Алгоритм формирования машинного представления числовых данных в формате с плавающей запятой

И. И. ЖУЛЬКОВСКАЯ, О. А. ЖУЛЬКОВСКИЙ

Днепродзержинский государственный технический университет

Исследован и описан алгоритм современного подхода к формированию машинного представления и хранения числовой информации в формате с плавающей запятой. Показано, что проблемы, связанные с точностью компьютерных вычислений, обусловлены ограничением разрядности аппаратной реализации ЭВМ.

Досліджено та описано алгоритм сучасного підходу до формування машинного уявлення і зберігання числової інформації у форматі з плаваючою комою. Показано, що проблеми, пов'язані з точністю комп'ютерних обчислень, обумовлені обмеженням розрядності апаратної реалізації ЕОМ.

Investigate and describe the algorithm of the modern approach to the formation of the machine representation and storage of digital information in floating point format. It is shown that the problems associated with the accuracy of computer calculations, due to the limited bit depth computer hardware implementation.

бщая характеристика проблемы. Нарастание объемов вычислений и задействованных при этом ресурсов ЭВМ, а также постоянно растущие требования к точности результатов теоретических исследований (математического моделирования) и проектирования сложных систем и динамических процессов повышает актуальность проблемы машинного представления и хранения числовой информации в современных ЭВМ (персональных компьютерах).

Численное решение большинства практических задач главным образом сопряжено с выполнением операций над действительными (вещественными) числами или так называемыми числами с плавающей запятой (точкой).

Такое кодирование числовой информации в современных языках и системах программирования обусловлено введением и использованием специальных (сопроцессорных) типов данных, задействующих значительную часть ресурсов ЭВМ.

Вышесказанное требует от исследователя предметного понимания алгоритмов формирования машинного представления и хранения вещественных чисел, методов реализации этих алгоритмов в современных языковых системах с целью повышения эффективности (увеличения скорости обработки данных при уменьшении размеров машинного кода программы) проводимых численных исследований.

Краткий аналитический обзор. Как известно, до середины 80-х годов прошлого века прогрессировало несколько десятков архитектур ЭВМ, каждая из которых реализовывала свои способы представления в машинной памяти множества вещественных чисел. В 1985 году ассоциацией IEEE (Institute of Electrical and Electronics Engineers) был разработан и внедрен единый стандарт для представления в двоичном коде чисел с плавающей запятой. Этот, наиболее распространенный и используемый стандарт для вычислений с плавающей запятой, используется многими современными аппаратными и программными средствами ЭВМ. Полное название этого стандарта - *IEEE Standard for Binary Floating-Point Arithmetic*, *ANSI/IEEE Std* 754-1985 (*ANSI – American National Standards Institute*) [1–3]. Код данного стандарта в международной электротехнической комиссии *IEC - IEC* 60559:1989, *Binary Floating-Point Arithmetic for Microprocessor Systems*.

На данный момент ассоциация IEEE реорганизована из международной общественной инженерной организации в коммерческую структуру, в связи с чем авторские стандарты IEEE стали коммерческим продуктом и не находятся в свободном распространении и обращении.

Известные и доступные публикации на тему представления и хранения вещественных чисел в памяти компьютера рассматривают вопросы погрешностей результатов при вычислении функций, пути минимизации ошибок, связанных с округлением и потерей точности, создания тестов для реализаций математических функций и т.п. [4–6].

Таким образом, эти публикации, рассматривая стандарт IEEE, не содержат описания алгоритма формирования машинного представления и хранения в памяти компьютера числовой информации в формате с плавающей запятой.

Цель работы (постановка задачи) – исследование и описание современного алгоритма формирования машинного представления и хранения числовых данных в формате с плавающей запятой.

Результаты работы. Стандарт IEEE-754, прежде всего, позволил упростить процесс переноса существующего программного обеспечения на другие совместимые программно-аппаратные платформы, предоставил возможность разрабатывать новые, эффективные и переносимые вычислительные программы, содержащие стандартизованные элементарные функции, операции высокоточной арифметики и символьных вычислений. Стандартом также обеспечивается прямая поддержка диагностики времени исполнения, обработки исключительных ситуаций и интервальной арифметики. Программы, написанные в соответствии с IEEE-стандартом, возвращают идентичные результаты в любых совместимых программно-аппаратных средах.

