

ИНФОРМАЦИОННЫЕ
ТЕХНОЛОГИИ
И ТЕЛЕКОММУНИКАЦИИ

INFORMATION
TECHNOLOGIES
AND TELECOMMUNICATIONS

УДК 004.383.3

DOI: 10.17212/2782-2001-2025-4-85-104

Проектирование цифровых фильтров с масштабированием в системе остаточных классов*

П.А. ЛЯХОВ

РФ, 355017, г. Ставрополь, ул. Пушкина, 1, Северо-Кавказский федеральный университет

ljahov@mail.ru

В статье предложен метод проектирования цифровых фильтров с использованием масштабирования в системе остаточных классов. Рассматриваются архитектуры модульных умножителей для различных типов модулей: степеней двойки, чисел вида $2n - 1$, $2n + 1$, а также произвольных простых чисел. Для каждого случая предложены эффективные схемотехнические решения, включающие генераторы частичных произведений и специализированные схемы суммирования, адаптированные к особенностям соответствующих модулей. Также рассматриваются преобразования чисел в код D-1 и обратно, а также представление отрицательных значений в системе остаточных классов. Спроектирован цифровой фильтр с использованием масштабированных коэффициентов. Масштабирование позволяет уменьшить разрядность вычислений, что приводит к снижению затрат на аппаратную реализацию и энергопотребление. Разработка учитывает специфику модульной арифметики и особенности реализации в логических структурах программируемых логических интегральных схем. Описаны преобразования между кодами и представлены отрицательные значения, обеспечивающие корректную обработку в различных подмодулях. Предложенные архитектуры адаптированы к практическому применению и могут быть масштабированы под требования конкретных аппаратных платформ и вычислительных задач, включая энергозависимые, высокопроизводительные и многоканальные системы обработки данных. Приводятся результаты моделирования работы фильтров с различными наборами модулей на программируемых логических интегральных схемах в среде Xilinx Vivado. Анализ показывает, что уменьшение одного из модулей в 2...4 раза позволяет добиться сокращения аппаратных затрат до 4,3 % и энергопотребления до 3,4 % при приемлемом уровне вычислительной погрешности. Представленные подходы позволяют создавать производительные и экономичные цифровые фильтры для обработки сигналов в системе остаточных классов.

Ключевые слова: система остаточных классов, цифровой фильтр, масштабирование коэффициентов, модульное умножение, архитектура умножителя, программируемая логическая интегральная схема, параллельные вычисления, аппаратная оптимизация, проектирование цифровых фильтров, квантование коэффициентов фильтра

* Статья получена 22 апреля 2025 г.

ВВЕДЕНИЕ

Современные цифровые системы обработки сигналов предъявляют всё более жесткие требования к скорости и энергоэффективности вычислений [1]. Одним из ключевых компонентов таких систем являются цифровые фильтры, используемые для подавления шума, извлечения полезной информации и преобразования спектральных характеристик сигналов [2]. Повышение пропускной способности и снижение задержек при работе фильтров становится особенно актуальным в задачах реального времени (например, в радиосвязи [3], радарных системах [4], медицинской диагностике [5] и обработке видеоизображений [6]). Однако реализация высокоскоростных цифровых фильтров сопряжена с некоторыми проблемами. Прежде всего это высокая вычислительная сложность: при увеличении порядка фильтра растет количество операций умножения и суммирования, что требует значительных аппаратных ресурсов [7]. Способы реализации на основе фиксированной или плавающей запятой могут приводить к росту площади, энергопотребления и задержек. Кроме того, параллельная обработка данных затрудняется из-за наличия переносов при суммировании, что ограничивает потенциальную скорость выполнения операций [8].

Разработка высокоскоростных цифровых фильтров является многогранной задачей, требующей оптимизации на нескольких уровнях: алгоритмическом, архитектурном и схемотехническом [9].

С точки зрения аппаратной оптимизации использование специализированных вычислительных блоков, встроенных в современные программируемые логические интегральные схемы (ПЛИС) и интегральные схемы специального назначения (application-specific integrated circuit, ASIC), позволяет выполнять умножение и сложение с высокой скоростью. Применение конвейеризации [10] и распараллеливания операций ускоряет выполнение фильтрации, особенно в многоступенчатых и каскадных структурах. Также важна оптимизация критического пути, которая позволяет сократить задержки, ограничивающие тактовую частоту [11].

Для снижения вычислительной сложности применяют различные приемы. Одним из них является разложение операций умножения на сдвиги и сложения (мультипликативная декомпозиция), особенно эффективное при работе с коэффициентами, близкими к степеням двойки [12]. Также используется ограничение разрядности коэффициентов (квантизация), что снижает сложность операций при допустимом уровне искажения. Кроме того, в линейных фильтрах с конечной импульсной характеристикой часто используется симметрия коэффициентов, что уменьшает количество умножений [13].

Алгоритмическое масштабирование также вносит вклад в повышение производительности. Масштабирование коэффициентов фильтра позволяет сократить максимальное значение произведений, при этом уменьшается необходимая разрядность и ресурсоемкость арифметики. Помимо этого, масштабирование способствует упрощению модулярной арифметики в системах остаточных классов.

Одним из эффективных решений этих проблем является применение системы остаточных классов (СОК), которая позволяет реализовать параллельные и переносонезависимые вычисления (вычисления, в которых результат одной операции зависит от переноса (переполнения) из предыдущей, как правило, при выполнении арифметических операций над многобитными числами

(например, при сложении или вычитании)) [14]. Вместе с тем использование СОК требует адаптации базовых арифметических операций, в частности умножения, с учетом представления данных в рассматриваемой системе. Также важным направлением оптимизации является масштабирование коэффициентов фильтра, позволяющее уменьшить разрядность вычислений и, как следствие, упростить реализацию фильтра на уровне аппаратуры.

Система остаточных классов – это метод представления чисел, при котором одно большое число заменяется на набор остатков по выбранному набору взаимно простых модулей. Такое представление позволяет разложить вычисления на параллельные потоки меньшей разрядности, что дает ряд существенных преимуществ [15].

Система обеспечивает параллелизм и высокую производительность. Каждая операция выполняется независимо в подканале, соответствующем конкретному модулю. Арифметические операции, такие как умножение и сложение, выполняются параллельно без необходимости координации между каналами. Отсутствие переносов между разрядами упрощает и ускоряет схемы суммирования и умножения. В отличие от позиционной арифметики, где перенос влияет на скорость, в системе остаточных классов эта проблема устраняется, что особенно важно для высокочастотных цифровых систем. Масштабируемость достигается за счет возможности добавления новых модулей без изменения существующей архитектуры. Это удобно при проектировании многоразрядных цифровых фильтров и вычислительных систем с расширяемой разрядностью [16]. Использование избыточных модулей позволяет обнаруживать и корректировать ошибки. Такая устойчивость повышает надежность вычислений в критически важных приложениях, включая радиолокацию, криптографию и медицинские приборы. Модули, работающие с числами меньшей разрядности, требуют меньше аппаратных ресурсов и снижают энергопотребление по сравнению с традиционными арифметическими блоками, оперирующими длинными целыми числами. Некоторые фиксированные модули, такие как 2^n , $2^n - 1$ и $2^n + 1$, позволяют реализовать арифметику с помощью простых логических операций – побитовых и циклических сдвигов, а также сложения без переноса [17]. Это значительно упрощает аппаратную реализацию систем на их основе.

Таким образом, СОК является мощным инструментом при проектировании цифровых фильтров и других компонентов, где критичны скорость, параллелизм и эффективность. Однако ее применение требует специфических методов преобразования и синтеза арифметических операций, что и рассматривается в настоящей статье. Здесь предлагается метод проектирования цифровых фильтров с масштабированием в СОК. Будет продемонстрировано снижение затрат за счет уменьшения одного модуля в 2 и в 4 раза.

