«НАУКА. ИННОВАЦИИ. ТЕХНОЛОГИИ», №2, 2013 г.

УДК 681.3

Н. И. Червяков [N. I. Chervyakov],
М. Г. Бабенко [M. G. Babenko],
П. А. Ляхов [P. A. Lyakhov],
И. Н. Лавриненко [I. N. Lavrinenko]

ЭФФЕКТИВНЫЙ АЛГОРИТМ ДЕЛЕНИЯ ЧИСЕЛ В СИСТЕМЕ ОСТАТОЧНЫХ КЛАССОВ НА ОСНОВЕ ПРИБЛИЖЕННОГО МЕТОДА[1]

Effective algorithm of the numbers division in the system of residual classes on the basis of approximate method

В статье предложен новый алгоритм деления чисел в системе остаточных классов, основанный на использовании приближенного метода сравнения модулярных чисел. Показано, что предложенный алгоритм по вычислительной сложности лучше, чем известные аналоги. Предложена техническая реализация алгоритма и приведены примеры его работы.

Ключевые слова: алгоритм, система остаточных классов, модулярная арифметика, деление.

This paper proposes a new algorithm for division of numbers in the residue number system, based on the use of the approximate method of comparison of modular numbers. It is shown that the algorithm for computing complexity better than known analogs. Technical implementation of the proposed algorithm, and examples of his work are shown.

Key words: algorithm, residue number system, modular arithmetic, division.

Нетрадиционная система счисления на основе арифметики остаточных классов в качестве основы для вычислительной техники привлекает многих исследователей, и интерес к ней в последнее десятилетие резко возрастает, что подтверждается большим числом публикаций, посвященных практическому применению системы

1 Работа выполнена при финансовой поддержке Российского Фонда Фундаментальных Исследований, грант №13-07-00478-а.

остаточных классов (СОК) в цифровой обработке сигналов, системах обработки изображений, помехоустойчивом кодировании, криптографических системах, квантовых автоматах, нейрокомпьютерах, системах с массовым параллелизмом операций, облачных вычислениях и других [1–9]. В [10–15] исследованы модели отдельных вычислительных узлов специализированных процессоров, которые включают в себя такие элементы СОК, как: входные и выходные преобразователи; модульные сумматоры; схемы определения знака и переполнения; деления и масштабирования; расширения базы СОК; схемы округления, локализации и исправления ошибок и другие, которые могут быть реализованы на классической, нейросетевой или квантовой вычислительной базе.

СОК дает преимущества быстрого сложения и умножения по сравнению со всеми остальными системами счисления, что обуславливает большой интерес к СОК в тех областях, где требуются большие объемы вычислений. Однако некоторые операции, такие как сравнение и деление чисел весьма сложны в СОК. Нахождение более эффективных алгоритмов деления позволит найти новые перспективные области применения СОК.

Большинство известных алгоритмов деления в СОК могут быть разделены на две категории: с использованием умножения и с использованием вычитания. Большинство алгоритмов на основе умножения предварительно вычисляют обратную величину делителя, после чего эта величина умножается на делимое. Алгоритмы на основе вычитания используют вычитание кратных делителя из делимого до тех пор, пока полученный результат не будет меньше делителя. Некоторые известные алгоритмы деления в СОК на основе умножения изложены в [16–17]. Все эти алгоритмы используют преобразование в обобщенную позиционную систему счисления (ОПСС) для нахождения обратной величины делителя и сравнения чисел. Эти алгоритмы, основанные на ОПСС, являются медленными и требуют выполнения большого количества арифметических действий. Алгоритмы на основе вычитания представлены в [18] которые не требуют вычислений в ОПСС, однако используют некоторые другие немодульные операции. Алгоритм с вычитанием, представленный в [18] кажется наиболее привлекательным для применения на практике, поскольку использует эффективный метод проверки знака числа. Известен алгоритм деления в формате СОК [19], который не использует двоичную систему счисления или ОПСС во время деления. Этот алгоритм требует дополнительного набора модулей СОК, что ведет к большой избыточности. Большинство су-

ществующих алгоритмов обладают различными недостатками, что делает их менее пригодными для решения задачи деления в СОК.

В этой статье мы предлагаем очень быстрый алгоритм деления в общем случае в СОК и его техническую реализацию с использованием только регистровых сдвигов и суммирований. Улучшенный алгоритм обладает следующими свойствами: очень быстр по сравнению с известными алгоритмами, не имеет ограничений на делимое и делитель (кроме равенства нулю делителя), не использует предварительную оценку частного, не использует обратную величину для делителя и не использует операцию расширения базы.

Модификация алгоритма деления в системе остаточных классов на основе приближенного метода сравнения модулярных чисел.

Рассмотрим модификацию алгоритма основного деления чисел, представленных в системе остаточных классов в случае, когда и делимое, и делитель представляют собой произвольные целые числа, и делитель не приводится к случаю попарно простого с модулями СОК [20]. Модификация основана на использовании делимого и делителя, представленных в относительных величинах.

Специализированные процессоры на основе арифметики СОК могут сыграть важную роль в высокоскоростных системах обработки данных в режиме реального времени. Операции сложения, вычитания и умножения, называемые *модульными операциями*, могут быть реализованы очень быстро, без распространения межразрядных переносов. Немодульные операции деления, сравнения чисел, определения знака и переполнения диапазона остаются сравнительно медленными. Любое улучшение скорости этих медленных алгоритмов значительно улучшает производительность многомодульных арифметико-логических устройств (АЛУ). Обычно при рассмотрении деления в СОК выделяют три категории: деление с нулевым остатком, масштабирование и деление в общем случае. Проблема деления в общем виде в СОК привлекает внимание многих исследователей для разработки высокопроизводительных многомодульных АЛУ. Известные алгоритмы деления в СОК, основанные на использовании преобразования в ОПСС, масштабировании, округлении, расширении и других операциях, являются медленными и требуют выполнения большого количества арифметических действий. Большинство известных ал-

горитмов основаны на сравнении делимого с делителем или с его удвоенным значением, которые представляют определенную сложность. В связи с этим возникает необходимость упростить структуру вычислений при сравнении модулярных чисел. Одним из направлений упрощения операции сравнения модулярных чисел является подход с использованием приближенного метода вычисления позиционной характеристики модулярного числа, который позволяет абсолютно правильно реализовать основные классы процедур принятия решений: проверка равенства (неравенства) двух значений; сравнение двух значений (больше, меньше) и другие, которые обеспечивают решение основного круга задач, возникающих при аппаратной или программной реализации вычислений в системе остаточных классов.

