**Операционное устройства и алгоритм Бута**

**Введение (сл 2)**

В классической фон-неймановской вычислительной машине функция арифметической и логической обработки данных возлагается на арифметико-логическое устройство (АЛУ). Учи­тывая разнообразие выполняемых операций и типов обрабатываемых данных, ре­ально можно говорить не о едином устройстве, а о комплексе специализирован­ных операционных устройств (ОПУ), каждое из которых реализует определенное подмножество арифметических или логических операций, предусмотренных сис­темой команд вычислительных машин. С этих позиций следует выделить:

- ОПУ целочисленной арифметики;

*-*ОПУ для реализации логических операций;

- ОПУ десятичной арифметики;

- ОПУ для чисел с плавающей запятой.

**Структуры ОПУ (сл 3)**

Операционное устройство предназначено для выполнения операции над операндами в соответствии с кодом выполняемой команды.

Набор элементов, на основе которых строятся структуры различных ОПУ, называется структурным базисом. Структурный базис ОПУ включает в себя:

- регистры, обеспечивающие кратковременное хранение слов данных;

- управляемые шины, предназначенные для передачи слов данных;

- комбинационные схемы, реализующие вычисление функций микроопераций и логических условий по управляющим сигналам от устройства управления. С практических позиций больший интерес представляют два иных вида структур ОПУ: жесткая (сл 4) и магистральная (сл 5).

<https://gigabaza.ru/images/9/17645/6d8c225d.jpg>

<https://gigabaza.ru/images/9/17645/621d511b.jpg>

**Базис целочисленных операционных устройств (сл 6)**

Для большинства современных ВМ общепринятым является такой формат с фиксированной запятой (ФЗ), когда запятая фиксируется справа от младшего разряда кода числа. По этой причине соответствующие операционные устройства называют целочисленными ОПУ. В форме с ФЗ могут быть представлены как числа без знака, когда все n позиций числа отводятся под значащие цифры, так и со знаком. В последнем случае старшим (n-1)-й разряд числа занимает знак числа (0 —плюс, 1 — минус), а под значащие цифры отведены разряды с (n- 2)-го по 0-й.

При записи отрицательных чисел используется дополнительный код, который для числа N получается по следующей формуле:

https://gigabaza.ru/images/9/17645/md5976c2.jpg

Если исключить логические операции, которые рассматриваются отдельно, целочисленное ОПУ должно обеспечивать выполнение следующих арифметических

операций над числами без знака и со знаком:

- сложение/вычитание;

- умножение;

- деление.

**Ускорение целочисленного умножения**

Методы ускорения умножения можно условно разделить на аппаратные и логические. Те и другие требуют дополнительных затрат оборудования, которые при использовании аппаратных методов возрастают с увеличением разрядности сомножителей. Аппаратные способы приводят к усложнению схемы умножителя, но не затрагивают схемы управления. Дополнительные затраты оборудования при реализации логических методов не зависят от разрядности операндов, но схема управления умножителя при этом утяжеляется. На практике ускорение умножения часто достигается комбинацией аппаратных и логических методов.

**Логические методы ускорения умножения (сл 8)**

Логические подходы к убыстрению умножения можно подразделить на две группы:

- методы, позволяющие уменьшить количество сложений в ходе умножения;

- методы, обеспечивающие обработку нескольких разрядов множителя за шаг.

Реализация и тех и других требует введения дополнительных цепей сдвига в регистры.

Рассмотрим алгоритм Бута, который входит в первую группу.

**Алгоритм Бута. (сл 9)**

Алгоритм умножения Бута это алгоритм умножения, который позволяет перемножить два двоичных числа в дополнительном коде. Алгоритм был разработан Эндрю Дональдом Бутом в 1951. Бут пользовался настольными вычислителями, которые выполняли операцию сдвига быстрее, чем операцию сложения, и создал алгоритм для увеличения скорости их работы.

