Методика выбора базисных оснований для рекурсивной модулярной арифметики

Р. А. Соловьёв, Д. В. Тельпухов
Институт проблем проектирования в микроэлектронике РАН, Москва, Россия
e-mail: turbo@ippm.ru, nofrost@inbox.ru

Соловьёв Р.А., Тельпухов Д.В. Методика выбора базисных оснований для рекурсивной модулярной арифметики // Вычислительные технологии. 2014. Т. 19, № 4. С. 99–106.

Предложена методика выбора базисных и рабочих оснований для рекурсивной модулярной арифметики, которая базируется на идее выразить систему модулей традиционной модулярной арифметики через систему субмодулей, имеющую меньшую размерность. Новое рекурсивное представление данных позволяет устранить часть известных недостатков традиционной модулярной арифметики. Рекурсивная модулярная арифметика в зависимости от выполняемых операций накладывает ограничения на систему модулей. Детально изучены эти ограничения и предложена методика построения модульного базиса для рекурсивной модулярной арифметики.

Ключевые слова: модулярная арифметика, система остаточных классов, конечные поля, китайская теорема об остатках, вычет.

Solovyev R.A., Telpukhov D.V. Methodology for basic selection in recursive residue number system // Computational Technologies. 2014. Vol. 19, No. 4. P. 99–106.

Recently, a new method for the representation of numbers in residue number system (RNS), called recursive residue number system (RRNS) was proposed. In this method, it becomes possible to represent large numbers using sets of modules with very small modules using the recursive decomposition. RRNS preserves all the advantages of traditional RNS related to parallel independent computations for each module. The method can increase the speed of computation by reducing complexity of arithmetic operations for each modular channel. But then RRNS increases the number of channels and imposes significant restrictions when choosing sets of modules. First-of-a-kind research on the choice of optimal sets of modules in RRNS is proposed in this paper. An algorithm for selecting a RRNS basis for the two given values: the required dynamic range (DR) and the maximum bit length of basic modules (MBLBM) (i. e. modules, which will be used for arithmetic operations in modular channels) are proposed in this paper. It is shown under which combinations of DR and MBLBM RRNS the proposed method has the advantage over traditional RNS. The graphs of data redundancy in RRNS and examples of RRNS for construction of sets of modules are presented. According to the results shown in the article it is appropriate to use MBLBM from 3 to 7 bits for typical tasks. Moreover, if DR is less than 64 bits, the most effective MRBM is from 3 up to 5 bits depending on the possible redundancy in the RRNS representation.

Keywords: modular arithmetic, residue number system, finite field, Chinese remainder theorem, remainder.

1. Введение в рекурсивную модулярную арифметику

Модулярная арифметика (система остаточных классов (СОК)) является непозиционной системой счисления. Представление числа в модулярной арифметике основано на понятии вычета и китайской теореме об остатках [1, 2]. Система остаточных классов определяется набором взаимно простых модулей $(p_1, p_2, \ldots, p_i, \ldots, p_n)$ с произведением $M = p_1 \cdot p_2 \cdot, \ldots, \cdot p_n$ так, что каждому целому числу x из отрезка $[0; M-1]$ ставится в соответствие набор вычетов $(x_1, x_2, \ldots, x_n)$, где $x_1 \equiv x \bmod p_1, x_2 \equiv x \bmod p_2, \ldots, x_n \equiv x \bmod p_n$. Китайская теорема об остатках при этом гарантирует взаимную однозначность представления для чисел из отрезка $[0; M-1]$.

Использование модулярной арифметики при проектировании специализированных вычислителей имеет преимущества, основными из которых являются параллелизм вычислений, возможность обнаружения и коррекции ошибок при выполнении арифметических операций и энергоэффективность вычислителей на базе СОК [3]. Получение этих преимуществ особенно важно для целого ряда новых приложений, в частности для активно развивающегося направления мобильных навигационных систем на основе распределённого множества сенсоров [4]. Известен также ряд недостатков: большие накладные расходы в виде преобразователей из позиционного кода в модулярный и обратно; представление модульных операций посредством операций двоичной арифметики, что приводит к избыточности оборудования при их реализации; неравномерность модульных вычислителей по сложности и времени выполнения операций.