Суть приближенного метода вычисления позиционной характеристики модулярного числа и его использование для деления модулярных чисел основана на использовании относительных величин анализируемых чисел к полному диапазону, определяемому Китайской теоремой об остатках, которая связывает позиционное число a с его представлением в остатках $(\alpha_1,\alpha_2,...,\alpha_n)$, где α_i — наименьшие неотрицательные вычеты числа, относительно модулей системы остаточных классов $p_1,p_2,...,p_n$ следующим выражением:

$$a=\left|\sum_{i=1}^{n}\frac{P}{p_i}\left|P_i^{-1}\right|_{p_i}\alpha_i\right|_P, \qquad (1)$$

где $P=\prod_{i=1}^{n}p_i$,

p_i — модули СОК,

$\left|P_i^{-1}\right|$ — мультипликативная инверсия P_i относительно p_i,

$P_i=\frac{P}{p_i}=p_1p_2...p_{i-1}p_{i+1}...p_n$.

Если разделить левую и правую части выражения (1) на константу P, соответствующую диапазону чисел, то получим приближенное значение

$$\left|\frac{a}{P}\right|_1=\left|\sum_{i=1}^{n}\frac{\left|P_i^{-1}\right|_{p_i}}{p_i}\alpha_i\right|_1\approx\left|\sum_{i=1}^{n}k_i\alpha_i\right|_1, \qquad (2)$$

где $k_i = \dfrac{\left|P_i^{-1}\right|_{p_i}}{p_i}$ — константы выбранной системы,

α_i — разряды числа, представленного в СОК по модулям p_i,

где $i = 1,2,...,n$, при этом значение сумм (1) и (2) будет в интервале [0,1).

Конечный результат суммы определяется после суммирования и отбрасывания целой части числа с сохранением дробной части суммы. Дробная величина $F(a)=\left|\dfrac{a}{P}\right|_1 \in [0,1)$ содержит как информацию о величине числа, так и о его знаке. Если $\left|\dfrac{a}{P}\right|_1 \in \left[0,\dfrac{1}{2}\right)$, то число a — положительное и $F(a)$ равна величине числа a, разделенной на P. В противном случае a — отрицательное число, и $1- F(a)$ показывает относительную величину числа a. Округление величины $F(a)$ до 2^{-t} бита будем обозначать как $[F(a)]_{2^{-t}}$.

Точное значение величины $F(a)$ определяется неравенствами $[F(a)]_{2^{-t}} < F(a) < [F(a)]_{2^{-t}} + 2^{-t}$. Целая часть числа, полученная в результате суммирования констант k_i, представляет собой ранг числа, то есть такую непозиционную характеристику, которая показывает, сколько раз диапазон системы P был превзойден при переходе от представления чисел в системе остаточных классов к его позиционному представлению. При необходимости определение ранга может производиться непосредственно в процессе выполнения операции суммирования констант k_i. Дробная часть может быть записана также как $A \bmod 1$, потому что $A = \lfloor A \rfloor + A \bmod 1$. Количество разрядов дробной части числа определяется максимально возможной разностью между соседними числами. При точном сравнении, которое широко используется при делении чисел, необходимо вычислить значение, которое является эквивалентом преобразования из СОК в позиционную систему счисления. Для решения задачи сравнения чисел a и b достаточно знать приблизительно относительные значения чисел $\left|\dfrac{a}{P}\right|_1$ и $\left|\dfrac{b}{P}\right|_1$ по отношению к динамическому диапазону [0,1), которое выполняется достаточно просто, но при этом верно определяются соотношения $A = B$, $A > B$ или $A > B$.

Пример 1. Пусть дана система оснований $p_1 = 2$, $p_2 = 3$, $p_3 = 5$, $p_4 = 7$, объем диапазона $P = 2 \cdot 3 \cdot 5 \cdot 7 = 210$. Допустим, что в заданной СОК будут представлены только положительные числа. Определим величины $P_1 = \dfrac{P}{p_1} = 105$, $P_2 = \dfrac{P}{p_2} = 70$,

$P_3 = \frac{P}{p_3} = 42$, $P_4 = \frac{P}{p_4} = 30$ и сравним два числа $a = 25$ и $b = 30$, представленные в СОК по основаниям p_1, p_2, p_3, p_4. Определим числа a и b в СОК: $a = (1,1,0,4)$, $b = (0,0,0,2)$. Для реализации предлагаемого алгоритма найдем константы $k_i = \frac{\left|P_i^{-1}\right|_{p_i}}{p_i}$:

$$k_1 = \frac{\left|\frac{1}{|105|_2}\right|_2}{2} = \frac{1}{2} = 0{,}5;\ k_2 = \frac{\left|\frac{1}{|70|_3}\right|_3}{3} = \frac{1}{3} \approx 0{,}3333;$$

$$k_3 = \frac{\left|\frac{1}{|42|_5}\right|_5}{5} = \frac{3}{5} = 0{,}6;\ k_4 = \frac{\left|\frac{1}{|30|_7}\right|_7}{7} = \frac{4}{7} \approx 0{,}5714.$$

Далее, используя выражение (2), найдем функции $F(a)$ и $F(b)$:

$$F(a) = \frac{a}{P} \approx \left|1 \cdot 0{,}5 + 1 \cdot 0{,}3333 + 0 \cdot 0{,}6 + 4 \cdot 0{,}5714\right|_1 \approx 0{,}1189,$$

$$F(b) = \frac{b}{P} \approx \left|0 \cdot 0{,}5 + 0 \cdot 0{,}3333 + 0 \cdot 0{,}6 + 2 \cdot 0{,}5714\right|_1 \approx 0{,}1428.$$

Так как $\left|\frac{b}{P}\right|_1 > \left|\frac{a}{P}\right|_1$ то есть 0.1428 > 0,1189, то $b > a$, и действительно 30 > 25.

