УДК 004.222:004.43

# ОЦЕНКА ДОПОЛНИТЕЛЬНОЙ ПОГРЕШНОСТИ РЕЗУЛЬТАТОВ ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ ИНФОРМАЦИОННО-ТЕЛЕМАТИЧЕСКИХ СИСТЕМ

**О. В. Мнушка, асп.,**
**Харьковский национальный автомобильно-дорожный университет**

***Аннотация.*** *Проведен анализ источников дополнительной погрешности результатов имитационного моделирования информационно-телематических систем, обусловленных форматом представления вещественных данных в памяти компьютера и реализацией математических и специальных функций в стандартных и прикладных библиотеках языков программирования C и C++.*

***Ключевые слова:*** *имитационное моделирование, погрешность, точность, IEEE 754, специальные функции.*

# ОЦІНЮВАННЯ ДОДАТКОВОЇ ПОХИБКИ РЕЗУЛЬТАТІВ ІМІТАЦІЙНОГО МОДЕЛЮВАННЯ ІНФОРМАЦІЙНО-ТЕЛЕМАТИЧНИХ СИСТЕМ

**О. В. Мнушка, асп.,**
**Харківський національний автомобільно-дорожній університет**

***Анотація.*** *Проведено аналіз джерел додаткової похибки результатів імітаційного моделювання інформаційно-телематичних систем, обумовлених форматом надання дійсних даних у пам'яті комп'ютера та реалізацією математичних і спеціальних функцій у стандартних та прикладних бібліотеках мов програмування C і C++.*

***Ключові слова:*** *імітаційне моделювання, похибка, точність, IEEE 754, спеціальні функції.*

# ESTIMATION OF THE COMPLEMENTARY ERROR OF INFORMATION AND TELEMATICS SYSTEMS SIMULATION

**O. Mnushka, P. G.,**
**Kharkiv National Automobile and Highway University**

***Abstract.*** *The paper deals with the sources of additional errors in the results of simulation of information and telematics systems, caused by the format of real data in the computer memory and implementation of mathematical and special functions in the standard and applied libraries of the C and C++ programming languages.*

***Key words:*** *simulation, tolerance, accuracy, IEEE 754, special functions.*

## Введение

Имитационное моделирование компонентов, узлов, информационно-телематических и мехатронных систем в целом позволяет ускорить процесс модернизации существующих и внедрения новых систем. Математическое и специальное программное обеспечение, получившее распространение для решения подобных задач, в ряде случаев не гарантирует получение адекватных результатов моделирования, что обусловлено известными ограничениями современных вычислительных систем на представление и обработку данных.

Основная масса программного обеспечения, ориентированного на научные вычисления, написана на языках программирования C и

C++, которые наряду с фортраном и python являются основными языками программирования для решения вычислительных задач.

Из-за специфики работы с данными и реализации стандартных и прикладных библиотек C и C++ возникают дополнительные погрешности вычислений, а в ряде случаев и грубые ошибки, понимание природы которых должно обеспечить повышение достоверности моделирования и уменьшения накладных расходов при разработке и внедрении новых информационно-телематических, встраиваемых, мобильных и др. систем.

**Анализ публикаций**

Вещественные данные в памяти вычислительной системы представлены в нормализованном виде в формате IEEE 754 $\pm M \cdot q^{\pm p}$, где $M$ – мантисса, $q$ – основание системы счисления, $p$ – порядок (экспонента) (рис. 1) [1].

| Знак | Экспонента | | | Мантисса | | | |
|---|---|---|---|---|---|---|---|
| S | k | … | 0 | (h) | n | … | 0 |

Рис. 1. Формат вещественных чисел IEEE 754

Стандарт IEEE 754-2008 определяет четыре базовых бинарных формата и три десятичных формата, способы реализации различных арифметических операций, обработку исключительных ситуаций и т. д. Стандартные размеры мантиссы, которая определяет точность, 9 (half), 23 (single), 53 (double) и 113 (quad) двоичных или $\log_{10} 2^n$ десятичных разрядов. В ряде форматов для увеличения точности используется скрытый разряд ($h$). На платформе Intel™ с сопроцессором (FPU) × 87 для повышения точности вычислений используется расширенный (extended, M = 64) формат.