Для преодоления части указанных недостатков был предложен новый подход к проектированию модулярных вычислителей, который получил название *рекурсивная модулярная арифметика* (РМА) [5]. Идея РМА основана на принципе глубокого распараллеливания модульных операций модулярной арифметики с основаниями $p_1, p_2, \ldots, p_n$ посредством рекурсивного редуцирования модульных операций по каждому рабочему основанию p_j $(i \le j \le n)$ к модульным вычислениям по предшествующим основаниям $p_1, p_2, \ldots, p_{i-1}$, имеющим то или иное технологическое преимущество, называемое базисным. При этом редукция допустима только при выполнении условия согласования вычислительных диапазонов по каждому рабочему модулю p_j с вычислительными диапазонами по соответствующим комплексам базисных оснований.

2. Представление данных в рекурсивной модулярной арифметике

Пусть заданы система модулей $(p_1, p_2, \ldots, p_i, \ldots, p_n)$ и некоторый вектор $\mathbf{A} = (a_1, a_2, \ldots, a_i, \ldots, a_n)$. Выразим a_i через систему субмодулей $(p_{i,1}, p_{i,2}, \ldots, p_{i,k})$, для которой $P_i = p_{i,1} \cdot p_{i,2} \cdot, \ldots, \cdot p_{i,k}$ и значение a_i строго меньше P_i. Тогда $a_i = (a_{i,1}, a_{i,2}, \ldots, a_{i,k})$, и в этом случае вектор $\mathbf{A}$ можно представить в следующем виде: $(a_1, a_2, \ldots, a_{i-1}, (a_{i,1}, a_{i,2}, \ldots, a_{i,k}), a_{i+1}, \ldots, a_n)$, где элемент a_i заменён на соответствующий вектор.

Описанная процедура является основой при построении структуры РМА. Совокупность k младших модулей $(q_1, q_2, \ldots, q_k)$ называется системой базовых модулей с произведением $Q = q_1 \cdot q_2 \cdot, \ldots, \cdot q_k$. Все следующие модули $(p_1, p_2, \ldots, p_m)$ с произведением $P = p_1 \cdot p_2 \cdot, \ldots, \cdot p_m$ называются рабочими и последовательно выражаются через предыдущие. Модуль p_1 выражается через вектор $\mathbf{r}_1 = (q_1, q_2, \ldots, q_k)$, модуль p_2 — через вектор $\mathbf{r}_2 = (q_1, q_2, \ldots, q_k, p_1)$ и т. д. Таким образом формируется рекурсивное представ-

ление каждого последующего модуля через предыдущие. Суммарное число модулей в системе $n = k + m$. Полученное преобразование представлено в формуле (1) и схематично изображено на рис. 1:

$$X \to \left(q_1, q_2, \ldots, q_k, \overbrace{(q_1, q_2, \ldots, q_k)}^{\mathbf{r}_1}, \overbrace{\left(q_1, q_2, \ldots, q_k, \overbrace{(q_1, q_2, \ldots, q_k)}^{\mathbf{r}_1} \right)}^{\mathbf{r}_2} \right). \qquad (1)$$

Поясним процедуру рекурсивных преобразований на примере. Рассмотрим в качестве базисных модулей трёхбитные простые числа $q_1 = 5$, $q_2 = 7$. Очевидно, что вычетами по модулям 5 и 7, для которых $Q = 5 \cdot 7 = 35$, можно однозначно представить любой вычет по модулю $p_1 = 31$. В то же время вычетами по модулям 5, 7 и 31, где вычеты по модулю 31 представимы по модулям 5 и 7, можно однозначно представить любой вычет по модулю $p_2 = 997$. Динамический диапазон для заданной системы модулей $M = P \cdot Q = 5 \cdot 7 \cdot 31 \cdot 997 = 1\,081\,745$.

Разложим число $A = 1100$, заданное в позиционной системе счисления, в вектор по обычному модулярному базису: $\mathbf{A} = (|1100|_5, |1100|_7, |1100|_{31}, |1100|_{997}) = (0, 1, 15, 103)$. Далее разложим число A по рекурсивному базису: $\mathbf{A} = \big(|1100|_5, |1100|_7, \big(||1100|_{31}|_5, ||1100|_{31}|_7\big), \big(||1100|_{997}|_5, ||1100|_{997}|_7, \big(|||1100|_{997}|_{31}|_5, |||1100|_{997}|_{31}|_7\big)\big)\big) = (0, 1, (0, 1), (3, 5, (0, 3)))$.

