УДК 519.684.6; 004.272.2; 004.31 doi 10.26089/NumMet.v21r217

ВОЗМОЖНОСТИ МНОГОЯДЕРНЫХ ПРОЦЕССОРОВ MALT В ЗАДАЧАХ ОБРАБОТКИ ИЗОБРАЖЕНИЙ

Н. Г. Михеев[1], В. А. Антонюк[2], С. Г. Елизаров[3], Г. А. Лукьянченко[4]

В статье рассматриваются результаты экспериментальной оценки производительности и энергоэффективности многоядерных процессоров MALT в задачах обработки изображений на примере фильтрации изображения с помощью оператора Собеля. Измерения осуществлялись с использованием низкоуровневого эмулятора MALTemu, прототипа процессора в ПЛИС и экспериментальной СБИС модели MALT–Cv2 Rev1. Полученные результаты сравниваются с аналогичными результатами для процессоров общего назначения (последовательная реализация) и графических процессоров с поддержкой технологии CUDA.

Ключевые слова: многоядерный процессор, параллельные вычисления, обработка изображений, оператор Собеля, производительность, энергоэффективность, MALT, CUDA.

1. Введение. Одним из наиболее активно развивающихся классов задач, связанных с большими данными и высокопроизводительными вычислениями, являются задачи в области обработки изображений. Компьютерная обработка изображений — одна из ключевых технологий в самых различных областях: от беспилотных автомобилей и летательных аппаратов до систем безопасности [1–3]. В последнее время производительность отдельных процессорных ядер практически не растет. Общая производительность вычислительных систем повышается за счет увеличения количества процессорных ядер. Кроме того, для повышения производительности вычислений в задачах, связанных с обработкой изображений, все чаще используются специализированные ускорители разной степени универсальности: графические процессоры (GPU) [4], программируемые логические интегральные схемы (ПЛИС или FPGA) [5], тензорные процессоры (TPU) [6], специализированные микросхемы (ASIC) [7]. Таким образом, современные вычислительные системы становятся не только многоядерными, но и гетерогенными (неоднородными): для решения различных подзадач используются различные вычислительные устройства, и тип этих устройств зависит от решаемой задачи. В данной работе будут исследоваться возможности специализированного многоядерного процессора MALT [8] для решения задач обработки изображений.

Системы обработки изображений можно разделить на два типа: в системах первого типа обработка производится на конечном устройстве (к таким системам относятся, например, автономные летательные аппараты, поскольку результат обработки требуется получить за минимальное и гарантированное время), в системах второго типа изображение с конечного устройства передается в центр обработки данных для дальнейшего анализа. И в том и в другом случае важна не только производительность, но и энергоэффективность, т.е. производительность на единицу потребляемой электрической мощности. В первом случае энергоэффективность важна, поскольку от нее часто зависит время автономной работы подобных систем, во втором — согласно результатам исследований, затраты на оплату электроэнергии являются одной из основных статей расхода современных ЦОД и составляют до 35% от всех расходов [9]. О важности энергоэффективности современных вычислительных систем свидетельствует и растущий интерес к так называемым «зеленым» вычислениям (Green computing), в том числе с использованием графических процессоров [10] и других ускорителей [11].

Алгоритмы, используемые в современных системах компьютерного зрения, достаточно сложны, но, в большинстве случаев, их можно разложить на относительно простые базовые операции. Одним из наибо-

[1] Московский государственный университет им. М.В. Ломоносова, Физический факультет. ГСП–1, Ленинские горы, д. 1, стр. 2, 119991, Москва; аспирант, e-mail: ng.mikheev@physics.msu.ru

[2] Московский государственный университет им. М.В. Ломоносова, Физический факультет. ГСП–1, Ленинские горы, д. 1, стр. 2, 119991, Москва; доцент, e-mail: antonyuk@physics.msu.ru

[3] Московский государственный университет им. М.В. Ломоносова, Физический факультет. ГСП–1, Ленинские горы, д. 1, стр. 2, 119991, Москва; ст. науч. сотр., e-mail: elizarov@physics.msu.ru

[4] Московский государственный университет им. М.В. Ломоносова, Физический факультет. ГСП–1, Ленинские горы, д. 1, стр. 2, 119991, Москва; науч. сотр., e-mail: lukyanchenko@physics.msu.ru

лее популярных классов базовых операций являются сверточные операции, т.е. операции, предполагающие свертку изображения (двумерного массива) с некоторым ядром (другим двумерным массивом меньшего размера). Помимо достаточно большого класса фильтров, свертка изображений активно используется в сверточных нейронных сетях [12], которые получили в настоящее время широкое распространение. Поэтому эффективная реализация операции свертки важна для значительного количества используемых на сегодняшний день алгоритмов.