При работе с числами в формате IEEE 754 возникает ряд проблем, связанных с ограниченной разрядной сеткой и некорректным округлением результатов вычислений [2].

Для преодоления недостатков формата IEEE 754 в работе [3] предлагается так называемый постбинарный формат хранения чисел. В настоящее время отсутствует его аппаратная и программная поддержка, что не дает возможности оценить его преимущества и недостатки.

Для снижения дополнительной погрешности вычислений используют специальные приемы – интервальную и обычную арифметику произвольной точности [3], которые реализованы в библиотеках ЯП С:
– GNU MP (https://gmplib.org/);
– GNU MPFR (http://www.mpfr.org/);
– MPFI (http://perso.ens-lyon.fr/ nathalie. revol/software.html ) и др.

Помимо погрешностей вычислений, связанных с форматом хранения данных, различают погрешности вычислений в библиотеках языков программирования [5], обусловленные способами реализации элементарных и специальных функций. Общие подходы к разработке и тестированию функций без привязки к определенному языку программирования представлены в стандарте ISO/IEC 10967. Information technology. Language independent arithmetic [6], при этом библиотеки существующих ЯП уже реализованы без учета требований данного стандарта.

Следует отметить, что, как правило, при реализации стандартных и прикладных библиотек не указываются величина дополнительной погрешности, закон их изменения и т. п. В настоящее время погрешность машинных вычислений определяют в единицах ULP (unit in the last place, unit of least precision) [7], которая для арифметических операций не превышает 0,5 ULP, а для трансцендентных функций – ~ (0,5…1) ULP.

Таким образом, существующие на сегодняшний день прикладные библиотеки C и C++ обеспечивают различную погрешность вычислений одних и тех же функций. С учетом того, что эти библиотеки могут иметь различный набор функций, существует проблема их совместного использования и оценки погрешности результатов смешанных вычислений.

**Цель и постановка задачи**

Повышение достоверности результатов имитационного моделирования информационно-телематических систем на основе оценки дополнительной погрешности результатов, обусловленной спецификами реализации специальных математических функций в библиотеках языков программирования C и C++ и представления вещественных данных в памяти компьютера.

### Оценка дополнительной погрешности результатов имитационного моделирования

Рассмотрим проблемы, обусловленные форматом хранения данных и способами реализаций специальных функций в библиотеках ЯП С и С++, которые возникают при имитационном моделировании (ИМ) и выражаются в виде дополнительной погрешности вычислений.

В работе [8] для определения вероятности ошибки в цифровой спутниковой информационно-телематической системе используется отношение (1) с подынтегральной дополнительной функцией ошибки $erfc(x)$ (2) [9]

$$P(x;y)=\mathbf{E}\left[\mathrm{erfc}\left(\rho(x;y)S+\rho(x;y)Z(x;y)\right)\right], \quad (1)$$

где $\mathbf{E}[.]$ – символ математического ожидания; $\rho$ – отношение сигнал-шум; $S=\sin(\pi/M)$; $M$ – число положений фаз фазовой манипуляции; $Z$ – помеха (случайная величина).

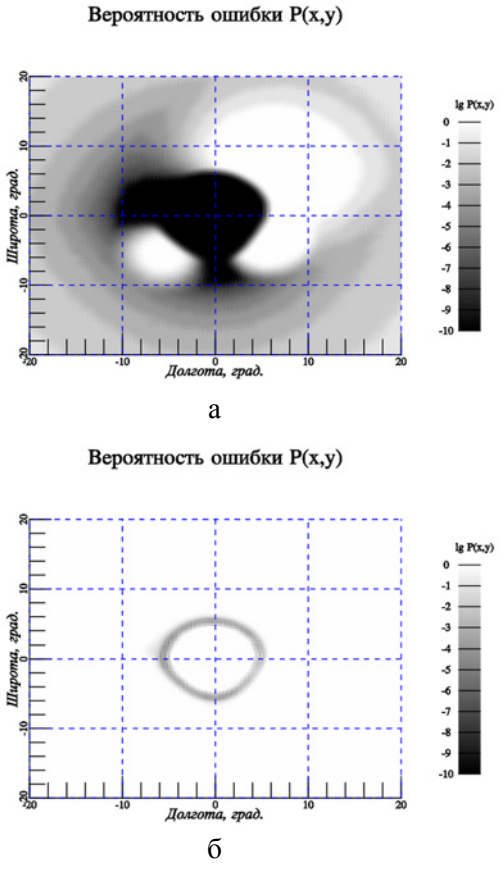
$$erfc(x)=1-erf(x)=\frac{2}{\sqrt{\pi}}\int_x^{\infty}e^{-t^2}dt, \quad (2)$$

