УДК 004.272.2

Модулярная логарифметика – новые возможности для проектирования модулярных вычислителей и преобразователей (краткий обзор)

А.Л. Стемпковский, В.М. Амербаев, А.И. Корнилов

Учреждение Российской академии наук Институт проблем проектирования в микроэлектронике РАН, korn06@alphachip.ru

***Аннотация* — В сообщении привлекается внимание разработчиков к модулярной логарифметике. Указанной арифметике уделяется большое внимание начиная с 90-х годов прошлого столетия. В сообщении отмечаются конструктивные особенности модулярной логарифметики, которые требуют разработки новых более прогрессивных методов проектирования модулярной арифметики.**

***Ключевые слова* — модулярная арифметика, логарифметика поля GF(p), дискретный логарифм, логарифм Якоби.**

I. Введение

Общеизвестно, что десятичная система счисления – величайшее достижение в истории человеческой мысли, существенно упростила счет и вычисления, что послужило революционным толчком для технического прогресса, а после изобретения бесконечных десятичных дробей она приобрела статус универсальной системы счисления архимедовой математики. Двоичная система счисления составила основу современного технического прогресса не только в области проектирования и производства вычислительных средств на современной элементной базе микро- и наноэлектроники, но и в сфере современных информационных технологий. Вместе с тем внутренние проблемы развития технического прогресса стимулируют как рост вычислительной потребности, так и совершенствование вычислений при решении тех или иных специальных задач. Так, после того, как десятичная система счисления была привнесена в 16 веке в Европу, бурное развитие получили астрономия и мореплавание, которые в 17 веке стимулировали рост вычислительной потребности. В результате возник логарифм, как средство, облегчающее трудоемкость операций умножения и деления чисел в десятичной системе счисления. Бурный рост вычислительной потребности во второй половине XX века поставил новые острые проблемы в области вычислительных технологий и технологий проектирования. Это:

- ускорение вычислений (методы распараллеливания как на уровне алгоритмов и программ, так и на уровне машинных кодов);
- отказоустойчивость арифметических вычислений (разработка систем арифметичной самокоррекции, т.е. систем обнаружения и исправления ошибок в режиме компьютинга – непрерывного вычислительного процесса);
- разработка реконфигурируемых вычислительных структур;
- разработка систем автономного компьютинга, т.е. вычислительных систем длительного автономного существования;
- разработка потоковых вычислений;
- разработка арифметико-логических средств повышения «процента выхода годных кристаллов» на этапе производства изделий на кристалле;
- криптографическая стойкость специализированных вычислительных систем открытого доступа и т.п.

Все эти проблемы возникают при решении задач проектирования широкого спектра цифровых устройств специального назначения. Однако межразрядные связи позиционной арифметики делают практически невозможным эффективное решение их на уровне машинных кодов. Что же касается надежности вычислений, то здесь остается, по существу, единственный путь – кратное резервирование аппаратуры.

В связи с этим актуальна задача – разработать такую систему компьютерного счета, в рамках которой открывались бы пути практически приемлемых решений перечисленных выше проблем. Одним из претендентов на такого рода систему компьютерного счета является модулярная арифметика, возникшая более полувека назад. Ей присущи такие свойства, как параллелизм и арифметичная самокоррекция машинных кодов. Однако, такие важные операции машинной арифметики, как деление, формирование признака переполнения, округление, перевод из одной системы счисления в другую, а также алгоритмы декодирования системы самокоррекции являются, существенно, последовательно - параллельными. Перечисленные

операции принято называть немодульными. Следовательно, класс вычислительных задач, где могут быть достигнуты успехи средствами модулярной арифметики, оконтуривается двумя требованиями к вычислительным процессам: низким процентным составом немодульных операций и/или допускающими сокращение немодульных операций посредством технически (экономически) допустимого увеличения вычислительного диапазона цифрового устройства.

