УДК 004.424.2 DOI 10.18522/2311-3103-2020-7-130-142

М.Д. Чекина

РЕАЛИЗАЦИЯ ФРАКТАЛЬНОГО СЖАТИЯ И ДЕКОМПРЕССИИ ИЗОБРАЖЕНИЙ ПАРАЛЛЕЛЬНО-КОВЕЙЕРНЫМ СПОСОБОМ НА РЕКОНФИГУРИРУЕМЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ

Фрактальные алгоритмы находят все большее количество областей применения – от компьютерной графики до моделирования сложных физических процессов, но для их программной реализации требуются значительные вычислительные мощности. Фрактальное сжатие изображений отличается высокой степенью компрессии данных при хорошем качестве восстановленного изображения. Целью данной работы является повышение производительности реконфигурируемых вычислительных систем (РВС) при реализации фрактального сжатия и декомпрессии изображений. В работе описаны разработанные методы фрактального сжатия и последующей декомпрессии изображений, реализованные параллельно-конвейерным способом для РВС. Основная идея параллельной реализации фрактального сжатия изображений сводится к параллельному выполнению попарного сравнения доменных и ранговых блоков. Для достижения наилучшей производительности необходимо одновременно сравнивать максимальное количество пар. При практической реализации фрактального сжатия изображений на РВС учитываются такие критические ресурсы, как количество входных каналов и количество логических ячеек ПЛИС. Для задачи фрактального сжатия изображения критическим ресурсом являются каналы данных, поэтому параллельная организация вычислений заменяется параллельно-конвейерной после выполнения редукцию производительности параллельной вычислительной структуры. Подача каждого операнда в вычислительную структуру осуществляется последовательно (побитово), что экономит вычислительный ресурс и уменьшает простой оборудования. Для хранения коэффициентов системы итерируемых функций, кодирующих изображение, введена структура данных, задающая отношения между номерами ранговых и доменных блоков и соответствующими параметрами. Для удобства последующей декомпрессии элементы массива, кодирующего сжатое изображение, упорядочены по номерам ранговых блоков, что позволяет избежать двойной косвенной адресации в вычислительной структуре. Представленный подход позволяет масштабировать параллельно-конвейерную программу на любое количество программируемых логических интегральных схем (ПЛИС). Практическая реализация фрактального сжатия изображений, выполненная на реконфигурируемом компьютере Терциус-2, содержащем восемь ПЛИС, обеспечивает ускорение в 15000 раз по сравнению с универсальным многоядерным процессором и в 18–25 раз по сравнению с существующими решениями для ПЛИС. Реализация декомпрессии изображения на реконфигурируемом компьютере показывает ускорение в 380 раз по сравнению с аналогичной реализацией для многоядерного универсального процессора.

Фракталы; фрактальное сжатие изображений; ПЛИС; реконфигурируемые вычислительные системы.

M.D. Chekina

THE PARALLEL-PIPELINED IMPLEMENTATION OF THE FRACTAL IMAGE COMPRESSION AND DECOMPRESSION FOR RECONFIGURABLE COMPUTING SYSTEMS

Fractal algorithms find an increasing number of areas of application - from computer graphics to modeling complex physical processes, but their software implementation requires significant computing power. Fractal image compression is characterized by a high degree of data compression with good quality of the reconstructed image. The aim of this work is to improve the performance of reconfigurable computing systems (RCS) when implementing fractal compression and decompression of images. The paper describes the developed methods of fractal compression and subsequent decompression of images, implemented in a parallel-pipeline method for RCS.

The main idea of parallel implementation of fractal image compression is reduced to parallel execution of pairwise comparison of domain and rank blocks. For best performance, the maximum number of pairs must be compared simultaneously. In the practical implementation of fractal image compression on the DCS, such critical resources as the number of input channels and the number of FPGA logical cells are taken into account. For the problem of fractal image compression, data channels are a critical resource; therefore, the parallel organization of computations is replaced by parallel-pipeline, after the performance reduction of the parallel computational structure is performed. Each operand goes into the computational structure sequentially (bit by bit) to save computational resources and reduce equipment downtime. To store the coefficients of the iterated functions system encoding the image, a data structure has been introduced that specifies the relation between the numbers of rank and domain blocks and the corresponding parameters. For the convenience of subsequent decompression, the elements of the array encoding the compressed image are ordered by the numbers of the rank blocks, which avoids double indirect addressing in the computational structure. Applying this approach for parallel-pipeline programs allows scaling computing structure to plurality programmable logic arrays (FPGAs). A practical implementation performed on a reconfigurable computer Tertius-2 containing eight FPGAs provides an acceleration of 15000 times compared to a universal multi-core processor and 18–25 times compared to existing solutions for FPGAs. The implementation of image decompression on a reconfigurable computer shows an acceleration of 380 times in comparison with the similar implementation for a multi-core general-purpose processor.

Fractals; fractal image compression; FPGA; reconfigurable computing systems.

Введение. Фрактальные алгоритмы находят применение в самых разнообразных классах задач: графике, сжатии изображений, анализе финансовых рынков, моделировании сложных физических процессов. Для эффективной программной реализации фрактальных алгоритмов требуются значительные вычислительные мощности. Стандартные методы и средства для многопроцессорных систем не обеспечивают удовлетворительную эффективность, т.к. специфика фрактальных алгоритмов подразумевает обмен большими массивами данных между вычислительными устройствами, из-за чего невозможно эффективное масштабирование параллельных программ. Реализации фрактальных алгоритмов для многопроцессорных систем на основе универсальных процессоров или графических ускорителей обеспечивают меньшую производительность, чем системы, построенные на основе ПЛИС [1].