Алгоритм деления целых чисел $\frac{a}{b}$ можно описать итеративной схемой, которая выполняется в два этапа. На первом этапе осуществляется поиск старшей степени 2^i при аппроксимации частного двоичным рядом. На втором этапе осуществляется уточнение аппроксимирующего ряда. Чтобы получить диапазон, больший чем P, можно выбрать значение $P = P' \cdot p_{n+1}$, то есть потребуется расширить базу СОК, добавив дополнительный модуль. Чтобы избежать этого расширения базы, которое является вычислительно сложной операцией, необходимо сравнивать не делимое с промежуточными делителями, а текущие результаты итерации (i) с предыдущими значениями итераций (i — 1). Это позволит выполнить условие $0 < b < P - 1$.

Известные алгоритмы деления определяют частное на основе итерации $A' = A - QD$, где A и A', соответственно, текущее и следующее делимое, D — делитель, Q_1 — частное, которое генерируется на каждой итерации из полного диапазона СОК, а не выбирается из небольшого множества констант. В предлагаемом алгоритме частное определяется на основе

итерации $r_i = A - b2^i$, где A — некоторое делимое, b — делитель, а 2^i является членом аппроксимирующего ряда частного.

Сравнение алгоритмов показывает, что делимое во всех итерациях не меняется, а делитель умножается на константу, что существенно уменьшает вычислительную сложность. Приведенный выше алгоритм легко модифицируется в систему остаточных классов с применением приближенного метода сравнения модулярных чисел. При итерационном процессе деления в позиционной системе счисления, для поиска старшей степени ряда аппроксимации частного и для уточнения аппроксимирующего ряда сравниваются делимое с удвоенными делителями или с суммой членов ряда. Применение этого принципа для СОК может привести к ошибке процесса деления, так как при переполнении динамического диапазона восстановленное число выходит за пределы рабочего диапазона, числа которого будут меньше делимого, что не соответствует действительности, так как на самом деле числа будут превышать диапазон P. Например, если модули СОК равны $p_1 = 2$, $p_2 = 3$, $p_3 = 5$, $p_4 = 7$, тогда диапазон $P = 2 \cdot 3 \cdot 5 \cdot 7 = 210$.

Допустим при восстановлении получили число $A = 220$. В СОК $A = 220 = = (0,1,0,3)$. Диапазон P превышен на число 10, которое в СОК равно (0,1,0,3). При использовании относительных значений, число $A = 220$ выражается как $A' = 10$, что не соответствует действительности. Для преодоления этой трудности необходимо в СОК сравнивать результаты текущих значений итераций с предыдущими, что позволяет правильно определить большее или меньшее число.

Итак, факт переполнения динамического диапазона в СОК можно использовать для принятия решения «больше–меньше». На первой итерации происходит сравнение делимого с делителем, а на остальных итерациях происходит сравнение удвоенных значений делителей $q_i b < q_{i+1} b$. На каждой новой итерации происходит сравнение текущего значения с предыдущим. Количество требуемых итераций зависит от величин делимого и делителя. Последовательное применение этой операции приводит к формированию последовательности целых чисел $bq_1 < bq_2 < ... < bq_n > bq_{n+1}$. Таким образом, алгоритм реализуется за конечное число итераций. Пусть на $n + 1$ итерации зафиксирован случай $bq_n > bq_{n+1}$, что соответствует переполнению диапазона СОК, то есть $bq_{n+1} > P$ и $a < bq_{n+1}$. На этом процесс формирования интерполяции частного двоичным рядом или набором констант в СОК завершается. Итак, процесс аппроксимации част-

ного может осуществляться путем сравнения только удвоенных соседних приближенных делителей.

Рассмотрим алгоритм деления на примере. Действия будут производиться как над десятичными числами, так и над числами, представленными в системе остаточных классов.

Пример 2. Найти частное $Q=\frac{a}{b}$ от деления числа a = 201 на число b = 8. Выберем СОК с основаниями 2,3,5,7, тогда P = $p_1 p_2 p_3 p_4$ = 210. Константы k_i, соответственно, равны: k_1 = 0,5; k_2 = 0,3333; k_3 = 0,6; k_4 = 0,5714.

Представим в СОК числа a и b.

$$a_{10} = 201 \to (1,0,1,5)_{СОК},$$

$$b_{10} = 8 \to (0,2,3,1)_{СОК}.$$

Относительные значения этих чисел, соответственно, равны:

$$F(a)=\left|\frac{a}{P}\right|_1 = \left|\sum_{i=1}^{4} k_i \alpha_i\right|_1 \approx |0,5\cdot 1+0,33\cdot 0+0,6\cdot 1+0,57\cdot 5|_1 = 0,95,$$

$$F(b)=\left|\frac{b}{P}\right|_1 = \left|\sum_{i=1}^{4} k_i \beta_i\right|_1 \approx |0,5\cdot 0+0,33\cdot 2+0,6\cdot 3+0,57\cdot 1|_1 = 0,03.$$

Решение. Деление a на b осуществляется по следующему алгоритму.

Для интер поляции частного определим степени 2^i, представленные в СОК.

I. Поиск старшей степени при аппроксимации частного двоичным рядом.

1. На первой итерации сравниваем $\left|\frac{a}{P}\right|_1$ и $\left|\frac{b}{P}\right|_1$. Если $\left|\frac{a}{P}\right|_1 < \left|\frac{b}{P}\right|_1$, то в память ничего не записывается, так как в этом случае делитель больше делимого, на этом процесс деления заканчивается, и частное равно 0. Если $\left|\frac{a}{P}\right|_1 = \left|\frac{b}{P}\right|_1$, то в память записываем константу $q_0 = 2^0$, а если $\left|\frac{a}{P}\right|_1 > \left|\frac{b}{P}\right|_1$, то реализуется итерационный процесс деления.