Таким образом, модулярная система счисления не может претендовать на статус универсальной системы счисления. Однако, ее использование при решении многокритериальных математических и технических задач проектирования вычислительных средств, как показывает опыт большого числа разработчиков [1], может содействовать повышению эффективности принятия проектных решений. Популярность модулярной арифметики в среде разработчиков специальных вычислительных средств велика – об этом говорит библиография работ, приведенная в сборнике научных трудов юбилейной международной научно-технической конференции «50 лет модулярной арифметике», проведенной в 2005 году в рамках V международной научно-технической конференции «Электроника и информатика 2005», МИЭТ, Москва [2]. Библиография содержит более 1500 работ. Спрос на модулярную арифметику также зависит от того, в какой степени и насколько удачно она может быть адаптирована к современным технологиям проектирования и производства вычислительных средств. Центральной задачей этой проблемы (назовем ее коротко - проблема адаптации) является задача сокращения «накладных расходов» на реализацию модульных операций. Эти расходы обусловлены тем, что арифметическая операция * (т.е. +, -, ×) над остатками *x, y mod p*, как над целыми числами, может приводить к выходу результата операции *x*y* за диапазон *Zp* и тогда требуется корректировка результата, т.е. взятие от числа *x*y* вычета $|x*y|_p$ по *mod p*. Операция взятия вычета $|x*y|_p$ выражается формулой:

$$|x * y|_p = x * y - \left\lfloor \frac{x*y}{p} \right\rfloor \cdot p \qquad (1)$$

Эта формула непосредственно связана с аксиомой Архимеда теории действительных чисел, которая предопределяет, так сказать аддитивный характер вычислений с остатками по *mod p*.

Технически, реализация операции по этой схеме требует:

а) реализации арифметической операции * над целыми числами *x* и *y*;

б) выделения целой части $\left\lfloor \frac{x*y}{p} \right\rfloor$;

в) умножения $\left\lfloor \frac{x*y}{p} \right\rfloor \cdot p$;

г) итогового вычитания, что, собственно, ввиду несоизмеримости модуля *p* со степенью двойки и приводит к дополнительным расходам.

II. Традиционные пути решения проблемы адаптации. Метод локальной близости оснований к степени двойки

Наиболее распространенным методом адаптации является выбор оснований модулярной арифметики максимально близкими к степеням двойки. Например, в работах [3], [4], [5] детально изучена система оснований вида $p_1=2^t-1$, $p_2=2^t$, $p_3=2^t+1$. В работах [6], [7] предлагается рассматривать систему оснований вида: $p_1=2^{n1}$, $p_2=2^{n2}-1$, $p_3=2^{n3}-1,\ldots$, $p_m=2^{nm}-1$, при этом для обеспечения попарной взаимной простоты модулей целесообразно руководствоваться правилом: числа 2^a-1 и 2^b-1 взаимно-простые тогда и только тогда, когда *a* и *b* взаимно просты.

Достоинство выбора модулей в виде $p=2^t\pm1$ состоит в том, что все модульные операции по этим модулям над остатками, представленными двоичным позиционным кодом, могут быть получены без формирования величины $\left\lfloor \frac{x}{p} \right\rfloor$, как этого требует формула типа (1). Этот факт следует из тождества:

$$|x|_{2^t\pm1} = \left| |x|_{2^t} \mp \left\lfloor \frac{x}{2^t} \right\rfloor \right|_{2^t-1} \qquad (2)$$

(Здесь, как и выше, символом $|x|_p$ обозначен вычет целого числа *x* по *mod p*). Как показывает формула (2) шаги б), в), г) вычислений по формуле (1) отсутствуют при вычислениях по формуле (2). По этой причине описанный метод естественно называть методом локальной адаптации арифметики вычетов к бивалентным технологиям.

Обобщением формулы (2) на случай модулей вида $p=2^t\pm k$ служит формула:

$$|x|_p = \left| |x|_{2^t} \mp k \cdot \left\lfloor \frac{x}{2^t} \right\rfloor \right|_p, \qquad (3)$$

которая при специальном выборе параметра *k* (например, $k = 2^s \mp 1,\ s < t$) также может обеспечить эффект локальной адаптации. Этот подход приобретает особое значение в тех случаях, когда простое число *p* – велико. Однако приведенные методы, вообще говоря, не устраняют возникновение «накладных расходов», а только лишь смягчают их, ибо для вычисления $|x*y|_p$ необходимо формировать, согласно формуле (1), величину $\left\lfloor \frac{x*y}{2^t} \right\rfloor$, представляющую собой аддитивную избыточность операции * по *mod* 2^t.