где $erf(x)=\frac{2}{\sqrt{\pi}}\int_0^{x}e^{-t^2}dt$ – функция ошибки

Для моделирования (1) была составлена программа на С++ с использованием библиотеки математических функций cmath и компилятора gcc/g++, который поддерживает расширенный набор математических и специальных функций в соответствии со стандартом С99.

Результаты моделирования отношения (1) с подынтегральной функцией $1-erf\ x$ («А») и подынтегральной функцией $erfc\ x$ («Б») приведены на рис. 2,а и 2,б соответственно. Результат «А» правильный, т. к. подтверждается результатами ИМ, полученными при использовании других математических моделей [10].

Проанализируем источники дополнительной погрешности, которые привели к искажению результатов моделирования.




а

б

Рис. 2. Результаты моделирования: а – с подынтегральной функцией $erfc\ x$; б – с подынтегральной функцией $1-erf\ x$

Рассмотрим формат хранения вещественных данных в памяти вычислительной системы. Как правило, в научных вычислениях используют формат double (например, GNU Scientific Library), однако в некоторых случаях точности, предоставляемой этим форматом, недостаточно. Существует множество синтетических примеров, например выражение (3) [4], которые позволяют исследовать точность вычислений в формате с плавающей точкой

$$d=173746\cdot\sin 10^{22}+94228\cdot\log_2 17{,}1-$$
$$-78487\cdot e^{0{,}42}= \quad (3)$$
$$=-1{,}3418189578296195\cdot 10^{-12}$$

В процессе его вычисления (3) происходит некорректное округление слагаемых. Анализ результатов вычисления выражения (3) (табл. 1) показывает, что в ряде случаев (рез. 5–9) двойной и расширенной точности оказывается недостаточно. Результаты полученные для Matlab (рез. 8) и С++ (рез. 9) полностью «выпадают», что можно объяснить спецификой

работы программ на этих языках с данными вещественных типов.

Также не дает точного ответа использование интервальной арифметики расширения языка C++ C-XSC [11]. Следует отметить, что реализация long double зависит от разработчика компилятора [12], результат вычислений для данных этого типа (рез. 5) выглядит правдоподобным, но ошибка присутствует уже во втором знаке после запятой. Результаты 1–4 являются правильными.

Таблица 1 Результаты вычисления выражения (3)

| № | Вариант программы | Точность (дес. разр.) | Результат |
|---|---|---|---|
| 1 | Maple 16 (mpfr) | 34 | -1.341818957…e-12 |
| 2 | C++(mpfr) | >34 | -1.341818957…e-12 |
| 3 | C++(mpfi, 64 байта) | >34 | -1.341818957…e-12 |
| 4 | C(quadmath, __float128) | 34 | -1.341818957…e-12 |
| 5 | C (long double) | 19 | -1.314504061…e-12 |
| 6 | C (double), | 16 | 2.910383045…e-11 |
| 7 | Scilab (double) | 16 | 2.910383045…d-11 |
| 8 | Matlab R2012b | 16 | 1.455191528…e-11 |
| 9 | C++ (long double, 5 байт) | 19 | 1.018557979…e-312 |
| 10 | C-XSC (l_real 16 байт) | 16 | 8.3456349137e-12 |

В основе реализации специальных функций (2) лежат разложения в ряд [9, 13] и рациональная аппроксимация (Паде-аппроксимация) [14].

