УДК 004.272.2

# Анализ и синтез арифметического узла проф. Поспелова Д.А. поля Галуа

В.М. Амербаев Е.С. Балака Р.А. Соловьев Д.В. Тельпухов

Институт проблем проектирования в микроэлектроник РАН, Ssapra@hotbox.ru

***Аннотация* — В статье рассматривается кодовая конструкция модульной арифметики, основанная на парном представлении операндов, впервые рассмотренная проф. Д.А. Поспеловым. Предлагается модифицировать данную конструкцию путем введения кодовой однотипности представления компанент пары операнда с целью улучшения архитектур устройств, построенных на ее основе, в частности, рассматривается арифметический узел по модулю (АУз). Для оценки эффективности разработанных способов построения модульных устройств проведен сравнительный анализ АУ с модульными аналогами.**

***Ключевые слова* — модулярная арифметика, арифметическое устройство по модулю.**

## I. Введение

Внимание специалистов в настоящее время все больше сосредотачивается на вопросах практической реализации достоинств модулярной арифметики и эффективного применения устройств на ее основе в задачах специального назначения. Основными проблемами, с которыми сталкиваются разработчики, являются большие накладные расходы на реализацию как модульных, так и немодульных операций модулярной арифметики по сравнению с аналогичными устройствами, построенными в позиционной арифметике. С целью решения сложившихся проблем разрабатываются специальные алгоритмы построения арифметических узлов модульной арифметики, учитывающие специфику выбранного модуля. Например, за счет использования специальных наборов оснований модулярной системы типа $2^t$-1, $2^t$, $2^t$+1, наиболее адаптированных к двоичным технологиям, появляется возможность проектировать эффективные прямые и обратные преобразователи из позиционной арифметики в модулярную и наоборот, не вносящие дополнительных задержек в работу схем [1]. Использование табличной арифметики на некотором наборе модулей позволяет реализовывать устройства с повышенным быстродейсвием [2].

В целях упрощения алгоритмов модулярной арифметики и их эффективной аппаратной реализации как с точки зрения площади занимаемой аппаратуры, так и с точки зрения быстродействия, исследователями предлагаются различные модулярные кодовые конструкции, адаптированные под решение определенного круга задач. Одно из предложений было высказано проф. Д.А. Поспеловым в его книге [3]. Однако с точки зрения аппаратного проектирования дальнейшего развития оно не получило.

## II. Кодовая конструкция проф. Поспелова Д.А.

Аддитивный характер вычислений в кольце вычетов $Z_p$ порождает дополнительные расходы на выполнение арифметических операций. Это обусловлено тем, что результат выполненной операции может выйти за диапазон $Z_p$, тогда требуется корректировка результата, т.е. взятие результата выполненной операции по модулю. Мультипликативная операция над остатками x, y mod p более трудоемка, поэтому наиболее эффективным способом избежать прямой реализации мультипликативной операции является переход к индексам вычетов по основанию первообразного корня, однозначно связанных с данным модулярным кодом. Описание индексной арифметики, а также проектные решения построения арифметических узлов на ее основе, в полной мере рассмотрены в работе [4]. В случае индексной арифметики операция «+» выполняется за один такт модульного суммирования, а операция «*» за такт модульного суммирования и *два* такта табличной операции. Для того, чтобы сбалансировать выполнение модульных операций, д.А. Поспелов ввел представление исходных операндов в виде пар $\left\langle |x|_p, ind_w |x|_p \right\rangle$, где $|x|_p$ есть вычет $x$ по mod $p$, $i = ind_w |x|_p$ - соответсвующий вычету $|x|_p$ индекс, при этом условно считается, что вычету 0 соответствует специальный символ $\lambda$, который обладает свойством $\lambda + i = i + \lambda = \lambda$ для любого индекса $0 \le i \le p$-2. Таким образом, все операции поля выполняются над парами: если требуется найти сумму двух операндов по модулю $p$, то суммируются по модулю $p$ первые компоненты пар; для формирования второй компоненты пары результата этот результат преобразуется в индекс путем выборки значения из таблицы индексов (рис. 1). Если требуется найти произведение двух операндов по модулю $p$, то суммируются по модулю $p$-1 вторые компоненты пар; для формирования первой компоненты пары результата этот результат преобразуется в антилогарифм (вычет) путем выборки значения из таблицы вычетов (рис. 2):

