http://vestnik.mrsu.ru

ISSN Print 2658-4123
ISSN Online 2658-6525

УДК 004.272.34 DOI: 10.15507/2658-4123.029.201902.187-204

Метод умножения с масштабированием результата для высокоточных модулярно-позиционных интервально-логарифмических вычислений

А. С. Коржавина[1*], **В. С. Князьков**[1, 2]
[1]*ФГБОУ ВО «Вятский государственный университет» (г. Киров, Россия)*
[2]*ФГБОУ ВО «Пензенский государственный университет» (г. Пенза, Россия)*
[*]*as_korzhavina@vyatsu.ru*

Введение. Для решения задач моделирования, критичных к ошибкам округления (включая задачи вычислительной математики, математической физики, оптимального управления, биохимии, квантовой механики, математического программирования и криптографии), требуется точность от 100 до 1 000 десятичных цифр и более. Главным недостатком программных библиотек высокоточных вычислений является неприемлемое для практических задач снижение быстродействия, в особенности при выполнении операции умножения. Одним из способов повышения скорости вычислений над длинными числами является использование системы счисления в остаточных классах. В данной статье рассматривается новый, более быстрый по сравнению с аналогами метод выполнения операции умножения длинных чисел с масштабированием результата за счет применения оригинальной гибридной модулярно-позиционной интервально-логарифмической формы представления чисел с плавающей точкой.
Материалы и методы. Для повышения скорости вычислений разработан новый способ организации числовой информации (модулярно-позиционная интервально-логарифмическая форма), в котором мантисса представлена в системе остаточных классов, а информация об абсолютной величине числа (характеристика) – в интервально-логарифмической системе счисления, что позволяет ускорить выполнение операций сравнения и масштабирования. Для сравнения модулярных чисел применяются положения интервального анализа; для масштабирования модулярных чисел – свойства логарифмической системы счисления, а также приближенные интервальные вычисления с использованием китайской теоремы об остатках.
Результаты исследования. Разработан и запатентован новый быстрый метод умножения модулярных чисел с плавающей точкой, проведена оценка быстродействия разработанного метода, выполнено сравнение данного метода с классическими и конвейерными методами умножения длинных чисел.
Обсуждение и заключение. Разработанный метод в 2,4–4,0 раза быстрее конвейерного метода умножения и в 6,4–12,9 раз – классических методов умножения.

Ключевые слова: система остаточных классов, высокоточное вычисление, умножение чисел, масштабирование, интервальная арифметика, сравнение чисел, логарифмическая система счисления

Финансирование: Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 18-37-00278.

Для цитирования: Коржавина А. С., Князьков В. С. Метод умножения с масштабированием результата для высокоточных модулярно-позиционных интервально-логарифмических вычислений // Инженерные технологии и системы. 2019. Т. 29, № 2. С. 187–204. DOI: https://doi.org/10.15507/2658-4123.029.201902.187-204

The Multiplication Method with Scaling the Result for High-Precision Residue Positional Interval Logarithmic Computations

A. S. Korzhavina[1*], **V. S. Knyazkov**[1, 2]
[1]*Vyatka State University (Kirov, Russia)*
[2]*Penza State University (Penza, Russia)*
[*]*as_korzhavina@vyatsu.ru*

Introduction. The solution of the simulation problems critical to rounding errors, including the problems of computational mathematics, mathematical physics, optimal control, biochemistry, quantum mechanics, mathematical programming and cryptography, requires the accuracy from 100 to 1 000 decimal digits and more. The main lack of high-precision software libraries is a significant decrease of the speed-in-action, unacceptable for practical problems, in particular, when performing multiplication. A way to increase computation performance over very long numbers is using the residue number system. In this work, we discuss a new fast multiplication method with scaling the result using original hybrid residue positional interval logarithmic floating-point number representation.
Materials and Methods. The new way of the organizing numerical information is a residue positional interval logarithmic number representation in which the mantissa is presented in the residue number system, and information on an absolute value (the characteristic) in the interval logarithmic number system that makes it possible to accelerate performance of comparison and scaling is developed to increase the speed of calculations; to compare modular numbers, the provisions of interval analysis are used; to scale modular numbers, the properties of the logarithmic number system and approximate interval calculations using the Chinese reminder theorem are used.
Results. A new fast multiplication method of floating-point residue-represented numbers is developed and patented; the authors evaluated the developed method speed-in action, compared the developed method with classical and pipelined multiplication methods of long numbers.
Discussion and Conclusion. The developed method is 2.4–4.0 times faster than the pipelined multiplication method, and is 6.4–12.9 times faster than classical multiplication methods.

Keywords: residue number system, high-precision computations, multiplication, scaling, interval arithmetic, comparison, logarithmic number system

Funding: The study was funded by the Russian Foundation for Basic Research (project No. 18-37-00278).

For citation: Korzhavina A.S., Knyazkov V.S. The Multiplication Method with Scaling the Result for High-Precision Residue Positional Interval Logarithmic Computations. *Inzhenernyye tekhnologii i sistemy* = Engineering Technologies and Systems. 2019; 29(2):187-204. DOI: https://doi.org/10.15507/2658-4123.029.201902.187-204

Введение

Рост вычислительных мощностей современных компьютеров делает возможным решение прикладных задач сверхбольшой размерности с огромным количеством вычислительных операций. Неконтролируемые ошибки округления, методологически присущие стандарту вещественных чисел IEEE 754, не позволяют решить проблему точности, достоверности и воспроизводимости вычислений при решении

задач данного класса [1–5]. В частности, для решения в современных постановках задач в области экспериментальной вычислительной математики [1; 2], математической физики [4], биохимии [3], астрофизики и получения достоверных результатов требуются операции с числами длиной от 100 до 1 000 десятичных цифр (с использованием специально разработанных программных библиотек высокоточных вычислений). В связи с этим актуальными направлениями исследований являются теория и способы практической реализации вычислительной математики многократной точности (высокоточная, или длинная, арифметика), оперирующей с числами произвольной длины в сверхбольших числовых диапазонах.

Для решения задач в сверхбольших числовых диапазонах в настоящее время применяются такие специализированные программные пакеты высокоточных вычислений, как ARPREC, MPFUN90, DDFUN, FMLIB, FMZM90, QD, GMP, MPFR++, NTL, PARI/GP, CLN, HPAlib, Predicates, GARPREC, GQD, MatLab, Matematica, Maple [6]. Перечисленные программные решения базируются на специально разработанных многоразрядных форматах (128-, 256-, 512-битная (и более) арифметика) в базисах классических позиционных систем счисления и правил вычислений стандарта IEEE 754. Эти решения, благодаря наличию высокоуровневых программных интерфейсов и широкого спектра реализованных библиотек математических функций, являются наиболее популярными.

Недостатком современных пакетов высокоточных вычислений является резкое снижение скорости вычислений при обработке длинных многоразрядных операндов. При выходе длины операндов за диапазон представления данных в стандарте IEEE 754 скорость вычислений снижается в десятки и тысячи раз [2; 7] из-за необходимости алгоритмической обработки цепочек межзнаковых переносов длинных операндов. В итоге время решения задачи становится неприемлемым для практической деятельности.

В связи с этим активно проводятся исследовательские и опытно-конструкторские работы по модернизации известных методов, созданию новых программно-эмулируемых и программно-аппаратных реализаций методов численной обработки информации для высокоточных и достоверных расчетов в сверхбольших числовых диапазонах.

Можно выделить два направления работ, направленных на повышение скорости вычислений при выполнении расчетов в сверхбольших числовых диапазонах. Первое направление ориентировано на модернизацию и создание новых технологий гибридных вычислений и обработки данных: численная и нечисленная обработка данных реализуется в гибридных системах кодирования с использованием «гибридных» наборов операций (сигнатур) и правил их выполнения. Математические методы и их алгоритмические решения для гибридных технологий вычислений ориентируются на программную реализацию на вычислительных платформах универсального назначения и, как правило, опираются на правила выполнения операций стандарта IEEE 754. Примером успешного использования этого подхода является библиотека высокоточной модулярно-позиционной арифметики [8], где использованы системы счисления в остаточных классах (СОК), вычисления в интервальной арифметике и позиционная система счисления стандарта IEEE 754.