В рамках данной работы реализован оператор Собеля [13], основанный на свертке изображения с двумя целочисленными ядрами размера 3×3. Оператор Собеля используется для поиска краев объектов на изображениях и является составной частью некоторых более сложных фильтров, например фильтра Кэнни [14]. Кроме того, данный оператор достаточно часто применяется для сравнения производительности различных вычислительных систем [15, 16].

2. Архитектура и особенности программирования многоядерных процессоров MALT. Архитектура MALT (Manycore Architecture with Lightweight Threads) [8] предполагает наличие большого числа (до тысячи) процессорных ядер с упрощенным набором команд (RISC) на одном кристалле. В текущей версии используются процессорные ядра с набором команд, совместимым с архитектурой MicroBlaze [17]. Каждое из ядер в нормальном режиме работает на частоте 1 ГГц и имеет 8 Кбайт локальной памяти. Энергопотребление такого ядра в активном режиме составляет примерно 60 мВт.

Помимо локальной памяти, ядра могут взаимодействовать с внешней глобальной памятью стандарта DDR посредством общего «умного» контроллера памяти. Ядра обмениваются данными между собой и с внешней общей памятью по внутрикристальной сети SPBUS. Ядра делятся на три типа: вычислительные, коммуникационные и управляющие. Сеть представляет собой набор цепочек, образованных вычислительными ядрами (рис. 1). Корневые ядра в каждой цепочке являются коммуникационными: они отвечают за связь цепочек друг с другом, а также с контроллером внешней памяти. Отдельные коммуникационные ядра могут использоваться для обслуживания внешних интерфейсов, например Ethernet. Одно из ядер является управляющим: все приложения запускаются на нем и, при наличии соответствующих инструкций, запускают код на остальных ядрах. Управляющее и коммуникационные ядра при необходимости можно также задействовать в вычислениях, но это повлечет за собой ряд ограничений, связанных с обменом данными по внутрикристальной сети. В некоторых модификациях процессоров, например MALT–C, к вычислительным ядрам могут подключаться SIMD ускорители, но их рассмотрение выходит за рамки настоящей работы.

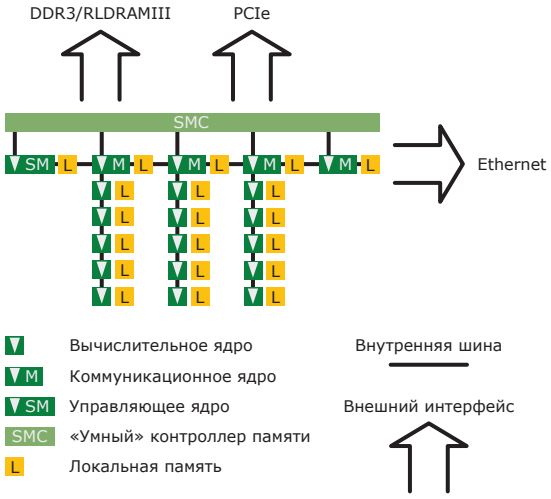

Рис. 1. Структура внутрикристальной сети 20-ядерного процессора MALT–Cv2

Разработка программ для процессоров MALT может осуществляться с использованием языков программирования C и C++ [18]. Общий алгоритм вычислений предполагает следующие этапы: запуск программы на управляющем ядре, запуск потоков на вычислительных ядрах (обычно запускается один и тот же код, обрабатывающий разные данные либо принимающий на вход разные значения параметров), копирование входных данных из общей памяти в локальную, непосредственно вычисления, обмен сообщениями с другими ядрами, копирование результатов вычислений в глобальную память, завершение потоков на вычислительных ядрах, завершение основной программы на управляющем ядре.

3. Реализация оператора Собеля. Оператор Собеля позволяет вычислить модуль градиента интенсивности изображения, что, в свою очередь, позволяет найти границы объектов на изображении. При вычислении оператора Собеля [13] сначала производится вычисление компонент G_x и G_y градиента изображения:

$$G_x = C(K_x, A), \qquad G_y = C(K_y, A),$$

где A — матрица интенсивностей изображения, $C(\cdot,\cdot)$ — корреляционная функция, K_x, K_y — матричное представление оператора градиента:

$$K_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}, \qquad K_y = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix}.$$

Результатом действия оператора Собеля в каждой точке изображения является норма вектора градиента. Норма может определяться по-разному. Одной из наиболее популярных является l_2–норма:

$$G_{l_2} = \sqrt{G_x^2 + G_y^2}.$$

Достаточно часто также используется и l_1–норма:

$$G_{l_1} = |G_x| + |G_y|.$$

Поскольку не все архитектуры оптимизированы для работы с числами с плавающей точкой (в частности, для вычисления квадратного корня), в рамках данной работы будет использоваться l_1–норма. Выходное изображение должно иметь тот же динамический диапазон, что и входное изображение, поэтому требуется осуществлять нормировку. Например, при кодировании интенсивности входного изображения целыми числами в диапазоне $[0, 255]$, наибольший градиент будет достигнут, если обрабатываемое окно имеет вид:

$$\begin{pmatrix} 0 & 0 & 255 \\ 0 & 0 & 255 \\ 255 & 255 & 255 \end{pmatrix}.$$

Для того, чтобы значение модуля градиента оказалось в диапазоне $[0, 255]$, требуется ввести нормировочный коэффициент $1/6$.