III. Исследования в области логарифметики вещественных чисел

Логарифметика поля вещественных чисел *R* есть арифметика, в которой каждое число представлено его логарифмом (по некоторому фиксированному основанию), а ее операции изоморфны операциям поля *R*.

Цель перехода к логарифметике: облегчить реализацию мультипликативных операций (умножение и деление) поля *R*.

Интерес к компьютерной логарифметике возник в конце XX и в начале XXI веков [8], [9]. Однако, если при переходе к логарифмам упрощаются мультипликтивные операции, то несколько сложнее реализуются аддитивные операции. Выигрыш в производительности ожидается лишь при решении специальных задач, в которых число аддитивных операций соизмеримо с числом мультипликативных.

При переходе к логарифметике поля *R* возникают две трудности (трудности трансляции). Первая трудность состоит в том, что логарифмическая функция определена на пложительных числах: $y = \lg_a x$ ($x > 0$). Распространение традиционных логарифмов на отрицательные числа осуществляется методом симметризации области определения.

В результате центральной симметризации возникает функция:

$$y = \operatorname{sgn} x \lg_a |x| \qquad (1)$$

График этой функции приведен на рис. 1. Область ее определения задается условием $|x| \neq 0$. Однако, она не является биективной.

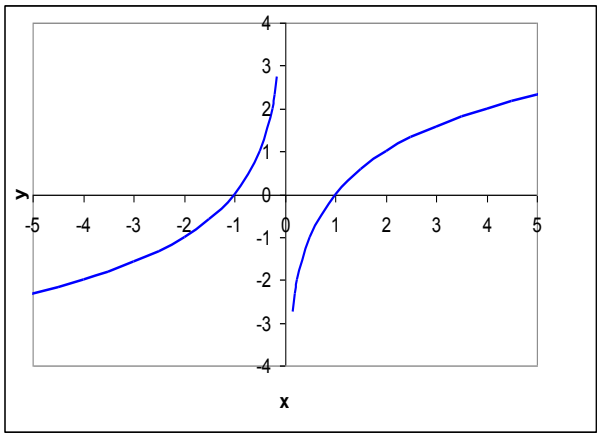


Рис. 1. График функции $y = \operatorname{sgn} x \lg_a |x|$

График ее показывает, что функция (1) имеет две «ветви» взаимной однозначности:

$$y = \begin{cases} \operatorname{sgn} \lg_a |x|, & |x| > 1; \\ 0, & |x| \le 1. \end{cases} \qquad (2)$$

и

$$y = \begin{cases} 0, & |x| > 1; \\ \operatorname{sgn} \lg_a |x|, & 0 < |x| \le 1. \end{cases} \qquad (3)$$

Функция (2) задает, так называемый, «расширенный логарифм», определенный на *R*, для которого отрезок *[-1, +1]* представляет «зону нуля», где нарушается условие строгой монотонности. Соответственно, «обратная» функция имеет вид: $y = \operatorname{sgn} x a^{|x|}$, график которой представлен на рис. 2.

Здесь точка $x = 0$ является точкой разрыва непрерывности.

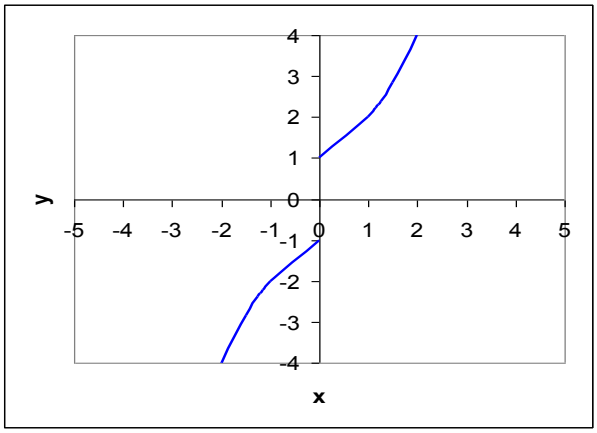


Рис. 2. График функции $y = \operatorname{sgn} x a^{|x|}$

В работе [8] описана логарифметика поля $\mathbb{R}$, построенная на понятии расширенного логарифма (2).