Фрактальное сжатие изображений является одной из самых востребованных задач на основе фрактальных методов. В его основе лежит обнаружение самоподобных участков в изображении. Впервые возможность применения систем итерируемых функций (СИФ) к проблеме сжатия изображения была исследована М. Барнсли и А. Слоуном [2]. Жакен А. представил метод фрактального кодирования, использующий доменные и ранговые блоки квадратной формы, покрывающие всё изображение. Этот подход стал основой для большинства методов фрактального кодирования и был усовершенствован Ю. Фишером [3], и рядом других исследователей

Приложения, реализующие фрактальное сжатие изображений для традиционных многопроцессорных систем, обладают рядом недостатков. Вычислительные системы на универсальных процессорах требуют больших накладных расходов на передачу данных между процессами, а также их синхронизацию, что приводит к простою части вычислительного оборудования [4]. Графические ускорители обладают большой вычислительной мощностью, но не могут функционировать как самостоятельные устройства. При выполнении фрактального алгоритма на системе, основным вычислителем которой является графический ускоритель, требуется постоянный обмен данными с центральным процессором и оперативной памятью. Пропускная способность этих каналов существенно ниже, чем частота работы графического ускорителя, что приводит к снижению производительности системы [5–8].

Существующие реализации фрактального сжатия изображений разработаны для реконфигурируемых акселераторов на основе только одной ПЛИС, которая не может функционировать как самостоятельное устройство, и требуют для работы наличия универсального процессора. Кроме того, все известные реализации не предусматривают возможность масштабирования алгоритмов на многокристальную систему [9–13].

Помимо описанных выше недостатков, как правило, параллельно реализуют только сжатие изображения, оставляя его декомпрессию для выполнения на персональном компьютере. Это объясняется асимметричностью алгоритма фрактального сжатия изображения, компрессия данных занимает в десятки раз больше времени, чем их декомпрессия, но для изображений с высоким разрешением обратная операция также будет происходить длительное время.

Реконфигурируемые вычислительные системы (РВС), объединяющие в единый ресурс множество ПЛИС [14], обладают высокой реальной производительностью и имеют большой потенциал для реализации фрактальных алгоритмов, но отсутствует инструментарий для работы с самоподобными структурами на РВС.

Метод реализации фрактального сжатия изображений для РВС параллельно-конвейерным способом. Фрактальная архивация основана на том, что с помощью трехмерных аффинных преобразований [15] изображение представляется в компактной форме. При этом изображение одновременно разбивается на два множества: неперекрывающихся ранговых блоков и перекрывающихся доменных блоков. Ранговые блоки могут быть одинакового размера, хотя более эффективно использовать адаптивное разбиение с переменным размером блоков. Это дает возможность плотно заполнять ранговыми блоками меньшего размера части изображения, содержащие мелкие детали.

После разбиения для каждого рангового блока перебирают все доменные блоки для поиска максимального соответствия между ними. Домены сжимают до размеров рангового блока и выполняют операции изменения ориентации. Далее вычисляют оптимальные значения коэффициентов преобразования для наилучшего соответствия ранговому блоку. Сначала вычисляется сдвиг по яркости:

$$v = \left(\sum_{i=1}^{m} \sum_{j=1}^{m} r_{ij} - u \sum_{i=1}^{m} \sum_{j=1}^{m} d_{ij} \right) / m^2 ,$$

здесь m – длина стороны рангового и усредненного доменного блока в пикселях, d_{ij} – значение яркости пикселя усредненного доменного блока, r_{ij} – значение яркости пикселя рангового блока, u – коэффициент сжатия яркости.

Затем для определения соответствия двух блоков между ними находят расстояние

$$E(R,D) = \sum_{i=1}^{m} \sum_{j=1}^{m} \left(u d_{ij} + v - r_{ij} \right)^2 .$$

Если для некоторого домена после применения к нему преобразования при соответствующих коэффициентов системы итерируемых функций (СИФ) его значение E(R,D) не превышает заданной допустимой погрешности, рассматриваемый ранговый блок считают покрытым данным доменным блоком – производится сохранение подходящих коэффициентов СИФ, и происходит переход к обработке следующего ранга.

Основная сложность фрактального сжатия заключается в том, что для нахождения соответствующих доменных блоков требуется полный перебор всех возможных пар, что требует больших затрат времени и вычислительного ресурса.

Общая идея параллельной реализации фрактального сжатия изображений сводится к параллельному выполнению попарного сравнения доменных и ранговых блоков.

Согласно представленному алгоритму, для выполнения фрактального сжатия изображения необходимо выполнить четыре базовых операции над всеми парами ранговых и доменных блоков: сжатие доменного блока до размеров рангового (W), поворот сжатого блока (F_i, где i=0..7), сдвиг по яркости (L), вычисление расстояния между преобразованным доменным блоком и ранговым (E).

При практической реализации фрактального сжатия изображений на РВС следует учитывать критические ресурсы: количество входных каналов и количество логических ячеек ПЛИС. Реализация параллельно-конвейерной программы проводилась на реконфигурируемом компьютере Терциус-2 [16], выпускаемом НИЦ суперЭВМ и нейрокомпьютеров, содержащем 8 ПЛИС XCKU095, каждая из которых обладает ресурсом в 1176000 LUT, и универсальный процессор Intel Core i5-6600K Skylake. Данный вычислительный блок содержит 16 блоков распределенной памяти с 32-битными каналами.

Для задачи фрактального сжатия изображения критическим ресурсом являются каналы данных, поэтому необходимо перейти от параллельной организации вычислений к параллельно-конвейерной, выполнив редукцию производительности параллельной вычислительной структуры [17]. Коэффициент редукции производительности параллельной вычислительной структуры по каналам для задачи фрактального сжатия изображений можно выразить формулой

$$K_c = \frac{BN_{im}n_D}{n_R C},$$

здесь N_{im} – количество пикселей в исходном изображении, n_D – количество пикселей в доменном блоке, n_R – количество пикселей в ранговом блоке, C – количество доступных каналов, B – количество бит, кодирующих один пиксель изображения.

