СИСТЕМЫ И СРЕДСТВА ИНФОРМАТИКИ

**Научный журнал Российской академии наук
(издается под руководством Отделения нанотехнологий
и информационных технологий РАН)**

Издается с 1989 года
Журнал выходит ежеквартально

**Учредитель:
Федеральный исследовательский центр
«Информатика и управление» Российской академии наук**

РЕДАКЦИОННЫЙ СОВЕТ

РЕДАКЦИОННАЯ КОЛЛЕГИЯ

Редакция

Журнал включен в базу данных Russian Science Citation Index (RSCI),
интегрированную с Web of Science
Журнал входит в систему Российского индекса научного цитирования (РИНЦ)
Журнал включен в базу данных CrossRef (систему DOI — Digital Object Identifier),
в базу данных Ulrich's periodicals directory
и в информационную систему «Общероссийский математический портал Math-Net.Ru»

Журнал реферируется в «Реферативном журнале» ВИНИТИ
и в системе Google Scholar

Журнал включен в сформированный Минобрнауки России Перечень рецензируемых научных изданий, в которых должны быть опубликованы основные научные результаты диссертаций на соискание ученой степени кандидата наук, на соискание ученой степени доктора наук

http://www.ipiran.ru/journal/collected

СИСТЕМЫ И СРЕДСТВА ИНФОРМАТИКИ

Том 31 № 4 Год 2021

СОДЕРЖАНИЕ

ОПТИМИЗАЦИЯ АППАРАТНОЙ ПОДДЕРЖКИ БЫСТРОГО ПРЕОБРАЗОВАНИЯ ФУРЬЕ В РЕКУРРЕНТНОМ СИГНАЛЬНОМ ПРОЦЕССОРЕ*

***Д. В. Хилько*[1], *Ю. А. Степченков*[2], *Ю. И. Шикунов*[3], *Ю. Г. Дьяченко*[4], *Г. А. Орлов*[5]**

Аннотация: Рассматривается поддержка быстрого преобразования Фурье (БПФ, *англ.* FFT — fast Fourier transform) в гибридной архитектуре рекуррентного обработчика сигналов (ГАРОС). Приводится анализ существующей реализации. Выявляются недостатки и их последствия. Предлагается оптимизированное решение, направленное на упрощение масштабирования как архитектуры, так и числа отсчетов БПФ.

Ключевые слова: цифровая обработка сигналов; быстрое преобразование Фурье; цифровой сигнальный процессор; Radix-2

DOI: 10.14357/08696527210407

1 Введение

В области цифровой обработки сигналов (ЦОС) приняты две наиболее распространенные процедуры: цифровая фильтрация и дискретное преобразование Фурье (ДПФ). «ДПФ позволяет анализировать, преобразовывать и синтезировать сигналы такими способами, которые невозможны при непрерывной (аналоговой) обработке» [1]. Дискретное преобразование Фурье используется практически во всех инженерных областях, а также в физике и технике.

Сама математическая процедура ДПФ очень неэффективна: требует проведения очень большого числа комплексных умножений, что делает прямое вычисление ДПФ нецелесообразным. Поэтому в 1965 г. Кули и Тьюки предложили алгоритм вычисления ДПФ [2], который известен как БПФ. Применение

*Исследование выполнено при поддержке Российского научного фонда (проект 19-11-00334).

[1]Федеральный исследовательский центр «Информатика и управление» Российской академии наук, dhilko@yandex.ru

[2]Федеральный исследовательский центр «Информатика и управление» Российской академии наук, YStepchenkov@ipiran.ru

[3]Федеральный исследовательский центр «Информатика и управление» Российской академии наук, YIShikunov@gmail.com

[4]Федеральный исследовательский центр «Информатика и управление» Российской академии наук, diaura@mail.ru

[5]Федеральный исследовательский центр «Информатика и управление» Российской академии наук, orlov.jaja@gmail.com

БПФ сделало возможным проведение Фурье-анализа с помощью цифровых сигнальных процессоров (ЦСП).

Дальнейшее развитие алгоритма БПФ привело к появлению целого семейства алгоритмов: Radix-2, Radix-22, Radix-4 и др. Однако даже с учетом значительно более высокой скорости вычисления алгоритм БПФ все еще требует значительных временны́х затрат, особенно при увеличении разрешающей способности алгоритма (512 отсчетов и больше). Поэтому современный ЦСП должен предоставлять набор инструментов для эффективного вычисления БПФ на широком наборе разрешающих способностей.