Количество бит, необходимых для представления числа 1100 в разных модулярных базисах, будет следующим:

Двоичный вид	Модулярный базис	Рекурсивный модулярный базис
21	21	24

Очевидно, что каждый из первых k элементов представляется в виде одного числа. Элемент $k + 1$ содержит k вычетов, поскольку выражается через k остатков системы субмодулей: $a_{k+1} = (a_{k+1,1}, a_{k+1,2}, \ldots, a_{k+1,k})$. Элемент под номером $k + 2$ содержит

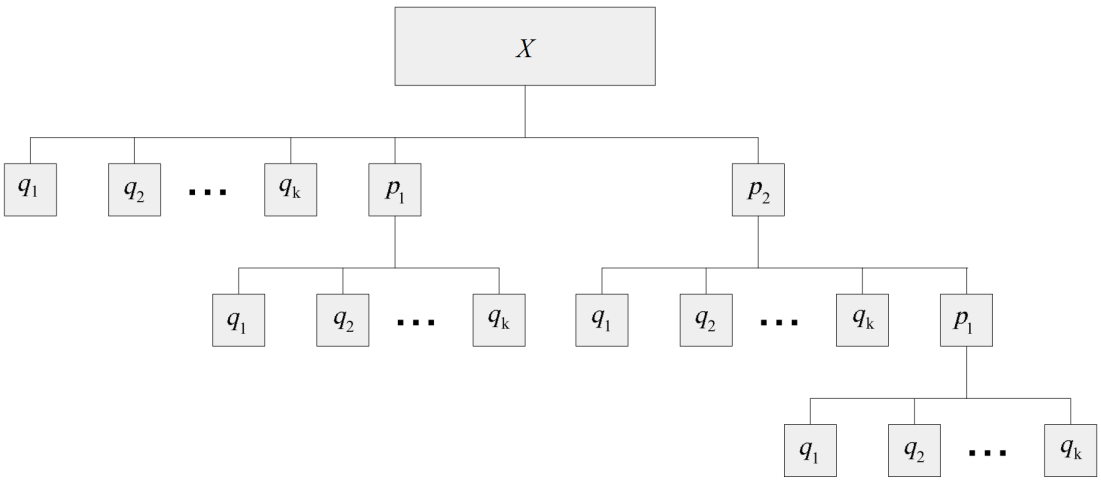

Рис. 1. Представление чисел в рекурсивной системе вычетов для $m = 2$, т. е. при задании двух рабочих модулей

$2k$ вычетов, так как выражается через k остатков системы субмодулей и k остатков вектора a_{k+1}. Таким образом, продолжая рассуждения, можно заключить, что число элементов L_i для вектора a_i выражается формулой

$$L_i = \begin{cases} 1, & i \le k, \\ 2^{i-k-1} \cdot k, & k < i \le n. \end{cases}$$

Общее же число элементов L вектора $\mathbf{A}$ можно рассчитать, воспользовавшись формулой суммы геометрической прогрессии [6]:

$$L = \sum_{i=1}^{i \le n} L_i = k \left(1 + \frac{2^{n-k} - 1}{2 - 1}\right) = 2^{n-k} \cdot k.$$

Количество бит, требуемых для представления числа в РМА, можно рассчитать по формуле

$$B = 2^{n-k} \cdot \sum_{i=1}^{i \le k} \lceil \log_2(q_i - 1) \rceil. \tag{2}$$

При вычислениях в традиционной модулярной арифметике (ТМА) каналом, определяющим быстродействие модулярной структуры, вероятнее всего будет канал по модулю 997, так как производительность зависит от значения модуля. В РМА все вычисления будут проводиться по трёхбитным модулям 5 и 7, что позволит достичь более высоких показателей быстродействия. Кроме того, при использовании рекурсивных структур наблюдается бóльшая степень распараллеливаемости и появляется регулярность.

Однако применение рекурсивной структуры при проектировании модулярных устройств накладывает ряд ограничений, которые приводят к усложнению устройств, выполненных по предлагаемой методологии. Рассмотрим эти ограничения.