Итак, стандарт IEEE–754 определяет:

1. Базовые и расширенные форматы чисел с плавающей запятой.
2. Операции сложения, вычитания, умножения, деления, квадратного корня, остатка и сравнения.
3. Преобразования между целочисленными и плавающими форматами.
4. Преобразования между различными плавающими форматами.

5. Преобразования между базовыми форматами чисел с плавающей запятой и десятичными строками.
6. Исключительные ситуации, возникающие при вычислениях с плавающей запятой, и их обработку.

В августе 2008 года ассоциация IEEE выпустила обновленный стандарт – ANSI/IEEE Std 754-2008, который включает и стандарт ANSI/IEEE Std 754-1985. Среди основных дополнений изданного стандарта: плавающие числа с десятичной экспонентой; определение ряда форматов с 32-битным шагом с возрастающей точностью для дополнения 32-битного формата с одинарной точностью и 64-битного формата с двойной точностью; обновленный набор операций.

Действительные числа в формате с плавающей запятой (или в экспоненциальной форме) в обобщенном виде представляются следующим образом:

$$A = (-1)^{\text{sign A}} \cdot M \cdot S^{p}, \qquad (1)$$

где sign *A* – знак числа (0 – для положительных, 1 – для отрицательных);

M – мантисса числа *A*;

S – основа системы счисления;

p – порядок числа *A*.

Двоичное число с плавающей запятой (binary floating point number) – битовая строка, характеризующаяся тремя составляющими: знаком (sign), знаковым порядком (signed exponent) и мантиссой (significand). Ее численное значение (если оно существует) равно произведению знака, мантиссы и двойки, возведенной в степень, равную порядку.

С математической точки зрения, для записи мантиссы и показателя степени порядка числа существует бесконечное количество разрядов. На практике же ресурсы ЭВМ ограничены, а при аппаратной реализации они ещё и фиксированы: регистры имеют конкретный размер, ALU (арифметико-логическое устройство) рассчитано на работу с определённым форматом данных и всё, что в него не «помещается», приходится так или иначе отбрасывать (округлять).

Ограничение разрядов при аппаратной реализации обусловлено техническими возможностями и экономией. Как известно, чем больше разрядов числа находится на обработке CPU (центральным процессором), тем меньше скорость операций сложения и умножения, тем больше количество транзисторов, приходящихся на один сумматор/умножитель. От количества транзисторов зависит размер кристалла, требуемое питание и охлаждение и, в конечном итоге, цена процессора.

Для увеличения количества значащих цифр при представлении вещественного числа и предотвращения переполнения при выполнении арифметических операций мантиссу нормализуют.

Как известно, число с плавающей запятой называется нормализованным, если старшая цифра его мантиссы является значащей (не ноль), т.е. значение мантиссы должно находиться в диапазоне $S^{-1} \le M < 1$. В двоичной системе счисления S=2 и мантисса приобретает значения из диапазона $0{,}5 \le M < 1$. Таким образом, мантисса любого числа в формате с плавающей запятой должна начинаться с единицы в двоичной системе счисления. Это классический метод, при котором результат нормализации представляется в виде правильной дроби, т.е. с единицей после запятой и с нулем в целой части числа (рассматривается двоичная система счисления).

Однако существуют и другие алгоритмы нормализации. Так, в стандарте IEEE-754 старшая единица мантиссы нормализованного числа является единицей целой части мантиссы, т.е. запятая в мантиссе фиксируется после старшей единицы. Таким образом, условием нормализации является: $1 \le M < S$ (или для двоичной системы: $1 \le M < 2$). То есть, если число нормализовано, целая часть *M* всегда будет равна 1 и значит можно сэкономить на этом бите – не хранить, а только подразумевать наличие так называемого скрытого бита (hidden bit). В таком виде операнд хранится в памяти машины. При извлечении из памяти в операционный автомат этот скрытый бит восстанавливается, т.е. присутствует в явном виде. Очевидны плюсы такого кодирования – не приходится хранить лишних нулей, в результате в фиксированную по размеру мантиссу помещается максимум значащих разрядов.

Таким образом, в мантиссе хранятся значащие цифры числа, а порядок определяет его величину. Порядок, как известно, может быть как положительным, так и отрицательным целым числом, определяющим положение запятой в действительном числе.