Рассматриваемая в статье ГАРОС и ее ключевые особенности (управление потоком самодостаточных данных и рекуррентность) представлены в [3, 4]. Синтезированный на ее основе ПЛИС-прототип (ПЛИС — программируемая логическая интегральная схема) [5] представляет собой ЦСП общего назначения, а его валидация осуществлялась на задаче распознавания изолированных слов. В состав архитектуры были введены механизмы поддержки вычисления БПФ [4]. Однако при увеличении разрешающей способности БПФ оказалось, что накладные аппаратные расходы слишком велики.

Цель статьи — представление результатов оптимизации архитектуры ГАРОС и сокращения ее аппаратных затрат в рамках ПЛИС при сохранении эффективности реализации БПФ с произвольным масштабированием его числа отсчетов.

2 Текущая поддержка в гибридной архитектуре рекуррентного обработчика сигналов

2.1 Описание алгоритма быстрого преобразования Фурье

Алгоритм БПФ — эффективный способ вычисления ДПФ с числом отсчетов, равным натуральным числам во второй степени. Дискретное преобразование Фурье $X(k)$, где $k = 0, \ldots, N-1$, имеет вид:

$$X(k) = \frac{1}{N} \sum_{n=0}^{N-1} x(n) \left[\cos\left(\frac{2\pi nk}{N}\right) - i \sin\left(\frac{2\pi nk}{N}\right) \right] .$$

Поворотные коэффициенты размещены на единичной окружности в комплексной плоскости. Они симметричны и периодичны, что позволяет существенно сократить число умножений, требуемых для проведения ДПФ.

Алгоритм вычисления БПФ по основанию 2 (Radix-2) разделяет проведение ДПФ на серию 2-точечных ДПФ. Каждое такое преобразование называется типовой операцией «бабочка». Для работы алгоритма требуется, чтобы число отсчетов N было натуральной степенью двойки $N = 2^s$, $s \in \mathbb{N}$. Тогда для вычисления БПФ потребуется провести s стадий.

Результаты вычисления на каждой стадии могут быть сохранены в тех же самых ячейках памяти, которые изначально хранили исходные входные отсчеты.

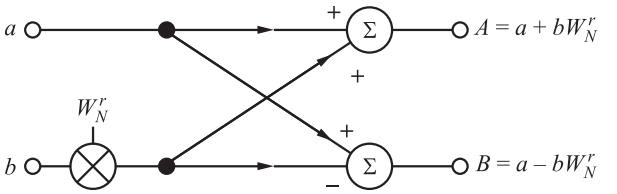

Рис. 1 Операция Radix-2 DIT «бабочка» [6, рис. 5.13]

Быстрое преобразование Фурье с прореживанием по времени (Decimation-in-time, DIT) вычисляет стадии, используя типовую операцию «бабочка», приведенную на рис. 1. Для корректной работы алгоритма необходимо загружать входные отсчеты в бит-реверсированном порядке.

Формулы вычисления комплексного БПФ DIT имеют вид:

$$\begin{aligned} R_a &= R_a + (R_a R_W - I_b I_w) = R_a + (C - D) = R_a + G; \\ I_a &= I_a + (R_b I_W + I_b R_w) = I_a + (E + F) = I_a + H; \\ R_b &= R_b + (R_a R_W - I_a I_w) = R_b - (C - D) = R_b - G; \\ I_b &= I_b + (R_a I_W + I_a R_w) = I_b - (E + F) = I_b - H, \end{aligned}$$

где R — действительная, а I — мнимая часть комплексного числа.

2.2 Описание существующей реализации

Текущая версия прототипа ГАРОС содержит средства аппаратной поддержки вычисления БПФ, которые обеспечивают выполнение 256-точечного Radix-2 DIT алгоритма в формате фиксированной точки с записью результатов на место входных данных (in-place-реализация).

Данные средства архитектуры характеризуются следующими особенностями:

- используется механизм упаковки четырех входных 16-битных данных в одном операнде, действительные и мнимые части — в разных разделах капсулы;
- упакованные данные хранятся в бит-реверсивном порядке;
- поворотные коэффициенты хранятся в блоке «Память констант секционная» (ПК_С): 128 действительных и 128 мнимых 16-битных констант;
- содержит 4 параллельных секции: 4 вычислительных блока (ВБ) и 4 копии ПК_С;
- в систему команд введена специальная инструкция «Butterfly», которая обеспечивает четырехстадийное вычисление Radix-2 DIT «бабочки».

