УДК 004.032.24/004.272.43

МОДЕЛИРОВАНИЕ СИСТЕМЫ ПЕРЕДАЧИ ДАННЫХ С МНОГОПОРОГОВЫМ ДЕКОДЕРОМ С ИСПОЛЬЗОВАНИЕМ OPENCL

Демидов Д.С., Овечкин Г.В.

ФГБОУ ВПО «Рязанский государственный радиотехнический университет», Рязань, e-mail: dmitri-demidiv@yandex.ru

В данной статье описана проблема обеспечения высокой достоверности данных, передаваемых по каналам связи, для решения которой применяются методы помехоустойчивого кодирования. Отмечается, что одними из лучших по соотношению эффективности и сложности реализации являются многопороговые декодеры (МПД) самоортогональных кодов. Показано, что использование компьютерного моделирования для оценки эффективности работы МПД при низких целевых вероятностях ошибки в ряде случаев оказывается невозможным из-за слишком большого времени работы модели. Предложено для повышения скорости моделирования использовать технику GPGPU (General-purpose computing for graphics processing units). Описаны существующие технологии, поддерживающие GPGPU. Обоснован выбор технологии OpenCL для реализации моделирования. Описаны основные технические решения, предложенные в ходе реализации десктоп-приложения для моделирования с использованием OpenCL. Результаты проведенных компьютерных экспериментов показали, что предложенное решение, использующее OpenCL, позволяет повысить скорость моделирования в 25 раз по сравнению с CPU-реализацией. Представлены пути дальнейшего развития разработанной модели.

Ключевые слова: помехоустойчивое кодирование, моделирование, многопороговый декодер, GPGPU, OpenCL

MODEL OF DATA TRANSMISSION SYSTEMS WITH MULTITHRESHOLD DECODER VIA OPENCL

Demidov D.S., Ovechkin G.V.

Federal Autonomous Educational Institution of Higher Education Ryazan State Radio Engineering University, Ryzan, e-mail: dmitri-demidiv@yandex.ru

This article is dedicated problems of transmission data quality. Point of transmission quality is raised by increase of data in the world. Such questions are described in error-correcting coding sphere. The best method ratio of efficiency and complexity in error-correcting coding sphere is multithreshold decoder. That article demonstrates impossibility of model multithreshold decoder operations for performance estimation because of high time costs. Consequently using of GPGPU (General-purpose computing for graphics processing units) is offered. Existing GPGPU technologies are described. Choosing of OpenCL for model multithreshold decoder operations is justified. Main problems and feature of model application (with OpenCL) are discussed. Examples of features with description are added to the article. Results of experiments showcases that using of OpenCL grants 25x increase of model speed compared with pure CPU results. Future point of model speed growth are described.

Keywords: error-correcting coding, modeling, multithreshold decoder , GPGPU, OpenCL

Быстрый рост объемов обработки данных, развитие цифровых систем вещания и вычислительных сетей предъявляют высокие требования к минимизации ошибок в используемых цифровых данных. Поэтому одной из важнейших задач является обеспечение высокой достоверности передачи данных. Исправлением ошибок после передачи по каналу занимается помехоустойчивое кодирование [2].

Одним из наиболее эффективных решений проблемы помехоустойчивого кодирования при высокой энергетической характеристике систем кодирования являются многопороговые декодеры, которые позволяют декодировать очень длинные коды с линейной от длины кода сложностью исполнения. В основе работы МПД лежит итеративное декодирование, что позволяет вплотную приблизиться к решению оптимального декодера в достаточно большом диапазоне кодовых скоростей и уровней шума в канале. При этом МПД сохраняет простоту и быстродействие обычного порогового декодера, что делает его очень привлекательным для применения в существующих и вновь создаваемых высокоскоростных системах связи [3].

Схема работы МПД представлена на рис. 1.

Главное достоинство МПД, которые были сначала разработаны для двоичных кодов, заключается в том, что используемые в них мажоритарные процедуры коррекции ошибок допускают полное распараллеливание выполняемых операций, что обеспечивает работу с теоретически максимально возможной производительностью.