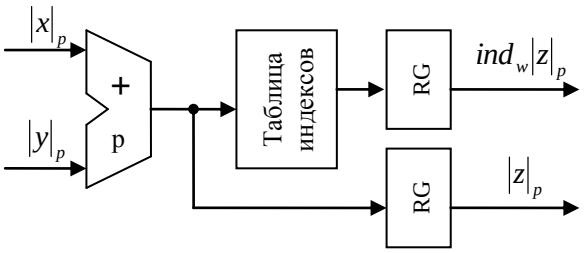


**Рис. 1. Структурная схема операции сложения в кодах Д.А. Поспелова**

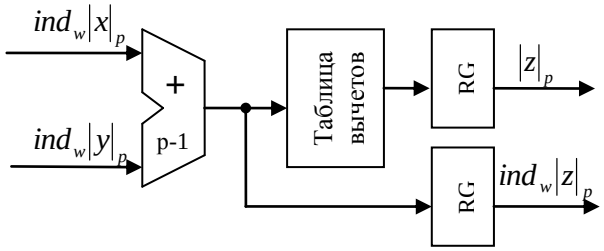


**Рис. 2. Структурная схема операции умножения в кодах Д.А. Поспелова**

Арифметику, построенную на парном представлении операндов, будем называть *бимодульной арифметикой поля GF(p).*

Таким образом, операции сложения и умножения сведены к операциям сложения по модулю *p* и модулю *p-1*, соответственно, и *одной* табличной операции выбора второй компоненты пары результата. Такое решение позволяет сократить время выполнения мультипликативной операции на один такт табличной операции и площадь на хранение двух таблиц преобразования в индексы, размерность каждой таблицы *2(p-1).* При этом Д.А. Поспелов утверждает [3, стр. 296], что несмотря на то, что логика операции умножения по модулю *p* стала более сложной, чем в обычной системе кода в остатках, выигрыш состоит в «однотипности оборудования для производства операций сложения и умножения». Данное утверждение справедливо в общем случае, когда сумматоры по модуля *p* и *(p-1)* проектируются по методу прямой логической реализации с использованием двоичных функциональных блоков. В этом случае суммирование по модулю m для двух операндов x и y, находящихся в диапазоне {0, 1,…, *m*-1}, выполняется по формуле:

$$|x+y|_m = \begin{cases} x+y, \textit{если } (x+y) < m, \\ x+y-m, \textit{если } (x+y) \geq m. \end{cases}$$

Данный метод построения сумматоров по модулю является наиболее распространенным. Во-первых, он позволяет в полной мере использовать современные наработки в области проектирования двоичных устройств. Во-вторых, предоставляет большую гибкость при реализации каждого компонента в составе всего вычислительного блока в зависимости от требований по занимаемой площади и быстродействию. В-третьих, метод универсален, т.е. независим от специфики используемого базового модуля m. Например, для малых значений модулей возможны построения с минимизацией их по площади. Для больших модулей, определяющих быстродействие всей модулярной системы в целом, возможно применение методов логического синтеза быстрых сумматоров на основе BDD-технологий [5].

Однако, как отмечалось во введении, в настоящее время все большую популярность завоевывают гибридные методы построения базовых арифметических узлов модулярной арифметики. Такие методы представляют собой комбинацию методов прямой логической реализации и методов на основе таблиц состояний. Использование гибридных методов обеспечивает компромисс между быстродействием и затратами на занимаемую площадь для некоторых значений модулей. В случае же реализации арифметики в кодах Д.А. Поспелова требуется оценивать проектирование сумматоров не только относительно базового модуля *p*, но и модуля *p-1*, иначе теряется основная идея данного кодирования, а именно однотипность оборудования.

Поэтому нами было предложено модифицировать код Д.А. Поспелова и развить его идею однотипности.