Нами разработаны три программные реализации оператора Собеля: для центральных процессоров общего назначения, для графических процессоров, поддерживающих технологию CUDA, и для многоядерных процессоров MALT.

Алгоритм для центральных процессоров подразумевает последовательное вычисление оператора для каждой точки изображения. Реализация для графических процессоров предполагает распределение изображения между вычислительными потоками по столбцам (рис. 2). Каждый поток последовательно вычисляет значение оператора для всех точек одного столбца изображения сверху вниз. При этом на каждом шаге блок нитей осуществляет чтение и запись последовательно расположенного набора точек в строке. Такой подход позволяет обеспечить эффективный обмен данными с глобальной памятью, поскольку изображение хранится в памяти построчно.

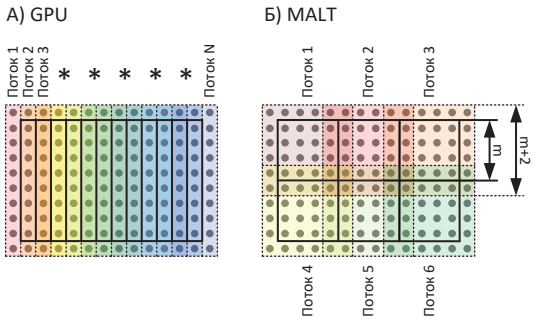

Рис. 2. Распределение точек изображения между потоками в случае использования графических процессоров (А) и в случае использования процессоров MALT (Б). Области, обрабатываемые разными потоками, выделены различными цветами. Исходное изображение разбивается на пересекающиеся области размером $(m+2) \times (m+2)$ точек (границы обозначены пунктирными линиями), выходное изображение собирается из областей размером $m \times m$ точек (сплошные линии)

В случае многоядерных процессоров MALT построчное чтение данных из глобальной памяти различными ядрами не приводит к повышению эффективности. В этом случае выгоднее разбивать изображения на квадратные области, копировать эти области в локальную память, производить вычисления в локальной памяти и копировать результат вычислений обратно в глобальную память. Использование квадратных областей размера $m \times m$ точек позволяет минимизировать количество граничных точек, для обработки которых требуются дополнительные данные. В ходе экспериментов изображение разбивалось на области размером 16×16 точек, которые распределялись между вычислительными ядрами. Поскольку для применения оператора Собеля к определенной

точке требуется информация и о соседних точках, между ядрами распределялись фрагменты исходного изображения размером $(m+2) \times (m+2)$ точек с пересечениями в граничных областях (рис. 2). Выходное изображение при этом собиралось из фрагментов $m \times m$ точек.

4. Экспериментальные оценки производительности и энергоэффективности. В рамках настоящей работы производилось исследование 20-ядерной модификации процессора MALT–Cv2 Rev1 (3 цепочки по 5 вычислительных ядер, 4 коммуникационных ядра и одно управляющее ядро). Для измерений использовались: 1) эмулятор MALTemu; 2) прототип процессора в ПЛИС; 3) экспериментальная СБИС. Эмулятор MALTemu [18] осуществляет низкоуровневую симуляцию работы основных компонентов процессора и предназначен главным образом для проверки переносимости ПО на реальные устройства. Среди прочего, эмулятор позволяет грубо оценить время выполнения кода, но, поскольку в эмуляторе не учитываются все аппаратные особенности, от полученных временны́х оценок не следует ждать полного соответствия аналогичным показателям для реальных устройств. Прототип процессора в ПЛИС, с точки зрения модели регистровых передач (RTL–модели), полностью эквивалентен экспериментальной СБИС. Благодаря этому пользовательские программы на языках C и C++ можно писать без учета специфики каждой из систем. Более того, на всех трех системах исполняются одни и те же бинарные файлы. Эксперименты с прототипом в ПЛИС производились с использованием системы на базе Xilinx Kintex Ultrascale XCKU115. Из-за высокой сложности модели для обеспечения хорошей стабильности работы частота тактирования прототипа в ПЛИС была ограничена значением 100 МГц. Исследования СБИС производились при частотах тактирования 100 МГц и 600 МГц. Компиляция программного кода осуществлялась средствами GCC 9.2.0 из состава MALT_SW 1.10 с уровнем оптимизации O2. Для оценки масштабируемости алгоритма измерения производительности осуществлялись для всех возможных значений количества используемых вычислительных ядер — от 1 до 15.