1. АРХИТЕКТУРЫ МОДУЛЬНЫХ УМНОЖИТЕЛЕЙ В СОК

Одним из ключевых компонентов цифровых фильтров в СОК являются модульные умножители, выполняющие операции умножения по каждому из модулей системы. Поскольку в СОК вычисления ведутся параллельно и независимо по каждому модулю, для каждого элемента набора модулей требуется своя специализированная схема умножителя. Архитектура таких умножителей

должна учитывать особенности арифметики по модулю, а также стремиться к минимизации аппаратных затрат и задержек. При этом важным фактором является структура самого модуля, поскольку от этого напрямую зависит выбор схемотехнических решений.

Разработка эффективных архитектур модульных умножителей направлена на оптимизацию ключевых операций: генерации частичных произведений, их суммирования и последующего приведения результата по модулю. Для модулей вида 2^n могут использоваться простые логические схемы, тогда как для модулей $2^n - 1$ и $2^n + 1$ необходимо учитывать циклический перенос или использовать кодовые преобразования. При работе с произвольными простыми модулями часто применяются методы кодирования, такие как Diminished-one (D-1) [18], а также схемы с предварительным инвертированием и коррекцией. Таким образом, модульные умножители в СОК представляют собой разнообразный класс архитектур, адаптированных под конкретную арифметику и ориентированных на высокую производительность и ресурсоэффективность.

1.1. УМНОЖИТЕЛЬ ПО МОДУЛЮ 2^n

Для суммирования частичных произведений в модульных умножителях широко применяются схемы сжатия с переносом CSA (Carry Save Adder, сумматор с сохранением переноса) и CPA (Carry Propagate Adder, сумматор с последовательным переносом). Сумматор CSA позволяет эффективно сократить несколько слагаемых до двух без распространения переноса, что ускоряет вычисления, тогда как CPA завершает суммирование с учетом переносов и формирует окончательный результат по модулю. Для выполнения умножения в СОК по модулю 2^n используют специальный генератор частичного произведения, сумматоры CSA и CPA по модулю 2^n (рис. 1, 2). Если число членов в генераторе частичных произведений (partial product generator, PPG) равно трем и более, то для сведения суммы к двум членам используют так называемые деревья CSA, для чего применяют CPA по модулю 2^n . Из рассматриваемого набора СОК соответствующим модулем является число 16 (2^4), для которого PPG состоит из четырех членов (рис. 3), сумматоров CSA и CPA. Архитектура умножителя по модулю 16 показана на рис. 4.

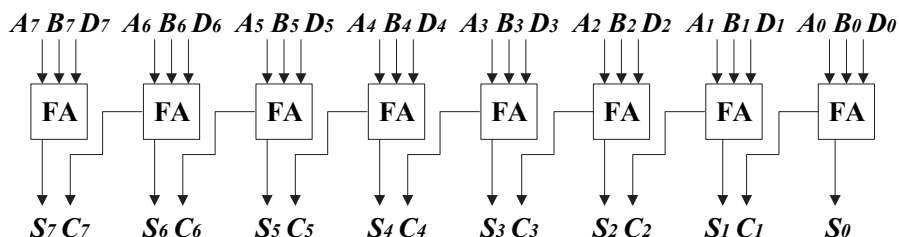


Рис. 1. Схема CSA по модулю 256

Fig. 1. The CSA scheme modulo 256

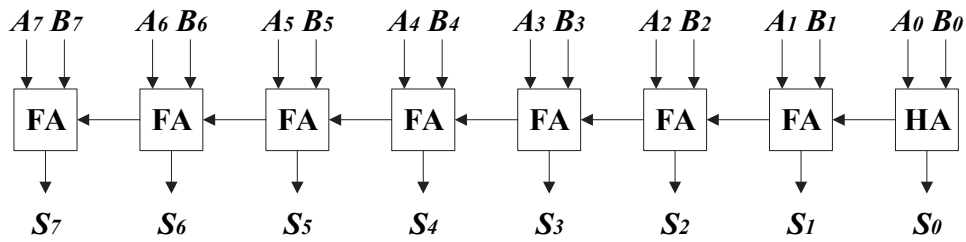


Рис. 2. Схема CPA по модулю 256

Fig. 2. The CPA scheme modulo 256

PPG mod 2^4				
3	2	1	0	n
x_3y_0	x_2y_0	x_1y_0	x_0y_0	pp_1
x_2y_1	x_1y_1	x_0y_1	0	pp_2
x_1y_2	x_0y_2	0	0	pp_3
x_0y_3	0	0	0	pp_4

Рис. 3. Схема PPG по модулю 16

Fig. 3. The PPG scheme modulo 16

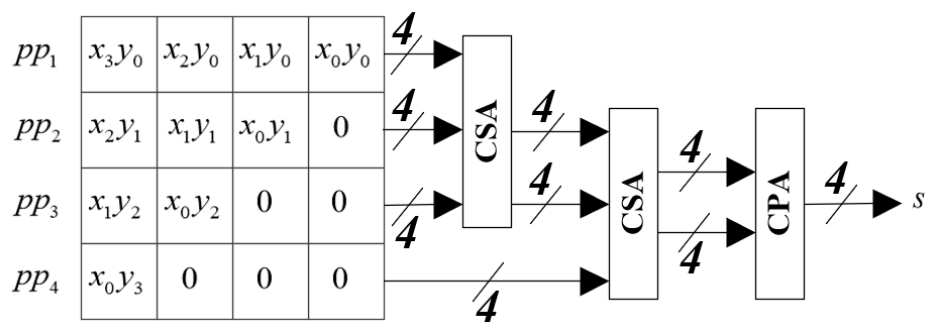


Рис. 4. Умножитель по модулю 16

Fig. 4. Multiplier modulo 16

Если один из множителей имеет знак «-», значение числа должно быть преобразовано в дополнение до двух. Для модуля значение должно быть преобразовано в его инверсию, т. е. все биты числа должны быть инвертированы, а затем просуммированы с единицей. Последняя операция требует суммирования

числа с одним битом. Для этого достаточно использовать цепочку полусумматора (НА) для меньшего потребления площади. Архитектура для представления отрицательного числа показана на рис. 5.

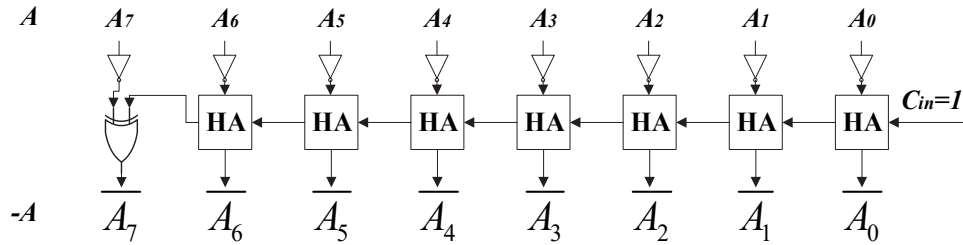


Рис. 5. Схема преобразования в отрицательное число по модулю 256

Fig. 5. A scheme for converting to a negative number modulo 256

1.2. УМНОЖИТЕЛЬ ПО МОДУЛЮ $2^n - 1$ И $2^n + 1$

Для выполнения умножения в СОК по модулю $2^n - 1$ используют специальный генератор частичного произведения (рис. 6), модульные сумматоры EAC-CSA (End-Around Carry CSA, циклический перенос старшего бита CSA) и EAC-CPA (End-Around Carry CPA, циклический перенос старшего бита CPA) (рис. 7, 8). На рис. 7 и 8 FA – обозначение полного сумматора. Если число членов в PPG (partial product generator, генератор частичных повреждений) равно трем или более, то для сведения суммы к двум членам используют так называемые деревья EAC-CSA, для которых применяется EAC-CPA для модуля $2^n - 1$; используют модули 3 ($2^2 - 1$) и 7 ($2^3 - 1$) из текущего набора СОК. Архитектуры умножителей для этих модулей показаны на рис. 9 и 10.