При реализации задачи на реконфигурируемом компьютере Терциус-2 с пословной подачей элементов доменных блоков значение коэффициента редукции составит K_c= 62500.

При всех преимуществах параллельной подачи всех элементов массива такой способ имеет ряд недостатков: параллельная подача всего доменного блока удобна только при фиксированном размере всех доменных блоков, если использовалось динамическое разбиение изображения, например, с помощью четверичного дерева, то может возникнуть потребность также и в динамическом выборе ширины канала для входных данных.

При пословной подаче элементов доменных блоков возникнет простой оборудования из-за исключения из работы тех доменов, для которых нашлось соответствие с ранговыми, что диктует необходимость изменить способ подачи данных. Если поразрядно (побитово) подавать значения яркости каждого из пикселей доменного блока с общим количеством точек 64, четыре одновременно подаваемых доменных блока займут 256 каналов.

При осуществлении последовательной (побитовой) подачи элементов массива пикселей коэффициент редукции производительности вычислительной структуры составит K_c= 1500000.

Операции сжатия и поворота можно провести один раз перед началом конвейера, подавая для дальнейших вычислений модифицированные доменные блоки D_i" (рис. 1).

В начале каждого конвейера будут проведены операции сжатия и смены ориентации, для каждой модификации вычислены сдвиг по яркости и расстояние.

Выполняя побитовую последовательную подачу данных для каждого доменного блока, мы сможем одновременно осуществлять обработку 256 доменных блоков.

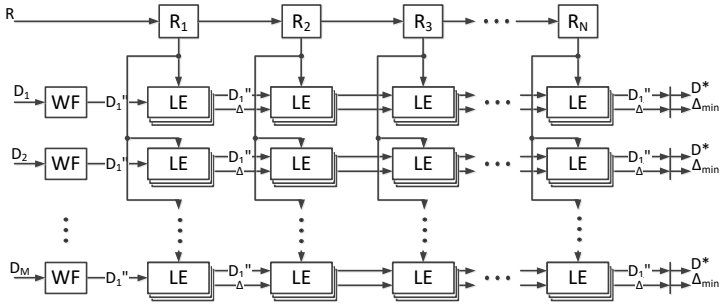

Рис. 1. Вычислительная структура с выносом блоков сжатия и поворота в начало конвейера

При такой подаче данных появляется возможность использовать одноразрядные арифметические блоки, не требующие больших затрат оборудования. При таких параметрах блок сжатия-поворота WF будет занимать 15 LUT, а 8 блоков LE займут 2120 LUT. Ресурс ПЛИС Kintex UltraScale XCKU095 – 1176000 LUT, на одном кристалле можно будет разместить не более 550 подобных вычислительных единиц.

Если осуществляется параллельная подача каждого восьмибитного слова, то возможна одновременная подача только 32-х доменных блоков. Арифметические блоки для работы с восьмибитными словами требуют большего ресурса (WF – 105 LUT, LE – 12904 LUT), поэтому на аналогичной ПЛИС возможно разместить не более 90 вычислительных единиц.

Доменные блоки могут быть пропущены через конвейер несколько раз. После нахождения совпадения домен исключается из потока. В этом случае при пословной подаче операндов будет простаивать большая часть оборудования, чем при использовании последовательной побитовой подачи.

Для хранения коэффициентов СИФ предлагается структура данных $X=(N_r,N_d,i,v)$, где N_r – номер рангового блока, N_d – номер доменного блока, i – поворотный коэффициент, v – сдвиг по яркости. В одном элементе такой структуры будет храниться информация о соответствии между конкретными ранговым и доменным блоками, что значительно облегчит впоследствии восстановление изображения по коэффициентам СИФ. Процесс формирования структуры X показан на рис. 2.

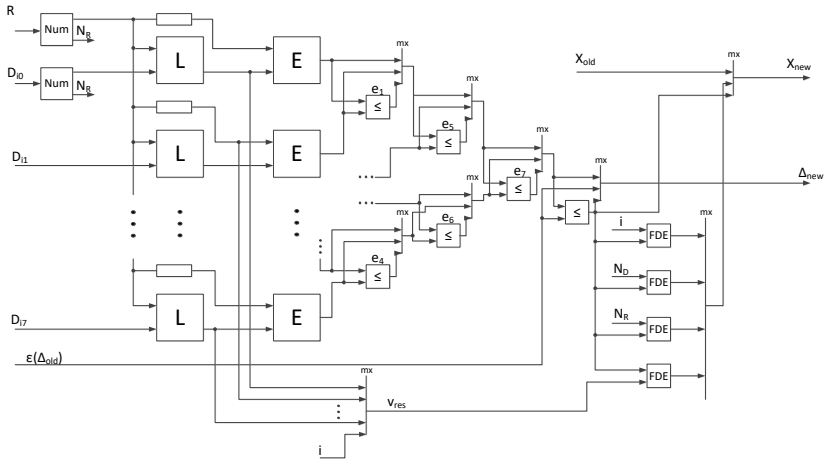

Рис. 2. Вычислительная структура, формирующая массив X, содержащий коэффициенты СИФ

Структура X ставит в соответствие друг другу массивы данных ранговых и доменных блоков. При этом соответствие не является биективным: одному ранговому блоку может соответствовать несколько доменных блоков, но не наоборот.