Результаты, полученные в случае вычислений на процессоре MALT, сравнивались с аналогичными результатами, полученными при использовании процессора общего назначения Intel Core i7–7820X (компилятор gcc 6.3.0, степень оптимизации O2), работающего на частоте 3,6 ГГц, а также с использованием графического процессора NVIDIA Tesla P4 (компилятор nvcc 10.0.130), работающего на частоте 1,59 ГГц. Объем глобальной памяти во всех трех случаях значительно превышал размер обрабатываемого изображения: максимальный объем глобальной памяти, доступной для процессоров MALT, определяется количеством адресов в 32-битном адресном пространстве; объем глобальной памяти в остальных системах также исчислялся гигабайтами. Для оценки энергоэффективности производились грубые измерения динамического энергопотребления — разности между мощностью, потребляемой устройством в режиме ожидания, и мощностью во время выполнения исследуемой задачи. Энергоэффективность оценивалась как отношение достигнутой скорости обработки кадров к потребляемой динамической мощности.

Для экспериментальных оценок производительности и энергоэффективности использовалось изображение в градациях серого размером 1024×768 точек с разрядностью 8 бит. Результат обработки сохранялся в том же формате. Под временем обработки подразумевалось время необходимое для обработки изображения, загруженного в глобальную оперативную память, и для сохранения результата обработки обратно в глобальную память. Время загрузки изображения из файла и сохранения результата обработки в файл при измерениях не учитывалось. В случае с обработкой на графическом процессоре под глобальной оперативной памятью понималась основная память ПК. Таким образом, время копирования данных из памяти ПК на видеокарту и обратно при измерениях учитывалось. Для оценки накладных расходов времена копирования данных между различными видами памяти измерялись отдельно от времени, затраченного непосредственно на вычисления. Времена выполнения копирований и вычислений, полученные по отдельности, суммировались и сравнивались с общим временем выполнения. На основе полученной разности оценивалась величина ошибки измерения времени.

Проверка результатов обработки во всех трех случаях производилась путем сравнения контрольных сумм.

Результаты измерений времени обработки изображений с использованием 15 ядер процессора MALT в сравнении с аналогичными измерениями для графического процессора и последовательной обработки с использованием процессора общего назначения приведены в таблице.

При работе процессора MALT на частоте тактирования 100 МГц низкоуровневый эмулятор MALTemu и прототип в ПЛИС демонстрируют близкие результаты на всех этапах обработки изображения. Различие абсолютных величин не превышает 10%. Распределение времени между работой с глобальной памятью

Результаты измерений производительности 20-ядерного процессора MALT–Cv2 Rev1, процессора общего назначения Intel Core i7–7820X и видеокарты NVIDIA Tesla P4 в задаче фильтрации изображения с использованием оператора Собеля

Процессор	MALT–Cv2 (эмулятор)		MALT–Cv2 (ПЛИС)	MALT–Cv2 (СБИС)		Intel Core i7–7820X	NVIDIA Tesla P4
Частота	100 МГц	600 МГц	100 МГц	100 МГц	600 МГц	3,6 ГГц	1,11 ГГц
Общее время обработки, мс	68 ± 4	$11,4 \pm 1,0$	62 ± 4	110 ± 12	36 ± 5	$2,7 \pm 0,1$	$1,00 \pm 0,03$
Время копирования данных из глобальной памяти в локальную, мс	$34,2 \pm 0,2$ ($50 \pm 3\%$)	$5,7 \pm 0,3$ ($50 \pm 2\%$)	$30,0 \pm 1,8$ ($49 \pm 3\%$)	38 ± 4 ($35 \pm 4\%$)	$7,9 \pm 1,0$ ($22 \pm 3\%$)	–	$0,22 \pm 0,01$ ($22 \pm 1\%$)
Время, затраченное на вычисления и работу с локальной памятью, мс	$26,3 \pm 1,5$ ($39 \pm 2\%$)	$4,4 \pm 0,2$ ($39 \pm 2\%$)	$25,2 \pm 1,5$ ($41 \pm 2\%$)	41 ± 4 ($38 \pm 4\%$)	$8,4 \pm 1,0$ ($23 \pm 3\%$)	–	$0,55 \pm 0,02$ ($55 \pm 2\%$)
Время копирования данных из глобальной памяти в локальную, мс	$7,7 \pm 0,4$ ($11 \pm 1\%$)	$1,3 \pm 0,1$ ($11 \pm 1\%$)	$6,6 \pm 0,4$ ($11 \pm 1\%$)	30 ± 3 ($27 \pm 3\%$)	$20,2 \pm 2,5$ ($56 \pm 7\%$)	–	$0,23 \pm 0,01$ ($32 \pm 1\%$)
Динамическое энергопотребление, Вт	–	–	–	0,15	0,85	16	38
Энергоэффективность, кадров в секунду / Ватт	–	–	–	61	32	23	26

и непосредственно вычислениями (соответствующие доли в процентах указаны в таблице в скобках под абсолютными значениями времени) сохраняется с точностью до нескольких процентов.