PPG mod 2^4-1				
3	2	1	0	n
x_3y_0	x_2y_0	x_1y_0	x_0y_0	pp_1
x_2y_1	x_1y_1	x_0y_1	x_3y_1	pp_2
x_1y_2	x_0y_2	x_2y_2	x_3y_2	pp_3
x_0y_3	x_3y_3	x_2y_3	x_1y_3	pp_4

Рис. 6. Схема PPG по модулю 15

Fig. 6. The. PPG scheme modulo 15

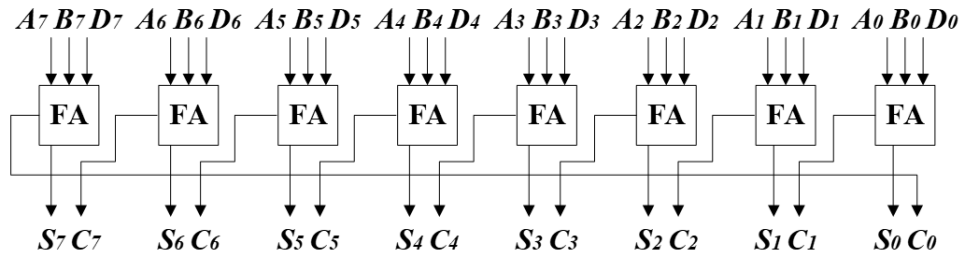


Рис. 7. Схема EAC-CSA по модулю 255

Fig. 7. The EAC-CSA scheme modulo 255

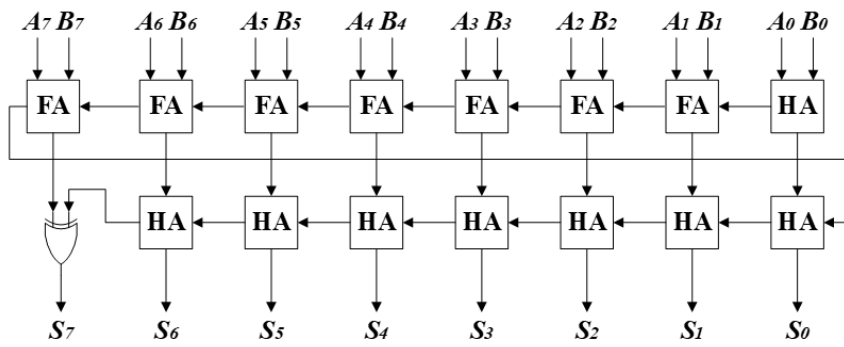


Рис. 8. Схема EAC-CPA по модулю 255

Fig. 8. The EAC-CPA scheme modulo 255

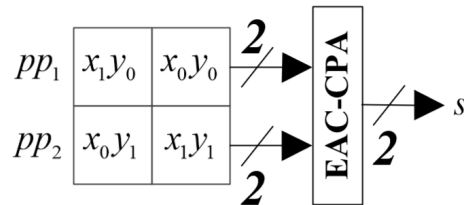


Рис. 9. Умножитель по модулю 3

Fig. 9. The multiplier modulo 3

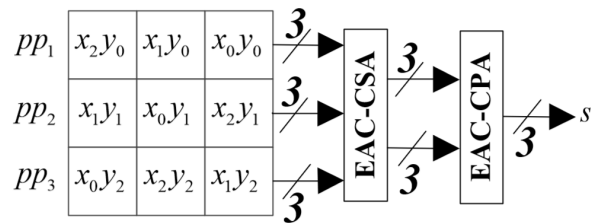


Рис. 10. Умножитель по модулю 7

Fig. 10. The multiplier modulo 7

Представление отрицательного числа по модулю $2^n - 1$ так же просто, как инвертирование всех битов числа. Для этого используются инверторы, независимые друг от друга. Архитектура преобразования в отрицательное число для этого модуля показана на рис. 11.

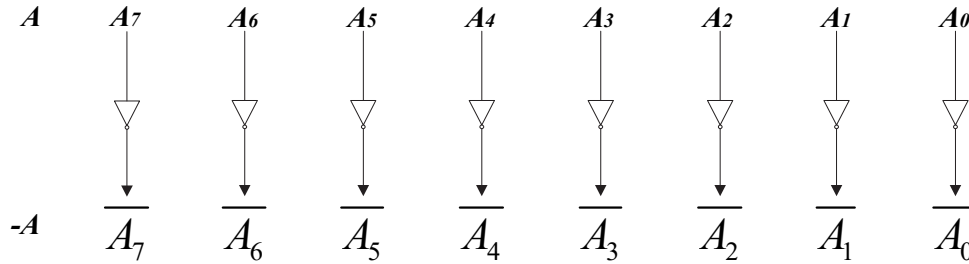


Рис. 11. Схема преобразования в отрицательное число по модулю 255

Fig. 11. A scheme for converting to a negative number modulo 255

Для выполнения операции умножения в СОК по модулю $2^n + 1$, в отличие от модулей 2^n и $2^n - 1$, необходимо преобразовать множители в кодировку Diminished-one (D-1), а затем выполнить генерацию частичных произведений (рис. 12), просуммировать значения с помощью блоков IEAC-CSA (инвертированный EAC-CSA) (рис. 13) для трех и более членов и IEAC-CRA (инвертированный EAC-CRA) для двух членов (рис. 14). Кроме членов из PPG, необходимо выполнить сложение со значениями множителей и поправочным коэффициентом cor , равным $\underbrace{00\dots00}_{n-bit}$.

PPG mod 2^4+1				
3	2	1	0	
$x'_3 y'_0$	$x'_2 y'_0$	$x'_1 y'_0$	$x'_0 y'_0$	pp_1
$x'_2 y'_1$	$x'_1 y'_1$	$x'_0 y'_1$	$\overline{x'_3 y'_1}$	pp_2
$x'_1 y'_2$	$x'_0 y'_2$	$\overline{x'_3 y'_2}$	$\overline{x'_2 y'_2}$	pp_3
$x'_0 y'_3$	$\overline{x'_3 y'_3}$	$\overline{x'_2 y'_3}$	$\overline{x'_1 y'_3}$	pp_4

Рис. 12. Схема PPG по модулю 17

Fig. 12. PPG scheme modulo 17

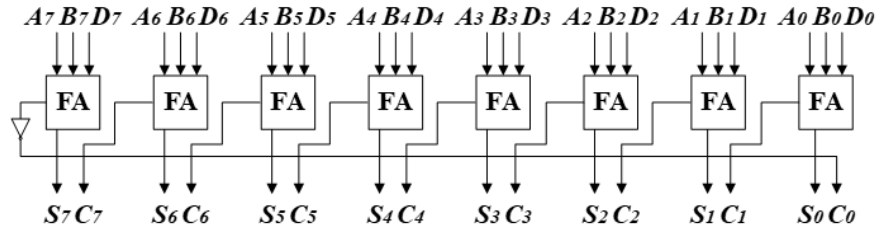


Рис. 13. Схема IEAC-CSA по модулю 257

Fig. 13. The IEAC-CSA scheme modulo 257

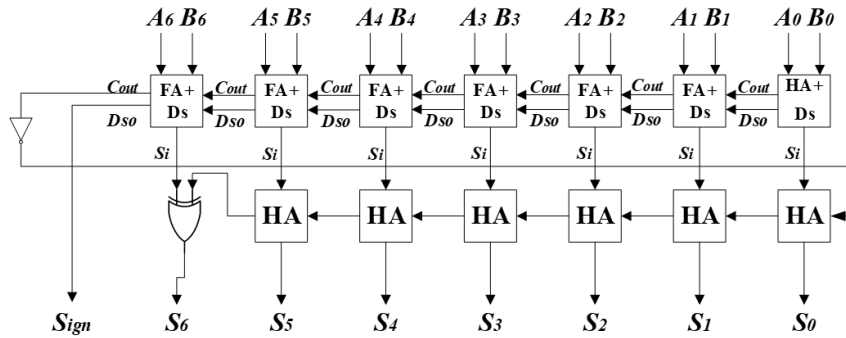


Рис. 14. Схема IEAC-CPA по модулю 129

Fig. 14. The IEAC-CPA scheme modulo 129

Чтобы преобразовать число в кодировку D-1, необходимо вычесть единицу из исходного числа, если значение не равно нулю, или преобразовать в набор 100...00, если оно равно нулю.