Основное препятствие, которое возникает на пути реализации поля $\mathbb{R}$ это сложность вычисления, так называемого, логарифма Якоби [10], вычисление которого является неотъемлемой частью реализации аддитивной операции в логарифметике. Другое препятствие – это сложность реализации операции округления данных в режиме фиксированной запятой. В этой связи уместно заметить, что, так называемый, режим плавающей запятой также базируется на логарифмическом представлении чисел, а переход к “мантиссам” чисел является оптимальным решением вычислительных проблем логарифметики поля $\mathbb{R}$.

IV. Модулярная логарифметика – преимущества в сравнении с традиционной (позиционной) логарифметикой

Пути преодоления отмеченных выше недостатков логарифметики поля $\mathbb{R}$ предоставляет модулярная арифметика, получившая широкое распространение у специалистов в области компьютерной арифметики и цифровой обработки сигналов [1], [2].

Для простоты изложения будем считать, что базисные основания $p_1, p_2, \ldots, p_n$ модулярной арифметики являются простыми числами [10].

Впервые модулярную логарифметику в рассмотренном варианте рассмотрел Д. А. Поспелов [12]. Здесь для упрощения мультипликативных операций модулярной арифметики используется традиционный подход – переход от вычетов к логарифмам (индексам). Однако ему присуща разбалансировка между затратами на аддитивные и мультипликативные операции. А именно, мультипликативная операция помимо аддитивной операции по *mod (p-1)* требует двух операций преобразования: а) от вычета (остатка) к индексу (логарифмирование), б) от индекса к вычету (антилогарифмирование).

Чтобы сбалансировать модульные операции Д. А. Поспелов предложил каждый вычет $|x|_{p_i}$ представлять парой $<|x|_{p_i}, \lg|x|_{p_i}>$, т.е. точкой на логарифмике - графике функции $y = \left|\lg|x|_p\right|_{p-1}$ (сингулярное значение логарифма при $|x|_p = 0$ распознается по первой компоненте пары). В таком случае все модульные операции как подчеркивает Д.А. Поспелов, приобретают однотипность. А именно, пусть $\alpha_x = <|x|_p, \lg|x|_p>$, $\alpha_y = <|y|_p, \lg|y|_p>$, тогда аддитивные и мультипликативные операции реализуются по однотипным схемам:

1) аддитивная операция:

$$z := \left||x|_p \pm |y|_p\right|_p \xrightarrow{\lg} \lg z \rightarrow < z, \lg z >$$

2) мультипликативная операция:

$$z := \left|\lg|x|_p + \lg|y|_p\right|_{p-1} \xrightarrow{\lg^{-1}} < \lg^{-1}(z), z >$$

(Здесь операции lg и $\lg^{-1}$ суть логарифмирование и потенцирование).

Источником накладных расходов такого подхода служит удвоение регистров операндов. Адаптацию к бивалентным технологиям здесь можно усмотреть по двум обстоятельствам: все модульные операции сведены к сумматорам двух типов - по *mod* p_i, *mod*$(p_i$*-1)*, при этом модули (p_i*-1)* разлагаются на более мелкие множители, благодаря чему достигается большая технологичная близость к степени двойки в сравнении с модулем p_i. В криптографии простые числа p такие, для которых (p*-1)* разлагаются на более, чем два простых сомножителя, называются гладкими. Естественно, число простых сомножителей числа (p*-1)* назвать степенью гладкости простого числа p. В класс арифметических задач модулярной арифметики, чем выше гладкость ее простых базисных модулей p_i, тем выше их технологичность, т.е. тем более модулярная арифметика оказывается адаптированной к бивалентным технологиям [13].

Приведенные выше примеры показывают, что «накладные расходы» на модульные операции обусловлены аддитивной природой целых чисел, источником которой служит аксиома Архимеда. Она же предопределяет аддитивный характер всех арифметических вычислений. В частности, в рамках модулярной арифметики все немодульные операции существенно обусловлены аддитивной природой чисел.

В свете сказанного можно утверждать, что позиционная система счисления (в отличие от непозиционной) максимально адаптирована к аддитивной природе целых чисел или, точнее, адекватна их аддитивной природе.

Для совершенствования адаптации модульных операций модулярной арифметики к бивалентным технологиям имеется еще один путь – отказаться на уровне модульных операций от аддитивных вычислений и перейти к мультипликативным (т.е. логарифметическим).