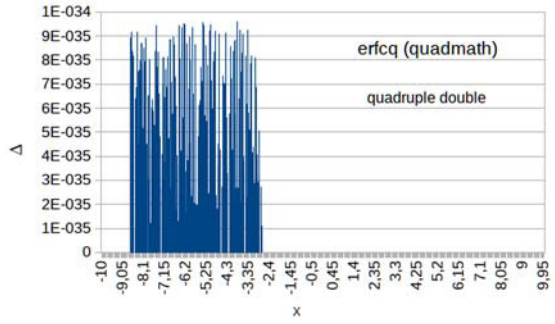
Автором были проанализированы исходные коды реализации специальных функций в библиотеках языков программирования fortran, C и C++ – C (glibc 2.18) и C++ (libstdc++ 4.8.2) и прикладных библиотеках – quadmath (128-битные числа), boost::math (boost 1.54) и GNU scientific library (gsl 1.15), mpfr 3.1.2. В основе реализаций (2) в стандартных библиотеках (glibc, libstdc++) и библиотеки quadmath используется реализация компании Sun Microsystems 1993 г. на основе (7.1.5) и (7.2.14) [13] и рациональной аппроксимации [14] с модификациями, связанными со схемой разбиения функции (2) на интервалы. В остальных библиотеках используются различные варианты рациональной аппроксимации выражения (2). В фортране, форте и др. используется напрямую аппроксимация [14]. Наиболее поздние реализации таких библиотек относятся к 2000-м годам, т. е. еще до введения стандарта ISO/IEC 10967.

Для анализа рассмотренных вариантов реализации функций был проведен численный эксперимент. Параметры компьютера – процессор Intel Core2 Duo T5750, ОЗУ 4 Гб, ОС Linux (i686, amd64, kernel 3.13), FreeBSD 9.2 (i386).

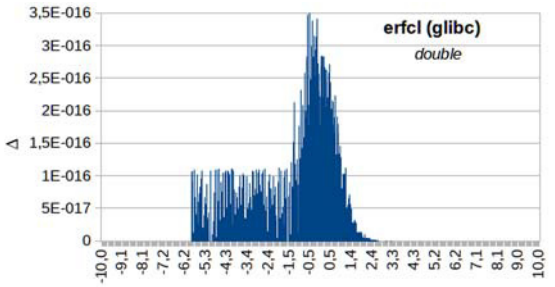
Для анализа погрешности вычислений, в соответствии с рекомендациями в работе [5], в качестве эталонной выбрана реализация (2) в библиотеке mpfr (вычисления с произвольной точностью и корректным округлением). Для определения абсолютной погрешности вычислений ($\Delta$) определялась погрешность вычислений функций *erf x* и *erfc x* с точностью 40 десятичных разрядов (д. р.) с использованием функций из библиотеки mpfr и соответствующих реализаций функций в анализируемых библиотеках с оптимальными типами данных – __float128 (16 байт) для библиотеки quadmath, long double (10 байт) и double (8 байт) – для реализаций в библиотеках glibc и boost, double – для реализации в библиотеке gsl соответственно.

Ввиду того, что все вычисления данных с плавающей точкой в программах на C/C++ производятся в данных двойной (*BSD, Win32) или расширенной (Linux) точности, погрешности вычислений с другими базовыми типами данных не анализировались.

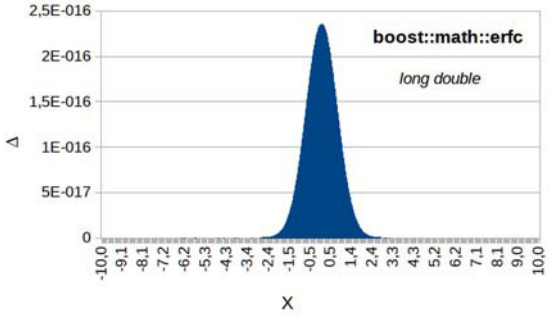
Получены значения погрешности вычисления (2) для различных библиотек (рис. 3):
а) для данных учетверенной точности (рис. 3,а, quadruple double, quadmath) абсолютная погрешность $\Delta$ не превышает $1\cdot10^{-34}$ для $X\in[-9;-3]$, что соответствует точности в 34 десятичных разряда;
б) для данных расширенной точности (long double) функция (2) реализована в glibc (рис. 3,б) и boost::math (рис. 3,в) абсолютная погрешность $\Delta$ составляет:
– $(1\ldots3{,}5)\cdot10^{-16}$ для $X\in[-6;2{,}5]$ и реализации glibc;
– $(0{,}1\ldots2{,}25)\cdot10^{-16}$ $X\in[-2{,}5;2{,}5]$ и реализации boost::math.