Вторым направлением работ для повышения скорости вычислений в сверхвысоких числовых диапазонах является разработка специализированных средств аппаратной поддержки операций над сверхбольшими операндами, разрядность которых многократно превышает разрядную сетку современных индустриальных процессоров. Попу-

лярной технологической базой для создания таких спецпроцессоров «длинной» арифметики являются программируемые логические интегральные схемы (FPGA) и системы на кристалле [9–14]. Применение таких спецпроцессоров позволяет сократить время расчетов по сравнению с программными решениями в несколько десятков раз, но недостатки, присущие позиционной длинной арифметике, сохраняются [13]. Эти методологические недостатки позиционной арифметики приводят к необходимости построения на аппаратном уровне исполнительных устройств высокой сложности, что в конечном итоге делает невозможным создание применимых технических решений. Данная проблема частично решается введением специализированных вычислительных конвейеров; однако, как показано в работе китайских ученых [12], подобный подход также ведет к резкому увеличению аппаратных затрат, поэтому на практике число ступеней сокращается до четырех сегментов.

В связи с этим при создании средств аппаратной поддержки длинной арифметики актуален подход, ориентированный на создание вычислительных платформ, поддерживающих на аппаратном уровне технологии гибридных вычислений, что позволяет сократить аппаратные затраты по сравнению с позиционной системой счисления. Серьезный вклад в развитие этого направления внесли И. Я. Акушский, Д. И. Юдицкий, В. М. Амербаев и целый ряд не менее значимых специалистов. Наиболее широко в системах гибридных вычислений используются системы счисления в остаточных классах [8; 15; 16] и логарифмические системы счисления [17–19]. Например, системы остаточных классов успешно используются для решения задач криптографии [20; 21] и цифровой обработки сигналов [22–24].

Основным недостатком систем счисления в остаточных классах является алгоритмическая сложность выполнения немодульных операций, таких как сравнение, деление, распознавание переполнения числового диапазона, масштабирование чисел, определение знака результата выполнения операции. При вычислениях в сверхбольших числовых диапазонах выполнение перечисленных операций приводит либо к сопоставимым с их программной реализацией временным затратам, либо к практически неприемлемым аппаратным затратам. Аналогичная ситуация происходит и при использовании логарифмических систем счисления, в которых для выполнения операции алгебраического сложения с высокой точностью требуется выполнить переход в позиционную систему счисления и наоборот. Соответственно, резко увеличивается время вычислений и растут аппаратные затраты на реализацию высокоскоростных преобразователей.

В данной статье рассматривается новый, более быстрый по сравнению с аналогами метод выполнения операции умножения длинных чисел с масштабированием результата за счет применения оригинальной гибридной модулярно-позиционной интервально-логарифмической формы представления чисел с плавающей точкой. Ряд результатов по модулярно-позиционным интервально-логарифмическим вычислениям опубликован авторами ранее [25].

Обзор литературы

Основной проблемой высокоточных расчетов в сверхбольших числовых диапазонах с применением вычислительных операций по правилам стандарта IEEE 754 является выполнение контроля ошибок округления, контроля переполнения диапазона и масштабирования чисел при выполнении аддитивных и мультипликативных операций. Особенно это касается длительных итерационных и автоматных вычислений с накоплением при обработке массивов данных большого объема. Накопление ошибок при некорректно организованном контроле приводит к получению недостоверных результатов. Для обес-

печения требуемой точности, достоверности и воспроизводимости расчетов в настоящее время применяются вычисления с использованием длинной позиционной арифметики, реализованной в современных специализированных библиотеках высокоточных вычислений. Основным недостатком современных библиотек длинной арифметики является неприемлемое для практики увеличение времени решения прикладных задач [4; 6]. Так, в работе китайских и итальянских ученых [7] для задач оптимального управления время вычислений возрастает с 5 с при использовании стандартного типа данных двойной точности до 980 с при использовании точности в 128 бит и до 35 ч – при использовании точности в 400 бит. Аналогичные результаты представлены в работе А. Вороса [2], где время вычисления возрастает с 4 минут до 22,5 дней. В задачах криптографии проблема ускорения арифметических операций над длинными целыми числами является не менее острой, чем в задачах моделирования [13].

Задача повышения скорости вычислений при выполнении расчетов в сверхбольших числовых диапазонах частично решается за счет применения специализированных процессоров-ускорителей для поддержки вычислений с использованием длинной арифметики. Например, японские ученые [9] представили семейство процессоров на базе FPGA, реализующих длинную арифметику типа «double-double» и «quad-double». Скорость решения задачи вычисления интегралов Фейнмана [10] с использованием данных процессоров приблизительно в 80–200 раз выше, чем скорость расчета с применением программных реализаций таких вычислений. Американскими учеными [11] приведены результаты реализации на FPGA целочисленных вычислений на длинных (разрядность – 64 000 бит) операндах в сравнении с вычислениями с применением библиотеки GMP: расчеты ускорились минимум в 5 раз при операциях сложения/вычитания и в 9 раз – при операции умножения.

Для ускорения выполнения операции умножения длинных чисел (1 024–2 048 бит) китайскими учеными [12] представлен конвейерный метод на базе 64-разрядных умножителей с глубиной конвейера до четырех ступеней (увеличение числа ступеней приводит к неоправданному росту аппаратных затрат). Как показывает анализ исследований, в области аппаратных решений умножителей в основном применяются базовые алгоритмы умножения квадратичной сложности в позиционной системе счисления [13; 14], поскольку аппаратная реализация асимптотически быстрых алгоритмов затруднена [26].

Для ускорения выполнения арифметических операций (кроме операции деления) над длинными целыми числами наиболее эффективными с точки зрения аппаратных затрат являются модулярные системы счисления. Например, исследователями [23] представлено устройство эллиптической криптографии, ускоряющее выполнение операции умножения Монтгомери с использованием 40 модулярных 15-битовых каналов. В работе австралийских ученых [16] модулярный вычислитель имеет 108 модулярных каналов с разрядной сеткой в 19 бит каждый, что позволяет работать в диапазоне чисел до 2 048 бит. Другими авторами [20] представлены модулярные устройства, позволяющие работать в диапазоне 1 024–4 096 бит.

Существенным недостатком СОК является сложность выполнения немодульных операций, таких как масштабирование, сравнение и определение переполнения диапазона представления чисел. При переполнении диапазона следует либо останавливать вычисления (так как будет получен некорректный результат), либо расширять диапазон представления чисел, либо выполнять масштабирование чисел (если это возможно).

Алгоритмы масштабирования в СОК представлены в достаточно большом количестве исследований. Разработанные методы масштабирования либо предназначены для специальных наборов модулей [27; 28], либо для масштабирования используются специальные подстановочные таблицы [22; 29; 30]. Последний подход практически неприемлем для масштабирования модулярных чисел при использовании произвольных наборов модулей большой разрядности из-за огромного (до Тбайта) объема подстановочных таблиц.

Основной сложностью при выполнении масштабирования является операция расширения базиса. Методы расширения базиса были исследованы авторами статьи ранее [25]. В результате исследований было установлено, что наиболее быстрые методы расширения базиса выполняются с использованием приближенной китайской теоремы об остатках – вычисления так называемой позиционной характеристики модулярного числа. Учеными [8] представлена интервальная позиционная характеристика (ИПХ) числа, в которой использованы преимущества интервальной арифметики [31; 32]. Использование ИПХ позволяет учитывать в явном виде ошибки округления, а также определять достоверность вычисления данной величины. Главным недостатком ИПХ является необходимость использования операций с плавающей точкой с направленным округлением, в то время как все остальные операции в СОК выполняются над целыми числами малой разрядности.

Использование логарифмической системы счисления (ЛСС) позволяет упростить выполнение операций умножения и деления, включая масштабирование [17; 33]. ЛСС превосходят по скорости и энергоэффективности арифметику с плавающей точкой на низкой разрядности операндов: до 16 бит – на любом наборе арифметических операций [18], до 32 бит – с преобладанием операций умножения и деления [19]. Дальнейшее увеличение разрядности ЛСС приводит к экспоненциальному росту сложности выполнения операций сложения и вычитания, поэтому при больших разрядностях ЛСС значительно уступает позиционной арифметике и используется только в приложениях, не требующих высокой точности [19].

В данной статье предлагается объединить преимущества СОК, ЛСС и интервальных вычислений: для высокоточных вычислений в сверхбольших числовых диапазонах рекомендуется модулярно-позиционная интервально-логарифмическая форма представления чисел и апробация эффективности таких гибридных вычислений на примере выполнения операции умножения с масштабированием.