Развитие идеи однотипности состоит в том, чтобы аддитивные и мультипликативные операции модульной арифметики выполнялись не только аппаратно однотипно, но и однотипно в кодовом представлении операндов. Это достигается путем перехода от представления компонент пар операндов по модулям *p* и *p-1* к однородному представлению по модулю *p-1*. Введем понятие модифицированного вычета по модулю:

$$|\tilde{x}|_p = \lambda_p \delta\left((p-1)-|x|_p\right) + \left||x|_p\right|_{p-1} \hat{\delta}\left((p-1)-|x|_p\right), \text{ где}$$

$\delta(u) = \begin{cases} 1, & u = 0, \\ 0, & \textit{иначе}, \end{cases}$ - функция Кронекера,

$\hat{\delta}(u) = 1 - \delta(u)$ - кофункция Кронекера.

Для представления второй компоненты пары операнда будем использовать дискретно-логарифметическое представление. Этот способ представления является эффективным при использовании небольших значений модулей порядка 8 бит, что является достаточным при проектировании большинства вычислительных устройств, предназначенных для решения специальных задач из области применения модулярной арифметики. Тем самым вторая компонента пары операндов будет иметь вид:

$$\log_w|\tilde{x}|_p = \delta\left(|\tilde{x}|_p\right)\lambda_p + \hat{\delta}\left(|\tilde{x}|_p\right) ind_w|x|_p,$$

где $ind_w|\tilde{x}|_p$ - индекс вычета $|\tilde{x}|_p$ по основанию *w*, т.е.

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

В этом случае полагается, что при t-битном простом числе $p$ константный символ $\lambda_p = 2^t - 1$. Очевидно, что при любом простом $p$ $\lambda_p \notin Z_{p-1}$.

Распишем, как в бимодульной арифметике будут выполняться арифметические операции. Реализация мультипликативных операций останется без изменений:

если $< |\tilde{x}|_p, \log_w |\tilde{x}|_p >$, $< |\tilde{y}|_p, \log_w |\tilde{y}|_p >$, то

$$(x \cdot y) \bmod p \longrightarrow$$

$$< \log^{-1}\left( \left| \log_w |\tilde{x}|_p + \log_w |\tilde{y}|_p \right|_{p-1} \right), \left| \log_w |\tilde{x}|_p + \log_w |\tilde{y}|_p \right|_{p-1} >,$$

т.е.

$$(x \cdot y) \bmod p = \begin{cases} \lambda_p, \text{если } \delta\left(\log_w |\tilde{x}|_p - \lambda_p\right) \vee \delta\left(\log_w |\tilde{y}|_p - \lambda_p\right) = 1, \\ \left| \left| \log_w |\tilde{x}|_p + \log_w |\tilde{y}|_p \right|_{p-1} \right|_{p-1}. \end{cases}$$

Логика выполнения аддитивных операций усложнится за счет введения дополнительных логических функций, связанных с переходом к однородному представлению. Аддитивные операции выполняются согласно выражению:

если $< |\tilde{x}|_p, \log_w |\tilde{x}|_p >$, $< |\tilde{y}|_p, \log_w |\tilde{y}|_p >$, то

$$(x + y) \bmod p \longrightarrow < |\tilde{x} + \tilde{y}|_{p-1}, \log_w (|\tilde{x} + \tilde{y}|_{p-1}), \text{т.е.}$$

$$(x + y) \bmod p = \begin{cases} p - 2, \ \textit{если} \ \tilde{x} = \tilde{y} = p - 1, \\ \lambda_p, \ \textit{если} \ \tilde{x} + \tilde{y} = p - 1, \\ \tilde{x} - 1, \textit{если} \ \tilde{x} \neq 0 \ \textit{и} \ \tilde{y} = \lambda_p, \\ \tilde{y} - 1, \textit{если} \ \tilde{y} \neq 0 \ \textit{и} \ \tilde{x} = \lambda_p \\ \tilde{x} + \tilde{y}, \ \textit{если} \ \tilde{x} + \tilde{y} < p - 1, \\ |\tilde{x} + \tilde{y}|_{p-1} - 1, \ \textit{если} \ \tilde{x} + \tilde{y} > p - 1. \end{cases}$$

Тем самым переход к однородному представлению операндов позволил выполнять аддитивные и мультипликативные операции на одном оборудовании, а именно на сумматоре по модулю *p-1*. Это положительно влияет и на требования к затратам по занимаемой площади и на контроль выполнения арифметических операций, при этом сохраняется требование однотипности используемого оборудования.