Метод параллельной реализации для РВС декомпрессии изображения сжатого фрактальным способом. Декомпрессия изображения, сжатого фрактальным способом, производится за счет итерационного применения найденных ранее коэффициентов системы итерируемых функций к произвольному начальному изображению. В качестве начального может быть взято любое изображение, в соответствии с теоремой о сжимающем отображении [18] итерационный процесс будет сходиться к неподвижному изображению. Изображение из коэффициентов СИФ восстанавливается следующим образом:

1. Создаются два произвольных изображения одинакового размера A и B.
2. На изображение A накладывается сетка ранговых блоков, аналогичная той, на которую было разбито сжатое изображение. С изображением B проводится аналогичная операция, но на него накладывается сетка из доменных блоков.
3. Для каждого доменного блока области B проводится соответствующее аффинное преобразование ранговых областей изображения A, описанное коэффициентами СИФ. Результат помещается в область B. После преобразования получается совершенно новое изображение.
4. Данные из области B копируются в область A.
5. Шаги 3 и 4 повторяются до тех пор, пока изображения A и B не станут неразличимыми.

Номера соответствующих друг другу ранговых и доменных блоков содержатся вместе с коэффициентами СИФ в структурах X, множество которых подается на вход процедуры по восстановлению изображения. Для дальнейшей работы необходимо выделить некоторую рабочую область в памяти ПЛИС или внешней памяти, где будет производиться итерационная декомпрессия изображения.

Предполагается, что входные данные X для процедуры декомпрессии, полученные в ходе процедуры сжатия изображения, не упорядочены, поэтому возникает проблема обращения к конкретной области памяти устройства, а также наложения сеток доменных и ранговых блоков на рабочую область. Для решения этой проблемы, как правило, используется косвенная адресация: адреса нужных ранговых и доменных блоков вычисляются, исходя из их номеров, содержащихся во входных данных X.

Традиционная реализация процедуры декомпрессии не может быть реализована на РВС в силу невозможности одновременной записи и чтения из нее потока доменных и ранговых блоков. Информационные потоки доменных и ранговых блоков будут постоянно прерываться для осуществления процедуры записи в память результата. На рис. 3 показано, что при традиционной реализации процедуры декомпрессии блоки RAM, содержащие доменные и ранговые блоки, должны выступать как приемник и как источник данных.

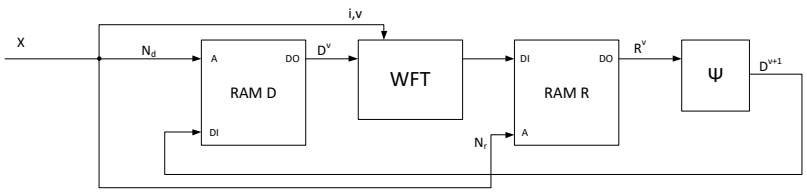

Рис. 3. Тривиальная реализация процедуры декомпрессии изображения

Так, из RAM D элементы доменных блоков поступают в вычислительную структуру WFT (рис. 4), осуществляющую аффинные преобразования.

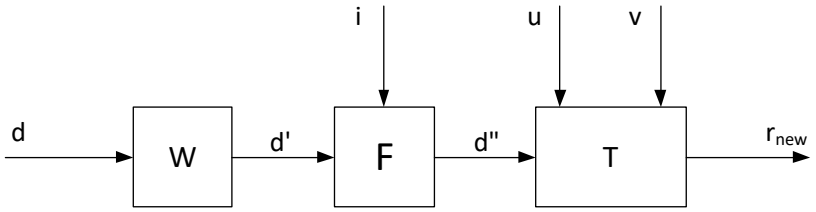

Рис. 4. Составляющие блока WFT

Здесь блоки W и F соответствуют операциям сжатия и поворота. На вход блока F подается один из коэффициентов СИФ – номер поворота доменного блока. В блоке T осуществляется обратный сдвиг по яркости, структура которого показана на рис. 5.

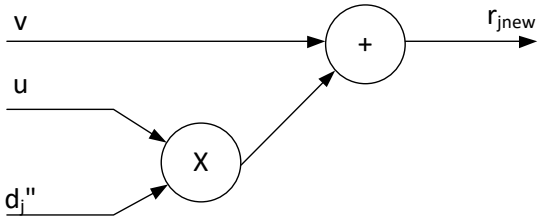

Рис. 5. Вычислительная структура блока T

Преобразованные данные записываются в RAM R, которое, в свою очередь, является источником данных для получения $D^{\nu+1}$ – элементов доменного блока для нового временного слоя итерационного процесса.

Через вычислительную структуру будет проходить не поток операндов, а единичные операнды. Это, в свою очередь, приводит к увеличению скважности и падению реальной производительности системы. Подобная вычислительная структура практически не поддается распараллеливанию из-за невозможности параллельного доступа к одной памяти вследствие косвенной адресации.

Для такой структуры характерна высокая скважность порядка 300 тактов, и скорость обработки данных снижается в 300 раз. Для ее снижения вычислительную структуру, производящую декомпрессию изображения, можно разделить на два кадра, первый из которых реализует функцию поиска значений рангового блока на основе предыдущих значений доменного $R^{\nu} = F\left(X, D^{\nu-1}\right)$, а второй кадр обеспечивает обновление доменных блоков на основе полученных ранговых $D^{\nu} = \Psi\left(X, R^{\nu}\right)$, здесь ν – номер временного слоя в итерационном процессе. При реализации такого подхода по две памяти для ранговых и доменных блоков используются либо как приемник, либо как источник данных. Такая структура соответствует классической формулировке алгоритма восстановления изображения (рис. 6).

Номер доменного блока N_d, содержащийся в X, формирует адрес чтения из памяти RAM D, и элементы доменных блоков из нее поступают в функцию WFT вместе с коэффициентами (i,v) СИФ, также содержащимися в X.

После преобразования данные записываются в память RAM R по адресу, сформированному согласно поступившему на вход номеру рангового блока N_r.