Ключевой элемент аппаратной поддержки БПФ — инструкция «Butterfly». Эта инструкция на аппаратном уровне интерпретируется ВБ как готовый

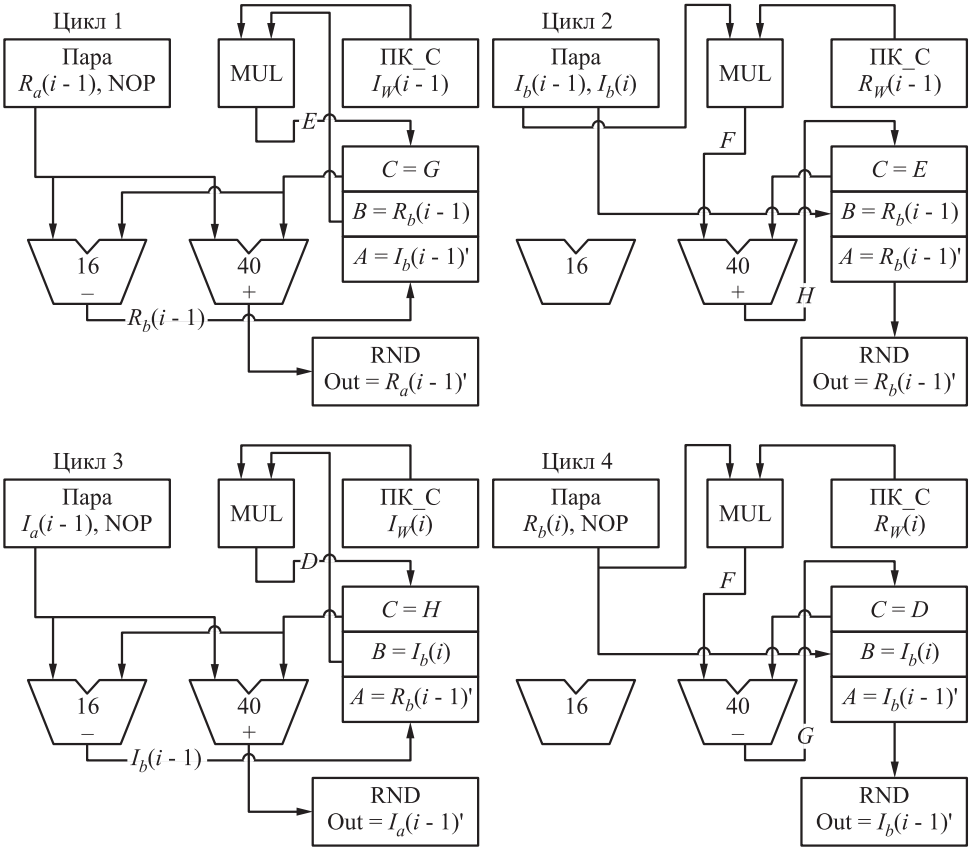

Рис. 2 Существующая схема инструкции «Butterfly»

четырехцикловый сценарий функционирования: каждому циклу выполнения инструкции определена своя схема вычислений. На рис. 2 представлена схема четырехцикловой инструкции: в установившемся режиме после наполнения конвейера «бабочка» вычисляется за 4 цикла.

Метрика Instruction Level Parallelism (ILP) для ВБ ГАРОС имеет значение 4 (1 умножитель, 1 блок сдвига и округления, 1 арифметико-логическое устройство 16-битное, 1 арифметическое устройство 40-битное). За 4 цикла ВБ может максимально выполнить 16 операций. Данная инструкция обеспечивает выполнение 14 операций. Коэффициент эффективности использования ILP составляет 87,5%.

Данное решение было апробировано на моделях архитектуры различного уровня, а также на ПЛИС-прототипе ГАРОС в процессе предваритель-

ной оценки производительности набора алгоритмов из BDTIMark2000 [7]. Несмотря на то что эффективность вычисления БПФ на одной секции ГАРОС сравнима с TMSC55x, данное решение требует аппаратной оптимизации.

2.3 Проблемы аппаратной поддержки быстрого преобразования Фурье в гибридной архитектуре рекуррентного обработчика сигналов

Основные проблемы аппаратной поддержки БПФ в ГАРОС:

(1) избыточные накладные расходы хранения поворотных коэффициентов, связанные с особенностями разделенной по секциям ПК_С;

(2) хранение отсчетов в капсуле в упакованных операндах.