Моделирование

Для оценки эффективности помехоустойчивых кодов и методов их декодирования при большом уровне шума требуется

использовать моделирование, поскольку аналитическая оценка вероятности ошибки в таких условиях является обычно очень неточной. Особенностью современных систем передачи и хранения данных является низкая целевая вероятность ошибки. Например, в оптических системах связи необходимо обеспечить вероятность ошибки менее 10^{-11}. Для оценки подобной вероятности ошибки с помощью моделирования нужно выполнить передачу как минимум 10^{12} битов. Вместе с тем моделирование кодирования, передачи и декодирования подобных объемов данных является очень затратной с временной точки зрения операцией. К примеру, при использовании вычислительных ресурсов центрального процессора (CPU) удалось получить скорость передачи данных по каналу связи с МПД только на уровне 320 кбит/с. Для эксперимента использовался процессор *Intel(R) Core(TM) i3 2,53 GHz* (двухъядерный процессор). Во время моделирования удалось достичь практически 100 % загрузки обоих ядер CPU. При такой скорости моделирование передачи 10^{12} бит через канал связи займет около 868 часов. Подобные временные затраты являются неприемлемыми.

processing unit). В случае достаточного параллелизма реализуемых вычислений применима техника **General-purpose computing forgraphics processing units (GPGPU)** [4]. Такую технику можно использовать и для моделирования работы МПД.

Отметим, что время моделирования можно грубо оценить как

$$t = \frac{K \cdot V \cdot N}{q \cdot n}.$$

Здесь t – время моделирования; V – число передаваемых бит; N – число элементарных операций, требующихся на передачу одного бита; n – число ядер в процессоре; q – тактовая частота процессора; K – коэффициент накладных расходов. При количестве ядер процессора n больше единицы коэффициент будет меньше единицы. Это связано с ростом числа инфраструктурных операций, выполняемых CPU для синхронизации параллельной работы ядер. Данные инфраструктурные операции будут уменьшать общий объем полезных операций, выполняемых GPU. При использовании процессором одного ядра K будет равен единице, так как все вычислительные ресурсы будут направлены на целевые операции.

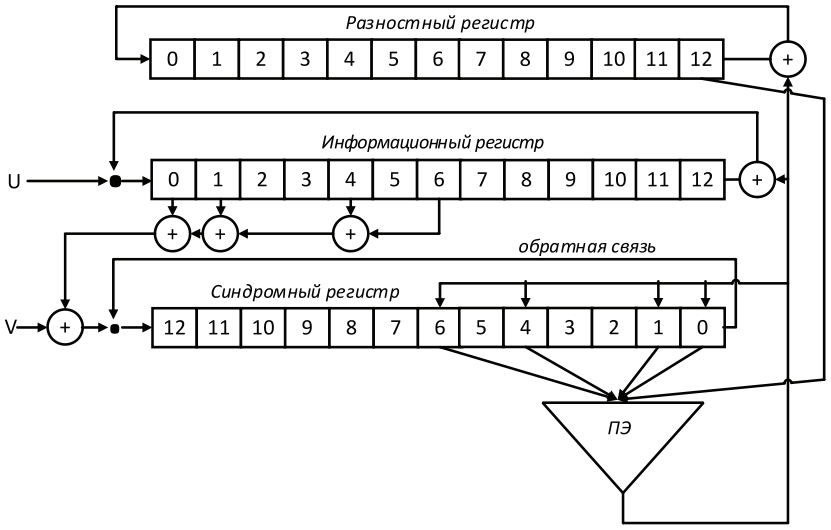

Рис. 1. Многопороговый декодер блокового кода

Исходя из вышесказанного необходимо найти возможности по увеличению скорости моделирования систем передачи данных.

Задача ускорения вычислений требует использования принципиально других подходов к организации вычислений. Перспективным направлением ускорения вычислений является использование незадействованных ресурсов гетерогенных компьютерных систем, в частности вычислительных ресурсов графических процессоров **GPU (graphic**

Обычно вопрос времени моделирования описывался простым правилом – чем больше частота, тем больше производительность. Однако бесконечный рост тактовой частоты невозможен. Это связано с конструктивными сложностями, в частности с проблемой перегрева процессора. Исходя из этого можно сделать вывод, что альтернативным ресурсом повышения скорости моделирования является увеличение числа ядер в процессоре. Однако потенциал роста числа ядер в CPU

ограничен. CPU с числом ядер 16 и больше являются большой редкостью и имеют высокую стоимость. Однако большое число ядер можно найти в GPU. В GPU число ядер может достигать нескольких тысяч, что позволяет рассмотреть использование GPU в качестве перспективного вычислительного механизма для решения вышеописанной задачи.

Использование GPGPU

В настоящее время для вычислений на GPU активно продвигаются две технологии **OpenCL** и **CUDA**.

OpenCL (OpenComputingLanguage) – это технология, реализующая параллельные компьютерные вычисления на различных типах графических и центральных процессоров.