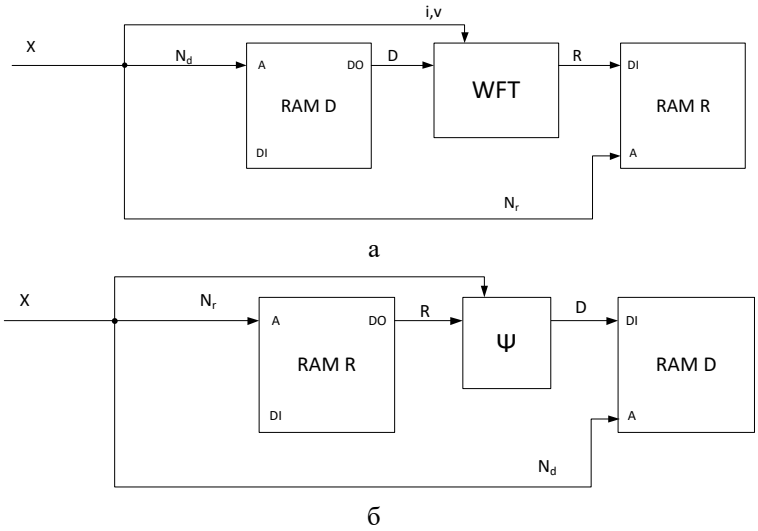

Рис. 6. Вычислительная структура с двумя проходами а) прямой ход – поиск элементов ранговых блоков, б) обратный ход – перенос элементов ранговых блоков в доменные

Следует отметить, что в данном варианте применяется двойная косвенная адресация для блоков RAM, содержащих ранговые и доменные блоки восстанавливаемого изображения, поэтому распараллелить вычислительную структуру по данным становится практически невозможно из-за неизбежности одновременных обращений к одному каналу памяти.

Однако можно предварительно выполнить сортировку элементов массива X, представляющих собой набор $\left(N_r, N_d, i, v\right)$, по номерам ранговых блоков N_r, упростив исходный набор из четырех параметров до трех $X_R = \left(N_d, i, v\right)$. В этом случае мы перейдем от двойной косвенной адресации к одинарной косвенной адресации. Это позволит улучшить распараллеливание по данным. На рис. 7 показана вычислительная структура, использующая в качестве входных данных отсортированные X_R.

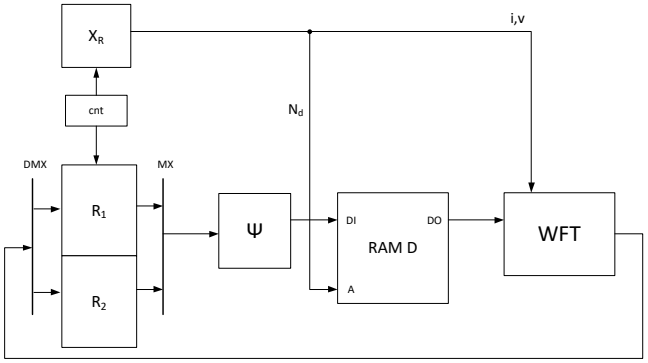

Рис. 7. Использование упорядоченных значений X_R

На вход вычислительной структуры поступают элементы X_R, которые синхронизированы с ранговыми блоками, хранящимися во внешней памяти устройства. Номер доменного блока N_d формирует адрес чтения в двухпортовой памяти RAM D, и элементы соответствующего доменного блока вычитываются для преобразования функцией WFT. После выполнения всех операций преобразованные данные записываются во внешнюю память. Использование двухпортовой памяти позволяет выполнять независимо операции записи и чтения данных. Содержимое обновленных ранговых блоков вычитываются из внешней памяти и поступает на вход RAM D для осуществления следующих итераций.

Подобный подход позволяет избежать двойной косвенной адресации, которая использовалась для ранговых и доменных блоков, перейдя к одинарной – только для доменных блоков. Поскольку ранее было проведено упорядочивание элементов X, поступающих на выход схемы, то новые ранговые блоки будут появляться в том порядке, в котором они расположены внутри изображения, что избавляет от необходимости использовать для них косвенную адресацию, оставляя ее применение только для доменных блоков.

Очевидным способом сортировки X является сортировка массива в потоке, преобразовывая структуры вида $X = (N_r, N_d, i, v)$ к виду $X_R = (N_d, i, v)$, т.е. номер элемента нового массива X будет соответствовать номеру рангового блока. Но такой подход потребует дополнительных операций и приведет к лишним временным затратам, поэтому целесообразно осуществлять сортировку непосредственно при формировании X во время кодирования изображения.

Выше описана вычислительная структура X, формирующая соответствие между коэффициентами СИФ и номерами ранговых и доменных блоков. При этом элементы X не были упорядочены по какому-либо признаку. Поскольку ранговые блоки поступают в вычислительную схему по порядку, то возможно формирование элементов X соответственно номерам ранговых блоков, тогда номер каждого из элементов X будет идентичен номеру рангового блока, который был использован при формировании данного элемента.

За счет использования КРП возможно организовать до 256 потоков данных, осуществляя их подачу побитно. При этом для взаимодействия с памятью, содержащей в себе доменные блоки, придется осуществлять буферизацию параллельно идущих потоков данных. На рис. 8 показана вычислительная структура для нескольких параллельно обрабатываемых ранговых блоков. Из X, поступающего на вход, выделяются номера доменных блоков N_d. Они управляют выбором доменного блока, поступающего в блоки WFT для дальнейших преобразований. Также из X на входы блоков WFT поступают коэффициенты i и v. После преобразования данные записываются в соответствующие ранговые блоки. Затем порядок чтения и записи меняется, и теперь память, содержащая обновленные данные ранговых блоков, становится источником. Из нее данные поступают на вход памяти, хранящей доменные блоки для их обновления.