В строгом смысле под термином модулярная логарифметика кольца $\mathbb{Z}_{p_1 p_2 \ldots p_n}$ понимается арифметика кольца, порождаемая прямым произведением логарифметик полей Галуа GF(p_i) ($1 \le i \le n$) [9].

Логарифметика поля GF(p) разработана выдающимися математиками К. Ф. Гауссом, К. Г. Я. Якоби и их последователями в первой половине XIX [10], [14].

Эта арифметика является расширением индексной арифметики поля GF(p) на сингулярную точку (т.е. нулевую точку поля GF(p)). Операции логарифметики конструируются так, чтобы логарифметика поля GF(p) была изоморфна арифметике конечного поля GF(p).

Рассмотрим дискретный логарифм над полем GF(p), определенный по формуле:

$$\lg_w |x|_p = (2^t - 1)\delta(|x|_p) + ind_w |x|_p \, \hat{\delta}(|x|_p). \quad (5)$$

Здесь:

- $\delta(|x|_p) = \begin{cases} 1, |x|_p = 0, \\ 0, |x|_p \neq 0; \end{cases}$
- $t = [lg_2(p)]$ - битность модуля p, где $\lceil \alpha \rceil$ - наименьшее целое число, превосходящее α;
- $\hat{\delta}(|x|_p)$ - кофункция Кронекера, т.е. $\hat{\delta}(|x|_p) = 1 - \delta(|x|_p)$;

- w – примитивный элемент поля GF(p);

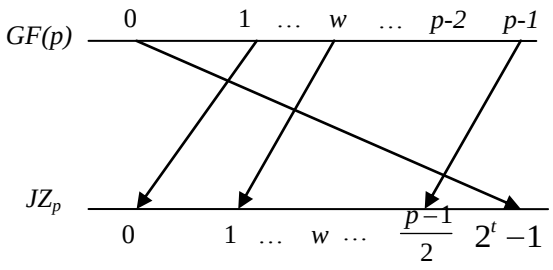
- $ind_w|x|_p$ - индекс вычета $|x|_p$ по основанию w, т.е.

$$|x|_p \neq 0 \Leftrightarrow \left|w^{ind_w|x|_p}\right|_p = |x|_p$$

- $|x|_p = 0$ - точка сингулярности логарифма.

Замечание. Выбор показателя t символа сингулярности $2^t - 1$ экономичен тем, что он не увеличивает числа бит для своего представления в сравнении с битностью двоичного представления элементов поля GF(p) и имеет стандартную форму для любого простого числа p.

Согласно определению, расширенный логарифм $y = \lg_w |x|_p$ взаимнооднозначно отображает поле *GF(p)* на множество $JZ_p = \{0,1,2,...,p-2,2^t-1\}$. Характерными точками этого отображения являются точки *0, 1, w, p-1* $\in GF(p)$,которые при любом простом $p>2$ и любом w отображаются соответственно в точки $2^t - 1, 0, 1, \frac{p-1}{2} \in JZ_p$:



В силу биективности функция $y = \lg_w |x|_p$ порождает на JZ_p структуру поля, изоморфную полю *GF(p)*.

Обозначим символами $\boxplus$ $\boxtimes$, соответственно, аддитивную и мультипликативную операции поля JZ_p.

Согласно определению, если $\alpha = \lg_w |a|_p$, $\beta = \lg |b|_p$, то

$$\alpha \boxplus \beta = \lg_w |a \cdot b|_p ,$$

$$\alpha \boxtimes \beta = \lg_w |a + b|_p$$

Процедура изоморфного конструирования операции логарифметики поля GF(p) описана в трудах [15], [16], [17]. Существенно, что они включают в себя процедуры взаимодействия с сингулярными значениями расширенного логарифма, в отличии от индексной арифметики, рассмотренной в [17], где исключены из вычислений взаимодействия с сингулярными точками области определения и сингулярными значениями области значений дискретного логарифма.

Обозначим символом $\mathcal{L}_p$ логарифметику поля GF(p). По построению $\mathcal{L}_p$ является полем изоморфным полю GF(p). Следовательно, модулярная логарифметика $\mathcal{L}_{p_1} \times \mathcal{L}_{p_2} \times ... \times \mathcal{L}_{p_m}$ изоморфна модулярной арифметике кольца $\mathbb{Z}_{p_1} \times \mathbb{Z}_{p_2} \times ... \times \mathbb{Z}_{p_m}$.