Материалы и методы

В статье предлагается новый способ представления целых и вещественных чисел для выполнения высокоточных и достоверных вычислений в сверхбольших числовых диапазонах: гибридная модулярно-позиционная интервально-логарифмическая форма представления чисел. Вещественные числа представляются следующим образом.

1. Мантисса вещественного числа представляется в виде целого числа в системе остаточных классов набором n остатков $\langle m_1, m_2, ..., m_n \rangle$ от деления позиционного значения мантиссы на каждый из n модулей $\{p_1, p_2, ..., p_n\}$

$$M \overset{СОК}{\rightarrow} \langle m_1, m_2, \ldots, m_n \rangle,$$

где $m_i = M \bmod p_i \equiv |M|_{p_i}$ – i-й остаток от деления числа M по i-му модулю p_i:

$$m_i = |M|_{p_i} = M - \left[\frac{M}{p_i} \right] \cdot p_i,$$
$$i = 1, 2, \ldots, n,$$

где $\left[\frac{M}{p_i} \right]$ – целая часть частного $\frac{M}{p_i}$;

$\{p_1, p_2, ..., p_n\}$ – набор оснований или базис СОК. При этом диапазон представления модулярных мантисс определяется произведением всех модулей СОК, то есть $M \in [0; P = p_1 \cdot p_2 \cdot \ldots \cdot p_n)$. Для кодирования цифр мантиссы используются целые числа без знака, представленные в позиционной системе счисления, но операции над цифрами мантиссы выполняются по правилам модулярной арифметики. Любая модульная операция $\circ \in \{+, -, \times\}$ над двумя числами $\langle x_1, x_2, ..., x_n\rangle$ и $\langle y_1, y_2, ..., y_n\rangle$, представленными в СОК, выполняется независимо по каждому модулю:

$$\{z_1, z_2, \ldots, z_n\} = \{|x_1 \circ y_1|_{p_1}, |x_2 \circ y_2|_{p_2}, \ldots, |x_n \circ y_n|_{p_n}\}.$$

2. Характеристика абсолютной величины мантиссы вещественного числа представляется в виде логарифмического интервала (в интервально-логарифмической системе счисления):

$$M \overset{\text{ИЛСС}}{\rightarrow} \{\underline{L_M} = \underline{\log_b M}; \overline{L_M} = \overline{\log_b M}\},$$

где $\underline{\log_b M}$, $\overline{\log_b M}$ – логарифм числа по основанию b, вычисленный с округлением к $-\infty$ и $+\infty$ соответственно; M – модуль числа, представленный в позиционной системе счисления. Для кодирования характеристики мантиссы вещественного числа используется двоичная позиционная система счисления, но операции над значениями характеристик чисел выполняются по правилам интервальной арифметики и логарифметики. Результат умножения двух логарифмических интервалов $[\underline{L_X} = \underline{\log_b X}; \overline{L_X} = \overline{\log_b X}]$ и $[\underline{L_Y} = \underline{\log_b Y}; \overline{L_Y} = \overline{\log_b Y}]$ определяется следующим образом:

$$\underline{L_Z} = \underline{\log_b X \cdot Y} = \underline{\log_b X + \log_b Y} = \underline{L_X} + \underline{L_Y},$$
$$\overline{L_Z} = \overline{\log_b X \cdot Y} = \overline{\log_b X + \log_b Y} = \overline{L_X} + \overline{L_Y}.$$

3. Масштаб (порядок) числа представляется в позиционной системе счисления в виде целого числа со знаком; операции выполняются также в позиционной системе счисления.

4. Знак числа представляется в позиционной системе счисления в виде одноразрядного числа со знаком; причем знак равен –1, если число отрицательное, 1 – если число положительное, и 0 – в случае равенства числа нулю. Дополнительный признак нуля вводится с целью представления интервальной логарифмической характеристики нулевого операнда, для которого невозможно вычисление логарифма.

Таким образом, число в гибридной модулярно-позиционной интервально-логарифмической форме представляется в следующем виде:

$$X \overset{\text{МПИЛ-СС}}{\rightarrow} [m_1, m_2, \ldots, m_n, \underline{L}, \overline{L}, \lambda, \sigma],$$

где $M = m_1, m_2, ..., m_n$ – модулярная мантисса числа; λ – масштаб (порядок) числа; $\underline{L}, \overline{L}$ – границы интервальной логарифмической характеристики мантиссы числа; σ – знак числа.

При этом позиционное значение мантиссы вещественного числа X определяется как $[X \cdot b^e]$, где e – целое число, определяемое необходимой точностью. Например, мантисса вещественного числа, представленного в формате с плавающей точкой стандарта IEEE 754 как $X = (-1)^s \times 1.f \times 2^{E-E_0}$, где s – знак числа; $1.f$ – нормализованная мантисса; $E - E_0$ – порядок. При переводе в гибридную форму вычисляется так:

$$M = 2^t \times 1.f,$$

где t – разрядность мантиссы, определяемая конкретным типом данных.

Итак, позиционное значение данного числа определяется следующим образом:

$$X = \sigma \cdot b^{\lambda} \cdot \sum_{i=1}^{n} \left| m_i \cdot \left| P_i^{-1} \right|_{p_i} \right|_{p_i} \cdot P_i,$$

где $P_i = \frac{P}{p_i}$, $\left|P_i^{-1}\right|_{p_i}$ – мультипликативная инверсия P_i по модулю p_i, определяемая из соотношения $\left|P_i^{-1} \cdot P_i\right|_{p_i} \equiv 1;\ i \in [1, n]$; n – количество модулей.

При выполнении арифметических операций над числами, представленными в виде (1), вероятен выход за границы диапазона представления модулярных мантисс. При переполнении диапазона следует выполнить масштабирование чисел.

Масштабирование модулярных чисел выполняется на основании общего алгоритма масштабирования: пусть K – коэффициент масштабирования; Y – результат масштабирования числа X коэффициентом K; тогда результат масштабирования вычисляется по формуле:

$$Y = \frac{X - |X|_K}{K},$$

где $|X|_K$ – остаток от деления числа X по модулю K.

Для случая масштабирования модулярных чисел коэффициентом, взаимно простым с основаниями СОК, используется итерационный алгоритм на основе алгоритма, предложенного сингапурскими и австралийскими учеными [27; 29].

1. Определение $|X|_K$, или так называемый этап расширения базиса, – получение остатка x_{n+1} от деления числа, представленного в СОК остатками $x_1, x_2, ..., x_n$ по модулям $p_1, p_2, ..., p_n$, на число $p_{n+1} = K$.

2. Непосредственно масштабирование по каждому модулю выполняется по формуле:

$$y_i = \left|\left|x_i - |X|_K\right|_{p_i} \cdot \left|K^{-1}\right|_{p_i}\right|_{p_i},$$

где $|K^{-1}|_{p_i}$ – мультипликативная инверсия по модулю p_i коэффициента K.

Основные алгоритмы расширения базиса, анализ их вычислительной сложности были рассмотрены авторами в прошлой работе [25]. В данной статье используется быстрый метод масштабирования на основании китайской теоремы об остатках (КТО).

Согласно КТО, позиционное значение числа $X \in [0, P)$, представленного в СОК остатками $\langle x_1, x_2, ..., x_n \rangle$ по основаниям $\{p_1, p_2, ..., p_n\}$, вычисляется по формуле:

$$X = \left|\sum_{i=1}^{n} \left|x_i \cdot \left|P_i^{-1}\right|_{p_i}\right|_{p_i} P_i\right|_P = \sum_{i=1}^{n} \left|x_i \cdot \left|P_i^{-1}\right|_{p_i}\right|_{p_i} \cdot P_i - R \cdot P,$$

где $P_i = \frac{P}{p_i}$, $|P_i^{-1}|_{p_i}$ – мультипликативная инверсия P_i по модулю p_i; $i \in [1, n]$; n – количество модулей; R – позиционный индекс.

Зная значение коэффициента R, можно вычислить остаток от деления по новому основанию без перевода модулярного числа в позиционное представление:

$$|X|_{p_{n+1}} = \left|\sum_{i=1}^{n} \left|x_i \cdot \left|P_i^{-1}\right|_{p_i}\right|_{p_i} \cdot |P_i|_{p_{n+1}}\right|_{p_{n+1}} - \left|R \cdot |P|_{p_{n+1}}\right|_{p_{n+1}}\Bigg|_{p_{n+1}}.$$

Для вычисления коэффициента R авторами разработан алгоритм с использованием целочисленных интервалов на основе приближенной интервальной оценки величины:

$$\tilde{X} = \sum_{i=1}^{n} \left|x_i \cdot P_i^{-1}\right|_{p_i} \cdot \frac{1}{p_i} = \frac{X + R \cdot P}{P} = R + \frac{X}{P},$$

где целую часть величины $\tilde{X}$ определяет коэффициент R, а дробную – значение X/P. Процесс вычисления коэффициента R с использованием вещественных интервалов с направленным округлением и необходимые условия корректности вычислений представлены в работе К. Исупова и В. Князькова [8]; метод вычисления коэффициента R с использованием целочисленных интервалов описан в патенте [34].

