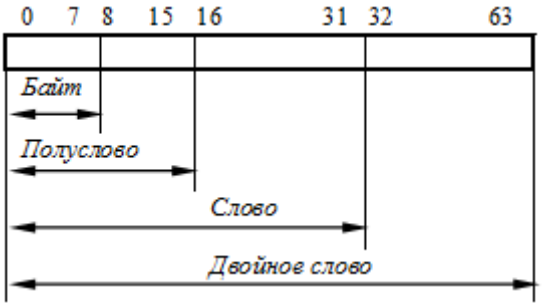


Рис. 1 – Форматы представления данных в ЭВМ

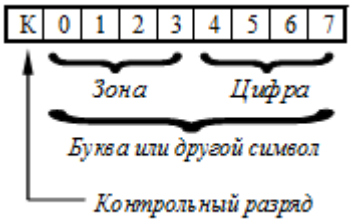


Рис. 2 – Разрядная сетка байта

Если в процессе обработки информации в байте окажется четное количество единиц, то будет зафиксирован машинный сбой и выдан сигнал обнаружения ошибки. Количество контрольных разрядов К в слове равно количеству содержащихся в нем байтов.

Адресом операнда считают адрес левого байта. Адрес (граница) операнда с фиксированной длиной всегда должен быть кратен количеству содержащихся в нем байтов: адрес полуслова – двум (например, 0010, 0100, 0110 и т.д.); слова – четырем(0100, 1000, 1100 и т.д.); двойного слова – восьми (1000, 10000, 11000 и т.д.). Такое ограничение дает возможность упростить процедуру обращения к оперативной памяти. При задании адреса операнда фиксированной длины указывается адрес левого (старшего) байта, а длина операнда определяется кодом операции в команде.

Адрес поля, т.е. адрес левого байта, может быть произвольным. Поля различной длины располагаются в ЭВМ друг за другом в порядке возрастания их адресов. Количество составляющих поле байтов задается в команде.

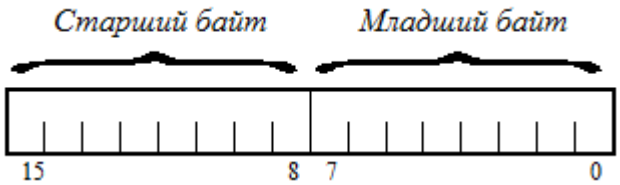


Рис. 3 – Формат данных в ЭВМ

1.2. Формы представления чисел в ЭВМ

При размещении обрабатываемой информации в ЭВМ следует учитывать необходимость контроля ее обработки и адресации в ячейки ОП. Это вызывает определенные требования к организации разрядной сетки ЭВМ. Под разрядной сеткой ЭВМ понимают количество разрядов, необходимых для размещения в ячейках оперативной памяти полного машинного слова. Для каждого типа ЭВМ она имеет строго определенное количество разрядов.

В ЭВМ используют две формы представления чисел в разрядной сетке: с фиксированной и плавающей запятой.

Представление чисел с фиксированной запятой. Число с фиксированной запятой – естественная форма представления числа, когда положение запятой в разрядной сетке строго фиксируется. Обычно она фиксируется перед старшим или после младшего разрядов. Если запятая фиксируется перед старшим разрядом, то числа в ЭВМ представляются как правильные дроби; если после младшего – как целые числа. ЭВМ, в которых используется такая форма записи чисел, называют машинами с фиксированной запятой.

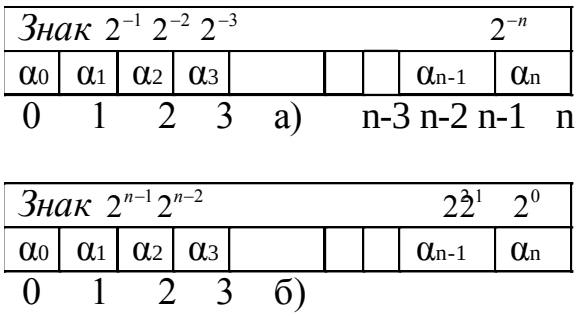


Рис.4 Схемы разрядной сетки ЭВМ с фиксированной запятой

На рис.4, *а* приведена разрядная сетка ЭВМ для представления чисел с фиксированной запятой перед старшим разрядом.

Разряды такой сетки нумеруются слева направо, начиная с нулевого, который называется знаковым разрядом. В этом разряде 0 соответствует плюсу, а 1 – минусу. На разрядной сетке указан вес каждого разряда. Максимальное машинное число по абсолютной величине, т.е. без учета знака, равно

$$|X|_{макс} = ,\overbrace{1111...11}^{n} = 1 - 2^{-n}, \qquad (1)$$

где n – количество разрядов числа.

Минимальное, отличное от нуля машинное число

$$|X|_{мин} =, \overbrace{0000...01}^{n} = 2^{-n}. \qquad (2)$$

Диапазон чисел всех возможных величин в данном случае определяется неравенством: $2^n \le |X| \le (1-2^{-n})$. (3)

Анализируя это неравенство, можно отметить следующее:

1.Диапазон представления чисел в машинах с фиксированной запятой сравнительно невелик.

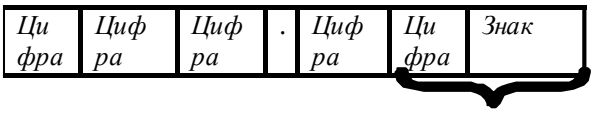
2.Число, абсолютное значение которого меньше минимального машинного слова (2^{-n}), будет записано в ЭВМ в виде нуля. Такое число называется машинным нулем, так как на самом деле оно не равно нулю, но для его изображения не достаточно разрядов в машинном слове.