В основе алгоритма Бута [63] лежит следующее соотношение, характерное для

последовательностей двоичных цифр:

https://gigabaza.ru/images/9/17645/m2c60fb4.jpg

где m и k — номера крайних разрядов в группе из последовательных единиц.

**Алгоритм(сл 10)**

Алгоритм Бута включает в себя циклическое сложение одного из двух заранее установленных значений A и S с произведением P, а затем выполнение арифметического сдвига вправо над P. Пусть m и r — множимое и множитель соответственно, а x и y представляют собой количество битов в m и r.

1. Установить значения A и S, а также начальное значение P. Каждое из этих чисел должно иметь длину, равную (x + y + 1).
   1. A: Заполнить наиболее значимые (левые) биты значением m. Заполнить оставшиеся (y + 1) бит нулями.
   2. S: Заполнить наиболее значимые биты значением (−m) в дополнительном коде. Заполнить оставшиеся (y + 1) бит нулями.
   3. P: Заполнить наиболее значимые x бит нулями. Справа от них, заполнить биты значением r. Записать 0 в крайний наименее значимый (правый) бит
2. Определить значение двух наименее значимых (правых) битов P.
   1. Если их значение равно 01, вычислить значение P + A. Переполнение игнорировать.
   2. Если их значение равно 10, вычислить значение P + S. Переполнение игнорировать.
   3. Если их значение равно 00, действие не требуется. P используется без изменений на следующем шаге.
   4. Если их значение равно 11, действие не требуется. P используется без изменений на следующем шаге.
3. Выполнить операцию арифметического сдвига над значением, полученным на втором шаге, на один бит вправо. Присвоить P это новое значение.
4. Повторить шаги 2 и 3 y раз.
5. Отбросить крайний наименее значимый (правый) бит P. Это и есть произведение m и r.

## Пример

Вычислить 3 × (−4). В этом случае **m** = 3, **r** = −4, *x* = 4, *y* = 4:

* A = 0011 0000 0
* S = 1101 0000 0
* P = 0000 1100 0
* Выполним цикл 4 раза :
  1. P = 0000 110**0 0**. Крайние два бита равны 00.
     + P = 0000 0110 0. Арифметический сдвиг вправо.
  2. P = 0000 011**0 0**. Крайние два бита равны 00.
     + P = 0000 0011 0. Арифметический сдвиг вправо.
  3. P = 0000 001**1 0**. Крайние два бита равны 10.
     + P = 1101 0011 0. P = P + S.
     + P = 1110 1001 1. Арифметический сдвиг вправо.
  4. P = 1110 100**1 1**. Крайние два бита равны 11.
     + P = 1111 0100 1. Арифметический сдвиг вправо.
* Произведение равно 1111 0100 (−12 в десятичной системе)

**Модифицированный алгоритм Бута (сл 11)**

Так же стоит сказать, что на практике большее распространение получила модификация алгоритма Бута, где количество операций сложения при любом сочетании единиц и нулей в множителе всегда равно n/2. В модифицированном алгоритме производится перекодировка цифр множителя из стандартной двоичной системы (0, 1) в избыточную систему (-2. -1, 0, 1, 2), где каждое число представляет собой коэффициент, на который умножается множимое перед добавлением к СЧП. Одновременно анализируются три разряда множителя bi+1bibi-1 (два текущих и старший разряд из предыдущей тройки) и, в зависимости от комбинации 0 и 1 в этих разрядах, выполняется прибавление или вычитание множимого, прибавление или вычитание удвоенного множимого, либо никакие действия не производятся (табл. 7.1).

Таблица 7.1. Логика модифицированного алгоритма Бута

**Список литературы.**

<https://pue8.ru/protsessory/655-operatsionnyj-avtomat-algoritm-umnozheniya-buta.html>

<https://pandia.ru/text/78/369/1681.php>

<https://dic.academic.ru/dic.nsf/ruwiki/1722628>

<https://life-prog.ru/1_57408_algoritm-buta.html>

<https://poznayka.org/s104257t1.html>

<https://studfile.net/preview/4339723/page:9/>