Процедура восстановления изображения, сжатого фрактальным способом, с трудом поддается распараллеливанию на РВС. Предложенный выше подход не является оптимальным решением этой задачи, поэтому поиск максимально оптимального способа ее распараллеливания может стать предметом дальнейших исследований.

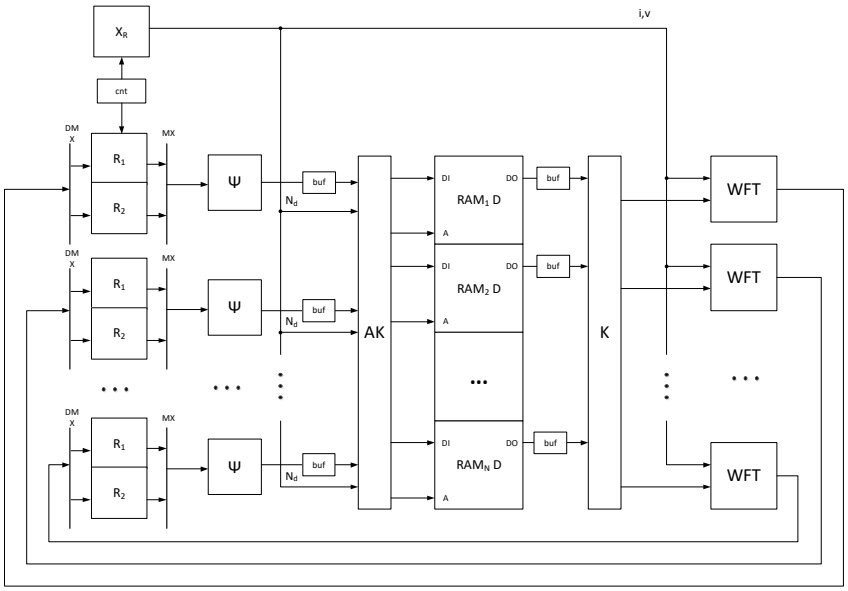

Рис. 8. Распараллеливание процедуры декомпрессии изображения

Оценка производительности фрактального сжатия и декомпрессии изображений на РВС. Время работы алгоритма фрактального сжатия изображений для РВС Терциус-2 можно оценить по формуле

$$T = \frac{1}{\mu U}\left(Ld + \left(Lt_{WF} + QLt_{LE} \right) N_D \frac{N_R}{QCF} \right),$$

где Lt_{WF} – латентность блока сжатия и поворота, Lt_{LE} – латентность блоков, определяющих сдвиг по яркости и расстояние между доменным и ранговым блоками, N_D - количество доменных блоков, N_R – количество ранговых блоков, F – количество ПЛИС, Ld – количество тактов, необходимое для загрузки одного доменного блока, Q – количество блоков LE в конвейере, μ – частота работы ПЛИС, U – коэффициент использования оборудования, показывающий соотношение времени полной загрузки вычислительного ресурса к общему времени работы.

При прохождении каждого доменного блока через вычислительную структуру он будет исключён из конвейера при совпадении с заданной погрешностью с ранговым блоком. Вероятность данного события оценивается как 0,5, что соответствует значению коэффициента использования оборудования U=0,5. Если доменный блок проходит через вычислительную структуру несколько раз для разных групп ранговых блоков, то вероятность того, что он будет исключён, и коэффициент простоя оборудования примет значение в интервале от 0,5 до 1, что в среднем даст значение U=0,75.

Для рассматриваемого примера время сжатия 4-мегапиксельного изображения на реконфигурируемом компьютере Терциус-2 при пословной подаче восьмибитных операндов массива составит около 7,5 секунд, а при последовательной побитовой подаче данных для той же задачи – приблизительно 1,5 секунды, за счет многократного увеличения числа блоков конвейерной обработки.

Фрактальное сжатие аналогичного изображения задачи на универсальном 4-ядерном процессоре Intel Core i7-4770 с тактовой частотой 3,50 GHz будет выполняться более шести часов (23000 секунд). Реализация фрактального сжатия

изображений для реконфигурируемого компьютера показывает ускорение в 15000 раз по сравнению с аналогичной реализацией для многоядерного универсального процессора.

При проведении сравнения предложенного подхода с существующими однокристальными реализациями фрактального сжатия [1, 5-9] ускорение составит 18-25 раз при тех же параметрах.

Выполним оценку времени, которое займет декомпрессия цветного четырехмегапиксельного изображения в формате RGB, разбитого на 250 000 квадратных ранговых блоков со стороной 4 пикселя. Время работы можно оценить по формуле

$$T = 1,3t\left(Ld + \frac{3LtN_R I}{CF} \right),$$

где t – величина обратная частоте работы ПЛИС, Ld – количество тактов, необходимое для загрузки одной порции входных данных, N_R – количество ранговых блоков, Lt – латентность вычислительной схемы, I – количество итераций, требуемых для восстановления изображения, C – количество каналов для подачи данных, F – количество задействованных ПЛИС. Для примера, указанного выше, оценочное время работы на реконфигурируемом компьютере Терциус-2 составит примерно 0,003 секунды, тогда как время выполнения этой же задачи на универсальном четырехъядерном процессоре Intel Core i7-4770 составит примерно 1,15 секунд. Таким образом, реализация декомпрессии изображения на РВС показывает ускорение в 380 раз по сравнению с аналогичной реализацией для многоядерного универсального процессора.

Заключение. Описанный выше подход к реализации фрактального сжатия изображений позволяет получать параллельно-конвейерные программы для РВС различных конфигураций, а также дает большие возможности для масштабирования алгоритма, позволяя наращивать производительность вычислительной системы с увеличением количества кристаллов ПЛИС.