3.Число, полученное в результате вычислений по абсолютному значению, не должно превышать максимального машинного числа $(1-2^{-n})$. Если число выходит за верхний предел $(1-2^{-n})$, то целая часть его не может быть расположена в машинном слове и поэтому теряется, что приводит к искажению результата. В этом случае говорят, что произошло переполнение разрядной сетки.

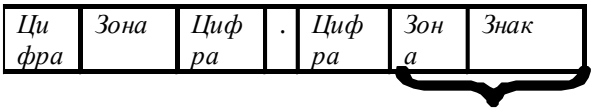
Чтобы не допустить в процессе вычислений переполнения разрядной сетки, при подготовке задачи к решению на ЭВМ вводятся так называемые масштабные коэффициенты, с помощью которых числа, участвующие в вычислениях, и результаты расчета по модулю не превышают максимального машинного числа. Масштабный коэффициент m связывает реальное истинное число Y и его машинное изображение (код) X следующим соотношением:

$$|X| = m|Y| < 1. \qquad (4)$$

Десятичные числа в ЭВМ представляются только в естественной форме. Десятичные числа кодируются тетрадами в коде 8-4-2-1 и используется два формата для их представления: упакованный (уплотненный) и распакованный (зонный). В упакованном формате в одном байте размещаются две десятичные цифры. Код знака размещается в правых четырех разрядах младшего байта (рис.5,а).



а) *Младший байт*



б) *Младший байт*

Рис.5 Форматы десятичных чисел в ЭВМ

В распакованном формате младшие четыре разряда байта заняты цифрой, а остальные разряды содержат служебный символ (зону). В младший байт записываются код знака и код младшей десятичной цифры (рис.5,б).

Десятичные числа вводятся обычно в распакованном формате, а при выполнении операций переводятся по специальной программе в упакованный формат; при выводе из ЭВМ десятичные числа вновь преобразуются в распакованный формат по специальной программе.

Упакованный формат представления десятичных чисел позволяет эффективно использовать ОП, уменьшить процесс выполнения арифметических операций и ускорить процесс обмена информацией между ОП и периферийными устройствами.

1.3. Операция сложение-вычитание чисел в современных ЭВМ.

Важнейшей функцией большинства вычислительных устройств является выполнение арифметических операций. В связи с этим в ЭВМ выделяют специальный функциональный блок — арифметическое устройство (АЛУ), предназначенный для выполнения операций над числовыми кодами Числа, участвующие в арифметических операциях, выполняемых цифровым автоматом, называются операндами. Операционный автомат, выполняющий операцию сложения двух чисел изображен на рис.9.

Для позиционных систем счисления с естественными весами все допустимые числа являются полиномами по степеням p (основания системы счисления). Следовательно, все арифметические действия в этом случае выполняются по правилам алгебраического сложения, умножения и деления полиномов.

Основной операцией в ЭВМ является операция сложения. По способу ее выполнения АЛУ могут быть параллельного, последовательного и параллельно-последовательного действия.

В АЛУ последовательного действия производится последовательное суммирование всех n разрядов a_i и b_i слагаемых A и B.

Последовательное суммирование операндов должно выполняться на основании следующего равенства:

$$A_i + B_i = П_i p + S_i \qquad (5)$$

При этом перенос Пi из разряда с номером i принимает значения

$$Пi = \begin{cases} 0, & a_i + b_i \le p - 1 \\ 1, & a_i + b_i \ge p \end{cases} \qquad (6)$$

При сложении полиномов должны суммироваться все члены с одинаковыми степенями, то (5) перепишется в виде

$$A_i + B_i + П_{i-1} = П_i p + S_i \qquad (7)$$

Если для суммы установлена та же длина слова, что и для слагаемых, то правильное представление значений суммы будет существовать только при $П_n = 0$ и для ее определения потребуется n тактов машинного времени *(n* тактов суммирования). В случае $П_n = 1$ потребуется $n + 1$ такт суммирования. Из (6) также следует, что значение старшей цифры суммы зависит от значения всех предыдущих разрядов слагаемых.

Формирование одного разряда суммы *Si* и переноса из значений цифр слагаемых и переноса с предыдущего разряда производится с помощью одноразрядного сумматора по основанию *p*.

АЛУ параллельного действия содержит параллельный сумматор, в котором операция сложения одновременно выполняется над всеми разрядами суммируемых чисел *A* и *B,*

следовательно, время выполнения операции сложения составляет один такт машинного времени.

Параллельному способу выполнения операций соответствует минимальное время сложения при максимальном объеме оборудования (требуется *n* одноразрядных сумматоров). Последовательному способу, наоборот, характерно максимальное время выполнения операции при минимальных затратах оборудования (один одноразрядный сумматор).

Арифметические устройства параллельно-последовательного действия занимают промежуточное положение между двумя первыми типами АЛУ в отношении времени выполнения операции сложения и используемого оборудования. В таком АЛУ за один такт машинного времени находится сумма *m* разрядов слагаемых (чаще всего 8-разрядных слов, которые называются байтами).

1.4. Сложение и вычитание двоичных чисел с фиксированной запятой в прямом коде.

Правила сложения чисел в прямом коде не отличаются от обычных правил сложения, т. е. если оба слагаемых имеют одинаковые знаки, то их числовые разряды складываются, а сумме приписывается знак одного из них. Если слагаемые имеют разные знаки, то из числовых разрядов большего по абсолютной величине числа вычитается меньшее, а сумме приписывается знак большего слагаемого. При этом числовые разряды кода обрабатываются отдельно от знаковых, так как последние не имеют веса.