Для выполнения вычитания чисел необходимо представить вычитаемое число в дополнительном коде [19], а затем прибавить его к первому числу. Значение числа «-1» в дополнительном коде будет равно $\underbrace{11\dots11}_{n-bit}$, т. е. необходимо

просуммировать это значение с исходным числом по модулю 2^n . Если исходное значение равно нулю, то сумма будет равна 11...11. Получается, что необходимо обнаружить все биты, чтобы узнать, равно это значение нулю или нет.

В качестве сумматора предлагается использовать схему сложения CPA. Поскольку нам известен второй член, схема упрощается за счет свойств логических выражений:

$$S = A \oplus 1 \oplus D = \bar{A} \oplus D = A \odot D; \quad (1)$$

$$C = ((A \oplus 1) \wedge D) \vee (A \wedge 1) = (\bar{A} \wedge D) \vee A = A \vee D, \quad (2)$$

где A , B и D – члены, S и C – результаты бита суммы и бита переноса, а $\oplus$, $\odot$, $\wedge$, $\vee$ и $\bar{A}$ – вентили Иключающее ИЛИ (XOR), Иключающее НЕ (XNOR), И, ИЛИ и НЕ. Архитектуры этих вычислений показаны на рис. 15. Последний бит переноса сумматора покажет, какое число было исходным,

и в зависимости от этого бита преобразуются остальные биты. Если исходное число не было равно нулю, исходное число уменьшится на единицу, старший бит переноса будет равен единице, а знак определения нуля будет равен нулю. Если исходное число было равно нулю, старший бит переноса будет равен нулю, знак обнаружения нуля будет равен единице, а остальные биты будут установлены в ноль. Для исправления числа используется логическая операция И над каждым битом, кроме знакового, так как по его свойству $A \wedge 1 = A$; $A \wedge 0 = 0$. Старший бит (обнаружение нуля) вычисляется с помощью логической операции НЕ над старшим битом переноса. На рис. 16 показана архитектура преобразователя в D-1.

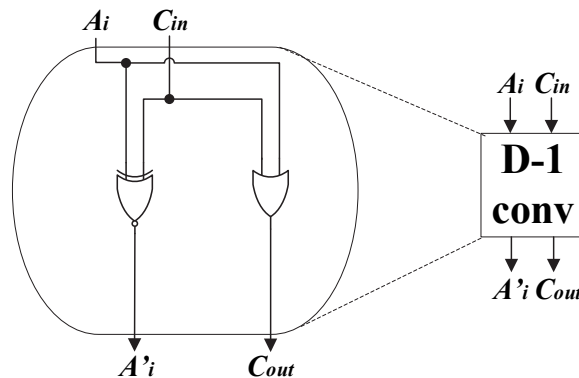


Рис. 15. Схемы расчета для преобразования числа в кодировку Diminished-one

Fig. 15. Calculation schemes for converting a number to a Diminished-one encoding

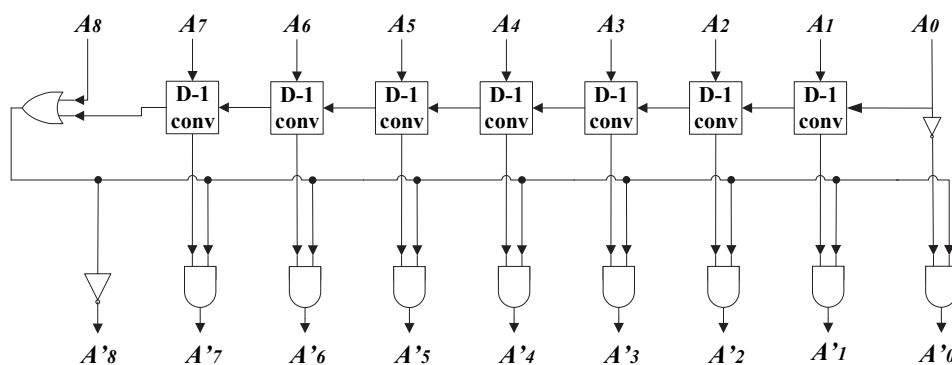


Рис. 16. Схема преобразования чисел в Diminished-one на основе CPA

Fig. 16. A number conversion scheme in Diminished-one based on CPA

Для преобразования значения из Diminished-one в исходное значение RNS по модулю $2^n + 1$ использовалась архитектура последовательных НА, поскольку нам достаточно сложить число в Diminished-one без старшего бита с инверсией старшего бита (определение нуля). Схема преобразователя из D-1 представлена на рис. 17.

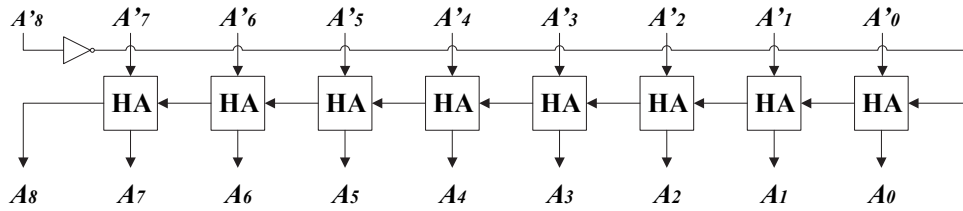


Рис. 17. Схема преобразования числа из кодировки Diminished-one по модулю 257, основанная на блоках HA

Fig. 17. A scheme for converting a number from Diminished-one modulo 257, based on HA blocks

Для модуля $2^n + 1$ представление отрицательного числа в D-1 зависит от старшего бита числа. Если он равен нулю, то все биты, кроме старшего, инвертируются, в противном случае ничего не меняется. Для этого используется логический элемент NOR, который обладает следующим свойством:

$$\overline{A \vee 1} = \overline{A} \wedge \overline{1} = \overline{A} \wedge 0 = 0; \quad (3)$$

$$\overline{A \vee 0} = \overline{A} \wedge \overline{0} = \overline{A} \wedge 1 = \overline{A}. \quad (4)$$

В первом случае все биты установлены в ноль, но изначально они были равны нулю, поэтому никаких изменений здесь не произошло. Во втором случае все биты, кроме самого старшего, инвертированы. Архитектура для представления отрицательного числа показана на рис. 17. Числа 5 и 17 являются подходящими модулями для используемого набора RNS. Архитектуры умножителей для этих модулей показаны на рис. 18 и 19.

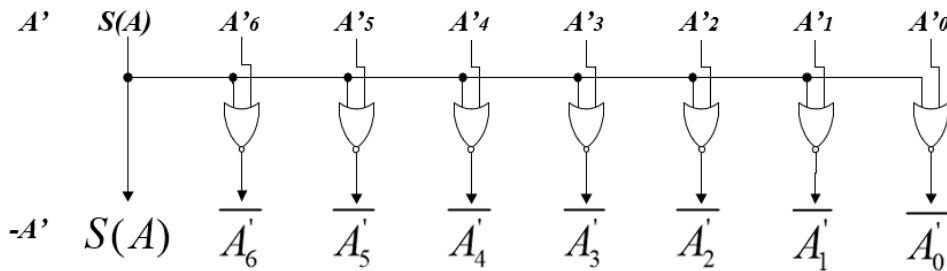


Рис. 18. Преобразование отрицательного числа по модулю 129

Fig. 18. Converting a negative number modulo 129

В табл. 1 представлен сравнительный анализ всех рассмотренных умножителей по различным модулям СОК (2^n , $2^n - 1$, $2^n + 1$). Несмотря на очевидную более высокую сложность реализации вычислений по модулю $2^n + 1$, далее будет продемонстрирована высокая эффективность в применении модулей такого вида в цифровых фильтрах для повышения коэффициента параллелизма вычислений в СОК.