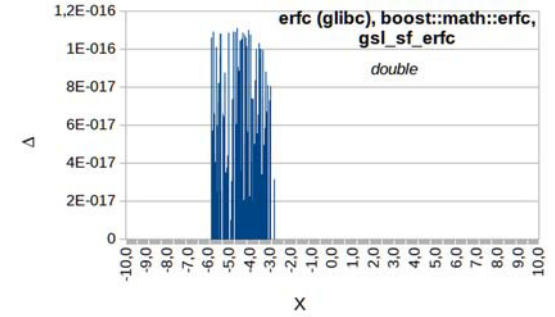

а




б




в




г

Рис. 3. Погрешность вычисления функции $erfc\,x$: а – для данных учетверенной точности; б – для данных расширенной точности, реализована в glibc; в – для данных расширенной точности, реализована в boost::math; г – для данных двойной точности

Абсолютная погрешность вычислений для данных этого типа составляет приблизительно 16 десятичных разрядов, что на три порядка хуже достижимой точности, которая составляет 19 десятичных разрядов.

в) для данных двойной точности (double) $\Delta \approx (1{,}1\ldots1{,}2)\cdot 10^{-16}$ в диапазоне изменения аргумента $X \in [-6;-3]$. В данном случае погрешность вычислений одного порядка указана с достижимой точностью (рис 3,г).

Для вычислительных платформ IA32 (i686) и amd64 для рассматриваемых функций получены и проанализированы значения погрешности вычислений. Специфика реализации вычислений над данными с плавающей точкой для данных платформ, обусловленная аппаратной реализацией модулей FPU и SSE/SSE2, может проявляться в том случае, когда производится компиляция программ с параметрами компилятора, заданными по умолчанию. В компиляторах gcc/g++ для 32-битной платформы для этих целей используется сопроцессор, а на 64-битной – механизм SSE/SSE2-расширений [15].

Для анализируемых функций дополнительная погрешность вычислений составляет:

а) $|\Delta| < 5\cdot 10^{-16}$ для функции $erf\,x$ из библиотеки gsl и аргумента $X \in [-1{,}7;1{,}8]$ (рис. 4,а);

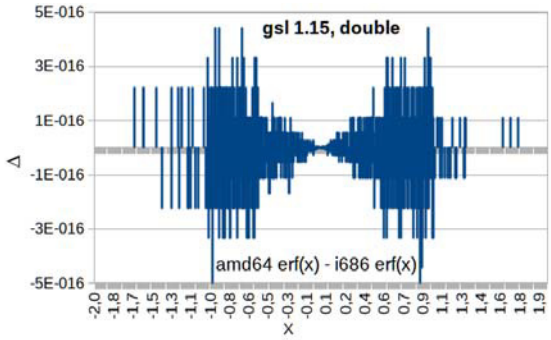
б) $|\Delta| < 1{,}2\cdot 10^{-16}$ для функции $erf\,x$ из библиотеки glibc и аргумента $X \in [-1{,}7;1{,}8]$ (рис. 4,б);

в) $\Delta \le -1{,}2\cdot 10^{-16}$ для функции $erfc\,x$ из библиотеки glibc и аргумента $X \in [-6;-2]$ и $|\Delta| \le 1{,}5\cdot 10^{-16}$ для $X \in [-2;2]$ (рис. 4,в).

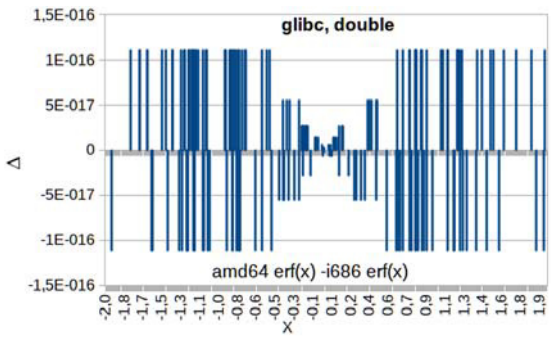
Для библиотек boost и quadmath вычисления для обеих рассмотренных платформ проводятся с одинаковой точностью.