## III. Арифметическое устройство по модулю p

В работе [6] обозначены новые технологические возможности для совершенствования всех процедур модулярной арифметики, основанные на понятии вычислительного элемента (ВЭ) как минимальной вычислительной единицы, входящей в состав модулярной вычислительной системы. Авторы работы указывают на то, что для выполнения требований оптимальности и эффективности вычислительных архитектур в специализированных приложениях целесообразно использовать аппаратно адаптируемые к задачам вычислительные элементы. По сути, ВЭ является самостоятельным вычислительным устройством, выполняющим прикладные и системные задачи. Основными узлами, входящими в ВЭ, являются устройство управления, оперативно-запоминающее устройство, постоянно-запоминающее устройство, а также арифметический узел по модулю $p$ (АУз). Работа АУз рассматривается в двух режимах: 1) режим определения результата операций модульного сложения и вычитания; 2) режим определения результата операции модульного умножения.

На основе изложенной выше теории разработан вариант построения АУз по модулю $p$ для ВЭ бимодульной арифметики. На рис. 3 представлена его структурная схема.

## IV. Результаты синтеза

Для оценки аппаратных и временных затрат на проектирование арифметического узла по модулю бимодульной арифметики, а также для сравнительного анализа с аналогами, предложенные решения были реализованы в виде RTL-моделей, описанных на языке Verilog. Сравнение характеристик АУз бимодульной арифметики (БМА) проводилось с аналогами, функционирующими в традиционной модульной арифметике (МА), логарифметике (ЛогМА) и бимодульной арифметике по Поспелову Д.А. (тБМА). Структурный синтез проводился в базисе 45 нм библиотеки стандартных ячеек Nangate Open Cell Library с использованием САПР Synopsys Design Compiler. Моделирование и верификация Verilog проектов проводилась средствами ModelSim-Altera Starter Edition. Результаты моделирования схем по задержке на критическом пути представлены в таблице 1.

Таблица 1

*Сравнение АУ по быстродействию*

| | Максимальная задержка блока АУ, нс | | |
|---|---|---|---|
| | 3-4 бит | 5-6 бит | 7-8 бит |
| тБМА | 0,3 | 0,8 | 1,5 |
| МА | 0,4 | 0,9 | 1,6 |
| ЛогМА | 0,3 | 0,7 | 1,4 |
| БМА | 0,3 | 0,7 | 1,4 |

Результаты моделирования схем по занимаемой площади представлены на рис. 4.

Анализ результатов синтеза показал, что бимодульный АУз является более экономичным с точки зрения аппаратных затрат относительно аналогов при базовом модуле размерностью до 7 бит, свыше 7 бит оно проигрывает логарифметическому АУз, но сопоставимо с ним по быстродействию. Также бимодульное АУз выигрывает по показателям занимаемой площади относительно АУз, построенных на традиционной модульной и рассмотренной Д.А. Поспеловым арифметиках.