Результаты исследования

Умножение двух чисел, представленных в гибридной модулярно-пози-

ционной интервально-логарифмической форме с плавающей точкой, выполняется с использованием гибридной технологии вычислений следующим образом.

Для вычисления произведения $Z = \left[\langle z_1, z_2, \ldots, z_n \rangle, \underline{L_Z}, \overline{L_Z}, \lambda_Z, \sigma_Z\right]$ чисел $X = \left[\langle x_1, x_2, \ldots, x_n \rangle, \underline{L_X}, \overline{L_X}, \lambda_X, \sigma_X\right]$ и $Y = \left[\langle y_1, y_2, \ldots, y_n \rangle, \underline{L_Y}, \overline{L_Y}, \lambda_Y, \sigma_Y\right]$ необходимо:

– вычислить знак произведения $\sigma_Z = \sigma_X \cdot \sigma_Y$ путем алгебраического умножения знаков сомножителей;

– вычислить верхнюю границу интервальной логарифмической характеристики результата $\underline{L_Z} = \underline{L_X} + \underline{L_Y}$ путем алгебраического сложения значений нижних границ $\underline{L_X}$ и $\underline{L_Y}$ ИЛХ операндов в позиционной системе счисления;

– вычислить верхнюю границу ИЛХ результата $\overline{L_Z} = \overline{L_X} + \overline{L_Y}$ путем алгебраического сложения значений нижних границ $\overline{L_X}$ и $\overline{L_Y}$ ИЛХ операндов в позиционной системе счисления;

– вычислить порядок результата $\lambda_Z = \lambda_X + \lambda_Y$ путем алгебраического сложения порядков сомножителей;

– выполнить умножение модулярных мантисс путем нахождения значений $z_i = \left|x_i \cdot y_i\right|_{p_i} = x_i \cdot y_i - \frac{x_i \cdot y_i}{p_i} \cdot p_i$ для всех $i \in [1; n]$; при этом вычисления выполняются над операндами, представленными в позиционной системе счисления по правилам модулярной арифметики.

В данной статье отсутствует описание обработки исключительных ситуаций, таких как получение машинного нуля, переполнение и т. п. Более подробно метод описан в патенте [34].

Следует отметить, что поскольку мантиссы чисел, представленные в СОК, ограничены диапазоном [0; P), то при выполнении умножения двух мантисс результат может выйти за пределы диапазона представления, то есть $M_Z = M_X \cdot M_Y \geq P$. Для того чтобы мантисса результата была представима в СОК, необходимо выполнить операцию масштабирования:

$$\frac{M_X \cdot M_Y}{b^a},$$

где $a = \left[\log_b \frac{M_X \cdot M_Y}{P}\right]$; M_X, M_Y – позиционные значения мантисс; P – позиционное значение диапазона представления; [ ] означает округление к наибольшему целому.

Рассмотрим предельный случай, когда числа из диапазона [0; P) могут появиться с равной вероятностью. Вероятность того, что произведение двух мантисс выйдет за пределы диапазона представления модулярных мантисс, то есть $M_X \cdot M_Y \geq P$, равна

$$p(M_Z \geq P) \approx \frac{(P-1)^2 - (P-1)\ln(P-1)}{P^2} \approx$$

$$\approx \frac{P - \ln P}{P} \approx 1.$$

Это означает, что в предельном случае каждая операция умножения требует выполнения операции масштабирования, и при использовании позиционных характеристик (как точных, так и приближенных и интервальных) для определения коэффициента масштабирования a необходимо производить трудоемкую операцию вычисления логарифма.

В случае использования ИЛХ коэффициент a рассчитывается следующим образом:

$$a = \overline{L_X} + \overline{L_Y} - L_P,$$

где $\overline{L_X}$, $\overline{L_Y}$ – верхние границы интервальных логарифмических характеристик чисел X и Y; $L_P = \log_b P$ – константа для конкретного диапазона представления.

Таким образом, при использовании ИЛХ для вычисления коэффициента

масштабирования не требуется преобразования в позиционную систему счисления и вычисления логарифма.

При умножении модулярных мантисс целесообразно выполнять масштабирование обоих операндов до непосредственного выполнения умножения; причем, если величина обоих операндов превышает значение $\sqrt{P}$, следует распределять коэффициент масштабирования между операндами таким образом, чтобы отмасштабированные операнды не превышали величину $\sqrt{P}$:

$$b^a = b^{a_X} + b^{a_Y},$$

где b^{a_X} – масштабирующий коэффициент, применяемый к первому сомножителю; b^{a_Y} – масштабирующий коэффициент, применяемый ко второму сомножителю; a_X и a_Y – значения, определяемые соотношениями ИЛХ операндов следующим образом.

Пусть $L_1 = \overline{L_X} + \overline{L_Y} - L_P$, $L_2 = \overline{L_X} - \overline{L_Y}$; тогда $a_X = \frac{L_1 + L_2}{2}$, $a_Y = \frac{L_1 - L_2}{2}$. Если только один из операндов превышает величину $\sqrt{P}$, к нему необходимо применить масштабирующий коэффициент b^a.

Таким образом, если $\overline{L_Z} \geq L_P$, необходимо выполнить масштабирование мантисс операндов, после чего выполнить умножение отмасштабированных мантисс, а также скорректировать значение порядка результата $\lambda_z = \lambda_z + L_1$ и значение верхней и нижней границы интервальной логарифмической характеристики результата $\underline{L_Z} = \underline{L_Z} - L_1$, $\overline{L_Z} = \overline{L_Z} - L_1$.

Процесс масштабирования является итерационным, поскольку за один шаг выполняется масштабирование коэффициентом, не превышающим 2^q, где q – разрядность модулей СОК.

На каждом шаге масштабирования вычисляется значение коэффициента R и остаток от деления модулярного числа на $p_{n+1} = 2^\alpha$, где $\alpha \leq q$:

$$x_{n+1} = \left|M_X\right|_{2^\alpha} =$$

$$= \left|\left|\sum_{i=1}^{n}\left|x_i \cdot \left|P_i^{-1}\right|_{p_i}\right|_{p_i}\cdot\left|P_i\right|_{2^\alpha}\right|_{2^\alpha} - \left|R\cdot\left|P\right|_{2^\alpha}\right|_{2^\alpha}\right|_{2^\alpha}.$$

Затем выполняется масштабирование коэффициентом 2^α:

$$\tilde{x}_i = \left|\left|x_i - \left|M_X\right|_{2^\alpha}\right|_{p_i}\cdot\left|(2^\alpha)^{-1}\right|_{p_i}\right|_{p_i},$$

где $\left|(2^\alpha)^{-1}\right|_{p_i}$ – мультипликативная инверсия числа 2^α по модулю p_i – константа для конкретного значения модуля p_i.

Все вычисления производятся над целыми числами, представленными в позиционной системе счисления, по правилам модулярной арифметики.

Если $a_X > q$, процедура масштабирования повторяется над уже масштабированной мантиссой $\tilde{x}_1, \tilde{x}_2, \supset, \tilde{x}_n$ и так далее, пока не будет выполнено полное масштабирование коэффициентом $2^{\alpha X}$. Аналогичным образом выполняется масштабирование второго сомножителя Y.

Подробный алгоритм деления модулярной мантиссы числа, представленного в модулярно-позиционной интервально-логарифмической форме, на число 2^α (масштабирование степенью двойки), также представлен в патенте [34].

Среднее время выполнения разработанного метода равно

$$T = p\left(M_Z \geq P\right)\cdot t_1 + \frac{n}{k}t_2,$$

где $p(M_Z \geq P)$ – вероятность того, что произведение двух мантисс выйдет за пределы диапазона представления модулярных мантисс (в предельном случае $p(M_Z \geq P) = 1$); t_1 – время выполнения операции масштабирования; t_2 – время выполнения операции умножения по модулю; k – количество параллельных модулярных каналов.