Разработка архитектур модульных умножителей для различных типов модулей в системе остаточных классов является основой эффективной реализации арифметических операций в цифровых фильтрах. Однако при увеличении количества модулей или разрядности входных данных аппаратные затраты и энергопотребление системы могут существенно возрасти. Одним из способов дальнейшего повышения эффективности реализации цифровых фильтров является оптимизация самих коэффициентов фильтра – их масштабирование с последующим округлением и переводом в целочисленную форму. Масштабирование позволяет уменьшить диапазон значений, используемых в вычислениях, за счет предварительного умножения коэффициентов на степени двойки и округления результата. Такое преобразование не только снижает разрядность арифметики, но и упрощает представление данных в системе остаточных классов, поскольку позволяет использовать модули меньшей величины. Это открывает возможность дополнительного сокращения аппаратной сложности и повышения энергоэффективности, особенно при реализации фильтра на ПЛИС.

Таблица 1

Table 1

Сравнительный анализ методов реализации умножителей по различным модулям СОК

Comparative analysis of methods for implementing multipliers for various RNS modules

Тип модуля	Ключевые особенности архитектуры умножителя	Метод обработки отрицательных чисел	Дополнительные преобразования	Пример модуля в работе
2^n	Использование стандартных блоков PPG, CSA, CPA. Наиболее простая реализация	Инверсия битов и прибавление единицы (цепочка полусумматоров)	Не требуются	16, 32, 64
$2^n - 1$	Применение сумматоров с циклическим переносом (EAC-CSA, EAC-CPA)	Простая инверсия всех битов числа	Не требуются	3, 7, 15, 63
$2^n + 1$	Использование инвертированных сумматоров с циклическим переносом (IEAC-CSA, IEAC-CPA). Требуется преобразование в код D-1	Инверсия битов (кроме старшего) в зависимости от значения старшего бита в представлении D-1	Обязательное преобразование в D-1 и обратно. Схемы на основе CPA и цепочек полусумматоров (НА)	5, 17, 65

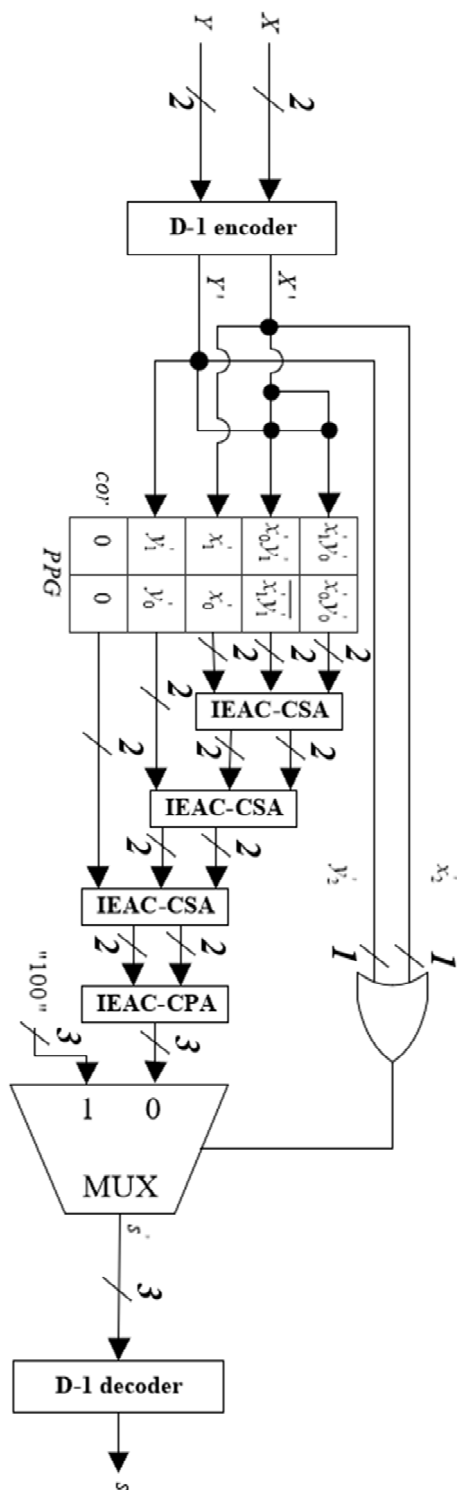


Рис. 19. Умножитель по модулю 5

Fig. 19. Multiplier modulo 5

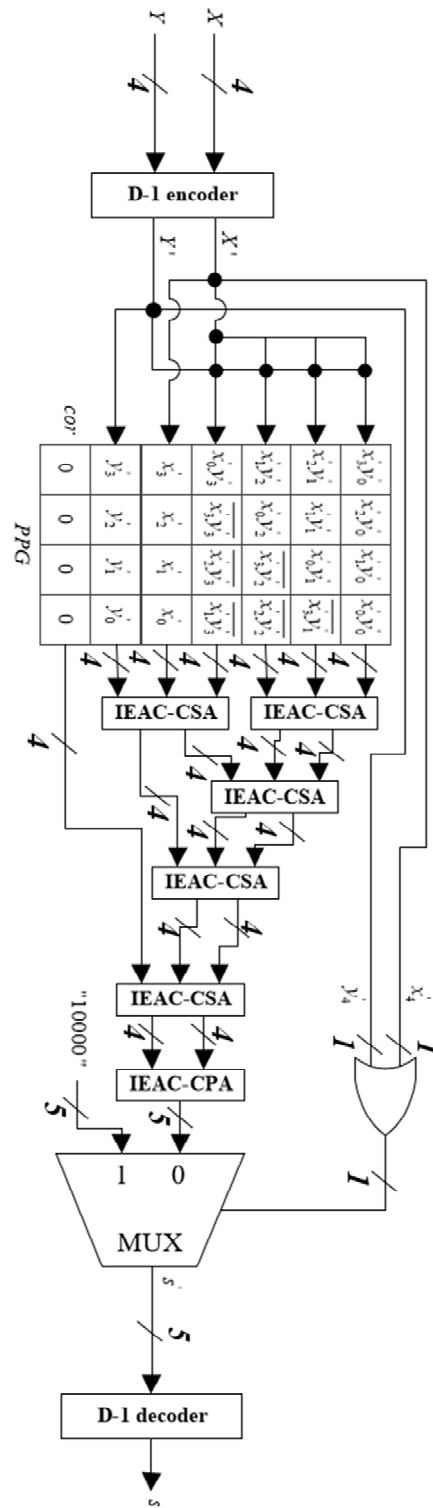


Рис. 20. Умножитель по модулю 17

Fig. 20. Multiplier modulo 17

Следующий раздел посвящен рассмотрению метода масштабирования коэффициентов, его влиянию на характеристики фильтра и результатам аппаратного моделирования на ПЛИС.

2. МАСШТАБИРОВАНИЕ КОЭФФИЦИЕНТОВ ФИЛЬТРА В СОК

Для эффективной аппаратной реализации линейной пространственной фильтрации на современных микроэлектронных устройствах все данные в их памяти должны быть представлены в дополнительном коде в формате с фиксированной точкой. Преобразуем цифровой фильтр F с коэффициентами f_i в целочисленное представление с помощью округления. Для этого умножаем их на 2^n (n – степень масштабирования) и округляем по формуле $f_i^* = \lceil 2^n f_i \rceil$. Далее, после обработки сигнала, преобразуем результат свертки следующим образом: делим на 2^n и округляем до меньшего значения. Таким образом, вычислительная погрешность минимизируется, так как погрешность округления в большую сторону компенсируется погрешностью округления в меньшую.