Данный подход можно использовать для реализации на реконфигурируемых вычислительных системах других фрактальных задач – построения фрактальных ландшафтов, фрактального анализа природных систем и финансовых рынков.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Doris Chen*. Deshanand Singh Fractal Video Compression in OpenCL: An Evaluation of CPUs, GPUs, and FPGAs as Acceleration Platforms // 2013 18th Asia and South Pacific Design Automation Conference. – P. 297-304.
2. *Barnsley M., Hurd L.P.* Fractal Image Compression. – Wellsley Massachusetts: A.K. Peters Ltd, 1993. – 224 p.
3. *Fisher Y.* Fractal Image Compression: Theory and Application. – Springer-Verlag, New York, 1995.
4. *Бойченко И.В., Кулбаев С.С., Немеров А.А., Голенков В.В.* Эксперимент по фрактальному сжатию rgb изображений на вычислительном кластере // Известия Томского политехнического университета. – 2012. – Т. 321, № 5. – С. 87-92.
5. *Munesh Singh Chauhan, Ashish Negi, Prashant Singh Rana.* Fractal Image Compression using Dynamically Pipelined GPU Clusters // Proceedings of the Second International Conference on Soft Computing for Problem Solving (SocProS, 2012), December 28-30, 2012.
6. *Mohammed Ismail B., Eswara Reddy B., Bhaskara Reddy T*. Cuckoo inspired fast search algorithm for fractal image encoding // Journal of King Saud University – Computer and Information Sciences, 2016.
7. *Md. Enamul Haque, Abdullah Al Kaisan, Mahmudur R Saniat, and Aminur Rahman*. GPU Accelerated Fractal Image Compression for Medical Imaging in Parallel Computing Platform // arXiv:1404.0774v1 [cs.DC] 3 Apr 2014.

8. *Munesh Singh Chauhan, Ashish Negi.* Fractals Image Rendering and Compression using GPUs // International Journal of Digital Information and Wireless Communications (IJDIWC) 2(1): 1-6 The Society of Digital Information and Wireless Communications, 2012. – ISSN 2225-658X.
9. *A-M.H.Y. Saad, M.Z.* Abdullah High-speed implementation of fractal image compression in low cost FPGA, Microprocessors and Microsystems 47. August 2016. – Doi: 10.1016/j.micpro.2016.08.004.
10. *A-M.H.Y. Saad, M.Z.* Abdullah High-Speed Fractal Image Compression Featuring Deep Data Pipelining Strategy // IEEE Access PP(99):1-1 · November 2018. – Doi: 10.1109/ACCESS.2018.2880480.
11. *Son T.N., Hung O.M., Xuan D.T., Tran V.L., Dzung N.T., Hoang T.M*. Implementation of Fractal image compression on FPGA // 4th International Conference on Communications and Electronics ICCE 2012, online IEEExplorer, Hue City, Vietnam. 1-3 Aug. 2012. – P. 339-344.
12. *Padmavati S.* Vaibhav Meshram FPGA Implementation for Fractal Quadtree Image Compression International Journal of Computer Sciences and Engineering. – Oct. 2018. – Vol. 6, Issue-10.
13. *Thai Nam Son Tran V Long, Hoang Manh Thang, Nguyen Tien Dzung*. Efficient implementation of a fractal color image compression on FPGA // 2013 International Conference of Soft Computing and Pattern Recognition (SoCPaR). – 2013. – Doi: 10.1109/SOCPAR.2013.7054124.
14. Thai Nam Son Thang Manh Hoang, Nguyen Tien Dzung Nguyen Hoang Giang Fast FPGA Implementation of YUV-based Fractal Image Compression // 2014 IEEE Fifth International Conference on Communications and Electronics (ICCE). – 2014. – Doi: 10.1109/CCE.2014.6916745.
15. *Гузик В.Ф., Каляев И.А., Левин И.И.* Реконфигурируемые вычислительные системы / под общ. ред. И.А. Каляева. – Ростов-на-Дону: Изд-во ЮФУ, 2016. – 472 с.
16. *Беклемишев Д.В.* Курс аналитической геометрии и линейной алгебры. – М.: Высш. шк., 1998. – 320 с.
17. *Левин И.И., Дордопуло А.И., Сорокин Д.А., Каляев З.В., Доронченко Ю.И.* Реконфигурируемые компьютеры на основе плис Xilinx Virtex Ultrascale // В сб.: Параллельные вычислительные технологии (ПаВТ'2019). Короткие статьи и описания плакатов XIII Международной научной конференции. – 2019. – С. 288-298.
18. *Дордопуло А.И., Левин И.И.* Методы редукции вычислений для программирования гибридных реконфигурируемых вычислительных систем // XII мультиконференция по проблемам управления (МКПУ-2019): Матер. конференции: в 4 т. – 2019. – С. 78-82.
19. *Колмогоров А.Н., Фомин С.В.* Элементы теории функций и функционального анализа. – 4-е изд. – М.: Наука, 1976. – 544 с.
20. *Левин И.И., Пелипец А.В.* Эффективная реализация распараллеливания на реконфигурируемых системах // Вестник компьютерных и информационных технологий. – 2018. – № 8. – С. 11-16.