Погрешности вычисления выражения $erfc\,x = 1 - erf\,x$ были рассчитаны для аргумента $X \in [-10;10]$ с шагом $\delta = 0{,}001$ и данных двойной и расширенной точности (рис. 5, в логарифмическом масштабе). При вычислениях использовались только оптимальные для данной библиотеки типы данных.

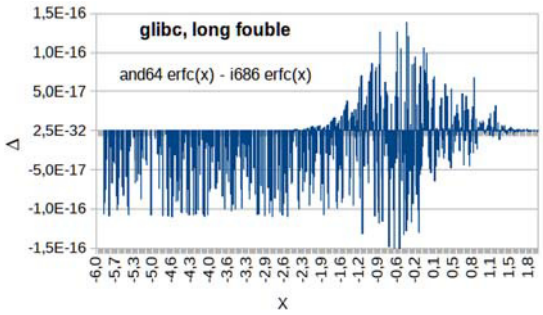
Для библиотек boost и glibc получены схожие результаты (рис. 5,а). Для данных расширенной точности для $X \in [-6;2]$ наблюдается существенный рост погрешности вычислений (на три порядка), а точность вычислений сопоставима с точностью вычислений с данными удвоенной точности.




а




б

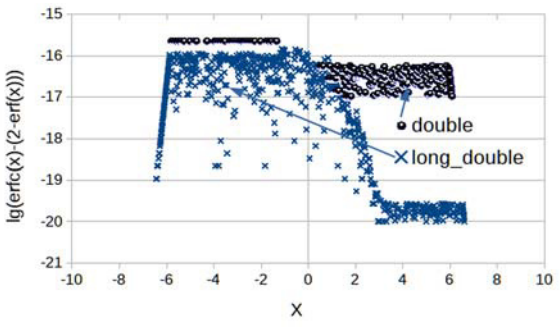



в

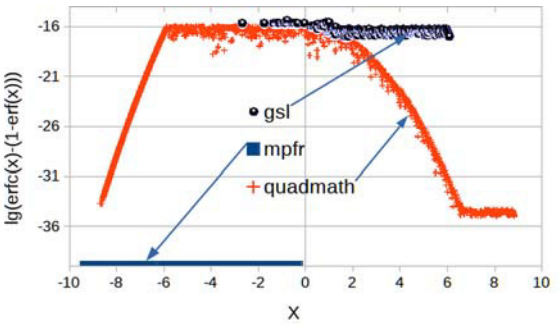
Рис. 4. Погрешность вычислений для платформ amd64 и i686: а – для функции *erf x* из библиотеки gsl; б – для функции *erf x* из библиотеки glibc; в – для функции *erfc x* из библиотеки glibc

Для данных удвоенной точности погрешность вычислений составляет 15–16 д. р., т. е. находится в стандартных границах для данных этого типа.

Для библиотек mpfr погрешность вычислений постоянна и сопоставима с максимально выбранной точностью вычислений, в данном случае 40 д .р. (рис. 5, б). Для библиотеки gsl небольшой рост погрешности вычислений наблюдается для $X \in [-2; 2]$, которая, в целом, находится на уровне погрешности вычислений для этого типа данных (рис. 5, б).




а




б

Рис. 5. Абсолютная погрешность $\Delta = erfc\ x - (1 - erf\ x)$: а – glibc, boost; б – gsl, mpfr, quadmath

Для библиотеки quadmath точность вычислений уменьшается в 2 раза для $X \in [-6; 2]$, а закон изменения погрешности такой же, как и для данных удвоенной точности (рис. 5, а), но дисперсия значительно меньше (рис. 5, б).

## Выводы

Характерная для современных цифровых вычислительных систем ограниченная размерная сетка, используемая для представления данных в их памяти, обуславливает дополнительную погрешность результатов ИМ, которая может оказаться существенной и соизмеримой с исходными данными.

Одни и те же ошибки в вычислениях, обусловленные форматом хранения данных, характерны и для универсальных математических пакетов, и для программ, реализующих алгоритмы решения частных задач (табл. 1).