Время обработки изображения при использовании СБИС, работающей на частоте 100 МГц и 600 МГц, почти в два раза больше времени, показанного для этой задачи эмулятором. При этом время взаимодействия с глобальной памятью и время обработки в пределах ядра на СБИС с увеличением частоты изменяются по-разному, в то время как процентное соотношение между этими величинами при увеличении частоты на эмуляторе сохраняется. Наибольшее замедление СБИС относительно эмулятора возникает при копировании результатов обработки из локальной памяти ядра в глобальную оперативную память. Это замедление связано с ограниченной производительностью контроллера памяти СБИС, используемой для экспериментов (модель, заложенная в эмулятор, особенности контроллера памяти СБИС не учитывала). Согласно информации от разработчиков, в последующих ревизиях СБИС планируется использование более быстрых контроллеров памяти.

Процессорные ядра MALT используют упрощенный набор команд, не используют кэширование, не имеют других механизмов аппаратной оптимизации и работают на более низкой частоте, поэтому они выполняют вычисления медленнее процессоров общего назначения. Обработка изображения с использованием 15 ядер процессора MALT на доступном сегодня образце СБИС занимает примерно в 12 раз больше времени по сравнению с последовательным выполнением задачи на одном ядре процессора общего назначения Intel Core i7–7820X при отличии частот тактирования в 6 раз, однако одно ядро процессора общего назначения для обработки одного изображения требует от 1,4 до 2,7 раз больше энергии, даже по сравнению с суммарным потреблением 15 вычислительных и 5 служебных ядер процессора MALT.

Графический процессор в данной задаче обладает еще более высокой производительностью и лучше процессоров общего назначения по энергоэффективности. Тем не менее, система с использованием графического процессора все же уступает процессору MALT по данному параметру. Доля времени, затраченного непосредственно на вычисления, в случае использования графического процессора несколько больше, чем в случае использования процессоров MALT. При этом в последнем случае время, затрачиваемое на обмен данными с глобальной памятью, уже удалось снизить в несколько раз за счет использования программно-аппаратного механизма блочных чтений. Однако предполагается, что данный параметр еще может быть улучшен в ходе дальнейших программных оптимизаций алгоритма за счет изменения структуры данных.

Несмотря на то, что процессор MALT в данном тесте показал самые низкие результаты по времени обработки изображения (36 мс в случае использования реальной СБИС и 11,4 мс в случае моделирования расчетов с использованием низкоуровневого эмулятора), достигнутой производительности вполне достаточно для обработки видеопотока в реальном времени без значительных задержек.

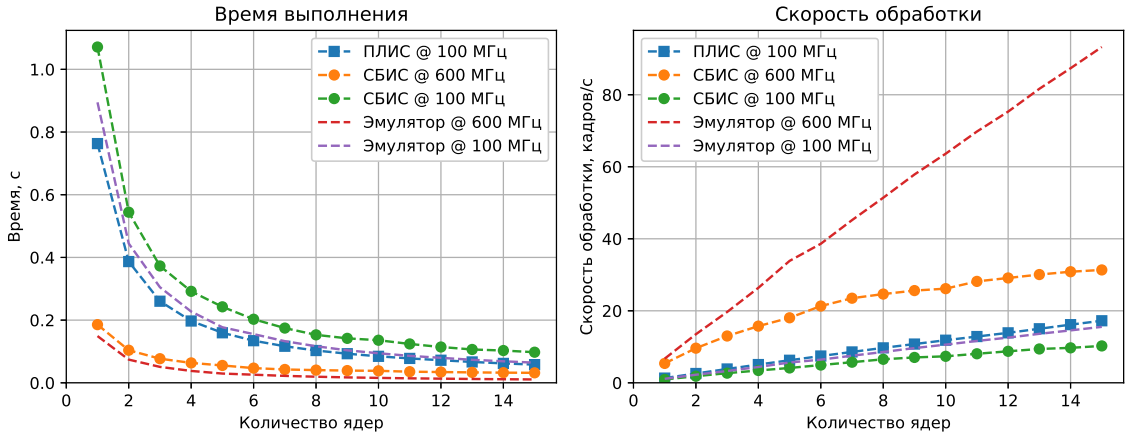

Рис. 3. Зависимость времени выполнения расчетов и скорости обработки изображений от количества вычислительных ядер процессора MALT–Cv2 Rev1 в случае вычислений с использованием эмулятора MALTemu, прототипа на ПЛИС и экспериментальной СБИС

Зависимости времени выполнения алгоритма и скорости обработки изображений от количества используемых для вычислений ядер процессора MALT представлены на рис. 3.