Пусть задана система базисных модулей $(p_1, p_2, \ldots, p_m)$ и необходимо представить вычеты по модулю p_{m+1} через вычеты по системе базовых модулей. Очевидно, что максимальный вычет по модулю p_{m+1} $\mathrm{MAX} = p_{m+1} - 1$. Зная это значение и последовательность выполняемых операций, можно рассчитать максимальное значение MAX результата арифметической операции по данному модулю. Очевидно, что для однозначного представления результата арифметических операций необходимо соблюдение неравенства $\mathrm{MAX} < Q$, где $Q = p_1 \cdot p_2 \cdot, \ldots, \cdot p_m$. Для остальных модулей расчёт производится аналогично.

В случае примера с базовыми модулями 2 и 3 $Q = 2 \cdot 3 = 6$. Наименьшее простое число (конечно, после 2 и 3) есть 5 ($\mathrm{MAX} = 4$). Здесь мы не можем выполнить ни операцию сложения, так как $2 \cdot \mathrm{MAX} > Q$ $(8 > 6)$, ни, тем более, операцию умножения, так как $\mathrm{MAX}^2 > Q$ $(16 > 6)$. Чтобы выполнить любую из арифметических операций (сложение или умножение), необходимо увеличить число Q (т. е. значения базисных модулей и/или их количество). Возьмём в качестве базисных модулей все взаимно простые трёхбитные числа 4, 5 и 7. В этом случае $Q = 4 \cdot 5 \cdot 7 = 140$. Чтобы выполнялось неравенство $\mathrm{MAX} < Q$, для операции умножения необходимо выполнение условия $\mathrm{MAX}^2 < Q$, или $p_{i-1} - 1 < \sqrt{Q}$ $(p_{i-1} - 1 < 11.8)$. Таким образом, выбираем ближайшее к 7 простое число $p_i = 11$. Далее совокупность рабочих модулей строится без каких-либо проблем с помощью аналогичного расчёта до тех пор, пока не будет достигнут требуемый вычислительный диапазон.

3. Сравнение ТМА и РМА

В предыдущем примере был рассмотрен РМА-базис (5, 7, (31, 997)), который покрывает динамический диапазон [0; 1081745). Однако сравнение его с модулярным базисом (5, 7, 31, 997) не является корректным, поскольку в традиционной модулярной арифметике базис строился бы исходя из принципа минимизации используемых модулей. Наиболее очевидным подходом в ТМА является стратегия последовательного выбора простых оснований вплоть до покрытия требуемого динамического диапазона. Например, для динамического диапазона 1 081 745 набор модулей в ТМА может быть следующим: (5, 7, 11, 13, 17, 19). Видно, что максимальный рабочий модуль в ТМА равен 19, тогда как в РМА он равен 7.

Базовые модули в РМА для заданного динамического диапазона можно выбирать по двум параметрам: максимальному значению старшего модуля и количеству базовых модулей. Для относительно небольших динамических диапазонов использование РМА может не дать существенного преимущества над ТМА. Например, для динамического диапазона 14 бит и разрядности базовых модулей 4 бита РМА будет иметь вид (7, 11, 13, (17)), тогда как ТМА — (2, 3, 5, 7, 11, 13). В этом случае старший модуль 13 в вычислениях один и тот же, однако накладные расходы на РМА больше, чем на ТМА.

Ниже представлено, при какой разрядности динамического диапазона (в битах) эффективнее применять РМА (расчёт сделан для случая, когда РМА поддерживает операцию умножения):

Максимальная разрядность базовых модулей РМА	3	4	5	6	7	8	9
Разрядность динамического диапазона	8	15	38	77	162	332	703

Видно, что в качестве базовых модулей РМА использовать простые числа большой разрядности целесообразно только в случае очень больших динамических диапазонов, необходимых, например, для многоразрядных вычислений в прикладной и вычислительной теории чисел [7, 8]. В остальных случаях ТМА будет эффективнее.

4. Алгоритм поиска рекурсивного модулярного базиса

При выборе оснований для РМА значение максимального базисного модуля фиксируется, в то время как динамический диапазон покрывается рабочими основаниями, которые не принимают непосредственного участия в выполнении модульных операций. В ТМА максимальное значение модуля увеличивается с ростом динамического диапазона (рис. 2).