Главное достоинство модулярной логарифметики $\mathcal{L}_{p_1} \times \mathcal{L}_{p_2} \times ... \times \mathcal{L}_{p_m}$ в сравнении с традиционной состоит в том, что она позволяет преодолеть барьер пиковой разрядности логарифметики $\mathbb{Z}_{2^n}$, возникающий в связи с возрастанием сложности вычисления логарифмов Гаусса – Якоби с возрастанием n. Модулярная арифметика позволяет преодолеть этот барьер при больших n за счет организации параллельных логвычислений в малых диапазонах чисел по малым диапазонам модулей кольца $\mathcal{L}_{p_1} \times \mathcal{L}_{p_2} \times ... \times \mathcal{L}_{p_m}$.

При этом выполняется условие:

$$p_1\, p_2 \ldots p_m > 2^n$$

Другие достоинства модулярной логарифметики слогаются из того, что она открывает новые технологичные возможности для совершенствования всех процедур модулярной арифметики. Рассмотрим (бегло) некоторые из них.

1. Технически все модульные операции по каждому модулю p логарифметики имеют одинаковую структуру, т.е. состоят из трех цифровых блоков:
   - простых булевых схем - идентификации сингулярности (так называемые, предикаторы сингулярности);
   - сумматоров по модулю p – 1;
   - одновходовых таблиц Якоби.

   Совокупно эти блоки объединяются в, так называемый, вычислительный элемент (ВЭ). В зависимости от функциональной принадлежности, ВЭ может иметь различную архитектуру.
2. Вычислительный элемент допускает универсализацию относительно семейства однотипных модулей $p_1, p_2, \ldots, p_n$. Семейство модулей $p_1, p_2, \ldots, p_n$ однотипно, если $[p_1 - 1, p_2 - 1, , \ldots, p_n - 1]$ (практически) близко по порядку к максимальному модулю.

   Универсальные вычислительные элементы составляют базу для построения надёжных архитектур модульных вычислений.
3. Универсальный вычислительный элемент, наделенный системой кодовой защищенности, является внутренним самокорректирующим элементом модулярной вычислительной системы. Кроме того сам модулярный код является самокорректирующимся. Таким образом, кодовая защищенность вычислителя модулярной логарифметики представляет собой прямое произведение двух типов кодовой защищенности: внутренней и внешней. Это открывает перспективы для разработки высоконадежных вычислительных структур на основе кодовой защищенности, с одной стороны, и для разработки теоретических исследований, связан-

ных с распространением идей и методов теории К. Шеннона достоверной передачи информации по каналам связи с шумом на надежность вычислительных каналов с шумом. Точнее, расширить классические исследования Винограда С. и Коэна Дж. Д., «Надежные вычисления при наличии шумов», М., Наука, 1968, 112 с., на случай шумящих каналов арифметической обработки данных.

4. Оптимальный выбор технологичных модулей и разработка библиотек функциональных блоков модулярной логарифметики для САПР Synopsys. Эта задача корректно ставится лишь в рамках модулярной логарифметики.
5. Разработка принципов конвейерной реализации всех немодульных операций модулярной логарифметики.
6. Выбор технологичных модулей p вида $4k+1$ (в силу теоремы Гаусса об изоморфизме) позволяет все достоинства модулярной логарифметики вещественно-значных величин распространить на модулярную логарифметику комплексно-значных величин. Таким образом, достигается расширение среды параллельных логвычислений над вещественными величинами на компьютерную арифметику логвычислений над комплексно-значными величинами.
7. Модулярная логарифметика впервые позволяет ставить задачу об эффективной адаптации модулярной арифметики к бивалентным технологиям проектирования методом варьирования выбора “гладких” технологичных модулей.
8. Разработка реконфигурируемых арифметических структур конвейерного типа и параллельного действия.
9. Разработка устройств быстрых линейных преобразований, а также устройств быстрых и надежных операций матричной алгебры. [18]

V. Заключение

Модулярная логарифметика не снимает трудностей аппаратной и временной реализации таких немодульных операций как округление, формирование знака числа, перевода двоичных кодов в модулярный и обратно, и других. Однако, позволяет глубже взглянуть на особенности надежных модулярных вычислений при наличии шумов в модульных каналах; позволяет строить оптимальные в смысле минимизации бивалентного дефекта модульные структуры вычислителей различного рода, используемых в системах цифровой обработки сигналов; позволяет эффективнее использовать идеи параллеллизма и организации вычислений, но уже на уровне машинных кодовых слов, заложенных в трудах отечественных ученых [19] – [22].