При вычислении суммы двух чисел возможны два случая: слагаемые имеют одинаковые знаки; слагаемые имеют различные знаки. В соответствии с этим алгоритмы получения суммы для каждого из вариантов значительно отличаются между собой. Так, алгоритм получения суммы двух чисел с одинаковыми знаками определяется следующим образом:

1) сложить два числа;

2) сумме присвоить знак одного из слагаемых.

В то же время алгоритм получения алгебраической суммы представлен на рис. 7 и записывается следующим образом:

1. Сравнить знаки слагаемых и, если они одинаковы, то выполнять сложение по первому алгоритму

2. Сравнить слагаемые по абсолютной величине, если знаки слагаемых разные.

3. Если есть необходимость, переставить числа местами (чтобы вычитать из большего меньшее).

4. Произвести вычитание двух чисел.

5. Результату присвоить знак большего слагаемого. Из этого следует, что первый алгоритм значительно проще второго. Следовательно, желательно преобразовать отрицательные числа таким образом, чтобы операцию вычитания заменить сложением, т. е. выполнять суммирование двух чисел следующим образом:

$$S = A + (- B) \qquad (8)$$

В прямом коде знаковый разряд и цифровую часть числа нельзя рассматривать как единое целое, а выполнение операции сложения затруднено тем, что необходимо кроме сумматора иметь еще в составе машины и вычитатель кодов чисел. Эти недостатки настолько серьезны, что прямой код для выполнения операции алгебраического сложения не применяется, но он удобен при выполнении операций умножения и деления.

Алгоритм сложение-вычитание чисел в прямом коде

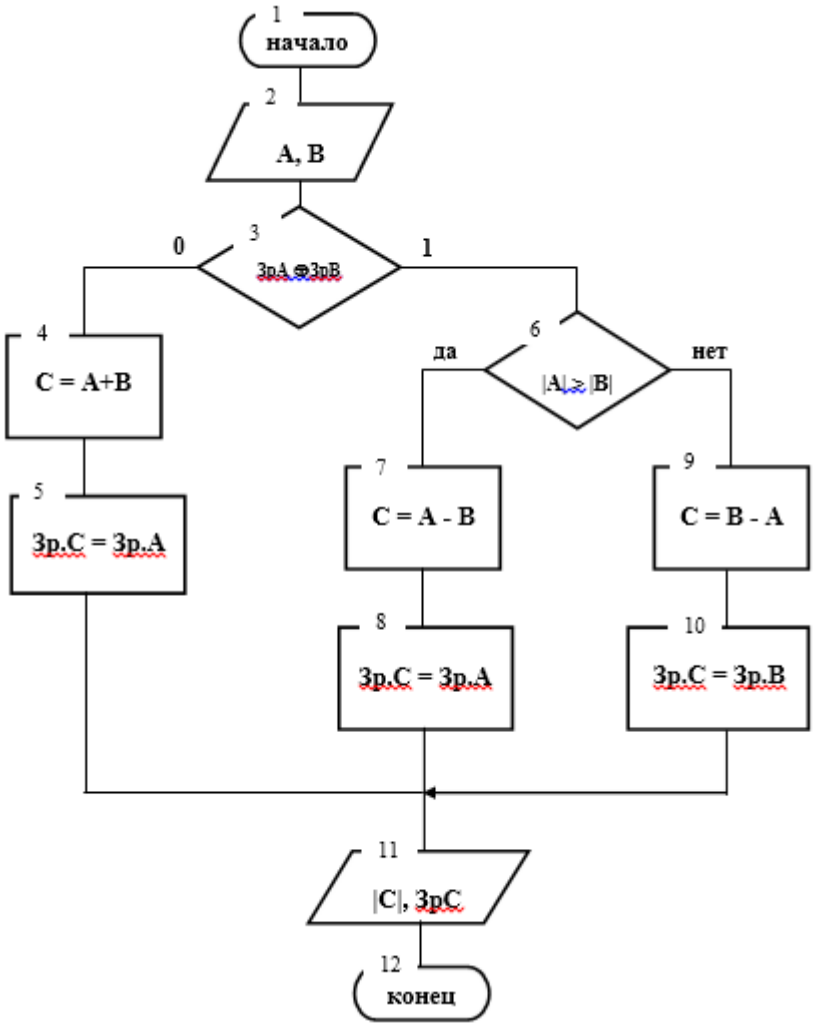


Рис.6 - Алгоритм сложение-вычитание чисел в прямом коде

Вычитание чисел в прямом коде, где обозначено А и В входные числа, С результат сложения, Зр.А, Зр.В, Зр.С – знаковые разряды чисел А, В и С.

1.5. Кодирование отрицательных чисел

Для выполнения операций в ЭВМ числа кодируются специальными машинными кодами. Используются прямой, дополнительный и обратные коды, позволяющие заменить операцию непосредственного вычитания операцией сложения чисел с целью упрощения АЛУ ЭВМ.

Прямой код. Он основан на представлении чисел в виде их абсолютного значения с кодом соответствующего знака: плюса или минуса.

Формула для образования прямого кода двоичного числа

A=0, $a_1\ a_2\ a_3 \ldots a_n$

имеет вид

$$A_{пр} = \begin{cases} A, если A \geq 0 \\ 1 - A, если A \leq 0. \end{cases} \qquad (9)$$

Пример1. 1.A=+0,1101;Апр=0,1101.

2.A=-0,10101;Апр=1-(-0,10101)=1,10101.

Из (9) следует, что нуль в прямом коде может быть положительным и отрицательным:

A=+0,00…00; Апр=0,00…00.

A=-0,00…00; Апр=1,00…00.

Прямой код используют для хранения чисел в ЗУ, в устройствах ввода и вывода, а также при выполнении операции умножения.