Одномерная линейная пространственная фильтрация основана на использовании операций сложения и умножения. При больших размерностях операндов эффективность вычислений может быть повышена за счет параллельной организации вычислений в СОК. Эта система представляет собой прямую сумму конечных колец и оперирует малыми остатками вместо обработки чисел большой разрядности в позиционных системах счисления, что открывает возможности для высокоскоростной параллельной обработки данных на арифметико-логическом уровне. Поэтому СОК целесообразно использовать для задач с преобладающим числом операций сложения, вычитания и умножения. Одной из таких задач является цифровая фильтрация сигналов. Для фильтрации с использованием модульных вычислений необходимо сначала перевести масштабированные коэффициенты в СОК с модулем $\{m_1, \dots, m_l\}$. Графическая интерпретация масштабирующих коэффициентов представлена на рис. 21.

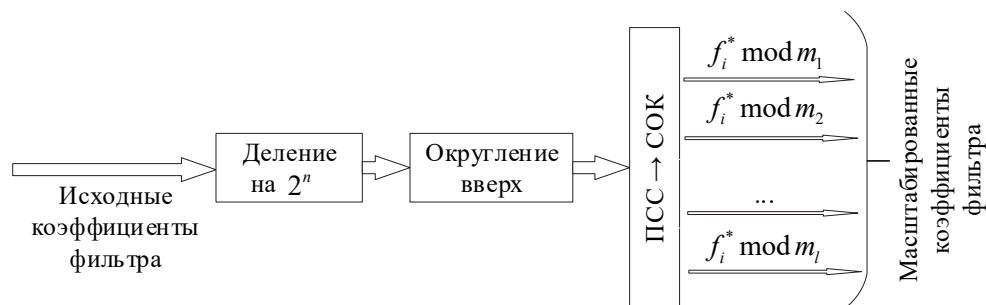


Рис. 21. Схема масштабирования коэффициентов фильтра обработки сигнала и их представление в системе остаточных классов

Fig. 21. A scaling scheme of signal processing filter coefficients and their representation in the residue class system

Схема обработки сигнала с использованием масштабированных коэффициентов одномерного фильтра и параллельных вычислений в СОК имеет вид, представленный на рис. 22.

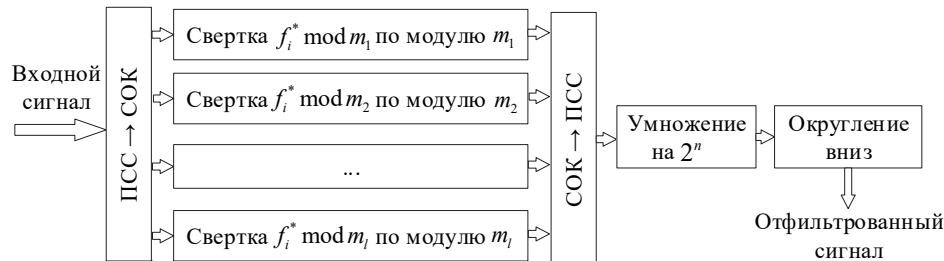


Рис. 22. Предлагаемая схема обработки сигнала одномерным фильтром с масштабированными коэффициентами и параллельными вычислениями в СОК

Fig. 22. The proposed scheme of signal processing by a 1D filter with scaled coefficients and parallel computing in the RNS

Разрядность коэффициентов фильтра n , эффективная с точки зрения вычислительной сложности устройств цифровой фильтрации сигналов и не приводящая к существенному ухудшению качества обработки сигналов, определяется в зависимости от характеристик используемого цифрового фильтра и требований к устройству.

3. АППАРАТНОЕ МОДЕЛИРОВАНИЕ ФИЛЬТРОВ С МАСШТАБИРОВАНИЕМ КОЭФФИЦИЕНТОВ В СОК

Для аппаратного моделирования использовался стандартный комплекснозначный фильтр 15-го порядка с конечной импульсной характеристикой и произвольными 16-битными коэффициентами, предназначенный для обработки 16-битного сигнала. Фильтры такого типа широко используются в современных беспроводных сенсорных сетях и мобильной связи. Рассмотрим работу такого фильтра в различных СОК с наборами модулей $\{127, 65, 64, 33, 31, 17\}$, $\{127, 65, 33, 32, 31, 17\}$ и $\{127, 65, 33, 31, 17, 16\}$. Разница между наборами заключается в модуле 2^n (64, 32 и 16). В первом наборе входной сигнал и данные коэффициента подаются без изменений, что повторяет известный метод реализации цифровых фильтров в СОК из работы [17]. Во втором и третьем наборах значения сигнала и коэффициента уменьшаются в 2 и 4 раза соответственно из-за разницы в модуле 2^n , а именно: $(64/32 = 2, 64/16 = 4)$. Указанное обстоятельство приводит к общему снижению точности вычислений при фильтрации в 2 и 4 раза соответственно.

Важной характеристикой аппаратной реализации является количество используемых Flip-Flops и LUTs. Так, Flip-Flops (защелки) представляют собой базовые элементы памяти, используемые для хранения одного бита информации и реализации регистров, счетчиков и других синхронных элементов. Они определяют объем используемой синхронной логики и влияют на глубину и стабильность вычислительного конвейера. LUTs (Look-Up Tables, таблицы подстановки) – это программируемые логические таблицы, применяемые для реализации булевых функций. По сути, они формируют основу комбина-

ционной логики на ПЛИС. Количество задействованных LUTs напрямую связано с логической сложностью схемы и объемом вычислений, выполняемых за один такт. Совокупность этих двух параметров позволяет количественно оценить ресурсоемкость цифровой схемы и проводить сравнение различных архитектур по их аппаратной эффективности.

Было проведено аппаратное моделирование фильтров на ПЛИС в Xilinx Vivado на целевой плате Xilinx Virtex 7 xc7vx485tffg1157-1, содержащей 607200 Flip-Flops и 303600 LUTs. Результаты аппаратного моделирования представлены в табл. 2. Проанализировав результаты, можно сделать вывод, что при уменьшении одного модуля в 2 раза удалось снизить затраты на оборудование на 2,2 %, затраты на электроэнергию – на 0,9 %. При уменьшении одного модуля в 4 раза удалось снизить затраты на оборудование на 4,3 %, затраты на электроэнергию – на 3,4 %.

Было использовано масштабирование для уменьшения диапазона СОК. Таким образом, можно уменьшить разрядность вычислений и тем самым сократить аппаратные и энергетические затраты до 10 % для масштабирования до 5 бит. Однако чем больше масштабируются биты, тем выше погрешность вычислений для выходного сигнала.

Продемонстрированный в табл. 2 результат открывает широкие возможности для применения фильтров с масштабированными коэффициентами в СОК на практике. Такие устройства могут найти применение для решения широкого класса задач цифровой обработки сигналов, изображений и видео, а также при разработке систем машинного обучения, особенно чувствительных к скорости обработки информации.

Таблица 2

Table 2

Результаты аппаратного моделирования фильтров с масштабированием коэффициентов в СОК

Results of hardware modeling of filters with coefficient scaling in RNS

Тип фильтра	Набор модулей	Flip-Flops	LUTs	Энергопотребление, мВт
Фильтр без масштабирования [17]	(127, 65, 64, 33, 31, 17)	4158	19 874	549,578
Масштабирование	(127, 65, 33, 32, 31, 17)	3997	19 445	544,360
	(127, 65, 33, 31, 17, 16)	3913	19 013	530,759

Таким образом, применение масштабирования в системе остаточных классов позволяет не только снизить ресурсоемкость реализации фильтров, но и повысить их энергоэффективность без значительного ущерба точности. Полученные результаты подтверждают практическую ценность предложенного подхода для использования в современных цифровых устройствах обработки сигналов.