Выбор базиса в РМА и ТМА начинается с нахождения динамического диапазона, который определяется решаемой задачей. Для РМА дополнительно необходимо определиться с максимальной разрядностью базовых модулей: чем меньше будет эта разрядность, тем больше бит потребуется для представления данных. Таким образом, возникает вопрос о поиске баланса между необходимым быстродействием и итоговой разрядностью представления.

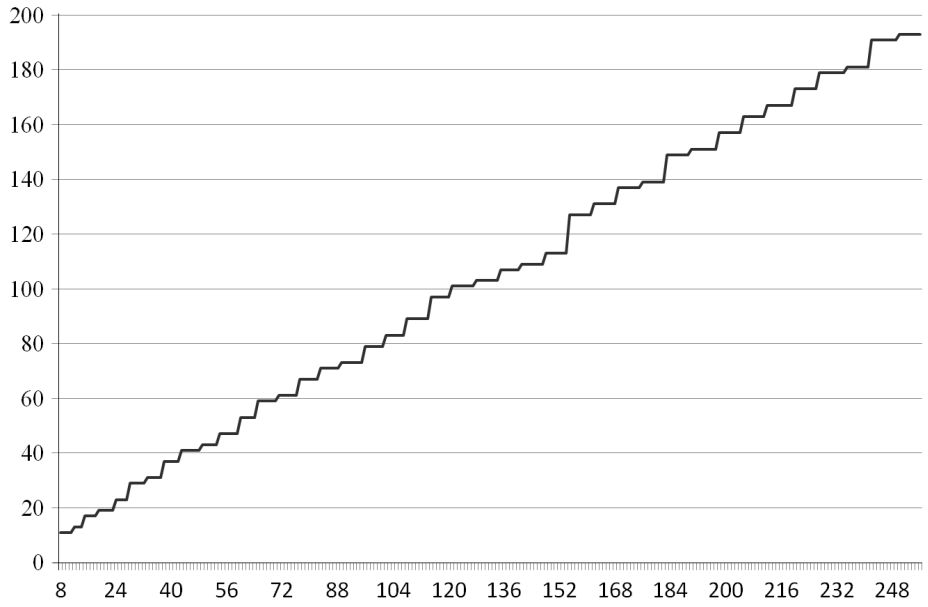

Рис. 2. Зависимость значения максимального модуля TMA от покрываемого динамического диапазона (в битах)

Итак, на вход алгоритма подаются DD — динамический диапазон в битах, $MBit$ — максимальная разрядность базисных рекурсивных модулей и операция (обычно умножение).

Шаг 1. В цикле перебираются количество модулей N рекурсивного базиса от 3 до 10.

Шаг 2. Выбираются максимальные простые числа $(p_1, p_2, \ldots, p_N)$ значением меньше 2^{MBit} в количестве N, если это возможно для заданного N, иначе — возвращаемся к шагу 1.

Шаг 3. Считаем произведение $M = p_1 \cdot p_2 \cdot, \ldots, \cdot p_N$ и далее в зависимости от операции последовательно выбираем рабочие модули. Для операции умножения каждый последующий модуль равен максимальному простому числу, удовлетворяющему условию $p_{new} - 1 < \sqrt{p_1 \cdot p_2 \cdot, \ldots, \cdot p_D}$. При этом, если p_{new} меньше или равно любому из p_i, то построить рекурсивный базис с такой конфигурацией невозможно. Поиск останавливается, когда произведение всех базисных и рабочих модулей покрывает требуемый динамический диапазон.

Шаг 4. Считаем количество бит B, требуемых для представления чисел в найденном базисе, по формуле (2). При $B < B_{\min}$ сохраняем этот набор модулей как оптимальный на данный момент и, если $N \leq 10$, переходим к шагу 1, иначе — возвращаем найденный оптимальный набор модулей и завершаем алгоритм.

5. Экспериментальные результаты

В рамках эксперимента проверялось, насколько эффективны с точки зрения разрядности данных могут быть рекурсивные базисы, полученные с помощью представленного алгоритма. Для этого была написана программа на языке программирования PHP, которая доступна онлайн [9].