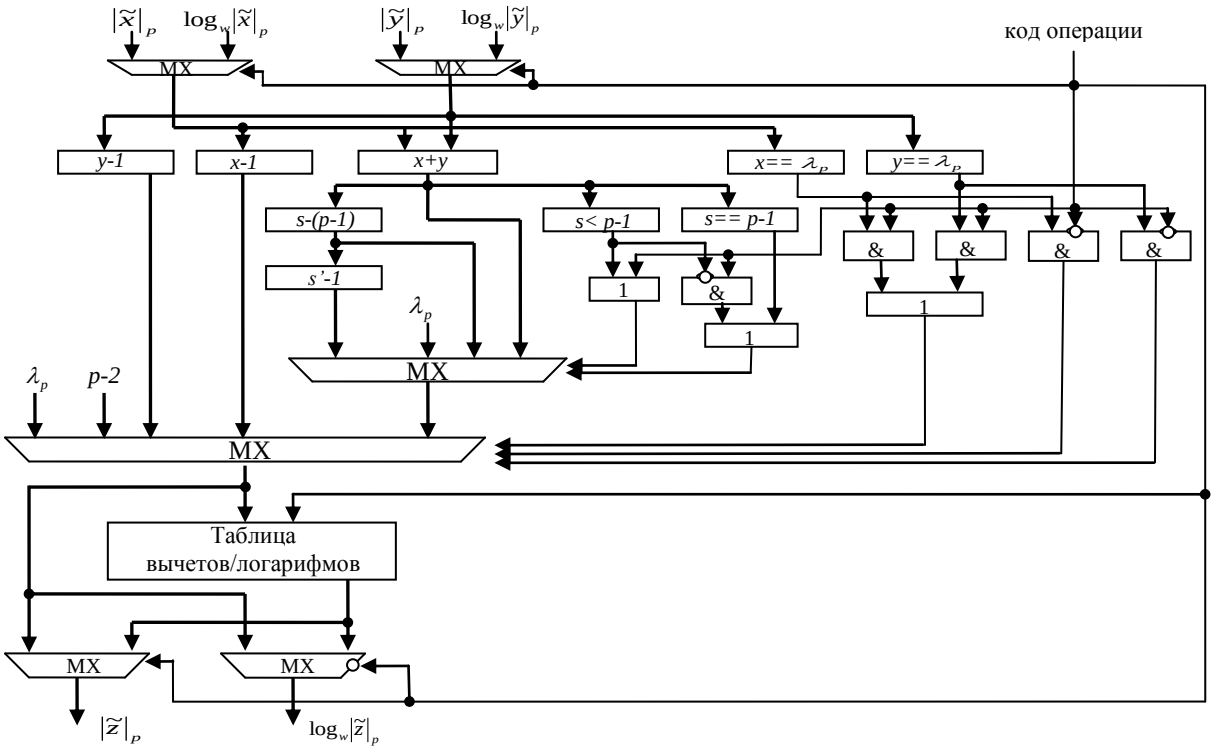


**Рис. 3. Структурная схема бимодульного арифметического устройства по модулю p**

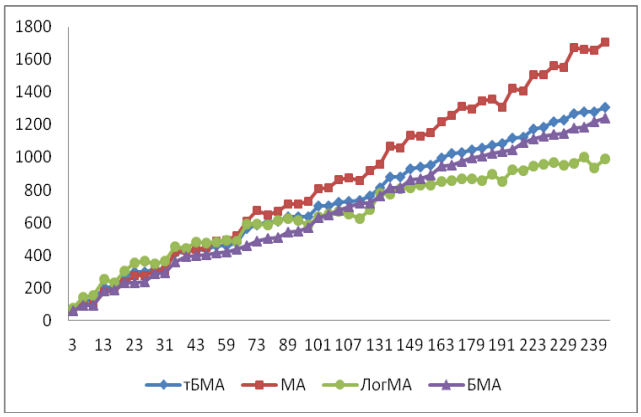


**Рис. 4. Результаты синтеза схем арифметического устройства по модулю по занимаемой площади**

Отметим, что бимодульная модулярная арифметика, за счет избыточности ВЭ по каждому модулю модулярной системы открывает новые технологические возможности в свете построения отказоустойчивых вычислительных устройств. Вопросы повышения помехоустойчивости и отказоустойчивости будут рассмотрены в следующей статье.

## Литература


[1] Leonel Sousa, Efficient Method for Magnitude Comparison in RNS Based on Two Pairs of Conjugate Moduli, // 18th IEEE Symposium on Computer Arithmetic. 2007.

[2] Ирхин В.П. Табличная реализация операций модулярной арифметики // Сб. науч.трудов Юбилейной Международной научно-технической конференции «50 лет модулярной арифметики». 2005. С. 268-273.

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

[4] Амербаев В.М., Малашевич Д.Б. Анализ эффективности реализации модульных операций индексной модулярной арифметики // Известия ВУЗов. Электроника. 2009. С. 54-57.

[5] Корнилов А.И., Исаева Т.Ю., Семенов М.Ю. Методы логического синтеза сумматоров с ускоренным переносом по модулю ($2^n$-1) на основе BDD-технологии // Известия ВУЗов. Электроника. 2004. № 3. С. 54-60.

[6] Стемпковский А.Л., Амербаев В.М., Корнилов А.И. Модулярная логарифметика –новые возможности для проектирования модулярных вычислителей и преобразователей // IV Всероссийская научно-техническая конференция «Проблемы разработки перспективных микро- и наноэлектронных систем – 2010». Сб. трудов. / под общ. ред. А.Л. Стемпковского. М.: ИППМ РАН, 2010. С. 368-373.