Дополнительный код. Формула для образования дополнительного кода двоичного числа А имеет вид

$$A_{доп} = \begin{cases} A, если A \geq 0 \\ 10 + A, если A < 0. \end{cases} \qquad (10)$$

Пример2. A=-0,101010; Адоп=10+(-0,101010)=1,010110.

Как видно из выражения (10), дополнительный код положительного числа полностью совпадает с изображением числа в прямом коде.

Сравнивая отрицательное число А с его дополнительным кодом Адоп, можно вывести следующее правило: чтобы записать отрицательное число в дополнительном коде, нужно в знаковом разряде этого числа поставить единицу, а во всех числовых разрядах нули заменить единицами, а единицы – нулями и к полученному результату прибавить единицу младшего разряда.

Пример3. Записать двоичное число A=-0,0101 в дополнительном коде.

Адоп=1,1010+0,0001=1,1011.

Чтобы преобразовать дополнительный код отрицательного числа в прямой код, необходимо в числовых разрядах этого числа заменить нули на единицы, а единицы на нули и прибавить к полученному результату единицу младшего разряда.

Пример4. Преобразовать дополнительный код Адоп=1,101011 отрицательного числа в прямой код.

Апр=1,0100+0,0001=1,0101.

В дополнительном коде отрицательный нуль отсутствует.

Обратный код. Формула для образования обратного кода имеет вид

$$A_{обр} = \begin{cases} A, если A \geq 0 \\ 10 + A - 10^{n}, если A \leq 0. \end{cases} \qquad (11)$$

Пример5. А=-0,100110.

Аобр=10-0,100110-0,000001=1,011001.

Сравнивая отрицательное число А с его обратным кодом Аобр, можно вывести следующее правило: чтобы записать отрицательное число в обратном коде, нужно в знаковом разряде этого числа поставить единицу, а в числовых разрядах нули заменить единицами, а единицы – нулями.

В обратном коде нуль изображается неоднозначно:

А=+0,00…00; Аобр=0,00…00;

А=-0,00…00; Аобр=1,11…11.

Обратный код положительного числа полностью совпадает с изображением числа в прямом коде.

1.6. Модифицированные коды.

С точки зрения построения АЛУ они удобны для выявления переполнения разрядной сетки, которое может получиться при сложении чисел. Эти коды отличаются от простых машинных кодов тем, что на изображение знака отводятся два разряда: плюс изображается двумя нулями, а минус – двумя единицами.

Преобразование двоичных чисел в модифицированные прямой, дополнительный и обратный коды производится по правилу, рассмотренным выше.

Пример6. Представить двоичные числа А1=+0,1101101 и А2=-0,1101101 в дополнительном и обратном модифицированных кодах.

$A^{м}_{1пр}$ =00,1101101; $A^{м}_{1доп}$ =00,1101101; $A^{м}_{1обр}$ =00,1101101.
$A^{м}_{2пр}$ =11,1101101; $A^{м}_{2доп}$ =11,0010011; $A^{м}_{2обр}$ =11,0010010.

Преобразование чисел в заданный код осуществляется автоматически как при вводе чисел в ЭВМ, так и при выполнении операций.

Сложение чисел в модифицированном дополнительном коде осуществляется по правилам двоичной арифметики. Единица переноса, возникающая в старшем знаковом разряде суммы, отбрасывается. Знаковым разрядом числа является второй слева от запятой разряд; первый разряд служит для анализа переполнения разрядной сетки.

Пример7. Сложить в модифицированном дополнительном коде двоичные числа А и В при условии:

1.A>0; B>0; A+B>0.

A=+0,1101; B=+0,0001;

$$\begin{array}{r} A_{доп}{}^{M} = 00{,}1101 \\ +\quad B^{M}{}_{доп} = 00{,}0001 \\ \hline (A+B)_{доп}{}^{M} = 00{,}1110; (A+B)_{пр}{}^{M} = 00{,}1110 \end{array}$$

2.A>0; B<0; A+B>0.

A=+0,1101; B=-0,0001;

$$\begin{array}{r} A_{доп}{}^{M} = 00{,}1101 \\ +\quad B^{M}{}_{доп} = 11{,}1111 \\ \hline (A+B)_{доп}{}^{M} = 100{,}1100; \\ \hline \uparrow \\ (A+B)^{M}{}_{пр} = 00{,}1100 \end{array}$$

Единица переноса из старшего знакового разряда не учитывается.

3.A<0;B>0;A+B<0.

A=-0,1101; B=+0,0001;

$$
\begin{array}{l}
\quad A_{доп}{}^{M} = 11{,}0011 \\
+\quad B^{M}{}_{доп} = 00{,}0001 \\
\hline
(A+B)_{доп}{}^{M} = 11{,}0100; \\
(A+B)_{пр}{}^{M} = 11{,}1100
\end{array}
$$

4.A<0;B<0;A+B<0.
A=-0,1101; B=-0,0001

$$
\begin{array}{l}
\quad A_{доп}{}^{M} = 11{,}0011 \\
+\quad B^{M}{}_{доп} = 11{,}1111 \\
\hline
(A+B)_{доп}{}^{M} = 111{,}0010; \\
\underline{\qquad\qquad}\uparrow \\
(A+B)^{M}{}_{пр} = 11{,}1110
\end{array}
$$

Единица переноса из старшего знакового разряда не учитывается.

Сложение в модифицированном обратном коде осуществляется так же, как и в дополнительном коде. Отличие состоит в том, что единицу переноса из старшего знакового разряда (если она появляется) необходимо прибавить к младшему разряду суммы, т.е. образуется циклический перенос.