Допустим, что делимое a и делитель b имеют следующие значения:

$$a_{10} = 201 \to (1,0,1,5)_{COK},\ b_{10} = 8 \to (0,2,3,1)_{COK}.$$

Найдем частное от деления $Q = \left\lfloor \frac{a}{b} \right\rfloor$.

Тогда $$\left|\frac{a}{P}\right|_1 = \left|\frac{201}{210}\right|_1 \approx 0{,}95,$$

$$\left|\frac{b}{P}\right|_1 = \left|\frac{8}{210}\right|_1 \approx 0{,}03,$$

отсюда 0,95 > 0,03, то есть

$$a > b \Rightarrow \begin{cases} 201 > 8 & -в\ десятичном\ виде \\ (1,0,1,5) > (0,2,3,1) & -в\ системе\ остаточных\ классов \end{cases}.$$

В память записываем константы, представленные в двоичном коде $q_1 = 2^0$ и в СОК $q_5 = (1,1,1,1)$.

2. Далее во всех остальных итерациях будем сравнивать текущие значения с предыдущими. Так, на второй итерации умножаем знаменатель на 2^1, то есть $q_1 = 2$, если $\left|\frac{b}{P}\right|_1 < 2 \cdot \left|\frac{b}{P}\right|_1$, то в память запишем 2^1.

Тогда $b_{1_{10}} = 8 \cdot 2 = 16$, $b_{1_{10}} = 16 \to (0,2,3,1) \cdot (2,2,2,2) = (0,1,1,2)$.

Сравнение дает следующий результат:

$$\left|\frac{bq_1}{P}\right|_1 = \sum_{i=1}^{n} k_i b_i \approx \left|0{,}5 \cdot 0 + 0{,}33 \cdot 1 + 0{,}6 \cdot 1 + 0{,}57 \cdot 2\right|_1 = 0{,}07.$$

Тогда $\left|\frac{b}{P}\right|_1 \approx 0{,}03$, $\left|\frac{bq_1}{P}\right|_1 \approx 0{,}07$, так как 0,03 < 0,07, то .

В память записываем число в двоичном коде $q_2 = 2^1$, а в СОК $q_2 = (0,2,2,2)$.

3. Аналогично получим результаты сравнения на следующих итерациях. На третьей итерации умножаем знаменатель b на $q_2 = 2^2$,
если $\left|\frac{bq_1}{P}\right|_1 < \left|\frac{bq_2}{P}\right|_1$, то в память запишем 2^2.

Так как $b_{2_{10}} = 8 \cdot 4 = 32$, $b_{2_{10}} = 32 \to (0,2,3,1) \cdot (0,1,4,4) = (0,2,2,4)$.

$$\left|\frac{bq_2}{P}\right|_1 = \sum_{i=1}^{n} k_i b_i \approx \left|0{,}5 \cdot 0 + 0{,}33 \cdot 2 + 0{,}6 \cdot 2 + 0{,}57 \cdot 4\right|_1 = 0{,}14.$$

Тогда $\left|\frac{bq_1}{P}\right|_1 < \left|\frac{bq_2}{P}\right|_1$, так как 0,07 < 0,14.

В память записываем число в двоичном коде $q_3 = 2^2$,
а в СОК $q_3 = (0,1,4,4)$.

4. На четвертой итерации получим результат
$b_{3_{10}} = 8 \cdot 8 = 64$; $b_{3_{10}} = 64 \to (0,2,3,1) \cdot (0,2,3,1) = (0,1,4,1)$.

$$\left|\frac{bq_3}{P}\right|_1 = \sum_{i=1}^{n} k_i b_i \approx \left|0{,}5 \cdot 0 + 0{,}33 \cdot 1 + 0{,}6 \cdot 4 + 0{,}57 \cdot 1\right|_1 = 0{,}3.$$

Тогда $\left|\frac{bq_2}{P}\right|_1 < \left|\frac{bq_3}{P}\right|_1$, так как 0,14 < 0,3.

В память записываем число в двоичном коде $q_4 = 2^3$, а в СОК $q_5 = (0,2,3,1)$.

5. На пятой итерации получим следующий результат:

$b_{4_{10}} = 8 \cdot 16 = 128$, $b_{4_{10}} = 128 \to (0,2,3,1) \cdot (0,1,1,2) = (0,2,3,2)$,

$$\left|\frac{bq_4}{P}\right|_1 = \sum_{i=1}^{n} k_i b_i \approx \left|0{,}5 \cdot 0 + 0{,}33 \cdot 2 + 0{,}6 \cdot 3 + 0{,}57 \cdot 2\right|_1 = 0{,}6.$$

Тогда $\left|\frac{bq_3}{P}\right|_1 < \left|\frac{bq_4}{P}\right|_1$, так как 0,3 < 0,6.

В память записываем число в двоичном коде $q_5 = 2^4$,
а в СОК $q_5 = (0,1,1,2)$.

6. На шестой итерации получим следующий результат:

$$b_{5_{10}} = 8 \cdot 32 = 256,\ b_{5_{10}} = 256 \to (0,2,3,1) \cdot (0,2,2,4) = (0,1,1,4),$$

$$\left|\frac{bq_5}{P}\right|_1 = \sum_{i=1}^{n} k_i b_i \approx \left|0{,}5 \cdot 0 + 0{,}33 \cdot 1 + 0{,}6 \cdot 1 + 0{,}57 \cdot 4\right|_1 = 0{,}21.$$

Тогда $\left|\frac{bq_5}{P}\right|_1 > \left|\frac{bq_4}{P}\right|_1$, так как 0,6 > 0,21.

Таким образом, произошло переполнение, и в память ничего не записывается. Процесс формирования аппроксимационного ряда частного завершается.