ЗАКЛЮЧЕНИЕ

В статье рассмотрен комплексный подход к проектированию цифровых фильтров с использованием СОК и масштабирования коэффициентов. Особое внимание уделено разработке архитектур модульных умножителей для различных типов модулей: степеней двойки, чисел вида $2^n - 1$, $2^n + 1$ и произвольных простых чисел. Для каждого типа модуля предложены схемотехнические решения, адаптированные к особенностям модульной арифметики, включая использование кодировки D-1, схем CSA/CPA и специализированных преобразователей для отрицательных чисел.

Полученные архитектуры демонстрируют высокую эффективность с точки зрения логической сложности и возможности параллельной реализации. Применение СОК позволило устранить проблему переносов между разрядами и обеспечить независимую обработку по каждому модулю, что особенно важно для реализации цифровых фильтров с высокой производительностью и низкими задержками. Такой подход особенно актуален при разработке систем реального времени, а также для внедрения в ПЛИС и другие аппаратные платформы с ограниченными ресурсами.

Важным этапом оптимизации стала процедура масштабирования коэффициентов фильтра. Преобразование вещественных коэффициентов в целочисленную форму с помощью масштабирования позволяет значительно сократить разрядность вычислений, а следовательно, уменьшить занимаемую площадь, потребление энергии и затраты на реализацию. Результаты аппаратного моделирования в среде Xilinx Vivado показали, что при снижении одного из модулей в 2 и 4 раза достигается уменьшение аппаратных затрат на 2,2 % и 4,3 % соответственно, а энергопотребление снижается до 3,4 % без существенного ухудшения точности фильтрации.

Таким образом, предложенный подход сочетает в себе преимущества модульной арифметики и алгоритмической оптимизации, обеспечивая эффективное проектирование цифровых фильтров в СОК.

СПИСОК ЛИТЕРАТУРЫ

1. *Оппенгейм А., Шафер Р.* Цифровая обработка сигналов. – М.: Litres, 2022.
2. *Зеленчук Н.А., Терехов А.В.* Быстрый алгоритм вычисления матрично-векторных произведений в задаче разложения функций в ряды Фурье на круге // Системы анализа и обработки данных. – 2024. – № 4 (96). – С. 21–34. – DOI: 10.17212/2782-2001-2024-4-21-34.
3. The impact of digital filters on the diagnosis of simulated root resorptions in digital radiographic systems / N. Oliveira-Santos, H. Gaêta-Araujo, D.C. Ruiz, E.H.L. Nascimento, W.G. Cral, C. Oliveira-Santos, F.C. Groppo // Clinical Oral Investigations. – 2022. – Vol. 26 (7). – P. 4743–4752.
4. *Бахчевников В.В., Деркачев В.А., Бакуменко А.Н.* Реализация согласованного фильтра в частотной области на ПЛИС // Известия ЮФУ. Технические науки. – 2023. – № 2 (232). – С. 156–165.
5. *Sundararajan D.* Digital signal processing: An introduction. – Springer Nature, 2024.
6. *Чурсин А.Г.* Оценка быстродействия цифровых фильтров с конечной импульсной характеристикой // Вестник Воронежского института МВД России. – 2021. – № 3. – С. 146–151.
7. *Гужов В.И., Ильиных С.П., Андриященко Е.В.* Алгоритм быстрого преобразования Фурье для восстановления изображений из голограмм, зарегистрированных с помощью фотоматриц произвольного размера // Системы анализа и обработки данных. – 2024. – № 1 (93). – С. 71–81. – DOI: 10.17212/2782-2001-2024-1-71-81.

8. Ушенина И.В. Использование синтеза высокого уровня для реализации на ПЛИС адаптивных LMS-фильтров, работающих в формате с плавающей запятой // Вестник Томского государственного университета. Управление, вычислительная техника и информатика. – 2022. – № 59. – С. 108–116.
9. Смирнов А.В. Оптимизация характеристик цифровых фильтров одновременно в частотной и временной областях // Российский технологический журнал. – 2020. – Т. 8, № 6. – С. 63–77.
10. Pavan S., Shibata H. Continuous-time pipelined analog-to-digital converters: A mini-tutorial // IEEE Transactions on Circuits and Systems II: Express Briefs. – 2021. – Vol. 68 (3). – P. 810–815.
11. Север К.О., Алексеев К.Н., Турулин И.И. Реализация эффективного разделимого двумерного цифрового фильтра на ПЛИС // Известия ЮФУ. Технические науки. – 2024. – № 3. – С. 158–168.
12. Bindima T., Elias E. Low-complexity 2-D digital FIR filters using polyphase decomposition and Farrow structure // IEEE Transactions on Circuits and Systems I: Regular Papers. – 2019. – Vol. 66 (6). – P. 2298–2308.
13. Belghadr A., Jaberipur G. Efficient variable-coefficient RNS-FIR filters with no restriction on the moduli set // Signal, Image and Video Processing. – 2022. – Vol. 16 (6). – P. 1443–1454.
14. Об аппаратно-программной защите приложений доверенного искусственного интеллекта в базисах RNS с интервально-позиционными характеристиками решающих правил / В.С. Князьков, А.И. Иванов, К.С. Исупов, М.М. Бутаев // Известия высших учебных заведений. Поволжский регион. Технические науки. – 2024. – № 4 (72). – С. 50–69.
15. Omondi A.R., Premkumar A.B. Residue number systems: Theory and implementation. – World Scientific, 2007. – 296 p.
16. Hiasat A.A reverse converter and sign detectors for an extended RNS five-moduli set // IEEE Transactions on Circuits and Systems I: Regular Papers. – 2016. – Vol. 64 (1). – P. 111–121.
17. Lyakhov P.A. Area-efficient digital filtering based on truncated multiply-accumulate units in residue number system $2^n-1, 2^n, 2^n+1$ // Journal of King Saud University – Computer and Information Sciences. – 2023. – Vol. 35 (6). – P. 101574.
18. Leibowitz L. A simplified binary arithmetic for the Fermat number transform // IEEE Transactions on Acoustics, Speech, and Signal Processing. – 1976. – Vol. 24 (5). – P. 356–359.
19. Parhami B. Computer arithmetic: Algorithms and hardware designs. – London: Oxford University Press, 2010. – 492 p.

Ляхов Павел Алексеевич, кандидат физико-математических наук, доцент, заведующий кафедрой математического моделирования факультета математики и компьютерных наук имени профессора Н.И. Червякова Северо-Кавказского федерального университета. Основное направление научных исследований – цифровая обработка сигналов и изображений, искусственный интеллект, нейронные сети, модульная арифметика, параллельные и высокопроизводительные вычисления, цифровые схемы и аппаратные ускорители. Имеет более 200 печатных работ. E-mail: ljahov@mail.ru

Lyakhov Pavel A., Ph.D. in physical and mathematical sciences, Head of the Department of Mathematical Modeling, North-Caucasus Federal University. His research interests are currently focused are digital processing of signals and images, artificial intelligence, neural networks, modular arithmetic, parallel and high-performance computing, digital circuits and hardware accelerators. He has more than 200 published works. E-mail: ljahov@mail.ru

DOI: 10.17212/2782-2001-2025-4-85-104

Designing digital filters with scaling in the residue number system*

P.A. LYAKHOV

North-Caucasus Federal University, 1 Pushkin Street, Stavropol, 664033, Russian Federation
ljahov@mail.ru

Abstract

This article proposed a method for designing digital filters using scaling in the residue number system. The architectures of modular multipliers for various types of modules are considered: powers of two, numbers of the form $2^n - 1$, $2^n + 1$, and arbitrary prime numbers. For each case, effective circuit solutions are proposed, including partial product generators and specialized summation circuits adapted to the features of the corresponding modules. Conversions of numbers to the D-1 code and back, as well as the representation of negative values in residue number system are also considered. A digital filter is designed using coefficient scaling. Scaling allows or a decrease in the bit depth of calculations, which leads to a reduction in the costs of hardware implementation and power consumption. The development takes into account the specifics of modular arithmetic and the implementation features in FPGA logical structures. Conversions between codes and the representation of negative values are described, ensuring correct processing in various submodules. The proposed architectures are adapted to practical use and can be scaled to the requirements of specific hardware platforms and computing tasks, including energy-dependent, high-performance and multi-channel data processing systems. The results of modeling the operation of filters with different sets of modules on FPGA in the Xilinx Vivado environment are presented. The analysis shows that a decrease in one of the modules by 2–4 times makes it possible to achieve a reduction in hardware costs of up to 4.3 % and power consumption of up to 3.4 % with an acceptable level of computational error. The presented approaches make it possible to create productive and economical digital filters for signal processing in a residual class system.