Пример8. Сложить в модифицированном обратном коде двоичные числа А и В при условии

1.A>0; B>0; A+B>0.
A=+0,1101; B=+0,0001;

$$
\begin{array}{l}
\quad A_{обр}{}^{M} = 00{,}1101 \\
+\quad B^{M}{}_{обр} = 00{,}0001 \\
\hline
(A+B)_{обр}{}^{M} = 00{,}1110; (A+B)_{пр}{}^{M} = 00{,}1110
\end{array}
$$

2.A>0; B<0; A+B>0.
A=+0,1101; B=-0,0001;

$$A_{обр}{}^{M} = 00{,}1101$$
$$+\quad B^{M}{}_{обр} = 11{,}1110$$
$$(A+B)_{обр}{}^{M} = 100{,}1011;$$

Цикл.перенос – ↓ ____ ↑

$$(A+B)^{M}{}_{обр} = 00{,}1100$$

$$(A+B)^{M}{}_{пр} = 00{,}1100$$

Единица переноса из старшего знакового разряда учитывается.

3.A<0;B>0;A+B<0.

A=-0,1101; B=+0,0001;

$$A_{обр}{}^{M} = 11{,}0010$$
$$+\quad B^{M}{}_{обр} = 00{,}0001$$
$$(A+B)_{обр}{}^{M} = 11{,}0011;$$

$$(A+B)_{пр}{}^{M} = 11{,}1100$$

4.A<0;B<0;A+B<0.

A=-0,1101; B=-0,0001

$$A_{обр}{}^{M} = 11{,}0010$$
$$+\quad B^{M}{}_{обр} = 11{,}1110$$
$$(A+B)_{обр}{}^{M} = 111{,}0000;$$