Для того чтобы не хранить в ячейке памяти знак, порядок числа представляется не как целое число со знаком в явном виде, а в виде беззнакового числа, называемого смещенным порядком или характеристикой. При этом характеристика отличается от порядка на некоторую фиксированную для данного формата величину, называемую смещением (или смещением порядка):

$$X = p + b, \qquad (2)$$

где b – смещение порядка.

Смещение выбирается так, чтобы минимальному значению порядка соответствовал ноль. То есть, если порядок колеблется от $+128_{10}$ до -127_{10}, то к порядку всегда прибавляют 127_{10}, и тогда он колеблется в пределах от 0 до $+255_{10}$. Таким образом, не приходится хранить знак числа, что равносильно введению соответствующего масштабного коэффициента. Несложно модифицировать алгоритмы операций над числами так, чтобы они его учитывали.

В общем виде, если экспонента действительного числа должна занимать *k* бит и $0 < X < 2^{k-1}$, то величина смещения порядка или смещения экспоненты (exponent bias) выбирается по формуле:

$$b = 2^{k-1} - 1. \qquad (3)$$

Стандарт IEEE–754 определяет четыре формата представления чисел с плавающей запятой:

– с одинарной точностью (single-precision);

– с двойной точностью (double-precision);

– с одинарной расширенной точностью (single-extended precision);

– с двойной расширенной точностью (double-extended precision).

Естественно, в памяти ЭВМ двоичное число с плавающей запятой представлено набором битов.

Так, к примеру, число в формате с одинарной точностью занимает в памяти n=32 бита (4 байта); под экспоненту выделено k=8 бит. Соответственно, для мантиссы используется 23 разряда, а смещение экспоненты равно 127.

Для чисел двойной точности n=64 (8 байт) и k=11. Для мантиссы используется 52 разряда, а смеще-

ние экспоненты равно 1023.

Для чисел расширенной двойной точности определенные значения k и n стандартами не определяются. Вместо этого вводятся следующие ограничения: $80 \le n \le 128$; $k \ge 15$. Так, например, для процессоров 32-битной архитектуры Intel n=80 (10 байт) и k=15. При этом для мантиссы используется 64 разряда, а смещение экспоненты равно 16383.

Кроме того, иногда используются числа четырехкратной точности, для которых n=128 (16 байт) и k=15. Здесь для мантиссы используется 112 разряда, а смещение экспоненты равно 16383. Данные такого формата стандартами не специфицированы, но имеют аналогичную ранее описанным форматам структуру.

Форматы отличаются друг от друга диапазоном представимых чисел (см. табл. 1).

Таблица 1. Граничные значения вещественных чисел

Число	Общий вид	Точность			
		Однократная	Двойная	Расширенная двойная	Четырехкратная
Минимальное положительное	2^{-b+1}	2^{-126}	2^{-1022}	2^{-16382}	2^{-16382}
Максимальное положительное	$2^{b-n+k+1}\times$ $\times(2^{n-k}-1)$	$2^{104}\times$ $\times(2^{24}-1)$	$2^{971}\times$ $\times(2^{53}-1)$	$2^{16319}\times$ $\times(2^{65}-1)$	$2^{16271}\times$ $\times(2^{113}-1)$
Максимальное, меньшее 1	$1-2^{-n+k}$	$1-2^{-24}$	$1-2^{-53}$	$1-2^{-65}$	$1-2^{-113}$
Минимальное, большее 1	$1+2^{-n+k+1}$	$1+2^{-23}$	$1+2^{-52}$	$1+2^{-64}$	$1+2^{-112}$

Основное применение в вычислительной технике и программирование получили 32-, 64- и 80-битовый форматы представления вещественных чисел. Так, например, в *C/C++* используют следующие вещественные типы данных: float (4 байта), double (8 байт) и long double (10 байт).