Реализации одних и тех же математических функций, в частности специальных, зависят от библиотек языков программирования и могут основываться на различных моделях. Дополнительная погрешность результатов ИМ различается как для распространенных библиотек C и C++, так и для одних и тех же

библиотек C и C++ при их использовании на различных вычислительных архитектурах (IA-32 и amd64) и операционных системах.

Для уменьшения дополнительной погрешности результатов ИМ следует увеличивать разрядную сетку и применять соответствующие библиотеки. При этом не следует ориентироваться на платформозависимые решения, такие как формат расширенной точности, который поддерживается x87 сопроцессорами, т. к. для компиляторов C и C++ он практически реализован только в GNU Linux. В других ОС, как правило, используют вычисления в формате с удвоенной точностью.

Применение библиотек вычислений с произвольной точностью может привести к существенному падению производительности вычислений.

Для компиляторов gcc и g++ можно порекомендовать использование библиотек quadmath и boost, которые обеспечивают наименьшую дополнительную погрешность вычислений.

**Литература**


1. IEEE Standard for Floating-Point Arithmetic. – New York, 2008. – 70 p.
2. Handbook of Floating-Point Arithmetic / [J.-M. Muller, N. Brisebarre, F. de Dinechin]. – Basel: Birkhäuser, 2009. – 572 p.
3. Аноприенко А. Я. Особенности представления вещественных чисел в постбинарных форматах / А. Я. Аноприенко, С. В. Иваница // Математичні машини і системи. – 2012. – № 3. – С. 49–60.
4. Why and How to Use Arbitrary Precision / [K. Ghazi, V. Lefèvre, Ph. Théveny, P. Zimmermann] // Computing in Science and Engineering. – 2010. – Vol. 12; № 3. – P. 62–65.
5. Кулямин В. В. Стандартизация и тестирование реализаций математических функций, работающих с числами с плавающей точкой / В. В. Кулямин // Программирование. – 2007. – № 33(3). – С. 1–29.
6. ISO/IEC 10967-1:2012. Information technology. Language independent arithmetic. Part 1: Integer and floating point arithmetic. – Geneva. – 144 p.
7. Muller J.-M. On the definition of ulp(x) [Электронный ресурс] / J.-M. Muller, INRIA Technical Report 5504. – INRIA: 2005. – Режим доступа: http://www.ens-lyon.fr/LIP/Pub/Rapports/RR/RR2005/RR2005-09.pdf. – 2014.
8. Мазманишвили А. С. Визуализация информационных характеристик электромагнитной обстановки в системах спутниковой связи / А. С. Мазманишвили, О. Я. Никонов // Вісник СумДУ. Серія «Технічні науки». – 2008. – № 4. – С. 30–37.
9. Справочник по специальным функциям с формулами, графиками и математическими таблицами / [М. Абрамовиц, И. Стиган]. – М.: Наука, 1979. – 832 с.
10. Jerushim M. Simulation of Communication System. Modeling, Methodology, and Technics / M. Jerushim, P. Balaban, K. Sam Shanmugan. – Kluwer Academic Publisher, 2000. – 908 p.
11. Krämer W. Multiple/arbitrary precision interval computations in C-XSC / W. Krämer // Computing. – Springer, 2012. – Vol. 94; Is. 2–4. – P. 229–241.
12. Мнушка О. Особенности реализации формата IEEE 754 чисел с плавающей точкой в компиляторах языка C/C++ / О. В. Мнушка // Радиоэлектроника и молодежь в XXI веке: материалы 15-го Юбилейн. Межд. молодеж. фор. Сб. матер. фор.; 18–20 апр. 2011. Т. 9. Межд. конф. «Информационные интеллектуальные системы» – Х.: ХТУРЭ, 2011. – С. 240-241.
13. Computer Approximations / [J. F. Hart, E. W. Cheney, C. L. Lawson]. – 1968. – 354 p.
14. Cody W. J. Rational Chebyshev approximations for the error function / W. J. Cody // Math. Comp. – 1969. – № 23. – P. 631–637.
15. Using the GNU Compiler Collection (GCC) [Электронный ресурс]. – Режим доступа: http://gcc.gnu.org/onlinedocs/gcc-4.8.2/gcc/ – 04.2014.


Рецензент: О. Я. Никонов, доцент, д. т. н., ХНАДУ.

Статья поступила в редакцию 28 февраля 2014 г.