Цикл.перенос – ↓ ____ ↑

$$(A+B)^{M}{}_{обр} = 11{,}0001$$

$$(A+B)^{M}{}_{пр} = 11{,}1110$$

Блок-схема алгоритма сложение чисел в дополнительном коде

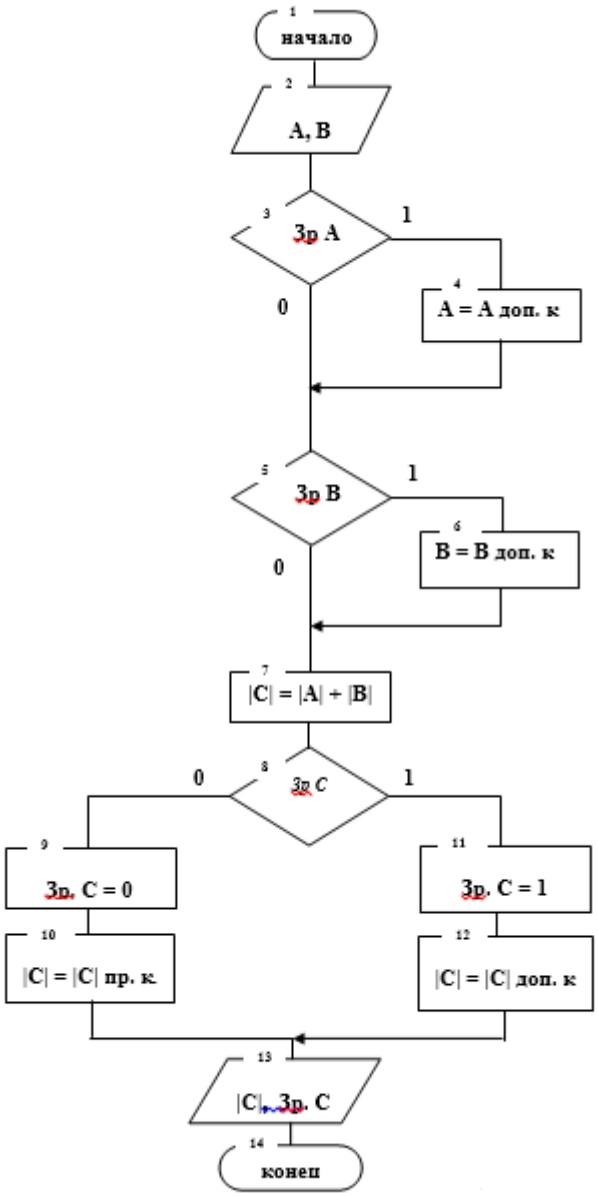


Рис.7 – Блок-схема алгоритма сложения чисел в дополнительном коде

Блок-схема алгоритма сложение чисел в обратном коде

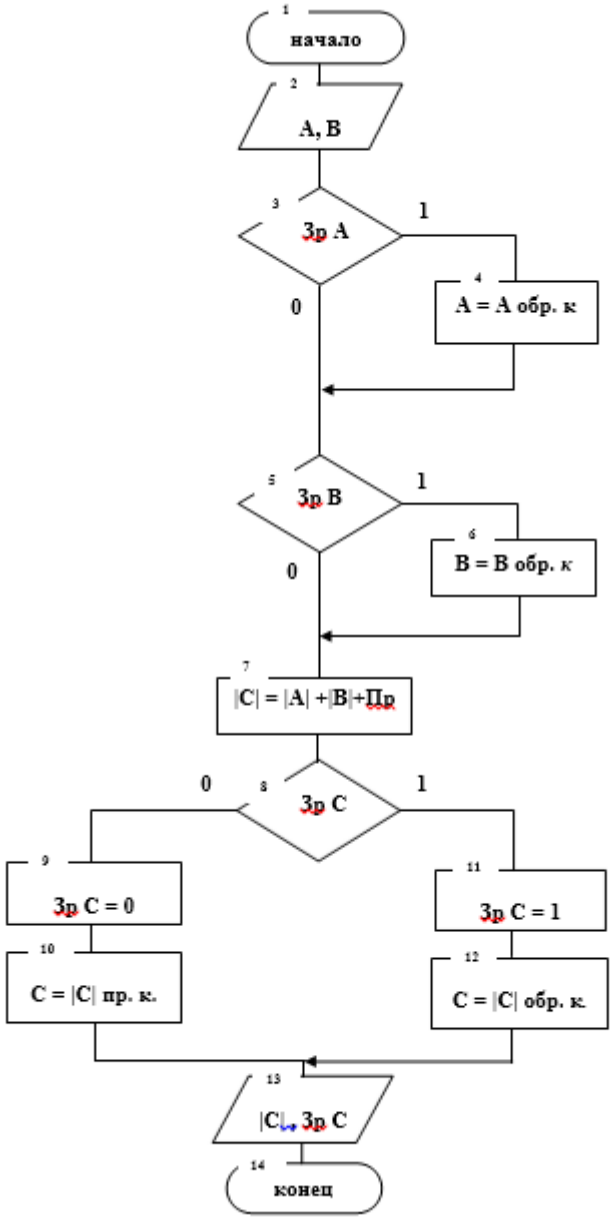


Рис.8 – Блок-схема алгоритма сложения чисел в обратном коде

Структурная схема операционного устройства выполняющего операцию сложение чисел в дополнительном и обратном коде.

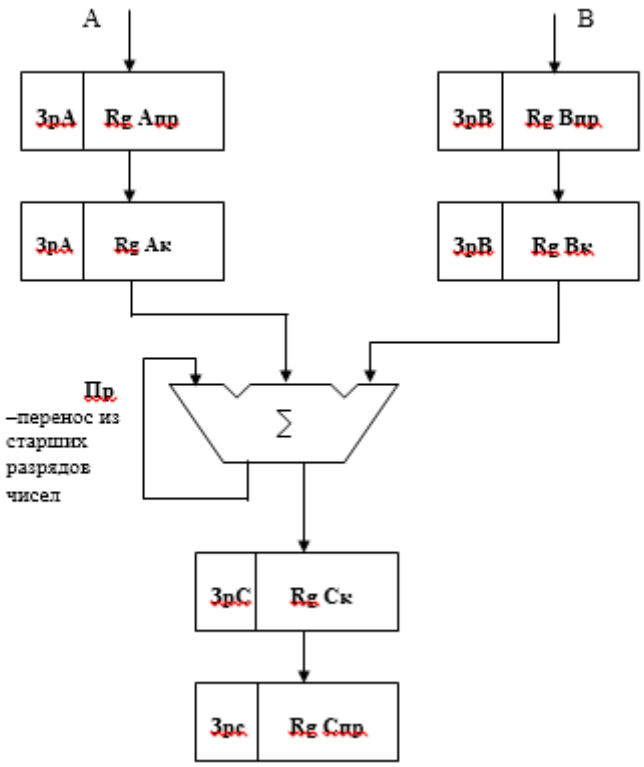


Рис. 9 – Сложение чисел в дополнительном и обрат ном коде

На рис.9 представлена структурная схема операционного автомата, выполняющего операции сложение-вычитание чисел в дополнительном и обратном коде, где представлено: ЗрА – знаковый разряд числа А, RgAпр – регистр для хранения модуля числа А в прямом коде, RgAк – регистр для хранения числа А в коде, ЗрВ – знаковый разряд числа В, RgBпр – регистр для хранения модуля числа В в прямом коде, RgBк – регистр для хранения числа В в коде, - сумматор, ЗрС –знаковый разряд числа С, RgCк – регистр для хранения результата С в коде, RgCпр – регистр для хранения результата С в прямом коде.

Работа алгоритма сложение чисел в дополнительном коде.

Блок 1 алгоритма является начальным (рис.7).

В блоке 2 алгоритма происходит ввод чисел А и В в десятичной системе счисления. В этом блоке числа переводятся в двоичную систему счисления соответствующими процедурами с учетом знаков операндов. Числа А и В будут представлены в

формате прямых кодов со своими знаками согласно формату, изображенному на рис.4.

В блоке 3 алгоритма анализируется знаковый разряд числа А. Если нулевой разряд α_0 равен нулю, то перевода в дополнительный код числа А не произойдет, в этом случае число А положительное. Если нулевой разряд α_0 равен единице, то в этом случае происходит преобразование отрицательного числа А в дополнительный код.

В блоке 4 алгоритма число А переводится в дополнительный код

А = А доп.к.

В блоке 5 алгоритма анализируется знаковый разряд числа В. Если нулевой разряд α_0 равен нулю, то перевода в дополнительный код числа В не произойдет, в этом случае число В положительное. Если нулевой разряд α_0 равен единице, то в этом случае происходит преобразование отрицательного числа В в дополнительный код.

В блоке 6 алгоритма число В переводится в дополнительный код

В = В доп.к.

В блоке 7 алгоритма происходит суммирование модулей чисел А и В. Результатом сложения чисел является модуль числа |С| = |А| + |В|. Знаковые разряды чисел в операции сложения участвуют. Если возникает перенос из старшего знакового разряда, то в дополнительном коде он не учитывается (отбрасывается).

В блоке 8 алгоритма анализируется знаковый разряд результата |С|. Если он равен нулю, то происходит переход на блок 9 алгоритма, в этом случае результатом является положительное число. Если знаковый разряд суммы равен единице, то осуществляется переход на блок 11 алгоритма, результат в этом случае - отрицательное число.

В блоке 9 алгоритма знаковому разряду результата С присваивается значение равное нулю З.р. С = 0.