Среднее время выполнения операции масштабирования определяется следующим образом:

$$t_1 = j \cdot \left(t_3 + \frac{n}{k} t_4 + \frac{n}{k} t_5 \right),$$

где t_3 – время выполнения операции расширения базиса; t_4 – время выполнения операции вычитания по модулю; t_5 – время выполнения операции умножения по модулю на константу; j – число итераций масштабирования.

Диапазон представления модулярных мантисс $P = \prod_{i=1}^{n} p_i \approx 2^{n \cdot q}$. Минимальный коэффициент масштабирования равен 2^1, максимальный равен $2^{\frac{n \cdot q}{2}}$. Максимальное количество шагов масштабирования примем равным $\frac{n}{2}$.

Таким образом, минимальное и максимальное время выполнения операции масштабирования равны соответственно:

$$t_{1min} = t_3 + \frac{n}{k} t_4 + \frac{n}{k} t_5,$$

$$t_{1max} = \frac{n}{2} \left(t_3 + \frac{n}{k} t_4 + \frac{n}{k} t_5 \right).$$

Время выполнения операции расширения базиса равно [34]:

$$t_3 = \frac{n}{k} t_5 + \frac{n}{k} t_6 + t_7,$$

где t_5 – время выполнения операции умножения по модулю на константу; t_6 – время выполнения операции скалярного произведения вектора на вектор-константу; t_7 – время выполнения операции сложения.

Время выполнения операции умножения по модулю двух произвольных чисел приблизительно в 2 раза выше, чем время выполнения стандартного целочисленного умножения; время выполнения операции умножения по модулю на константу приблизительно равно времени выполнения стандартного целочисленного умножения [13]. Таким образом, минимальное и максимальное время выполнения разработанного метода равно:

$$T_{min} = \left(\frac{n}{k} t_5 + \frac{n}{k} t_6 + t_7 + \frac{n}{k} t_4 + \frac{n}{k} t_5 \right) + \frac{n}{k} t_2 =$$

$$= \left(\frac{2n}{k} t_c + \frac{n}{k} t_c + t_c + \frac{n}{k} t_c \right) + \frac{2n}{k} t_c = \left(\frac{6n}{k} + 1 \right) t_c,$$

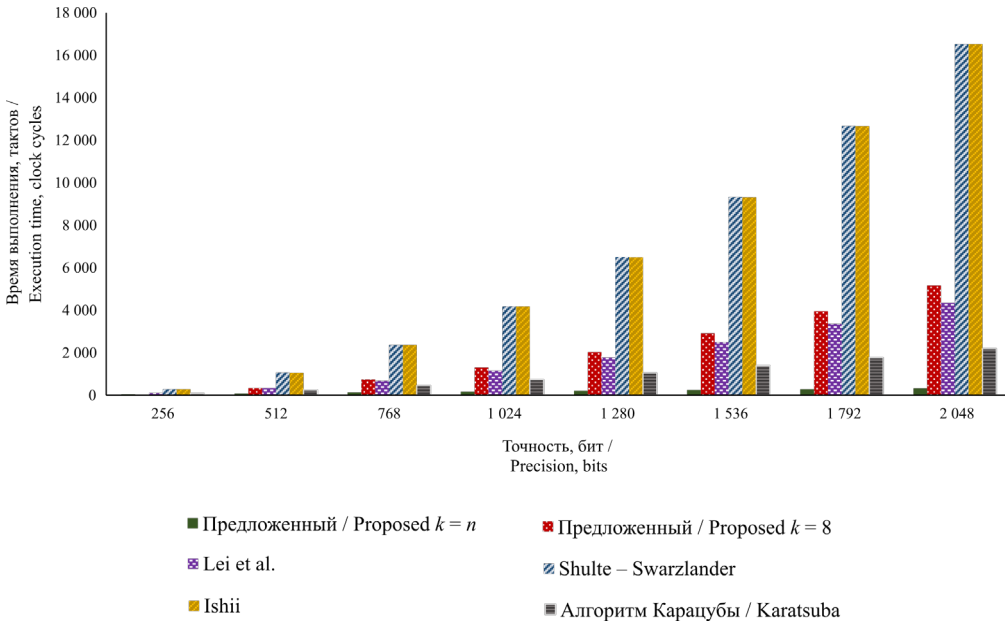
$$T_{max} = \frac{n}{2} \left(\frac{n}{k} t_5 + \frac{n}{k} t_6 + t_7 + \frac{n}{k} t_4 + \frac{n}{k} t_5 \right) + \frac{n}{k} t_2 =$$

$$= \frac{n}{2} \left(\frac{2n}{k} t_c + \frac{n}{k} t_c + t_c + \frac{n}{k} t_c \right) +$$

$$+ \frac{2n}{k} t_c = \left(\frac{5n^2 + 4n}{2k} + 1 \right) t_c,$$

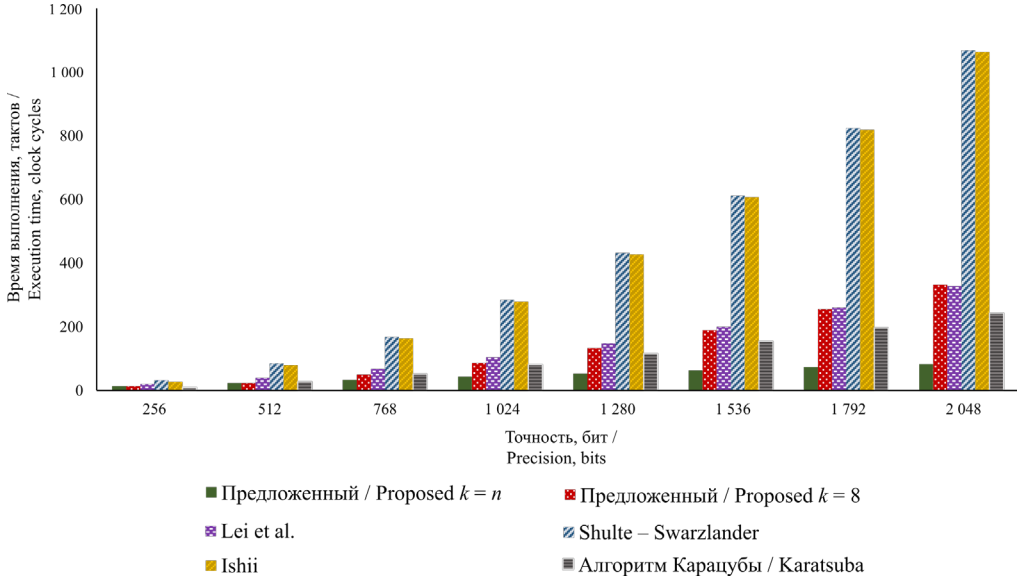
где t_7 – длительность такта.

Сравним разработанный метод с конвейерным методом умножения длинных чисел с плавающей точкой (обозначим его как Lei et al [12]), а также со стандартными методами умножения (обозначим их как Schulte – Swarzlander [14] и Ishii [13]). В качестве контроля рассмотрим асимптотически быстрый метод, используемый для организации некоторых целочисленных двоичных умножителей (алгоритм Карацубы [26]). В табл. 1 представлены оценки времени выполнения разработанного метода и аналогов; n – количество слов базовой длины, необходимых для представления длинного числа; k – количество параллельных модулярных каналов. В табл. 2 выполнено сравнение времени выполнения разработанного метода и аналогов для чисел разрядности 1 024 и 2 048 бит (n = 16 64-разрядных слов и n = 32 64-разрядных слова соответственно). На рис. 1; 2 представлены расчеты времени выполнения разработанного метода и аналогов для разрядности сомножителей от 256 до 2 048 бит с использованием 64-разрядных (рис. 1) и 16-разрядных слов (рис. 2) соответственно.