В случае вычислений на эмуляторе и ПЛИС зависимость скорости обработки изображений от количества ядер (в пределах 15) близка к линейной. Это свидетельствует о том, что алгоритм распараллелен достаточно хорошо, и при дальнейшем масштабировании системы производительность будет расти. Вычисления с использованием СБИС при любом количестве ядер занимают больше времени по сравнению с тем, что предсказывает эмулятор. Зависимость скорости обработки отличается от линейной. При использовании количества ядер более 8 наблюдается выход на насыщение. Это говорит о том, что не допускающая распараллеливания часть работы достаточно велика. В случае использования эмулятора и ПЛИС (при количестве ядер менее 15) насыщения на графике зависимости скорости обработки от количества ядер не наблюдается (рис. 3), что может быть связано с особенностями аппаратной реализации контроллера памяти СБИС.

5. Заключение. В рамках данной работы исследованы возможности многоядерных процессоров MALT в задачах обработки изображений. В качестве тестовой взята задача фильтрации изображений с помощью оператора Собеля. Разработана параллельная реализация данного оператора, оптимизированная для работы с ядрами общего назначения процессоров MALT. Исследования выполнены с использованием низкоуровневого эмулятора MALTemu, прототипа процессора в ПЛИС и экспериментальной СБИС модели MALT–Cv2 Rev1. Проведен сравнительный анализ полученных результатов с аналогичными для процессора общего назначения Intel Core i7–7820X и видеокарты NVIDIA Tesla P4.

По оценкам, полученным в симуляции, время обработки изображения размером 1024×768 точек при использовании 15 вычислительных ядер процессора MALT на частоте 100 МГц должно составлять 68 ± 4 мс. В случае использования прототипа в ПЛИС было достигнуто время обработки 62 ± 4 мс. Время обработки на СБИС оказалось значительно больше — 110 ± 12 мс. При работе на частоте тактирования 600 МГц время обработки изображения на СБИС так же превысило время, полученное в симуляции: 36 ± 5 мс против $11,4 \pm 1,0$ мс. Медленная работа СБИС объясняется низкой производительностью контроллера памяти, используемого в данной ревизии. В последующих ревизиях планируется использование контроллеров памяти с более высокой производительностью.

Достигнутая производительность приемлема для обработки видео в реальном времени. Тем не менее, производительность реализаций фильтра Собеля для процессора общего назначения Intel Core i7–7820X и графического процессора NVIDIA Tesla P4 оказалась выше (время обработки изображения размером 1024×768 точек составляет $2,7 \pm 0,1$ мс / изображение и $1,00 \pm 0,03$ мс / изображение соответственно).

Для задачи фильтрации изображений с помощью оператора Собеля сделаны оценки энергоэффективности. Наибольшая энергоэффективность достигнута при использовании процессоров MALT на частоте тактирования 100 МГц — 61 кадр в секунду / Ватт. Энергоэффективность графического процессора оказалась существенно ниже — 26 кадров в секунду / Ватт. В случае обработки изображений с использованием процессора общего назначения энергоэффективность еще ниже — 23 кадра в секунду / Ватт.

В ходе исследований получена зависимость скорости обработки изображений от количества используемых ядер процессора MALT в диапазоне от 1 до 15. Зависимости для эмулятора и прототипа в ПЛИС практически не отличаются от линейных. Этот факт позволяет рассчитывать на дальнейшее увеличение производительности в случае увеличения количества вычислительных ядер. Зависимость для СБИС отличается от линейной и уже при 15 ядрах близка к насыщению. Однако, как уже говорилось выше, это, скорее всего, связано с особенностями контроллера памяти конкретной экспериментальной СБИС.

С учетом полученных результатов можно сделать вывод о том, что процессоры MALT могут быть использованы для решения задач обработки изображений в случаях, когда наряду с производительностью требуется обеспечить высокую энергоэффективность системы. Использование подобных процессоров может быть оправдано в роботизированных системах с требованиями на большое время автономной работы. Также такие процессоры могут снизить затраты на содержание ЦОД, специализирующихся на обработке изображений.

Работа выполнена при частичной поддержке гранта РФФИ № 19–29–09044.