Упаковка данных снижает избыточность тегированных данных, что является несомненным преимуществом. Однако на последних двух стадиях БПФ контроллеру буферной памяти (БП) приходится считывать и записывать части упакованных данных (вместо целого операнда) по разным адресам. Это приводит к усложнению схемы памяти и алгоритма вычисления адресов, росту накладных расходов и потенциальному снижению частоты ее работы;

(3) нецикличная схема считывания действительных и мнимых частей отсчетов.

Из рис. 3 видно, что входные отсчеты считываются в следующей последовательности:

Цикл 1: $R_a(i)$. Цикл 2: $I_b(i)$, а $I_b(i+1)$ уже должен быть в памяти операндов **заранее**!

Цикл 3: $I_a(i)$. Цикл 4: $R_b(i+1)$.

Как видно, действительные и мнимые части считываются непоследовательно и даже из разных «бабочек» в ходе одного прогона инструкции. Это осложняет алгоритм вычисления адресов в контроллере БП;

(4) цикл № 2 инструкции «Butterfly» требует специального режима распаковки и рассылки упакованных данных. В ГАРОС за распаковку и рассылку упакованных данных отвечает компонент «Распределитель». Особенность цикла № 2 заключается в том, что на вход вычислительного блока должны прийти две мнимые части i-й и $(i+1)$-й «бабочек». Это поведение выбивается из основной схемы рассылки данных, что требует ввода специального режима, который используется в ГАРОС только в рамках одного алгоритма;

(5) изменение числа отсчетов БПФ требует переработки капсулы.

Анализ проблем текущей версии средств аппаратной поддержки БПФ в целом свидетельствует о большой степени аппаратной избыточности. Поэтому оптимизация архитектуры ГАРОС является актуальной задачей.

Можно выделить два основных направления оптимизации архитектуры: модификация инструкции «бабочка» и модификация БП.