Р и с. 1. Сравнение быстродействия разработанного метода с аналогами (с использованием 64-разрядных умножителей)

F i g. 1. The comparison of the developed method speed-in-action with analogues using 64 bit multipliers



Р и с. 2. Сравнение быстродействия разработанного метода с аналогами (с использованием 16-разрядных умножителей)

F i g. 2. The comparison of the developed method speed-in-action with analogues using 16 bit multipliers

Таблица 1
Table 1

Сравнение временной сложности разработанного метода и аналогов
Time complexity comparison with analogues

Метод умножения / Multiplication method	Время выполнения, тактов / Execution time, clock cycles
Lei at al.	$\frac{n^2}{4} + 2n + 8$
Schulte – Swarzlander	$n^2 + n + 12$
Ishii	$n^2 + n + 7$
Предложенный метод с масштабированием / Proposed method with scaling	$p_{n+1} = 2^{\alpha}$
Предложенный метод с масштабированием параллельный ($k = n$) / Proposed method with scaling parallel ($k = n$)	$\frac{5n+6}{2}$

Таблица 2
Table 2

Сравнение быстродействия разработанного метода и аналогов (в тактах)
The speed-in-action comparison (clock cycles) with analogues

Метод умножения / Multiplication method	Время выполнения, тактов / Execution time, clock cycles		Ускорение / Speed-up
	Аналоги / Analogues	Предложенный метод ($k = n$) / Proposed method ($k = n$)	
1 024 бит / 1 024 bit			
Lei at al.	104	43	**2,4**
Schulte – Swarzlander	284	43	**6,6**
Ishii	279	43	**6,4**
2 048 бит / 2 048 bit			
Lei at al.	328	83	**4,0**
Schulte – Swarzlander	1 068	83	**12,9**
Ishii	1 063	83	**12,8**

Обсуждение и заключение

Разработаны новые быстрые методы умножения модулярных чисел с плавающей точкой; проведена оценка быстродействия разработанных методов; выполнено сравнение с работами других авторов. Предложенные методы в 2,4–4,0 раза быстрее конвейерного метода умножения и в 6,4–12,9 раз быстрее классических методов умножения.

Показано, что при умножении двух модулярных чисел с плавающей точкой практически каждая операция умножения модулярных мантисс сопровождается немодульной операцией масштабирования, что существенно увеличивает общее время выполнения умножения. В связи с этим целесообразно продолжить исследования быстрых методов выполнения немодульных операций, в частности, операции масштабирования большими коэффициентами.

В данной статье не учитывается время выполнения операции преобра-

зования чисел в модулярно-позиционную интервально-логарифмическую форму представления. Авторы считают, что разработанный метод умножения будет использоваться для обработки больших объемов числовой информации, поступающей уже в необходимом формате; преобразование же данных из других форматов, в том числе стандартных, может быть осуществлено в параллельном или конвейерном режимах и не будет приводить к значительным затратам времени.

Авторами рассмотрен алгоритм масштабирования коэффициентом, равным 2^a; при этом интервальная характеристика представлена в виде логарифмов по основанию 2. В то же время разработанный способ представления информации и выполнения операции умножения может быть использован и при других значениях основания логарифма. Так, разработанные методы могут быть применены для операций над числами вида $M \cdot 10^E$. В таком случае коэффициент масштабирования будет равен 10^a. Данное преимущество может быть дополнительно использовано в задачах, критичных к ошибкам округления при вводе десятичной информации.

В качестве дальнейших исследований предполагается изучение и разработка быстрых методов выполнения немодульных операций расширения базиса и масштабирования, а также разработка арифметического модулярно-позиционного интервально-логарифмического устройства.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Reproducible and accurate matrix multiplication / R. Iakymchuk [et al.] // Scientific Computing, Computer Arithmetic, and Validated Numerics. SCAN 2015 / Eds. M. Nehmeier, J. Wolff von Gudenberg, W. Tucker. Lecture Notes in Computer Science. 2016. Vol. 9553. P. 126–137. DOI: https://doi.org/10.1007/978-3-319-31769-4_11

2. **Voros A.** Discretized Keiper/Li approach to the Riemann hypothesis // Experimental Mathematics. 2018. P. 1–18. DOI: https://doi.org/10.1080/10586458.2018.1482480

3. solveME: fast and reliable solution of nonlinear ME models / L. Yang [et al.] // BMC Bioinformatics. 2016. Vol. 17. P. 391. DOI: https://doi.org/10.1186/s12859-016-1240-1

4. **Panzer E.** Algorithms for the symbolic integration of hyperlogarithms with applications to Feynman integrals // Computer Physics Communications. 2015. Vol. 188. P. 148–166. DOI: https://doi.org/10.1016/j.cpc.2014.10.019

5. **Miltenberger M., Ralphs T., Steffy D. E.** Exploring the numerics of branch-and-cut for mixed integer linear optimization // Operations Research Proceedings 2017. Operations Research Proceedings (GOR (Gesellschaft für Operations Research e.V.)) / Eds. N. Kliewer, J. Ehmke, R. Borndörfer. Cham : Springer, 2018. P. 151–157. DOI: https://doi.org/10.1007/978-3-319-89920-6_21

6. Computer discovery and analysis of large Poisson polynomials / D. H. Bailey [et al.] // Experimental Mathematics. 2017. Vol. 26, no. 3. P. 349–363. DOI: https://doi.org/10.1080/10586458.2016.1180565

7. **Pan B., Wang Y., Tian S.** A high-precision single shooting method for solving hypersensitive optimal control problems // Mathematical Problems in Engineering. 2018. Vol. 2018. Article ID 7908378. DOI: https://doi.org/10.1155/2018/7908378

8. **Isupov K., Knyazkov V.** Interval estimation of relative values in residue number system // Journal of Circuits, Systems and Computers. 2018. Vol. 27, no. 1. P. 1850004. DOI: https://doi.org/10.1142/S0218126618500044

9. GRAPE-MPs: implementation of an SIMD for quadruple/hexuple/octuple-precision arithmetic operation on a structured ASIC and an FPGA / N. Nakasato [et al.] // 2012 IEEE 6th International Symposium on Embedded Multicore SoCs. IEEE, 2012. P. 75–83. DOI: https://doi.org/10.1109/MCSoC.2012.31

10. Application of GRAPE9-MPX for high precision calculation in particle physics and performance results / H. Daisaka [et al.] // Procedia Computer Science. 2015. Vol. 51. P. 1323–1332. DOI: https://doi.org/10.1016/j.procs.2015.05.317

11. **El-Araby E., Gonzalez I., El-Ghazawi T.** Bringing high-performance reconfigurable computing to exact computations // 2007 International Conference on Field Programmable Logic and Applications / Eds. K. Bertels [et al.]. IEEE, 2007. P. 79–85. DOI: https://doi.org/10.1109/FPL.2007.4380629

12. **Lei Y., Dou Y., Zhou J.** FPGA-specific custom VLIW architecture for arbitrary precision floating-point arithmetic // IEICE Transactions on Information and Systems. 2011. Vol. 94, no. 11. P. 2173–2183. DOI: https://doi.org/10.1587/transinf.E94.D.2173

13. Fast modular arithmetic on the Kalray MPPA-256 processor for an energy-efficient implementation of ECM / M. Ishii [et al.] // IEEE Transactions on Computers. 2017. Vol. 66, issue 12. P. 2019–2030. DOI: https://doi.org/10.1109/TC.2017.2704082

14. **Schulte M. J., Swartzlander E. E.** A family of variable-precision interval arithmetic processors // IEEE Transactions on Computers. 2000. Vol. 49, issue 5. P. 387–397. DOI: https://doi.org/10.1109/12.859535

15. **Asif S., Kong Y.** Highly parallel modular multiplier for elliptic curve cryptography in residue number system // Circuits, Systems, and Signal Processing. 2017. Vol. 36, issue 3. P. 1027–1051. DOI: https://doi.org/10.1007/s00034-016-0336-1

16. **Kong Y., Lai Y.** Low latency modular multiplication for public-key cryptosystems using a scalable array of parallel processing elements // 2013 IEEE 56th International Midwest Symposium on Circuits and Systems (MWSCAS). IEEE, 2013. P. 1039–1042. DOI: https://doi.org/10.1109/MWSCAS.2013.6674830

17. **Coleman J. N., Che Ismail R.** LNS with co-transformation competes with floating-point // IEEE Transactions on Computers. 2016. Vol. 65, issue 1. P. 136–146. DOI: https://doi.org/10.1109/TC.2015.2409059

18. **Kouretas I., Basetas C., Paliouras V.** Low-power logarithmic number system addition/subtraction and their impact on digital filters // IEEE Transactions on Computers. 2013. Vol. 62, issue 11. P. 2196–2209. DOI: https://doi.org/10.1109/TC.2012.111

19. The European logarithmic microprocesor / J. N. Coleman [et al.] // IEEE Transactions on Computers. 2008. Vol. 57, issue 4. P. 532–546. DOI: https://doi.org/10.1109/TC.2007.70791