На рис. 3, *а* приведено сравнение разрядности полученных базисов для PMA с ограничением по количеству бит на максимальный базисный модуль. Разрядность варьи-

руется от 3 до 7 бит. На этом же рисунке приведен аналогичный график для TMA. Как видно, для представления данных PMA ожидаемо требует больше бит, чем TMA. Для больших динамических диапазонов рост разрядности представления данных PMA с малобитными базисными основаниями становится очень большим.

На рис. 3, *б* для наглядности приведена часть графика, построенная для диапазона от 8 до 64 бит. Видно, что в этом случае оправданно использование даже трёхбитных (простые числа до 8) и четырёхбитных (простые числа до 16) базисов PMA.

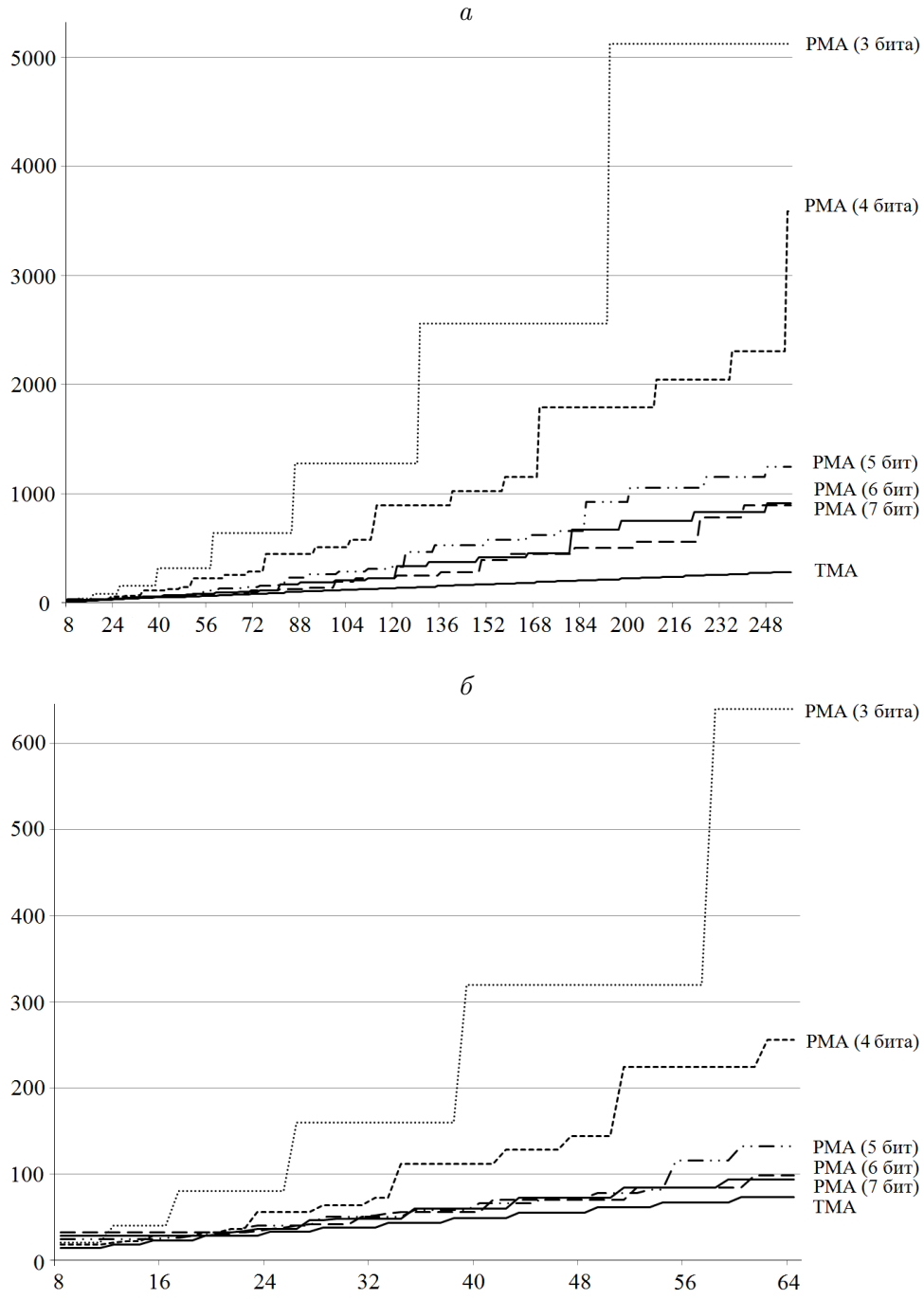

Рис. 3. Зависимость размерности данных PMA от требуемого динамического диапазона 8–256 бит (*а*), 8–64 бит (*б*)