В блоке 10 алгоритма результатом сложения чисел является положительное число, результат представлен в прямом коде |С| = |С| пр.к.

В блоке 11 алгоритма знаковый разряд результата |C| принимает значение единицы З.р С = 1.

В блоке 12 алгоритма результат сложения С преобразуется в дополнительный код |C| = |C|доп.к.

В блоке 13 алгоритма происходит выдача результата в виде модуля числа |C| и знакового разряда З.р. С.

Блок 14 алгоритма является конечным.

Работа алгоритма сложение чисел в обратном коде.

Блок 1 алгоритма является начальным (рис.8).

В блоке 2 алгоритма происходит ввод чисел А и В в десятичной системе счисления. В этом блоке числа переводятся в двоичную систему счисления соответствующими процедурами с учетом знаков операндов. Числа А и В будут представлены в формате прямых кодов со своими знаками согласно формату, изображенному на рис.4.

В блоке 3 алгоритма анализируется знаковый разряд числа А. Если нулевой разряд α_0 равен нулю, то перевода в обратный код числа А не произойдет, в этом случае число А положительное. Если нулевой разряд α_0 равен единице, то в этом случае происходит преобразование отрицательного числа А в обратный код.

В блоке 4 алгоритма число А переводится в обратный код А = А обр.к.

В блоке 5 алгоритма анализируется знаковый разряд числа В. Если нулевой разряд α_0 равен нулю, то перевода в обратный код числа В не произойдет, в этом случае число В положительное. Если нулевой разряд α_0 равен единице, то в этом случае происходит преобразование отрицательного числа В в обратный код.

В блоке 6 алгоритма число В переводится в обратный код В =В обр.к.

В блоке 7 алгоритма происходит суммирование модулей чисел А и В, а также переноса из старших знаковых разрядов Пр. Результатом сложения чисел является модуль числа |C| = |A| + |B| + Пр. Знаковые разряды чисел в операции сложения участвуют. Если возникает перенос из старшего знакового разряда, то в обратном коде он учитывается, т.е. суммируется с младшим разрядом числа С

.

В блоке 8 алгоритма анализируется знаковый разряд результата |C|. Если он равен нулю, то происходит переход на блок 9 алгоритма, в этом случае результатом является положительное число. Если знаковый разряд суммы равен единице, то осуществляется переход на блок 11 алгоритма, результат в этом случае - отрицательное число.

В блоке 9 алгоритма знаковому разряду результата C присваивается значение равное нулю З.р. C = 0.

В блоке 10 алгоритма результатом сложения чисел является положительное число, результат представлен в прямом коде |C| = |C| пр.к.

В блоке 11 алгоритма знаковый разряд результата |C| принимает значение единицы З.р C = 1.

В блоке 12 алгоритма результат сложения C преобразуется в обратный код |C| = |C|обр.к.

В блоке 13 алгоритма происходит выдача результата в виде модуля числа |C| и знакового разряда З.р. C.

Блок 14 алгоритма является конечным.

1.7. Переполнение разрядной сетки при сложении чисел в прямом коде.

При сложении двух двоичных чисел, по абсолютной величине меньших единицы, код суммы может по абсолютной величине превысить единицу или стать равным ей. В этом случае произойдет переполнение разрядной сетки, что приведет к ложному результату. Поясним это на примерах (табл.1) сложения двоичных чисел в дополнительных кодах.

В примерах 1 и 4 полученные суммы не соответствуют действительным, т.е. произошло переполнение разрядной сетки.

Переполнение разрядной сетки при образовании суммы по абсолютной величине больше 1 можно обнаружить двумя способами.

1. Сравнением знака полученной суммы со знаком действительной суммы. Как видно из примеров 1 и 4 (табл. 1), происходит полное искажение результата сложения как по знаку, так и по величине.

2. Анализом переносов, возникающих при сложении двоичных чисел. Признаками переполнения разрядной сетки являются либо наличие переноса в знаковый разряд при отсутствии переноса из разрядного знака (1-й пример), либо наличие переноса из знакового разряда суммы при отсутствии переноса в этом разряде (4-й пример). Если нет переносов из знакового разряда и в знаковый разряд суммы (3-й пример) или есть оба эти переноса (2-й пример), то переполнения разрядной сетки нет.

Таблица 1

1.A=+0,1101;	2.A=+0,1101;
B=+0,0111;	B=-0,0111;
A+B>0	A+B>0
Адоп=0,1101	Адоп=0,1101
+ Вдоп=0,0111	+ Вдоп=0,1001
(A+B)доп=1,0100	(A+B)доп=10,0110
(A+B)пр=1,1100	(A+B)пр=0,0110
3.A=-0,1101;	4.A=-0,1101;
B=+0,0111;	B=-0,0111;
A+B<0	A+B<0
Адоп=1,0011	Адоп=1,0011
+ Вдоп=0,0111	+ Вдоп=1,1001
(A+B)доп=1,1010	(A+B)доп=10,1100
(A+B)пр=1,0110	(A+B)пр=0,1100

1.8. Переполнение разрядной сетки при сложении чисел в модифицированных кодах.

Переполнение разрядной сетки при сложении модифицированных кодах обнаруживается способом сравнения знаковых разрядов полученной суммы. Поясним это на следующих примерах:

1.
$$A = +0{,}1101$$
$$B = +0{,}0111$$
$$A + B > 0$$
$$+\ A^{M}_{доп} = 00{,}1101$$
$$B^{M}_{доп} = 00{,}0111$$
$$(A+B)_{доп}{}^{M} = 01{,}0100$$

2.
$$A = -0{,}1101$$
$$B = -0{,}0111$$
$$A + B < 0$$
$$+\ A^{M}_{доп} = 11{,}0011$$
$$B^{M}_{доп} = 11{,}1001$$
$$(A+B)_{доп}{}^{M} = 110{,}1100$$
теряется ___ ↑

Как видно из примеров, знаковых разрядов полученной суммы положительных слагаемых имеем комбинацию «01» отрицательных – «10», являющихся признаками переполнения разрядной сетки.