VI. Литература

[1] Soderstrand M.A., Jenkins W.K., Jullien G.A., and Taylor F.J. Residue Number System Arithmetic: Modern Applications in Digital Signal Processing // IEEE Press, 1986.

[2] Юбилейная Международная научно-техническая конференция «50 лет модулярной арифметике»: Сб. научных трудов. М.: ОАО «Ангстрем», МИЭТ, 2006. 775 с.

[3] Корнилов А.И., Семенов М.Ю., Ласточкин О.В. // Принципы построения модулярных индексных умножителей. – Известия ВУЗов. Электроника. 2004. №2.

[4] Амербаев В.М., Стемпковский А.Л., Широ Г.Э. Модулярный быстродействующий согласованный фильтр // «50 лет модулярной арифметике»: Сб. научных трудов. М.: ОАО «Ангстрем», МИЭТ, 2006. С. 250 – 267.

[5] Корнилов А.И., Семенов М.Ю., Ласточкин О.В., Калашников В.С. Применение современных методов проектирования при реализации модулярных вычислительных процедур // «50 лет модулярной арифметике»: Сб. научных трудов. М.: ОАО «Ангстрем», МИЭТ, 2006. С. 369 – 383.

[6] Parhomi B., Computer arithmetic: algorithm and Hardware designs // Oxford University Press, 2000. № 4.

[7] Koren J., Computer Arithmetic Algorithms – Massachusetts, 2002.

[8] Шауман А.М., Основы машинной арифметики. – Ленинград: изд. ЛГУ, 1979.

[9] Arnold M.G., Residue Logarithmic number System, Theory and Implementation // Computer Arithmetic, 27-29 June 2005. P. 196-205.

[10] Лидл Р., Нидеррайтер Г. Конечные поля: в 2 т. / под общ. ред. В.И. Нечаева. М.: Мир, 1988.

[11] Виноградов И.М. Основы теории чисел. М.: Наука, 1972.

[12] Поспелов Д.А. Арифметические основы вычислительных машин дискретного действия. М.: Высш. шк., 1970.

[13] Амербаев В.М., Константинов А.В., Тельпухов Д.В. Бивалентный дефект модулярных кодов // Проблемы разработки перспективных микро- и наноэлектронных систем – 2008. Сб. научных трудов / под общ. ред. А.Л. Стемпковского. М.: ИППМ РАН, 2008. С. 462- 466.

[14] Математический Энциклопедический словарь. М.: Сов. энциклопедия, 1988. С. 141, 330.

[15] Zelniker G., Taylor F.J., A Reduced Complexity Finite Field ALU // JEEE Truns. Sirc. Syst. Dec. 1991. V. 38.

[16] Williams T.A. Circuit for Adding and/or Substracting Representation – U.S. Patent, № 4, 727, 508. Feb.23 1988.

[17] Preethy A.P., Padhakrishnan D. An RNS based logarithmic adder // JEEE Proceeding, Computer and Digital Techniques. July, 2000. V. 147, Issue 4. P. 283–296.

[18] Виноград С., Коуэн Дж. Д. Надежные вычисления при наличии шумов. М.: Наука, 1968. 112 с.

[19] Воеводин В.В. Вычислительная математика и структура алгоритмов. М.: Изд. МГУ, 2006.

[20] Современные проблемы вычислительной математики и математического моделирования: в 2 т. / Т. 1: Вычислительная математика, Т. 2: Математическое моделирование. М.: Наука, 2005.

[21] Бурцев В.С. Параллелизм вычислительных процессов и развитие архитектуры Супер ЭВМ. М.: Торус Пресс, 2006. 416 с.

[22] А.Л.Глебов, Гурарий М.М., Егоров Ю.Б., Жаров М.М., Русаков С.Г., Ульянов С.Л., Стемпковский А.Л. Актуальные проблемы моделирования в системах автоматизации схемотехнического проектирования // под. общ. ред. Стемпковского А.Л. М.: Наука, 2003. 430 с.