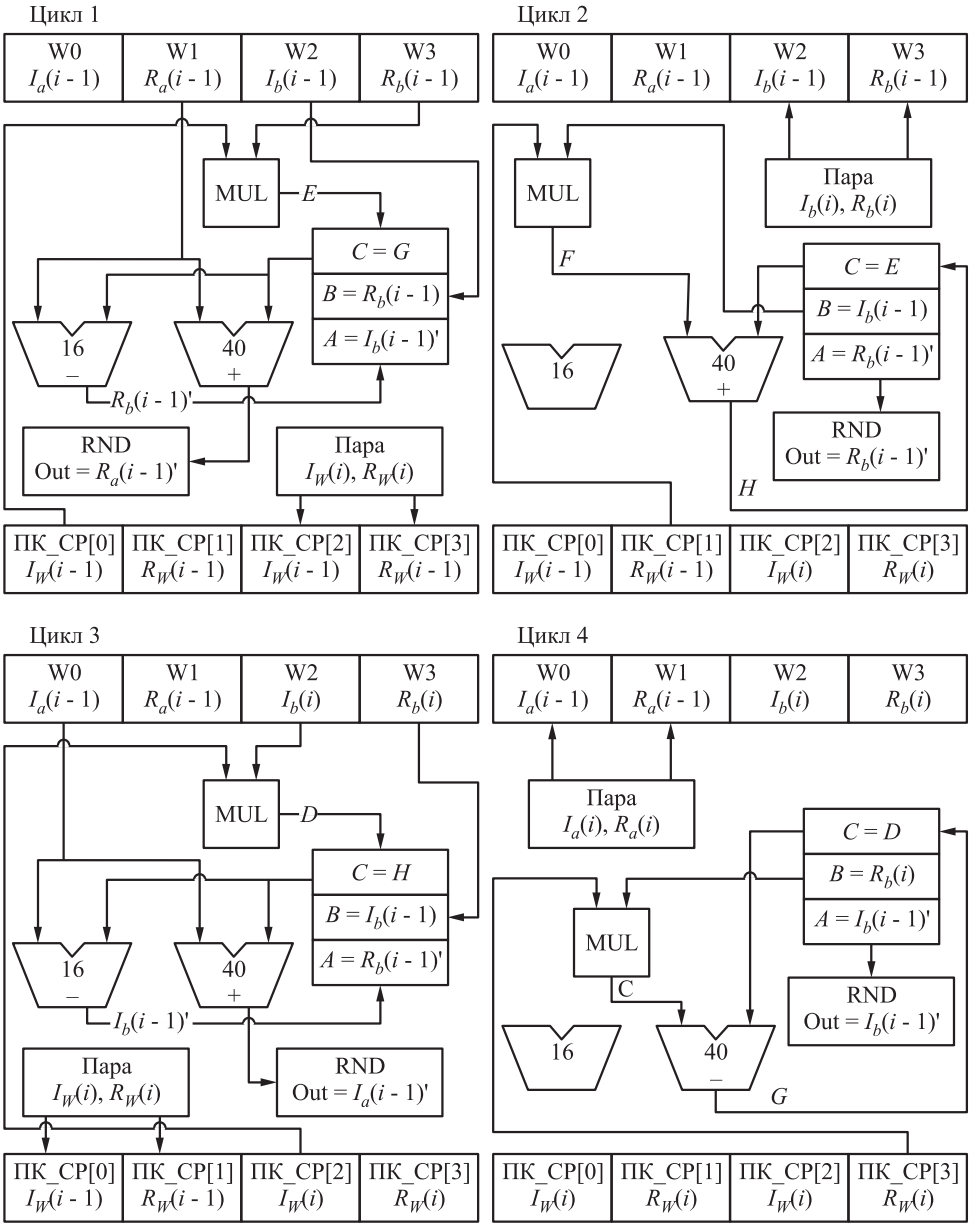

Рис. 3 Усовершенствованная схема инструкции «Butterfly»

3 Модифицированные средства аппаратной поддержки быстрого преобразования Фурье

3.1 Усовершенствование конвейеризованной инструкции

В предыдущем разделе был сделан вывод о необходимости переработки четырехстадийной инструкции «Butterfly». При этом технически ее характеристики не должны ухудшиться с точки зрения времени исполнения. Кроме того, переработанная схема инструкции должна также решать проблему 3, обозначенную в разд. 2.3. На рис. 3 представлены результаты переработки инструкции «бабочка».

В новой версии схемы инструкции поочередно считываются константы и данные, причем для циклов 1 и 3 считываются одни и те же константы, а данные считываются только для i-й бабочки (в отличие от предыдущей версии, где и для $(i+1)$-й). Таким образом, проблема 3 и, следовательно, проблема 4 решаются при внедрении данного решения.

С точки зрения использования ILP предлагаемое решение также обеспечивает выполнение 14 инструкций из 16 за 4 цикла и имеет коэффициент эффективности использования ILP, равный 87,5%.

3.2 Доработка вычислительных блоков

Для обеспечения возможности исполнения новой схемы инструкции «Butterfly» в состав вычислительных блоков ГАРОС необходимо ввести дополнительные входные пользовательские регистры W2 и W3. Таким образом регистровый файл будет расширен до четырех регистров. Для корректного выполнения БПФ исходной инициализации регистрового файла не требуется. Также в состав вычислительного блока входит память констант секционная регистровая (ПК_СР), имеющая объем в четыре 16-битных константы. В новой реализации необходимо обеспечить возможность записи сразу двух констант (ранее можно было записать только одну).

3.3 Модификация распределителя

Так как проблемы 3 и 4 решаются за счет изменения схемы инструкции, то из распределителя можно удалить поддержку специализированного режима распаковки операндов. Кроме того, особенности функционирования модифицированного контроллера БП требуют, чтобы распределитель на завершающей стадии вычисления алгоритма БПФ не замораживал конвейер, а генерировал пустые горсти операндов (что равносильно NOP-операциям).

3.4 Модификация буферной памяти

Самым существенным изменениям подверглась БП и ее контроллер. В состав БП необходимо ввести новый автономный банк памяти и блок управления

этой памятью, который включает в себя два работающих параллельно подблока: генератор адресов и контроллер чтения-записи.

1. **Блок памяти** состоит из блоков: IBlock и RBlock хранят мнимые и действительные части отсчетов преобразуемого сигнала соответственно; IwBlock и RwBlock хранят мнимые и действительные части поворотных коэффициентов W соответственно. Входные последовательности отсчетов должны быть записаны в бит-реверсированном порядке.