СПИСОК ЛИТЕРАТУРЫ

1. *Fan R., Jiao J., Ye H., et al.* Key ingredients of self-driving cars. ArXiv preprint: 1906.02939 [cs.RO]. Ithaca: Cornell Univ. Library, 2019. https://arxiv.org/abs/1906.02939
2. *Kanellakis C., Nikolakopoulos G.* Survey on computer vision for UAVs: current developments and trends // J. Intell. Robot. Syst. 2017. **87**. 141–168.
3. *Bučka P., Szabová M., Dekan M., et al.* Image processing of motion for security applications // European Scientific Journal. 2017. **13**, N 27. 44–58.
4. *Payne B.R., Belkasim S.O., Owen G.S., et al.* Accelerated 2D image processing on GPUs // Lecture Notes in Computer Science. Vol. 3515. Berlin: Springer, 2005. 256–264.
5. *Zohouri H.R.* High performance computing with FPGAs and OpenCL. ArXiv preprint: 1810.09773 [cs.DC]. Ithaca: Cornell Univ. Library, 2018. https://arxiv.org/abs/1810.09773
6. *Jouppi N.P., Young C., Patil N., et al.* In-datacenter performance analysis of a tensor processing unit. ArXiv preprint: 1704.04760 [cs.AR]. Ithaca: Cornell Univ. Library, 2017. https://arxiv.org/abs/1704.04760
7. *Valle M., Nateri G., Caviglia D.D., et al.* An ASIC design for real-time image processing in industrial applications // Proc. the European Design and Test Conference. New York: IEEE Press, 1995. 385–390.
8. *Елизаров С.Г., Лукъянченко Г.А., Марков Д.С. и др.* Программируемые на языках высокого уровня энергоэффективные специализированные СБИС для решения задач информационной безопасности // Системы высокой доступности. 2018. **14**, № 3. 40–48.
9. *Пирогова Л.А., Грекул В.И., Поклонов Б.Е.* Оценка совокупной стоимости владения центром обработки данных // Бизнес-информатика. 2016. **36**, № 2. 32–40.
10. *Handa P., Kalra M., Sachdeva R.* A survey on green computing using GPU in image processing // International journal of computer and technology. 2015. **14**, N 10. 6135–6141.
11. *Yanovskaya O., Yanovsky M., Kharchenko V.* The concept of green Cloud infrastructure based on distributed computing and hardware accelerator within FPGA as a Service // Proc. IEEE East-West Design and Test Symposium. New York: IEEE Press, 2014. 1–4.
12. *LeCun Y., Haffner P., Bottou L., Bengio Y.* Object recognition with gradient-based learning // Lecture Notes in Computer Science. Vol. 1681. Berlin: Springer, 1999. 319–345.
13. *Sobel I.* An Isotropic 3 × 3 Image Gradient Operator. Presentation at Stanford A.I. Project. 1968. https://en.wikipedia.org/wiki/Sobel_operator
14. *Canny J.* A computational approach to edge detection // IEEE Transactions on Pattern Analysis and Machine Intelligence. 1986. **8**, N 6. 679–698.
15. *Baumgartner D., Roessler P., Kubinger W., et al.* Benchmarks of low-level vision algorithms for DSP, FPGA, and Mobile PC processors // Embedded Computer Vision. Advances in Pattern Recognition. London: Springer, 2009. 101–120.
16. Geekbench 5 Compute Workloads. Toronto: Primate Labs Inc. 2019. https://www.geekbench.com/doc/geekbench5-compute-workloads.pdf
17. MicroBlaze Processor Reference Guide. San Jose: Xilinx Inc. 2008. https://www.xilinx.com/support/documentation/sw_manuals/mb_ref_guide.pdf
18. Руководство программиста MALT SW. М.: МГУ, 2019. https://maltsystem.ru/images/pdf/malt_sdk02.pdf

Поступила в редакцию
22 мая 2020

MALT Manycore Processors Capabilities in Image Processing Tasks

N. G. Mikheev[1], V. A. Antonyuk[2], S. G. Elizarov[3], and G. A. Lukyanchenko[4]

[1] *Lomonosov Moscow State University, Faculty of Physics; Leninskie Gory, Moscow, 119991, Russia; PhD Student, e-mail: ng.mikheev@physics.msu.ru*

[2] *Lomonosov Moscow State University, Faculty of Physics; Leninskie Gory, Moscow, 119991, Russia; Ph.D., Associate Professor, e-mail: antonyuk@physics.msu.ru*

[3] *Lomonosov Moscow State University, Faculty of Physics; Leninskie Gory, Moscow, 119991, Russia; Ph.D., Senior Scientist, e-mail: elizarov@physics.msu.ru*

[4] *Lomonosov Moscow State University, Faculty of Physics; Leninskie Gory, Moscow, 119991, Russia; Ph.D., Scientist, e-mail: elizarov@physics.msu.ru*

Received May 22, 2020

Abstract: In this paper we consider the experimental performance and energy efficiency evaluation in image processing tasks for the MALT manycore processors. The image filtering with the Sobel operator is used as an example. Measurements are conducted using the MALTemu low level emulator, an FPGA processor prototype and an experimental ASIC model MALT-Cv2 Rev1. The obtained results are compared with similar results for a general purpose CPU (sequential implementation) and a GPU with the CUDA technology support.