20. **Bigou K., Tisserand A.** Single base modular multiplication for efficient hardware RNS implementations of ECC // Cryptographic Hardware and Embedded Systems – CHES 2015 / Eds. T. Güneysu, H. Handschuh. Lecture Notes in Computer Science. 2015. Vol. 9293. P. 123–140. DOI: https://doi.org/10.1007/978-3-662-48324-4_7

21. **Bajard J.-C., Eynard J., Merkiche N.** Montgomery reduction within the context of residue number system arithmetic // Journal of Cryptographic Engineering. 2018. Vol. 8, issue 3. P. 189–200. DOI: https://doi.org/10.1007/s13389-017-0154-9

22. **Czyżak M., Smyk R., Ulman Z.** Pipelined scaling of signed residue numbers with the mixed-radix conversion in the programmable gate array // Poznan University of Technology Academic Journals. Electrical Engineering. 2013. No. 76. P. 89–99. URL: https://yadda.icm.edu.pl/baztech/element/bwmeta1.element.baztech-5d0a87e2-2459-476f-8c7e-2d72d07072f2/c/Czyzak.pdf

23. A fully RNS based ECC processor / S. Asif [et al.] // Integration. 2018. Vol. 61. P. 138–149. DOI: https://doi.org/10.1016/j.vlsi.2017.11.010

24. Pipelined FPGA coprocessor for elliptic curve cryptography based on residue number system / P. M. Matutino [et al.] // 2017 International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS) / Eds. Y. Patt, S. K. Nandy. IEEE, 2017. P. 261–268. DOI: https://doi.org/10.1109/SAMOS.2017.8344638

25. **Коржавина А. С., Князьков В. С.** Методы расширения базиса в системе остаточных классов: обзор и анализ вычислительной сложности // Современные наукоемкие технологии. 2017. № 12. С. 37–42. URL: https://www.top-technologies.ru/ru/article/view?id=36868

26. **Harvey D., van der Hoeven J., Lecerf G.** Even faster integer multiplication // Journal of Complexity. 2016. Vol. 36. P. 1–30. DOI: https://doi.org/10.1016/j.jco.2016.03.001

27. **Chang C. H., Low J. Y. S.** Simple, fast, and exact RNS scaler for the three-moduli set ($2^n - 1$, 2^n, $2^n + 1$) // IEEE Transactions on Circuits and Systems I: Regular Papers. 2011. Vol. 58, issue 11. P. 2686–2697. DOI: https://doi.org/10.1109/TCSI.2011.2142950

28. **Hiasat A.** Efficient RNS scalers for the extended three-moduli set ($2^n - 1$, 2^{n+p}, $2^n + 1$) // IEEE Transactions on Computers. 2017. Vol. 66, issue 7. P. 1253–1260. DOI: https://doi.org/10.1109/TC.2017.2652474

29. **Kong Y., Phillips B.** Fast scaling in the residue number system // IEEE Transactions on Very Large Scale Integration (VLSI) Systems. 2009. Vol. 17, issue 3. P. 443–447. DOI: https://doi.org/10.1109/TVLSI.2008.2004550

30. **Meyer-Base U., Stouraitis T.** New power-of-2 RNS scaling scheme for cell-based IC design // IEEE Transactions on Very Large Scale Integration (VLSI) Systems. 2003. Vol. 11, issue 2. P. 280–283. DOI: https://doi.org/10.1109/TVLSI.2003.810799

31. **Johansson F.** Arb: efficient arbitrary-precision midpoint-radius interval arithmetic // IEEE Transactions on Computers. 2017. Vol. 66, issue 8. P. 1281–1292. DOI: https://doi.org/10.1109/TC.2017.2690633

32. **Revol N.** Introduction to the IEEE 1788-2015 standard for interval arithmetic // Numerical Software Verification. NSV 2017 / Eds. A. Abate, S. Boldo. Lecture Notes in Computer Science. 2017. Vol. 10381. P. 14–21. DOI: https://doi.org/10.1007/978-3-319-63501-9_2

33. **Osinin I.** A modular-logarithmic coprocessor concept // 2017 International Conference on High Performance Computing & Simulation (HPCS). IEEE, 2017. P. 588–594. DOI: https://doi.org/10.1109/HPCS.2017.93

34. Способ организации выполнения операции умножения двух чисел в модулярно-логарифмическом формате представления с плавающей точкой на гибридных многоядерных процессорах : пат. 2666285 Рос. Федерация : МПК G 06 F 7/483, G 06 F 7/487 / Князьков В. С., Коржавина А. С. ; заявитель и патентообладатель ФГБОУ ВО «Вятский государственный университет». № 2017135775/22 ; заявл. 06.10.2017; опубл. 06.09.2018, Бюл. № 25.

Поступила 07.12.2018; принята к публикации 20.02.2019; опубликована онлайн 28.06.2019

Об авторах:

Коржавина Анастасия Сергеевна, старший преподаватель, кафедра электронных вычислительных машин, ФГБОУ ВО «Вятский государственный университет» (610000, Россия, г. Киров, ул. Московская, д. 36), ResearcherID: S-1877-2018, ORCID: https://orcid.org/0000-0001-8270-2097, as_korzhavina@vyatsu.ru

Князьков Владимир Сергеевич, главный научный сотрудник, НИИ прикладных и фундаментальных исследований, ФГБОУ ВО «Пензенский государственный университет» (440026, Россия, г. Пенза, ул. Красная, д. 40); профессор, кафедра электронных вычислительных машин, ФГБОУ ВО «Вятский государственный университет» (610000, Россия, г. Киров, ул. Московская, д. 36), доктор технических наук, ResearcherID: T-4089-2018, ORCID: https://orcid.org/0000-0003-3820-6541, kniazkov@list.ru

Заявленный вклад соавторов:

А. С. Коржавина – обзор литературы, разработка метода и анализ результатов; В. С. Князьков – формулировка и постановка задачи, научное руководство, обсуждение результатов.

Благодарности: Авторы благодарят анонимных рецензентов за их полезные комментарии.

Все авторы прочитали и одобрили окончательный вариант рукописи.

REFERENCES

1. Iakymchuk R., Defour D., Collange S., Graillat S. Reproducible and accurate matrix multiplication. In: Nehmeier M., Wolff von Gudenberg J., Tucker W. (eds) *Scientific Computing, Computer Arithmetic and Validated Numerics. SCAN 2015. Lecture Notes in Computer Science*. 2016; 9553:126-137. DOI: https://doi.org/10.1007/978-3-319-31769-4_11

2. Voros A. Discretized Keiper/Li approach to the Riemann hypothesis. *Experimental Mathematics*. 2018; 1-18. DOI: https://doi.org/10.1080/10586458.2018.1482480

3. Yang L., Ma D., Ebrahim A., Lloyd C.J., Saunders M.A., Palsson B.O. solveME: fast and reliable solution of nonlinear ME models. *BMC Bioinformatics*. 2016; 17:391. DOI: https://doi.org/10.1186/s12859-016-1240-1

4. Panzer E. Algorithms for the symbolic integration of hyperlogarithms with applications to Feynman integrals. *Computer Physics Communications*. 2015; 188:148-166. DOI: https://doi.org/10.1016/j.cpc.2014.10.019

5. Miltenberger M., Ralphs T., Steffy D. E. Exploring the numerics of branch-and-cut for mixed integer linear optimization. In: Kliewer N., Ehmke J., Borndörfer R. (eds) *Operations Research Proceedings 2017. Operations Research Proceedings (GOR (Gesellschaft für Operations Research e.V.))*. Cham: Springer; 2018. p. 151-157. DOI: https://doi.org/10.1007/978-3-319-89920-6_21

6. Bailey D.H., Borwein J.M., Kimberley J.S., Ladd W. Computer discovery and analysis of large poisson polynomials. *Experimental Mathematics*. 2017; 26(3):349-363. DOI: https://doi.org/10.1080/10586458.2016.1180565

7. Pan B., Wang Y., Tian S. A high-precision single shooting method for solving hypersensitive optimal control problems. *Mathematical Problems in Engineering*. 2018; 2018:7908378. DOI: https://doi.org/10.1155/2018/7908378

8. Isupov K., Knyazkov V. Interval estimation of relative values in residue number system. *Journal of Circuits, Systems and Computers*. 2018; 27(1):1850004. DOI: https://doi.org/10.1142/S0218126618500044