   Отсчеты имеют формат Q15, поэтому хранятся в 16-битных словах. Объем IBlock равен объему RBlock и составляет 1024 16-битных слова. Данные в IBlock и RBlock записываются со стороны управляющего устройства, поэтому в VHDL-реализации должен быть предоставлен соответствующий интерфейс доступа для записи и чтения данных блоков (так как in-place-реализация). Режим записи отсчетов как пакетный, так и одиночный.

   Поворотные коэффициенты также имеют формат Q15. Объем IwBlock равен объему RwBlock и составляет 512 16-битных слова. Эти коэффициенты являются константами и рассчитываются заранее. Запись констант осуществляется на этапе прошивки ПЛИС.

2. **Контроллер памяти для режима FFT** обеспечивает параллельное функционирование подблоков вычисления адресов и управления чтением и записью. Принимает данные от компонента «Импликатор» ГАРОС и передает их для записи в блок памяти.

   Инициализация блока осуществляется в момент считывания основным контроллером БП Acm: операнда, который имеет @Cdm = fft (т. е. перенастраивает режим памяти). После этого от операционного уровня архитектуры требуется подтверждение о том, что Acm: операнд получен и все нужные горсти сформированы, которое и инициирует выдачу операндов. После того как получено подтверждение от операционного уровня, начинается основной цикл функционирования.

   Размер Radix2 DIT БПФ определяется по значению поля количества итераций @Ci. Число стадий БПФ напрямую связано с количеством отсчетов по формуле «количество стадий» = $\mathrm{Log}_2(\mathrm{N})$. Тогда, например, для @Ci = 8 $N = 256$. Предлагаемая реализация БПФ контроллера в модели поддерживает настраиваемое количество секций исполнения алгоритма. Однако пока алгоритм рассчитан на 4 секции.

3. **Блок вычисления адресов** должен за 4 цикла вычислить адреса действительных и мнимых частей отсчетов, а также требуемых констант для текущих «бабочек». Особенность алгоритма заключается в том, что IBlock и RBlock адресуются одним и тем же адресом. Таким образом, необходимо вычислить 4 адреса констант и 4 адреса «бабочек».

   Особенностью схемы вычисления инструкции «Butterfly» является наличие четырех дополнительных циклов для запуска процесса. Однако для данного

блока в этом нет необходимости. Он должен сразу рассчитывать корректные адреса. Также данный блок должен работать «на шаг вперед» относительно блока чтения-записи, чтобы второй был всегда обеспечен новыми адресами.

Еще одна особенность алгоритма: адреса и параметры самых первых «бабочек» будут всегда идентичными вне зависимости от размера БПФ. Таким образом, первую порцию адресов нужно вычислять в процессе инициализации аппаратуры.

Наконец, для данного блока требуется особое поведение в завершающей стадии алгоритма, когда адреса для всех «бабочек» уже рассчитаны, но из-за отставания блока чтения-записи на 1 «бабочку», а также конвейера, блок должен продолжать работу вплоть до записи последнего выходного данного на последней стадии алгоритма.

4. **Блок управления чтением-записью** обеспечивает считывание по присланным адресам констант и отсчетов, требуемых для выполнения каждого конкретного цикла инструкции «бабочка». Кроме того, данный блок осуществляет запись выходных операндов, полученных от импликатора. Однако в этом процессе есть одна особенность. Запись выходных данных происходит по адресам «предыдущей бабочки», а значит, необходимо хранить эти адреса.

Рис. 4 Архитектура средств поддержки БПФ

Ранее было отмечено, что выходные данные от импликатора приходят с задержкой, т. е. данный блок должен рассчитывать реальные адреса записи выходных отсчетов наперед и хранить их в блоке FIFO (first in, first out), чтобы сохранить корректность in-place-реализации. Данный блок представляет собой конвейер длины 4; значит, для начала и завершения вычислений требуется наличие четырех дополнительных шагов для заполнения и опустошения FIFO соответственно.

Схема из четырех циклов выполнения инструкции «Butterfly» показывает, что последовательность поступления выходных отсчетов для одной секции следующая: Re, Re, Im, Im. Так как секций 4, получаем, что каждые 8 записей необходимо переключать блоки RBlock и IBlock соответственно. Результат работы блока на каждом шаге — два упакованных Apdi_x4: операнда, содержащих либо константы для четырех секций, либо отсчеты для четырех секций, т. е. константы поступают в рекуррентное операционное устройство стандартным механизмом формирования горстей. В процессе ожидания последних выходных отсчетов на завершении алгоритма блок не должен выдавать ничего. На рис. 4 представлена обновленная архитектура БП.