Итак, первое неравенство 0,95 > 0,03 определяет начало итерационного процесса деления, а последовательность последующих неравенств 0,07 > 0,03; 0,14 > 0,07; 0,3 > 0,14; 0,6 > 0,3, или 0,3 < 0,07 < 0,14 < 0,3 < 0,6 > 0,21 определяет количество итераций. Отсюда видно, что на шестом шаге заканчивается возрастающая последовательность. Этот факт сигнализирует об окончании итераций, так как полученный путем последовательного умножения на 2 делитель превысил делимое.

II. Уточнение аппроксимирующего ряда частного от деления a на b начнем со старшего q_n.

1. Из памяти выбираем старшую степень, то есть q_5 = (0,1,1,2), умножаем на знаменатель b = (0,2,3,1) и сравниваем с a.

Тогда $q_5 = (0,1,1,2) \cdot (0,2,3,1) = (0,2,3,2)$, а

$$\left|\frac{q_5}{P}\right|_1 \approx |\ 0{,}5\ \cdot 0 + 0{,}33 \cdot 2 + 0{,}6 \cdot 3 + 0{,}57 \cdot 2\ |_1 = 0{,}6$$

Так как 0,95 > 0,6, то в качестве старшей степени берем 2^4, а в СОК (0,1,1,2).

2. Из памяти выбираем степень 2^3 и вычисляем $(2^4 + 2^3) \cdot 8 = 192$, а в СОК $((0,1,1,2) + (0,2,3,1) \cdot (0,2,3,1) = (0,0,2,3)$.

Тогда $\left|\frac{q_5 + q_4}{P}\right|_1 \approx 0{,}5\ \cdot 0 + 0{,}33 \cdot 0 + 0{,}6 \cdot 2 + 0{,}57 \cdot 3\ |_1 = 0{,}91.$

Так как 0,91 > 0,6 и 0,95 > 0,91, то в качестве следующего члена ряда берем 2^3, а в СОК (0,2,3,1).

3. Из памяти берем 2^2 и вычисляем $(2^4 + 2^3 + 2^2) \cdot 8 = 224$.

Тогда ((0,1,1,2) + (0,2,3,1) + (0,1,4,4)) · (0,2,3,1) = (0,2,4,0),

$$\left|\frac{q_5 + q_4 + q_3}{P}\right|_1 \approx \left|0{,}5 \cdot 0 + 0{,}33 \cdot 2 + 0{,}6 \cdot 4 + 0{,}57 \cdot 0\right|_1 = 0{,}06.$$

Так как 0,06 < 0,91, то произошло переполнение диапазона P и степень 2^2, или в СОК (0,1,4,1) из аппроксимационного ряда исключается.

4. Из памяти берем 2^1 и вычисляем $(2^4 + 2^3 + 2^1) \cdot 8 = 208$.
Тогда ((0,1,1,2) + (0,2,3,1) + (0,2,2,2)) · (0,2,3,1) = (0,1,3,5),

$$\left|\frac{q_5 + q_4 + q_2}{P}\right|_1 \approx \left|0{,}5 \cdot 0 + 0{,}33 \cdot 1 + 0{,}6 \cdot 3 + 0{,}57 \cdot 5\right|_1 = 0{,}98.$$

Так как 0,95 < 0,98, то степень 2^1 или (0,2,2,2) из аппроксимационного ряда исключается.

5. Из памяти берем 2^0 и вычисляем $(2^4 + 2^3 + 2^0) \cdot 8 = 200$.
Тогда ((0,1,1,2) + (0,2,3,1) + (0,1,1,1)) · (0,2,3,1) = (0,2,0,4),

а $$\left|\frac{q_5 + q_4 + q_0}{P}\right|_1 \approx \left|0{,}5 \cdot 0 + 0{,}33 \cdot 2 + 0{,}6 \cdot 0 + 0{,}57 \cdot 4\right|_1 = 0{,}94.$$

Так как 0,95 > 0,94, поэтому в качестве младшей степени берем 2^0 или в СОК (1,1,1,1).

Следовательно, частное равно (0,1,1,2) + (0,2,3,1) + (1,1,1,1) = (1,1,0,4).

Для определения частного необходимо сложить оставшиеся члены аппроксимационного ряда. Из приведенного примера видно, что остались следующие члены ряда: (0,1,1,2), (0,2,3,1) и (1,1,1,1). Тогда частное определяется путем суммирования членов ряда

$$\frac{a}{b} = (0,1,1,2) + (0,2,3,1) + (1,1,1,1) = (1,1,0,4).$$

По остаткам частного восстановим позиционное число с помощью выражения (1), тогда

$$\frac{a}{b}=\left|\sum_{i=1}^{n}\frac{P}{p_i}\left|P_i^{-1}\right|_{p_i}\cdot\alpha_i\right|_P=\left|\frac{210}{2}\cdot1\cdot1+\frac{210}{3}\cdot1\cdot1+\frac{210}{5}\cdot3\cdot0+\frac{210}{7}\cdot4\cdot4\right|_P=$$

$$=\left|105+70+0+480\right|_P=\left|655\right|_{210}=25.$$

Действительно, $P_2=\frac{1}{p_2}=25$.

Результаты в СОК и в позиционной системе счисления совпадают, что говорит о правильности проведенного деления.

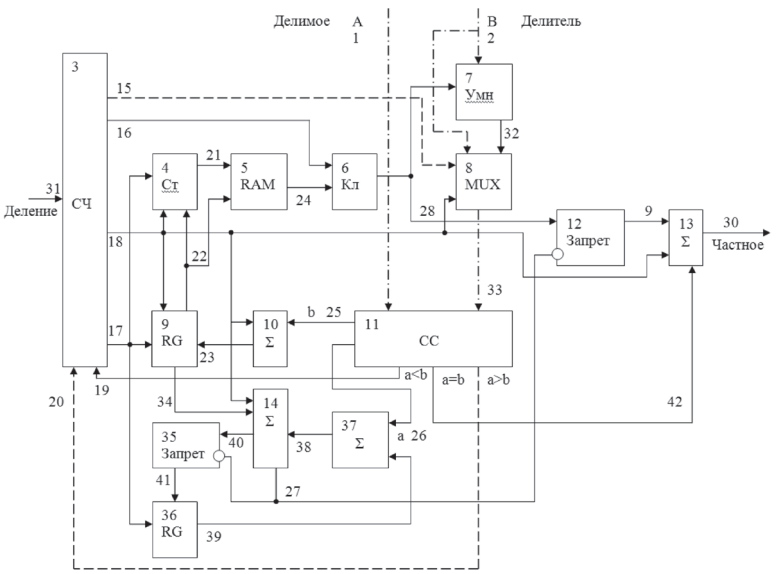
Техническая реализация алгоритма основного деления модулярных чисел.