9. Nakasato N., Daisaka H., Fukushige T., Kawai A., Makino J., Ishikawa T., et al. GRAPE-MPs: Implementation of an SIMD for quadruple/hexuple/octuple-precision arithmetic operation on a structured ASIC and an FPGA In*: 2012 IEEE 6th International Symposium on Embedded Multicore SoCs*. IEEE; 2012. p. 75-83. DOI: https://doi.org/10.1109/MCSoC.2012.31

10. Daisaka H., Nakasato N., Ishikawa T., Yuasa F. Application of GRAPE9-MPX for high precision calculation in particle physics and performance results. *Procedia Computer Science*. 2015; 51:1323-1332. DOI: https://doi.org/10.1016/j.procs.2015.05.317

11. El-Araby E., Gonzalez I., El-Ghazawi T. A. Bringing high-performance reconfigurable computing to exact computations. In: Bertels K., Najjar W., van Genderen A., Vassiliadis S. (eds.) *2007 International Conference on Field Programmable Logic and Applications.* 2007. p. 79–85. DOI: https://doi.org/10.1109/FPL.2007.4380629

12. Lei Y., Dou Y., Zhou J. FPGA-specific custom VLIW architecture for arbitrary precision floating-point arithmetic. *IEICE Transactions on Information and Systems*. 2011; 94(11):2173-2183. DOI: https://doi.org/10.1587/transinf.E94.D.2173

13. Ishii M., Detrey J., Gaudry P., Inomata A., Fujikawa K. Fast modular arithmetic on the Kalray MPPA-256 processor for an energy-efficient implementation of ECM. *IEEE Transactions on Computers*. 2017; 66(12):2019-2030. DOI: https://doi.org/10.1109/TC.2017.2704082

14. Schulte M.J., Swartzlander E.E. A family of variable-precision interval arithmetic processors. *IEEE Transactions on Computers*. 2000; 49(5):387-397. DOI: https://doi.org/10.1109/12.859535

15. Asif S., Kong Y. Highly parallel modular multiplier for elliptic curve cryptography in residue number system. *Circuits, Systems, and Signal Processing*. 2017; 36(3):1027-1051. DOI: https://doi.org/10.1007/s00034-016-0336-1

16. Kong Y., Lai Y**.** Low latency modular multiplication for public-key cryptosystems using a scalable array of parallel processing elements. In: *2013 IEEE 56th International Midwest Symposium on Circuits and Systems (MWSCAS)*. IEEE; 2013. p. 1039-1042. DOI: https://doi.org/10.1109/MWSCAS.2013.6674830

17. Coleman J.N., Che Ismail R. LNS with co-transformation competes with floating-point. *IEEE Transactions on Computers*. 2016; 65(1):136-146. DOI: https://doi.org/10.1109/TC.2015.2409059

18. Kouretas I., Basetas C., Paliouras V. Low-power logarithmic number system addition/subtraction and their impact on digital filters. *IEEE Transactions on Computers*. 2013; 62(11):2196-2209. DOI: https://doi.org/10.1109/TC.2012.111

19. Coleman J.N., Softley C.I., Kadlec J., Matousek R., Tichy M., Pohlet Z., et al. The European logarithmic microprocessor. *IEEE Transactions on Computers*. 2008; 57(4):532-546. DOI: https://doi.org/10.1109/TC.2007.70791

20. Bigou K., Tisserand A. Single base modular multiplication for efficient hardware RNS implementations of ECC. In: Güneysu T., Handschuh H. (eds) *Cryptographic Hardware and Embedded Systems – CHES 2015. Lecture Notes in Computer Science*. 2015; 9293:123-140. DOI: https://doi.org/10.1007/978-3-662-48324-4_7

21. Bajard J.-C., Eynard J., Merkiche N. Montgomery reduction within the context of residue number system arithmetic. *Journal of Cryptographic Engineering*. 2018; 8(3):189-200. DOI: https://doi.org/10.1007/s13389-017-0154-9

22. Czyżak M., Smyk R., Ulman Z. Pipelined scaling of signed residue numbers with the mixed-radix conversion in the programmable gate array. *Poznan University of Technology Academic Journals.*

Electrical Engineering. 2013; 76:89-99. Available at: https://yadda.icm.edu.pl/baztech/element/bwmeta1.element.baztech-5d0a87e2-2459-476f-8c7e-2d72d07072f2/c/Czyzak.pdf

23. Asif S., Hossain M.S., Kong Y., Abdul W. A fully RNS based ECC processor. *Integration*. 2018; 61:138-149. DOI: https://doi.org/10.1016/j.vlsi.2017.11.010

24. Matutino P. M., Araújo J., Sousa L., Chaves R. Pipelined FPGA coprocessor for elliptic curve cryptography based on residue number system. In: Patt Y., Nandy S.K. (eds.) *2017 International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS)*. 2017. p. 261-268. DOI: https://doi.org/10.1109/SAMOS.2017.8344638

25. Korzhavina A.S., Knyazkov V.S. [Base extension in residue number systems: a review and cost analysis]. *Sovremennyye naukoyemkiye tekhnologii* = Modern High Technologies. 2017; 12:37-42. Available at: https://www.top-technologies.ru/ru/article/view?id=36868 (In Russ.).

26. Harvey D., van der Hoeven J., Lecerf G. Even faster integer multiplication. *Journal of Complexity*. 2016; 36:1-30. DOI: https://doi.org/10.1016/j.jco.2016.03.001

27. Chang C.H., Low J.Y.S. Simple, fast, and exact RNS scaler for the three moduli set ($2^n - 1$, 2^n, $2^n + 1$). *IEEE Transactions on Circuits and Systems I: Regular Papers*. 2011; 58(11):2686-2697. DOI: https://doi.org/10.1109/TCSI.2011.2142950

28. Hiasat A. Efficient RNS scalers for the extended three-moduli set ($2^n - 1$, 2^{n+p}, $2^n + 1$). *IEEE Transactions on Computers*. 2017; 66(7):1253-1260. DOI: https://doi.org/10.1109/TC.2017.2652474

29. Kong Y., Phillips B. Fast scaling in the residue number system. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*. 2009; 17(3):443-447. DOI: https://doi.org/10.1109/TVLSI.2008.2004550

30. Meyer-Base U., Stouraitis T. New power-of-2 RNS scaling scheme for cellbased IC design. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*. 2003; 11(2):280-283. DOI: https://doi.org/10.1109/TVLSI.2003.810799

31. Johansson F. Arb: Efficient arbitrary-precision midpoint-radius interval arithmetic. *IEEE Transactions on Computers*. 2017; 66(8):1281-1292. DOI: https://doi.org/10.1109/TC.2017.2690633

32. Revol N. Introduction to the IEEE 1788-2015 standard for interval arithmetic. In: Abate A., Boldo S. (eds.) *Numerical Software Verification. NSV 2017. Lecture Notes in Computer Science*. 2017; 10381:14-21. DOI: https://doi.org/10.1007/978-3-319-63501-9_2

33. Osinin I. A modular-logarithmic coprocessor concept. In: *International Conference on High Performance Computing & Simulation (HPCS)*. IEEE. 2017. p. 588-594. DOI: https://doi.org/10.1109/HPCS.2017.93

34. Knyazkov V.S., Korzhavina A.S., inventors. The method of organization of multiplying operation of two numbers in floating-point modular-logarithmic format on hybrid multicore processors. Ru Patent 2666285. 2018 Sep 06. (In Russ.).

Received 07.12.2018; revised 20.02.2019; published online 28.06.2019

About authors:

Anastasia S. Korzhavina, Senior Lecturer, Chair of Electronic Computing Machines, Vyatka State University (36 Moskovskaya St., Kirov 610000, Russia), ResearcherID: S-1877-2018, ORCID: https://orcid.org/0000-0001-8270-2097, as_korzhavina@vyatsu.ru

Vladimir S. Knyazkov, Chief Researcher, Research Institute of Applied and Fundamental Research, Penza State University (40 Krasnaya St., Penza 440026, Russia); Professor, Chair of Electronic Computing Machines, Vyatka State University (36 Moskovskaya St., Kirov 610000, Russia), D.Sc. (Engineering), ResearcherID: T-4089-2018, ORCID: https://orcid.org/0000-0003-3820-6541, kniazkov@list.ru

Contribution of the authors:

A. S. Korzhavina – reviewing literature, developing the method and analyzing the results. V. S. Knyazkov – formulating the problem, scientific advising, discussing the results.

Acknowledgements: The authors would like to thank the reviewers for their helpful comments.

All authors have read and approved the final version of the paper.