4 Результаты испытаний программной и аппаратной моделей

В работе [7] дана оценка скорости вычисления 256-точечного БПФ на одной секции, которая составила $2N \log_2 N$. В процессе испытаний тестовые запуски капсулы осуществлялись на четырех секциях ГАРОС — как на предыдущей версии средств аппаратной поддержки, так и на модифицированной.

Сравнительные результаты испытаний

Параметр	Старая версия	Новая версия
Объем памяти для хранения констант (16-битных слов)	$256 \times 4 = 1024$	1024
Поддерживаемые размеры окон БПФ (точек)	256	от 8 до 1024
Размер капсулы (операндов)	132 для отсчетов + 23	23
Объем памяти для отсчетов (16-битных слов)	528	512
Benchmark (циклов)	$2N \log_2 N/4 = 1024$	$2N \log_2 N/4 = 1024$
Overhead (циклов)	44	16
Итого (циклов)	1068	1040

В таблице приводятся сравнительные результаты для двух версий. Данные приведены для $N = 256$ точек.

5 Заключение

Полученные результаты подтверждают успешность оптимизации и модификации средств аппаратной поддержки БПФ в ГАРОС. Новая версия средств

не только обладает большей гибкостью и скоростью, но и позволяет сократить размер капсулы. Полученная реализация средств поддержки позволила решить все перечисленные в разд. 2.3 проблемы.

Проблема 1 была решена путем размещения поворотных коэффициентов в отдельном блоке памяти, причем его чтение осуществляется по стандартному в ГАРОС механизму за счет формирования на выходе пары упакованных операндов. Более того, в том же самом объеме памяти было размещено гораздо больше различных значений поворотных коэффициентов, за счет чего повысилась гибкость и масштабируемость алгоритма.

Проблема 2 была решена также путем размещения отсчетов в отдельном блоке памяти и формирования пары упакованных операндов на выходе при его чтении. Более того, это автоматически решило проблему 5, так как теперь модификация капсулы — это всего лишь реконфигурация размера вычисляемого БПФ.

Проблемы 3 и 4 были решены за счет существенной переработки инструкции «Butterfly».

В различных областях ЦОС требуется вычислять не только БПФ с прореживанием по времени, но и по частоте, а также обратный БПФ. Кроме того, алгоритм по основанию 2 менее быстрый, чем по основанию 4, поэтому в будущем желательно обеспечить реализацию и других алгоритмов из семейства БПФ. Другим направлением работ в данном направлении должна стать апробация IP-блоков БПФ для эффективной аппаратной реализации «бабочек» и еще большего повышения производительности ГАРОС на этом классе задач.

Литература

1. *Lyons R. G.* Understanding digital signal processing. — 3rd ed. — Pearson Education, 2011. 966 p.
2. *Cooley J., Tukey J.* An algorithm for the machine calculation of complex Fourier series // Math. Comput., 1965. Vol. 19. No. 90. P. 297–301.
3. *Степченков Ю. А., Дьяченко Ю. Г., Хилько Д. В., Петрухин В. С.* Рекуррентная потоковая архитектура: особенности и проблемы реализации // Проблемы разработки перспективных микро-и наноэлектронных систем, 2016. № 2. С. 120–127.
4. *Хилько Д. В., Степченков Ю. А., Шикунов Д. И., Шикунов Ю. И.* Рекуррентная потоковая архитектура: технические аспекты реализации и результаты моделирования // Проблемы разработки перспективных микро-и наноэлектронных систем, 2016. № 2. Р. 128–135.
5. *Stepchenkov Yu., Morozov N., Khilko D., Shikunov Yu., Orlov G.* Hybrid multi-core recurrent architecture approbation on FPGA // IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering Proceedings. — Piscataway, NJ, USA: IEEE, 2019. P. 1705–1708.
6. Mixed-signal and DSP design techniques / Ed. W. Kester. — Analog Devices Inc., 2003, 410 p.
7. *Stepchenkov Yu. A., Khilko D. V., Shikunov Yu. I., Orlov G. A.* DSP filter kernels preliminary Benchmarking for recurrent data-flow architecture // IEEE Conference of

Russian Young Researchers in Electrical and Electronic Engineering Proceedings. — Piscataway, NJ, USA: IEEE, 2021. P. 2040–2044.