Улучшеннй алгоритм деления модулярных чисел на основе приближенного метода сравнения чисел состоит из следующих шагов.

1. Вычисляем приближенные значения делимого $F(a)$ и делителя $F(b)$, и сравниваем их. Если $F(a) < F(b)$, то процесс деления заканчивается и частное $\left\lfloor\frac{a}{b}\right\rfloor=0$. Если $F(a) = F(b)$, то процесс деления заканчивается и частное $\left\lfloor\frac{a}{b}\right\rfloor=1$. Если $F(a) > F(b)$, то осуществляется поиск старшей степени 2^k при аппроксимации частного двоичным кодом.
2. Сдвигаем функцию $F(b)$ влево до появления переноса старшего значащего разряда в знаковый разряд. Количество сдвигов определяет старшую степень, которая регистрируется счетчиком импульсов.
3. Из памяти выбираем константу 2^k (старшая степень ряда), умножаем ее на делитель $F_1(b) = b2^k$ и подаем на вход схемы сравнения. Константы $2^j \bmod p_i$, где $i=\overline{1,n}$, $1\le j\le\log_2 P$ предварительно записаны в память.
4. Находим $\Delta_i = F(a) — F_1(b)$. Если в знаковом разряде Δ_i стоит «1», то соответствующая степень ряда отбрасывается, если стоит «0», то в сумматор частного добавляем значение члена ряда с этой степенью, то есть 2^k.
5. Сдвигаем $F_1(b)$ «вправо», и проверяем член ряда со степенью $2^{k–1}$.
6. Находим $\Delta_2 = \Delta_1 — F_1(b)$ и выполняем действия в соответствии с пунктом 4.

7. Аналогично проверяем все оставшиеся члены ряда до нулевой степени. Полученный остаток $\Delta_i = \Delta_{i-1} - F_{i-1}(b) \approx 0$. В случае когда делитель принимает минимальное значение, а делимое — максимальное, то порог Δ_i можно взять больше нуля, что позволит сократить количество итераций при делении большого делимого и маленького делителя. В процессе этих преобразований суммируем все разрешенные члены ряда.

Процесс сдвига и вычитания завершается проверкой нулевой степени ряда. В накопительном сумматоре суммируются только разрешенные члены аппроксимирующего ряда частного. После последнего шага работа алгоритма завершается.



Техническая реализация нового алгоритма деления показана на рисунке. Принцип работы данного изобретения излагается ниже. Устройство для деления модулярных чисел позволяет выполнять операцию деления при произвольных значениях делимого и делителя без каких-либо дополнительных предварительных операций, кроме операций, устанавливающих устройство в исходное состояние.

Режим аппроксимации частного двоичным рядом.

В исходном состоянии схема управления (СУ) 3 по шине 18 выдает сигнал «установка в 0» по которому регистр 9, счетчик 4 и сумматоры 10, 13, 14 устанавливаются в начальное состояние, а мультиплексор 8 (сигнал поступает на адресный вход) коммутирует шину 2 на вход схемы сравнения 11, шины 33.

Делимое и делитель, представленные в системе остаточных классов по модулям p_1, p_1,..., p_n, соответственно, по шинам 1 и 33, поступают на вход схемы сравнения 11, в которой происходит сравнение относительных значений делимого и делителя. Если $F(a)=\left|\frac{a}{P}\right|_1 < F(b)=\left|\frac{b}{P}\right|_1$, то схема сравнения формирует сигнал по шине 19, который поступает на вход схемы управления 3 и устройство устанавливается в начальное состояние. Частное (шина 30) равно нулю. Если $F(a)=\left|\frac{a}{P}\right|_1 = F(b)=\left|\frac{b}{P}\right|_1$, то схема сравнения выдает сигнал равенства делимого и делителя по шине 42 на вход сумматора 13, в который записывается константа $1 = (1,1,1,1)_{COK}$. Если $F(a)=\left|\frac{a}{P}\right|_1 > F(b)=\left|\frac{b}{P}\right|_1$, то схема сравнения формирует сигнал по шине 20, под действием которого схема управления 3 переводит устройство в режим аппроксимации ряда частного. Значения $k_i\alpha_i$ и $k_i\beta_i$ с выходов схем 5–1, 5–2, … , 5-n и 6–1, 6–2, … , 6-n схемы сравнения 11 по шинам 25 и 26, соответственно, поступают на вход сумматоров 10 и 37. Относительное значение делимого, представленное в дополнительном коде, выход сумматора 37 по шине 38 поступает на вход схемы вычитателя 14. Относительное значение делителя, выход сумматора 10 по шине 23 поступает на вход регистра сдвига 9. Под действием тактовых импульсов схемы управления (шина 17) происходит сдвиг содержимого регистра 9 и счет этих импульсов счетчиком 4. Как только старший значащий разряд делителя становится знаковым, т. е. произошло переполнение, то эта единица по шине 22 останавливает счет импульсов счетчиком 4 и активируется сигнал «разрешение считывания» информации из памяти 5. Счетчик 4 регистрирует состояние, формирующее высшую степень 2^k ряда частного. В регистре 36 в это время во всех разрядах записано значение «0» и он не оказывает никакого влияния на сумматор 37. На этом режим интерполяции частного заканчивается. Итак, для интерполяции частного потребовалась одна итерация сравнения, одна операция суммирования и одна операция сдвига на k разрядов.