При отсутствии в ЭВМ вычитающего устройства операция вычитания заменяется операцией сложения. При этом знак вычитаемого заменяется на противоположный:

$$A-B=A+(-)B \qquad (12)$$

Операции умножения и деления строятся на базе арифметических операций сложения и вычитания.

2. Задание

1. Составить программы на языке высокого уровня по представленным блок-схемам алгоритмов:

- сложение чисел с фиксированной точкой в дополнительном коде (рис.7);
- сложение чисел с фиксированной точкой в обратном коде (рис.8);

2. Составить программу на языке высокого уровня по представленному совмещенному алгоритму на рис.10.

3. Промоделировать (тестировать) программы на ПЭВМ.

4. Проанализировать результаты выполнения программ.

Алгоритмы, используемые при выполнении лабораторной работы представлены на рис.7, рис.8, рис 10 .

Совмещенный алгоритм сложение чисел в дополнительном и обратном коде.

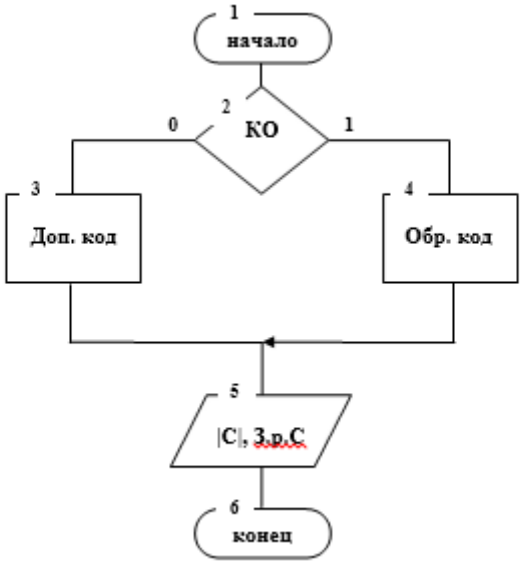


Рис.10 – Сложение чисел в дополнительном и обратном коде

На рис.10 обозначено: КО – код операции; если КО = 0, то происходит выполнение алгоритма сложение чисел в дополнительном коде, если КО = 1, то происходит выполнение алгоритма сложение чисел в обратном коде.

3. Содержание отчета

Отчет должен содержать:

- титульный лист;
- задание;
- описание переменных, используемых в программе;
- структурную схему операционного автомата;
- блок-схему алгоритма сложение-вычитание чисел в прямых кодах;
- описание работы алгоритма сложение-вычитание чисел в прямом коде;
- блок-схемы алгоритмов сложение чисел, представленных в дополнительном и обратном коде;
- описание работы алгоритмов сложение чисел, представленных в дополнительном и обратном коде;
- тексты программ;
- результаты выполнения программ.

Контрольные вопросы

1. Дайте определение системы счисления.
2. Достоинство и недостатки основных и вспомогательных систем счисления, используемых в ЭВМ.
3. Как представляются числа в ЭВМ. Как производится нумерация битов.
4. Какие существуют формы представления чисел в ЭВМ.
5. Как производится кодирование положительных и отрицательных чисел.
6. Что такое модифицированные коды. Приведите примеры.
7. В каком формате представляются числа в ЭВМ.
8. Как осуществляется операция сложение-вычитание чисел в прямом коде с фиксированной запятой.
9. Как осуществляется преобразование положительных и отрицательных чисел в дополнительный код.
10. Как осуществляется преобразование положительных и отрицательных чисел обратный код.

11. Как выполняется операция сложение чисел в дополнительном и обратном коде двоичных чисел с фиксированной запятой.

12. Назовите основные блоки и их функции операционного автомата, выполняющего операцию сложение чисел в дополнительном и обратном коде.

13. Что такое переполнение разрядной сетки. Укажите признаки переполнения разрядной сетки при сложении чисел.

14. Укажите признаки переполнения разрядной сетки в модифицированных кодах.

Библиографический список

1. Карцев М.А. Арифметика цифровых машин. –М.: Наука. 1969. – 575 с.
2. Самофалов К.Г., Романкевич А.М., Валуйский В.Н. Прикладная теория цифровых автоматов. – Киев: Высш. шк., 1987 – 374 с. : ил.
3. Савельев А.Я. Прикладная теория цифровых автоматов. –М.: Высш. шк., 1987. – 271 с: ил.
4. Каган Б.М. Электронные вычислительные машины и системы. – М.: Энергия 1979. - 528 с.
5. Майоров С.А., Новиков Г.И. Принципы организации цифровых машин. – Л.: Машиностроение. 1974. – 431 с.
6. Поспелов Д.А. Арифметические основы вычислительных машин дискретного действия. – М.: Высш. шк. 1970. – 307 с.
7. Соловьев Г.Н. Арифметические устройства ЭВМ. – М.: Энергия. 1978. –177 с.
8. Сергеев Н.П., Вашкевич Н.П. Основы вычислительной техники: Учеб. пособие для электротех. спец. вузов. – М.: Высш. шк., 1988. – 311 с.: ил.
9. Преснухин Л.Н., Нестеров П.В. Цифровые вычислительные машины. – М.: Высш. шк. 1981, - 511 с.