Итак, для формирования машинного представления вещественного числа в формате с плавающей запятой в памяти компьютера необходимо [7]:

а) перевести десятичное число в двоичный код;

б) нормализовать полученное двоичное число путем перемещения плавающей запятой вправо (для чисел по модулю <1) или влево (для чисел по модулю >1) так, чтобы старшая единица мантиссы являлась единицей целой части; количество перенесенных разрядов определяет положительный (при перемещении запятой влево) или отрицательный (при перемещении запятой вправо) порядок для дальнейшего расчета характеристики (смещенного порядка);

в) рассчитать характеристику числа, используя формулы (2) и (3) и данные табл. 2;

Таблица 2. Значения коэффициентов для формул (2), (3)

Формат *IEEE* (байт)	Поле k, бит	Поле M, бит
single-precision (4)	8	23
double-precision (8)	11	52
single-extended precision (10)	15	64
double-extended precision (16)	15	112

г) в отведенное форматом поле (см. рис.1) записать знак числа (0 – для положительных, 1 – для отрицательных значений), характеристику и нормализованную мантиссу (целая часть мантиссы не хранится в памяти ЭВМ);

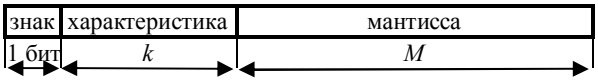

Рис.1. Основные форматы представления чисел с плавающей запятой в *IEEE*-стандарте

д) переставить байты таким образом, чтобы на первом месте стоял младший, а на последнем – старший байт числа (метод хранения числовых данных в современных ПК).

Рассмотрим пример представления в памяти современного ПК вещественного числа $-227{,}1875_{10}$ в 4-байтовом формате single-precision:

а) перевод $-227{,}1875_{10}$ в двоичный код:

$-227{,}1875_{10}=-11100011{,}0011_2$
$\quad 7\,6\,5\,4\,3\,2\,1\,0\,{-1}{-2}{-3}{-4}$

б) нормализация двоичного числа $-11100011{,}0011_2$:

$-11100011{,}0011_2=-1{,}10001100112\cdot 2^7 \Rightarrow p=7$;

$M=11000110011000000000000_2$

в) определение характеристики:

$X=2^{8-1}-1+7=134_{10}=10000110_2$

г) представление числа $-227{,}1875_{10}$ в IEEE-стандарте:

11000011 01100011 00110000 00000000

д) машинное представление числа $-227{,}1875_{10}$:

00000000 00110000 01100011 11000011

Представление в памяти вещественных чисел прочих форматов IEEE осуществляется по абсолютно аналогичному сценарию.

Выводы

В работе исследован и описан алгоритм современного подхода к формированию машинного представления и хранения числовой информации в формате с плавающей запятой. Рассмотрен пример представления в памяти современного ПК вещественного числа в формате single-precision. Показано, что проблемы, связанные с точностью компьютерных вычислений, обусловлены ограничением разрядности аппаратной реализации ЭВМ.

Вышесказанное требует от разработчиков программного обеспечения предметного понимания алгоритмов формирования машинного представления и хранения данных, методов компьютерной реализации этих алгоритмов с целью повышения эффективности проводимых численных исследований.

ЛИТЕРАТУРА

1. IEEE Standard for Binary Floating-Point Arithmetic. — New York, 1985. — 23 p.
2. IEEE Standard for Floating-Point Arithmetic. — New York, 2008. — 70 p.
3. IEEE floating point – Wikipedia, the free encyclopedia.– Режим доступа : http://en.wikipedia.org/wiki/IEEE_floating_point.
4. Оцінка точності обчислень спеціальних функцій при розробці комп'ютерних програм математичного моделювання / О. Я. Ніконов, О. В. Мнушка, В. М. Савченко // Вісник НТУ «ХПІ». Тематичний випуск : Інформатика і моделювання. — Харків : НТУ. — 2011. — № 17. — С. 115—121.
5. Кулямин В. В. Формальные подходы к тестированию математических функций // Труды ИСП РАН. — 2006. — № 10. — С. 69—114.
6. Аноприенко А. Я., Иваница С. В. Гибкая разрядность и постбинарные форматы представления вещественных чисел // Вестник Инженерной Академии Украины. Теоретический и научно-практический журнал Инженерной Академии Украины. Выпуск 1. — Киев, 2012. — С. 92—98.
7. Жульковський О. О., Жульковська І. І. Зображення чисел з плаваючою комою у сучасних комп’ютерах // Тези доп. Міждерж. науково-метод. конф. «Проблеми математичного моделювання». — Дніпродзержинськ : ДДТУ, 2011. — С.134.

пост.25.09.13