Операция сдвига эквивалентна одной итерации известного алгоритма. В каждую итерацию входит операция удвоения делителя и операция сравнения результата удвоения со значением делителя. Замена

абсолютных значений их относительными значениями позволила при интерполяции частного получить выигрыш по сравнению с рассмотренными выше алгоритмами в k раз, где k — количество сдвигов до появления переполнения.

Режим уточнения аппроксимирующего ряда частного от деления a на b.

Схема управления 3 формирует сигналы по шинам 15 и 17, которые подаются на адресные входы мультиплексора 8, и коммутирует умноженное значение делителя на высшую степень ряда частного (шина 32) на выход мультиплексора 8 и переводит регистр 9 в режим сдвига «вправо». Высшая степень ряда через ключ 6 (шина 28), который под действием двух сигналов, поступивших на его вход по шинам 16 и 24, подает на вход схемы умножителя 7 и схемы «запрета» 12 значения высшей степени ряда. Поступившие данные на вход схемы сравнения 11 сравниваются. Если $F(a)=\left|\frac{a}{P}\right|_1 < \left|\frac{q^k b}{P}\right|_1$, то схема сравнения 11 выдает сигнал по шине 9 и устанавливает устройство в исходное состояние. Если $F(a)=\left|\frac{a}{P}\right|_1 = \left|\frac{q^k b}{P}\right|_1$, то формируется сигнал $a = b$ и по шине 42 записывает в сумматор «единицу», и устройство переходит в исходное состояние. Если $F(a)=\left|\frac{a}{P}\right|_1 > \left|\frac{q^k b}{P}\right|_1$, то сигнал по шине 20 устанавливает устройство в режим уточнения аппроксимационного ряда частного.

Значение умноженного на высшую степень делителя и делимого, как в случае интерполяции ряда, поступают на вход сумматоров 10 и 37. Содержимое сумматора 10 по шине 23 подается на вход регистра 9, стирает старую информацию и записывает новое значение, и далее по шине 34 подается на вход сумматора 14, а содержимое сумматора 37 в дополнительном коде подается на вторые входы сумматора 14, где происходит суммирование (вычитание) умноженного делителя на высшую степень из содержимого делимого. Делимое и делитель, как и ранее, представлены своими относительными значениями. Если в знаковом разряде результата вычитания в сумматоре 14 стоит «ноль», то есть делимое больше делителя, тогда на запрещающих входах схем «запрета» 12 и 35 формируются «нули», и высшая степень частного по шине 28 через схему «запрета» 12 поступает на вход сумматора 13 по шине 9, а результат вычитания сумматора 14, то есть остаток делимого, приходящий на остальные степени по шине 40 через схему «запрета» 35, подается на вход регистра 36 и далее по шине 39 поступает на вход сумматора 37, где удаляется старое содержимое

и записывается новое значение. Далее происходит сдвиг «вправо» содержимого регистра 9, и процесс происходит аналогично вышеизложенному. При этом, если в знаковом разряде результата суммирования в сумматоре 14 будет стоять «1», то есть относительное значение делителя больше делимого, то появившаяся единица, поступающая на запрещающие входы схемы «запрета» 12 и 35 запрещает прохождение соответствующей степени на сумматор 13 и результат суммирования сумматора 14 на вход регистра 36, то есть регистр сохраняет прежнее значение результата суммирования. Таким образом, на вход сумматора 13 поступают только те уточненные степени, которые являются членами ряда частного. Процесс преобразования заканчивается после анализа степени 2^0. Таким образом, при уточнении итеративно удаляются лишние члены аппроксимационного ряда частного путем несложных преобразований, состоящих из операций сдвига и сложения. В известном алгоритме, при уточнении, используются такие сложные операции, как умножение и сравнение, которые входят в каждую итерацию.

Итак, основное деление модулярных чисел осуществляется примерно за 2 итерации сравнения и $k-1$ операций сдвига и сложения, а в известном алгоритме необходимо $2k$ итераций сравнения, k операций умножения и $2k$ операций суммирования. Выигрыш в скорости деления модулярных чисел достигает примерно k итераций. Это лучший на сегодняшний день алгоритм основного деления модулярных чисел. Это достоинство предложенного алгоритма по сравнению с известным достигается тесной связью архитектурных вычислений с аппаратной реализацией, что позволило значительно сократить вычислительную сложность деления модулярных чисел. Предложенный алгоритм отличается от известных простотой его реализации, который требует меньшего объема вычислений по сравнению с существующими алгоритмами.

Нами предложен новый алгоритм деления модулярных чисел на основе использования приближенного метода сравнения чисел, которые являются самыми быстрыми алгоритмами деления в СОК на сегодняшний день. Вычислительная сложность улучшенного нового алгоритма по сравнению с известными модификациями уменьшена примерно в 5 раз. Предложенная техническая реализация алгоритма позволяет сократить в k раз количество итераций сравнения и заменить итерационные умножения суммированиями со сдвигом. Перечисленные свойства указывают на существенное преимущество предложенного алгоритма перед известными ранее.

Противоречие между вычислительной сложностью определения основных проблемных процедур в СОК и их быстродействием разрешена путем замены абсолютных величин делимого и делителя их относительными значениями и простотой их вычисления, что позволяет повысить скорость выполнения операции деления в СОК.

Некоторая возможная вариативность нового алгоритма деления может повысить его эффективность в тех приложениях, где часто появляется необходимость в делении, которое ранее определяло наибольшую составляющую алгоритмической сложности и сдерживало широкое применение СОК при разработке новых классов вычислительных систем. Внедрение полученных результатов позволит снять это ограничение и расширить область применения модулярной арифметики.