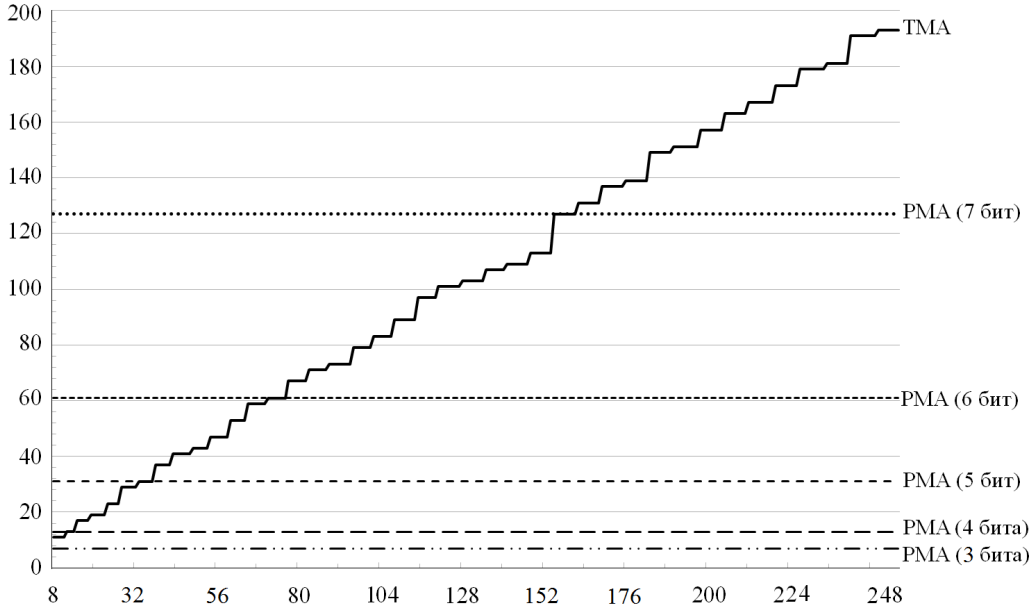

Рис. 4. Значения максимальных модулей из базисных оснований PMA и TMA

Для сравнения эффективности PMA на рис. 4 приведены графики максимального значения базисных модулей в PMA и TMA. Для PMA эти графики горизонтальны, поскольку значения базисных модулей ограничены сверху. При этом отчетливо видно, где именно с точки зрения скорости вычислений в модулярных каналах использование PMA в сравнении с TMA становится более эффективным.

Список литературы

[1] Ленг С. Алгебра. М.: Мир, 1968.

[2] Uspensky J.V., Heaslet M.A. Elementary Number Theory. New York: McGraw-Hill, 1939.

[3] Эрдниева Н.С. Использование системы остаточных классов для маломощных приложений цифровой обработки сигналов // Инженерный вестник Дона. 2013. № 2.

[4] Соловьёв А.Н., Алексеев В.Е. Безгироскопная инерциальная система на основе акселерометров // Нано- и микросистемная техника. 2012. № 4(141). С. 26–31.

[5] Амербаев В.М., Стемпковский А.Л., Соловьёв Р.А. Принципы рекурсивных модулярных вычислений // Информ. технологии. 2013. № 2. С. 22–27.

[6] Abramowitz M., Stegun I.A. Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables, 9th Printing. New York: Dover, 1972.

[7] Инютин С.А. Модулярные вычисления в сверхбольших компьютерных диапазонах // Изв. вузов. Электроника. 2001. № 6. С. 81–87.

[8] Инютин С.А. Вычислительные задачи большой алгоритмической сложности и модулярная арифметика // Вестник Тюменского гос. ун-та. 2002. № 3. С. 14–23.

[9] ИППМ РАН. Генератор рекурсивных базисов [Электронный ресурс]. http://vscripts.ru/2014/recursieve_rns_basis_generator.php

Поступила в редакцию 29 мая 2014 г.