Поступила в редакцию 22.09.21

HARDWARE SUPPORT OF FAST FOURIER TRANSFORM OPTIMIZATION IN A RECURRENT SIGNAL PROCESSOR

D. V. Khilko, Yu. A. Stepchenkov, Yu. I. Shikunov, Yu. G. Diachenko, and G. A. Orlov

Federal Research Center "Computer Science and Control" of the Russian Academy of Sciences, 44-2 Vavilov Str., Moscow 119133, Russian Federation

Abstract: The paper covers the fast Fourier transform (FFT) support in the hybrid recurrent signal processor architecture. An analysis of the existing implementation is presented. Disadvantages and their ramifications are identified. An optimized solution is proposed to ease the scaling of both the architecture and the number of FFT samples.

Keywords: digital signal processing; fast Fourier transform; digital signal processor; Radix-2

DOI: 10.14357/08696527210407

Acknowledgments

The research was supported by the Russian Science Foundation (project No. 19-11-0034).

References

1. Lyons, R. G. 2011. *Understanding digital signal processing*. 3rd ed. Pearson Education. 966 p.
2. Cooley, J., and J. Tukey. 1965. An algorithm for the machine calculation of complex Fourier series. *Math. Comput.* 19(90):297–301.
3. Stepchenkov, Yu. A., Yu. G. Diachenko, D. V. Khilko, and V. S. Petrukhin. 2016. Rekurrentnaya potokovaya arkhitektura: osobennosti i problemy realizatsii [Recurrent data-flow architecture: Features and realization problems]. *Problemy razrabotki perspektivnykh mikro- i nanoelektronnykh system* [Problems of perspective micro- and nanoelectronic systems development] 2:120–127.
4. Khilko, D. V., Yu. A. Stepchenkov, D. I. Shikunov, and Y. I. Shikunov. 2016. Rekurrentnaya potokovaya arkhitektura: tekhnicheskiye aspekty realizatsii i rezul'taty modelirovaniya [Recurrent data-flow architecture: Technical aspects of implementation and modeling results]. *Problemy razrabotki perspektivnykh mikro- i nanoelektronnykh system* [Problems of perspective micro- and nanoelectronic systems development] 2:128–135.

5. Stepchenkov, Yu., N. Morozov, D. Khilko, Yu. Shikunov, and G. Orlov. 2019. Hybrid multi-core recurrent architecture approbation on FPGA. *IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering Proceedings*. Piscataway, NJ: IEEE. 1075–1078.
6. Kesten, W., ed. 2003. *Mixed-signal and DSP design techniques*. Analog Devices Inc. 410 p.
7. Stepchenkov, Yu. A., D. V. Khilko, Yu. I. Shikunov, and G. A. Orlov. 2021. DSP filter kernels preliminary benchmarking for recurrent data-flow architecture. *IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering Proceedings*. Piscataway, NJ: IEEE. 2040–2044.

Received September 22, 2021

Contributors

Khilko Dmitri V. (b. 1987) — senior scientist, Institute of Informatics Problems, Federal Research Center "Computer Science and Control" of the Russian Academy of Sciences, 44-2 Vavilov Str., Moscow 119333, Russian Federation; dhilko@yandex.ru

Stepchenkov Yuri A. (b. 1951) — Candidate of Science (PhD) in technology, leading scientist, Institute of Informatics Problems, Federal Research Center "Computer Science and Control" of the Russian Academy of Sciences, 44-2 Vavilov Str., Moscow 119333, Russian Federation; YStepchenkov@ipiran.ru

Shikunov Yury I. (b. 1995) — engineer-researcher, Institute of Informatics Problems, Federal Research Center "Computer Science and Control" of the Russian Academy of Sciences, 44-2 Vavilov Str., Moscow 119333, Russian Federation; yishikunov@gmail.com

Diachenko Yuri G. (b. 1958) — Candidate of Science (PhD) in technology, senior scientist, Institute of Informatics Problems, Federal Research Center "Computer Science and Control" of the Russian Academy of Sciences, 44-2 Vavilov Str., Moscow 119333, Russian Federation; diaura@mail.ru

Orlov Georgy A. (b. 1994) — engineer-researcher, Institute of Informatics Problems, Federal Research Center "Computer Science and Control" of the Russian Academy of Sciences, 44-2 Vavilov Str., Moscow 119333, Russian Federation; orlov.jaja@gmail.com