REFERENCES

1. *Doris Chen*. Deshanand Singh Fractal Video Compression in OpenCL: An Evaluation of CPUs, GPUs, and FPGAs as Acceleration Platforms, *2013 18th Asia and South Pacific Design Automation Conference*, pp. 297-304.
2. *Barnsley M., Hurd L.P.* Fractal Image Compression. Wellsley Massachusetts: A.K. Peters Ltd, 1993, 224 p.
3. *Fisher Y.* Fractal Image Compression: Theory and Application. Springer-Verlag, New York, 1995.
4. *Boychenko I.V., Kulbaev S.S., Nemerov A.A., Golenkov V.V.* Eksperiment po fraktal'nomu szhatiyu rgb izobrazheniy na vychislitel'nom klastere [Experiment on fractal compression of rgb images on a computing cluster], *Izvestiya Tomskogo politekhni-eskogo universiteta* [Bulletin of the Tomsk Polytechnic University], 2012, Vol. 321, No. 5, pp. 87-92.
5. *Munesh Singh Chauhan, Ashish Negi, Prashant Singh Rana.* Fractal Image Compression using Dynamically Pipelined GPU Clusters, *Proceedings of the Second International Conference on Soft Computing for Problem Solving (SocProS, 2012), December 28-30, 2012.*
6. *Mohammed Ismail B., Eswara Reddy B., Bhaskara Reddy T*. Cuckoo inspired fast search algorithm for fractal image encoding, *Journal of King Saud University – Computer and Information Sciences, 2016*.
7. *Md. Enamul Haque, Abdullah Al Kaisan, Mahmudur R Saniat, and Aminur Rahman*. GPU Accelerated Fractal Image Compression for Medical Imaging in Parallel Computing Platform, *arXiv:1404.0774v1 [cs.DC] 3 Apr 2014*.

8. *Munesh Singh Chauhan, Ashish Negi.* Fractals Image Rendering and Compression using GPUs, *International Journal of Digital Information and Wireless Communications (IJDIWC) 2(1): 1-6 The Society of Digital Information and Wireless Communications, 2012*. ISSN 2225-658X.
9. *A-M.H.Y. Saad, M.Z.* Abdullah High-speed implementation of fractal image compression in low cost FPGA, Microprocessors and Microsystems 47. August 2016. Doi: 10.1016/j.micpro.2016.08.004.
10. *A-M.H.Y. Saad, M.Z.* Abdullah High-Speed Fractal Image Compression Featuring Deep Data Pipelining Strategy, *IEEE Access PP(99):1-1 · November 2018*. Doi: 10.1109/ACCESS.2018.2880480.
11. *Son T.N., Hung O.M., Xuan D.T., Tran V.L., Dzung N.T., Hoang T.M*. Implementation of Fractal image compression on FPGA, *4th International Conference on Communications and Electronics ICCE 2012, online IEEExplorer, Hue City, Vietnam. 1-3 Aug. 2012*, pp. 339-344.
12. *Padmavati S.* Vaibhav Meshram FPGA Implementation for Fractal Quadtree Image Compression, *International Journal of Computer Sciences and Engineering*, Oct. 2018, Vol. 6, Issue-10.
13. *Thai Nam Son Tran V Long, Hoang Manh Thang, Nguyen Tien Dzung*. Efficient implementation of a fractal color image compression on FPGA, *2013 International Conference of Soft Computing and Pattern Recognition (SoCPaR)*, 2013. Doi: 10.1109/SOCPAR.2013.7054124.
14. Thai Nam Son Thang Manh Hoang, Nguyen Tien Dzung Nguyen Hoang Giang Fast FPGA Implementation of YUV-based Fractal Image Compression, *2014 IEEE Fifth International Conference on Communications and Electronics (ICCE)*, 2014. Doi: 10.1109/CCE.2014.6916745.
15. *Guzik V.F., Kalyaev I.A., Levin I.I.* Rekonfiguriruemye vychislitel'nye sistemy [Reconfigurable computing systems], ed. by I.A. Kalyaeva. Rostov-on-Don: Izd-vo YUFU, 2016, 472 p.
16. *Beklemishev D.V.* Kurs analiticheskoy geometrii i lineynoy algebry [A course in analytic geometry and linear algebra]. Moscow: Vyssh. shk., 1998, 320 p.
17. *Levin I.I., Dordopulo A.I., Sorokin D.A., Kalyaev Z.V., Doronchenko Yu.I.* Rekonfiguriruemye komp'yutery na osnove plis Xilinx Virtex Ultrascale [Xilinx Virtex Ultrascale PLD-based reconfigurable computers], *V sb.: Parallel'nye vy-chislitel'nye tekhnologii (PaVT'2019). Korotkie stat'i i opisaniya plakatov XIII Mezh-dunarodnoy nauchnoy konferentsii* [Parallel computational technologies (PCT 2019)], 2019, pp. 288-298.
18. *Dordopulo A.I., Levin I.I.* Metody reduktsii vychisleniy dlya programmirovaniya gibridnykh rekonfiguriruemykh vychislitel'nykh sistem [Computation reduction methods for programming hybrid reconfigurable computing systems], XII mul'tikonferentsiya po problemam upravleniya (MKPU-2019): Mater. konferentsii [Materials of the XII multiconference on management problems 2019]: in 4 vol., 2019, pp. 78-82.
19. *Kolmogorov A.N., Fomin S.V.* Elementy teorii funktsiy i funktsional'nogo analiza [Elements of the theory of functions and functional analysis]. 4th ed. Moscow: Nauka, 1976, 544 p.
20. *Levin I.I., Pelipets A.V.* Effektivnaya realizatsiya rasparallelivaniya na rekonfiguriruemykh sistemakh [Efficient Parallel Execution on Reconfigurable Systems], *Vestnik komp'yuternykh i informatsionnykh tekhnologiy* [Herald of computer and information technologies], 2018, No. 8, pp. 11-16.

Статью рекомендовал к опубликованию д.т.н. Э.В. Мельник.

Чекина Мария Дмитриевна – Южный федеральный университет; e-mail: chekina @superevm.ru; 347900, г. Таганрог, пер. Итальянский, 106; тел.: +79281541526; аспирант.

Chekina Marija Dmitrievna – Southern Federal University; e-mail: chekina @superevm.ru; 106, Italyansky lane, Taganrog, 347900, Russia; phone: +79281541526; graduate student.