Литература

1. **Molahosseini A. S., Sorouri S., Zarandi A. A.** E. Research challenges in next-generation residue number system architectures // Computer Science & Education (ICCSE), 7th International Conference. 2012. P. 1658–1661.
2. **Yatskiv V., Su Jun, Yatskiv N., Sachenko A., Osolinskiy O.** Multilevel method of data coding in WSN // Intelligent Data Acquisition and Advanced Computing Systems (IDAACS), IEEE 6th International Conference. 2011. P. 863–866.
3. **Su Jun, Zhengbing Hu.** Method and dedicated processor for image coding based on residue number system // Modern Problems of Radio Engineering Telecommunications and Computer Science (TCSET), International Conference. 2012. P. 406–407.
4. **Су Цзунь.** Многоуровневый метод кодирования данных в беспроводных сенсорных сетях // Электротехнические и компьютерные системы. 2011. №04(80). С. 213–218.
5. **Ляхов П.А., Червяков Н.И.** Реализация многоканальных фильтров в системе остаточных классов // Инфокоммуникационные технологии. 2011. Т. 9. №2. С. 4–7.
6. **Chervyakov N. I., Veligosha A. V., Tyncherov K. T., Velikikh S.A.** Use of modular coding for high-speed digital filter design // Cybernetics and Systems Analysis. 1998. No. 34. P. 254–260.
7. **Zheng XD., Xu J., Li W.** Parallel DNA arithmetic operation based on n-moduli set // Applied Mathematics and Computation. 2009. 212(1). P. 177–184.
8. **Gomathisankaran M., Tyagi A., Namuduri K.** HORNS: A homomorphic encryption scheme for Cloud Computing using Residue Number System // Information Sciences and Systems (CISS), 45th Annual Conference. 2011. P. 1–5.

9. **Alia G., Martinelli E.** NEUROM: a ROM based RNS digital neuron // Neural Networks. 2005. № 18. P. 179–189.
10. **Червяков Н. И., Сахнюк П. А., Шапошников А. В., Макоха А. Н.** Нейрокомпьютеры в остаточных классах. М.: Радиотехника, 2003. 272 с.
11. **Червяков Н. И., Сахнюк П. А., Шапошников А. В., Ряднов С. А.** Модулярные параллельные вычислительные структуры нейропроцессорных систем. М.: ФИЗМАТЛИТ, 2003. 288 с.
12. **Галушкин А. И., Червяков Н. И., Евдокимов А. А., Лавриенко И. Н., Лавриенко А. В., Шаров Д. А.** Применение искусственных нейронных сетей и системы остаточных классов в криптографии. М.: ФИЗМАТЛИТ, 2012. 280 с.
13. **Червяков Н. И.** Организация арифметических расширителей в микропроцессорных системах, базирующихся на множественном представлении информации // Управляющие системы и машины. 1987. №1. С. 26–29.
14. **Червяков Н. И.** Методы и принципы построения модулярных нейрокомпьютеров // 50 лет модулярной арифметике, сборник трудов Юбилейной Международной научно-технической конференции. Москва, ОАО «Ангстрем», МИЭТ. 2006. С. 239–249.
15. **Червяков Н. И.** Реализация высокоэффективной модулярной цифровой обработки сигналов на основе программируемых логических интегральных схем // Нейрокомпьютеры: разработка, применение. 2006. № 10. С. 24–36.
16. **Червяков Н. И., Лавриненко И. Н.** Модулярные методы и алгоритмы деления на основе спуска Ферма и итераций Ньютона // Инфокоммуникационные технологии. 2009. Т. 7, № 4. С. 9–12.
17. **Червяков Н. И., Лавриненко И. Н., Лавриненко С. В., Мезенцева О. С.** Методы и алгоритмы округления, масштабирования и деления модулярных чисел // 50 лет модулярной арифметике, сборник трудов Юбилейной Международной научно-технической конференции. Москва, ОАО «Ангстрем», МИЭТ. 2006. С. 291–310.
18. **Червяков Н. И.** Методы, алгоритмы и техническая реализация основных проблемных операций, выполняемых в системе остаточных классов // Инфокоммуникационные технологии. 2011. Т. 9. № 4. С. 4–12.
19. **Chin-Chen Chang, Yeu-Pong Lai.** A division algorithm for residue numbers // Applied Mathematics and Computation. 2006. V. 172, No. 1. P. 368–378.
20. **Синьков М. В., Синькова Т. В., Федоренко А. В., Чапор А. А.** Нетрадиционная система остаточных классов и ее основоположник И. Я. Акушский. [Online] www.icfcst.kiev.ua/Symposium/Proceedings2/Sinkov.rtf.

ОБ АВТОРАХ

Червяков Николай Иванович
Доктор технических наук, профессор, завкафедрой прикладной математики и математического моделирования. Северо-Кавказский федеральный университет, Институт математики и естественных наук.

Chervyakov Nikolay Ivanovich
North Caucasus Federal University, Institute of Mathematics and Natural Sciences, Professor, Head of the Applied Mathematics and Mathematical Modeling Department, Doctor of Technical sciences.

E-mail **k-fmf-primath@stavsu.ru.**

Бабенко Михаил Григорьевич
Кандидат физико-математических наук. Северо-Кавказский федеральный университет, Институт математики и естественных наук, кафедра прикладной математики и математического моделирования.

Babenko Makhail Grigor'evich
North Caucasus Federal University, Institute of Mathematics and Natural Sciences, Departmentof the Applied Mathematics and Mathematical Modeling Department, Candidate of Physical and Mathematical Sciences.

E-mail **whbear@yandex.ru.**

Ляхов Павел Алексеевич (автор для переписки)
Кандидат физико-математических наук. Северо-Кавказский федеральный университет, Институт математики и естественных наук, кафедра прикладной математики и математического моделирования.

Lyakhov Pavel Alekseevich
North Caucasus Federal University, Institute of Mathematics and Natural Sciences, Department of the Applied Mathematics and Mathematical Modeling Department, Candidate of Physical and Mathematical Sciences.

E-mail **ljahov@mail.ru.**

Лавриненко Ирина Николаевна
Кандидат физико-математических наук, доцент.Северо-Кавказский федеральный университет, Институт математики и естественных наук, кафедра прикладной математики и математического моделирования.

Lavrinenko Irina Nikolaevna
North Caucasus Federal University, Institute of Mathematics and Natural Sciences, Department of the Applied Mathematics and Mathematical Modeling Department, Candidate of Physical and Mathematical Sciences, Associate professor.

E-mail **k-fmf-primath@stavsu.ru.**