Keywords: manycore processor, parallel computing, image processing, Sobel operator, performance, energy efficiency, MALT, CUDA.

References

1. R. Fan, J. Jiao, H. Ye, et al., *Key Ingredients of Self-Driving Cars*, ArXiv preprint: 1906.02939 [cs.RO] (Cornell Univ. Library, Ithaca, 2019). https://arxiv.org/abs/1906.02939.

2. C. Kanellakis and G. ,Nikolakopoulos, "Survey on Computer Vision for UAVs: Current Developments and Trends," J. Intell. Robot. Syst. **87**, 141–168 (2017).

3. P. Bučka, M. Szabová, M. Dekan, et al., "Image Processing of Motion for Security Applications," Europ. Sci. J. **13** (27), 44–58 (2017).

4. B. R. Payne, S. O. Belkasim, G. S. Owen, et al., "Accelerated 2D Image Processing on GPUs," in *Lecture Notes in Computer Science* (Springer, Berlin, 2005), Vol. 3515, pp. 256–264.

5. H. R. Zohouri, *High Performance Computing with FPGAs and OpenCL*, ArXiv preprint: 1810.09773 [cs.DC] (Cornell Univ. Library, Ithaca, 2018). https://arxiv.org/abs/1810.09773

6. N. P. Jouppi, C. Young, N. ,Patil, et al., *In-Datacenter Performance Analysis of a Tensor Processing Unit*, ArXiv preprint: 1704.04760 [cs.AR] (Cornell Univ. Library, Ithaca, 2017). https://arxiv.org/abs/1704.04760

7. M. Valle, G. Nateri, D. D. Caviglia, et al., "An ASIC Design for Real-Time Image Processing in Industrial Applications," in *Proc. Europ. Design Test Conf., Paris, France, March 6–9, 1995* (IEEE Press, New York, 1995), pp. 385–390).

8. S. G. Elizarov, G. A. Lukyanchenko, D. S. Markov, et al., "Programmable in High-Level Languages Energy-Efficient Specialized VLSI for Solving Information Security Problems," Sistemy Visokoi Dostupnosti **14** (3), 40–48 (2018).

9. L. A. Pirogova, V. I. Grekul, and B. E. ,Poklonov, "Estimated Aggregate Cost of Ownership of a Data Processing Center," Biznes Inform., No. 2, 32–40 (2016).

10. P. Handa, M. Kalra, and R. Sachdeva, "A Survey on Green Computing Using GPU in Image Processing," Int. J. Comp. Tech. **14** (10), 6135–6141 (2015).

11. O. Yanovskaya, M. Yanovsky, and V. Kharchenko, "The Concept of Green Cloud Infrastructure Based on Distributed Computing and Hardware Accelerator within FPGA as a Service," in *Proc. IEEE East-West Design Test Symp., Kiev, Ukraine, September 26–29, 2014* (IEEE Press, New York, 2014), pp. 1–4).

12. Y. LeCun, P. Haffner, L. Bottou, and Y. Bengio, "Object Recognition with Gradient-Based Learning," in *Shape, Contour and Grouping in Computer Vision* (Springer, Berlin, 1999), Vol. 1681, pp. 319–345.

13. I. Sobel, *An Isotropic 3×3 Image Gradient Operator*, Presentation at Stanford A.I. Project (1968). https://en.wikipedia.org/wiki/Sobel_operator

14. J. Canny, "A Computational Approach to Edge Detection," IEEE Transact. Pattern Analysis Mach. Intell. **8** (6), 679–698 (1986).

15. D. Baumgartner, P. Roessler, W. Kubinger, et al., "Benchmarks of Low-Level Vision Algorithms for DSP, FPGA, and Mobile PC Processors," in *Embedded Computer Vision. Advances in Pattern Recognition* (Springer, London, 2009), pp. 101–120.

16. Geekbench 5 Compute Workloads (Primate Labs, Toronto, 2019). https://www.geekbench.com/doc/geekbench5-compute-workloads.pdf

17. MicroBlaze Processor Reference Guide (Xilinx, San Jose, 2008). https://www.xilinx.com/support/documentation/sw_manuals/mb_ref_guide.pdf

18. MALT SW Programmer's Guide (Lomonosov State University, Moscow, 2019). https://maltsystem.ru/images/pdf/malt_sdk02.pdf