Keywords: residue number system, digital filter, coefficient scaling, modular multiplication, multiplier architecture, FPGA, parallel computing, hardware optimization, digital filter design, filter coefficient quantization

REFERENCES

1. Oppenheim A.V., Schaffer R.W. *Tsifrovaya obrabotka signalov* [Digital Signal Processing]. Moscow, Litres, 2022. (In Russian).
2. Zelenchuk N.A., Terekhov A.V. Bystryi algoritm vychisleniya matrichno-vektornykh proizvedenii v zadache razlozheniya funktsii v ryady Fur'e na krug [A fast algorithm for computing matrix-vector products in the problem of function decomposition into Fourier series on a circle]. *Sistemy analiza i obrabotki dannykh = Analysis and Data Processing Systems*, 2024, no. 4 (96), pp. 21–34. DOI: 10.17212/2782-2001-2024-4-21-34.
3. Oliveira-Santos N., Gaêta-Araujo H., Ruiz D.C., Nascimento E.H.L., Cral W.G., Oliveira-Santos C., Groppo F.C. The impact of digital filters on the diagnosis of simulated root resorptions in digital radiographic systems. *Clinical Oral Investigations*, 2022, vol. 26 (7), pp. 4743–4752.
4. Bakhchevnikov V.V., Derkachev V.A., Bakumenko A.N. Realizatsiya soglasovannogo fil'tra v chastotnoi oblasti na PLIS [Implementation of matched filter in frequency domain on FPGA]. *Izvestiya Yuzhnogo federal'nogo universiteta. Tekhnicheskie nauki = Izvestiya SFedU. Engineering Sciences*, 2023, no. 2 (232), pp. 156–165.
5. Sundararajan D. *Digital signal processing: An introduction*. Springer Nature, 2024.
6. Chursin A.G. Otsenka bystrodeistviya tsifrovyykh fil'trov s konechnoi impul'snoi kharakteristikoi [Performance evaluation of digital filters with finite impulse response]. *Vestnik Voronezhskogo*

* Received 22 April 2025.

instituta MVD Rossii = *Vestnik of Voronezh institute of the Ministry of the interior of Russia*, 2021, no. 3, pp. 146–151.

7. Guzhov V.I., Ilinykh S.P., Andryushchenko E.V. Algoritm bystrogo preobrazovaniya Fur'e dlya vosstanovleniya izobrazhenii iz gologramm, zaregistrirovannykh s pomoshch'yu fotomatriks proizvol'nogo razmera [Fast Fourier transform algorithm for image reconstruction from digital holograms recorded using photomatrices of arbitrary size]. *Sistemy analiza i obrabotki dannykh = Analysis and Data Processing Systems*, 2024, no. 1 (93), pp. 71–81. DOI: 10.17212/2782-2001-2024-1-71-81.

8. Ushenina I.V. Ispol'zovanie sinteza vysokogo urovnya dlya realizatsii na PLIS adaptivnykh LMS-fil'trov, rabotayushchikh v formate s plavayushchei zapyatoi [High-level synthesis for FPGA implementation of adaptive floating-point LMS filters]. *Vestnik Tomskogo gosudarstvennogo universiteta. Upravlenie, vychislitel'naya tekhnika i informatika = Tomsk State University Journal of Control and Computer Science*, 2022, no. 59, pp. 108–116.

9. Smirnov A.V. Optimizatsiya kharakteristik tsifrovyykh fil'trov odnovremenno v chastotnoi i vremennoi oblasti [Optimization of digital filters performances simultaneously in frequency and time domains]. *Rossiiskii tekhnologicheskii zhurnal = Russian Technological Journal*, 2020, vol. 8, no. 6, pp. 63–77.

10. Pavan S., Shibata H. Continuous-time pipelined analog-to-digital converters: A mini-tutorial. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 2021, vol. 68 (3), pp. 810–815.

11. Sever K.O., Alekseev K.N., Turulin I.I. Realizatsiya effektivnogo razdelimogo dvumernogo tsifrovogo fil'tra na PLIS [Implementation of an efficient separable vector digital filter on FPGA]. *Izvestiya Yuzhnogo federal'nogo universiteta. Tekhnicheskie nauki = Izvestiya SFedU. Engineering Sciences*, 2024, no. 3, pp. 158–168.

12. Bindima T., Elias E. Low-complexity 2-D digital FIR filters using polyphase decomposition and Farrow structure. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 2019, vol. 66 (6), pp. 2298–2308.

13. Belghadr A., Jaberipur G. Efficient variable-coefficient RNS-FIR filters with no restriction on the moduli set. *Signal, Image and Video Processing*, 2022, vol. 16 (6), pp. 1443–1454.

14. Knyazkov V.S., Ivanov A.I., Isupov K.S., Butayev M.M. Ob apparatno-programmnoi zashchite prilozhenii doverennogo iskusstvennogo intellekta v basisakh RNS s interval'no-pozitsionnymi kharakteristikami reshayushchikh pravil [On the hardware and software protection of trusted artificial intelligence applications in RNS bases with interval positional characteristics of decision rules]. *Izvestiya vysshikh uchebnykh zavedenii. Povolzhskii region. Tekhnicheskie nauki = University proceedings. Volga region. Engineering sciences*, 2024, no. 4 (72), pp. 50–69.

15. Omondi A.R., Premkumar A.B. *Residue number systems: Theory and implementation*. World Scientific, 2007. 296 p.

16. Hiasat A. A reverse converter and sign detectors for an extended RNS five-moduli set. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 2016, vol. 64 (1), pp. 111–121.

17. Lyakhov P.A. Area-efficient digital filtering based on truncated multiply-accumulate units in residue number system $2^n-1, 2^n, 2^n+1$. *Journal of King Saud University – Computer and Information Sciences*, 2023, vol. 35 (6), p. 101574.

18. Leibowitz L. A simplified binary arithmetic for the Fermat number transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 1976, vol. 24 (5), pp. 356–359.

19. Parhami B. *Computer arithmetic: Algorithms and hardware designs*. London, Oxford University Press, 2010. 492 p.

Для цитирования:

Ляхов П.А. Проектирование цифровых фильтров с масштабированием в системе остаточных классов // Системы анализа и обработки данных. – 2025. – № 4 (100). – С. 85–104. – DOI: 10.17212/2782-2001-2025-4-85-104.

For citation:

Lyakhov P.A. Proektirovanie tsifrovyykh fil'trov s masshtabirovaniem v sisteme ostatochnykh klassov [Designing digital filters with scaling in the residue number system]. *Sistemy analiza i obrabotki dannykh = Analysis and data processing systems*, 2025, no. 4 (100), pp. 85–104. DOI: 10.17212/2782-2001-2025-4-85-104.