CUDA (ComputeUnifiedDeviceArchitecture) – программно-аппаратная архитектура параллельных вычислений, которая позволяет существенно увеличить вычислительную производительность благодаря использованию графических процессоров фирмы **Nvidia** [1].

Обе описанные технологии представляют схожий функционал для активного и эффективного использования вычислительных ресурсов GPU. Существенным отличием является то, что CUDA поддерживается и продвигается преимущественно компанией Nvidia, следовательно, для моделирования с использованием CUDA понадобится персональный компьютер с GPU производства именно этой компании. Данный факт существенно сужает парк компьютеров, который может использоваться для моделирования. В то же время OpenCL поддерживается практически всеми производителями GPU (включая Nvidia), ничуть не уступая в эффективности. Исходя из этого выбор OpenCL для моделирования работы является очевидным.

Синтаксис языка программирования OpenCL базируется на стандарте C99, но имеет ряд специфических и очень важных изменений. Подобные изменения накладывают ряд значительных ограничений [5]:

– невозможно динамическое выделение памяти;

– невозможно использовать массивы с размерностью более единицы;

– невозможно использование рекурсий;

– отсутствие встроенного генератора случайных чисел.

Данные ограничения привели к необходимости нестандартных инженерных решений при программировании работы МПД на языке OpenCL.

Одно из решений связано с разработкой структуры, реализующей работу двумерного массива, который необходим при моделировании работы МПД. Структура представляет из себя одномерный массив, который содержит элементы нескольких одномерных массивов и некоторую инфраструктурную информацию. Структура, представленная на рис. 2, состоит из 4 частей:

– общее количество одномерных массивов в структуре;

– общая длина структуры;

– длины одномерных массивов, содержащихся в структуре;

– элементы одномерных массивов, входящих в структуру.

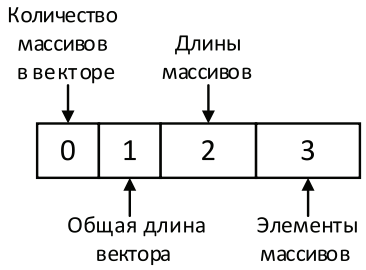

Рис. 2. Схема структуры для хранения одно- или двумерных массивов

Для одномерного массива длиной 2, содержащего элементы [0, 3], структура будет иметь вид, представленный на рис. 3.

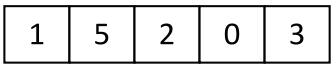

Рис. 3. Структура, хранящая одномерный массив

Если нужно использовать двумерный массив, то необходимо в структуру просто поместить два одномерных массива. Для двух одномерных массивов длиной 2, содержащих элементы [0, 3] и [1, 11], структура будет иметь вид, представленный на рис. 4.

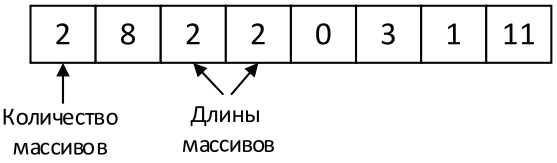

Рис. 4. Двумерный массив в структуре

Так же был реализован генератор случайных чисел на основе регистров сдвига с обратной связью. Принцип действия генератора заключается в смещении и маскировании случайной величины. Начальное случайное значение задается при старте приложения.

Реализация генератора на языке OpenCL, для заполнения массива случайных чисел, представлена в листинге.

Листинг

```
// Заполнение buffer значениями 0 или 1.
// Для генерирования случайных чисел используется полиномиальный генератор
__kernel void fill(__global int* buffer, __global int* kts)
{
        int k = 13;

        intgid = get_global_id(0);

        long ndsl3, lll;

        unsigned long ndsl1 = (unsigned long)kts[gid];
        unsigned long ndsl2 = 7799l;
        unsigned long nmsk3 = 0x001fffffl;
        unsigned long nmsk4 = 0x7fffffffl;
        unsigned long msk = 1023;
        unsigned long maxRandom = 0x7fff;

        for (int index = 0; index < k; index++)
        {
                for (inti = 0; i< 6; i++)
                {
                        lll = ndsl1^ndsl2;
                        ndsl1 = ndsl2&nmsk3;
                        ndsl3 = (ndsl2 >> 21) ^ (lll<< 10);
                        ndsl2 = ndsl3 & nmsk4;
                }

                floatrnd = 1.0 * ndsl2 / nmsk4;

                buffer[gid * k + index] = rnd> 0.5 ? 1 : 0;
                }

}
```

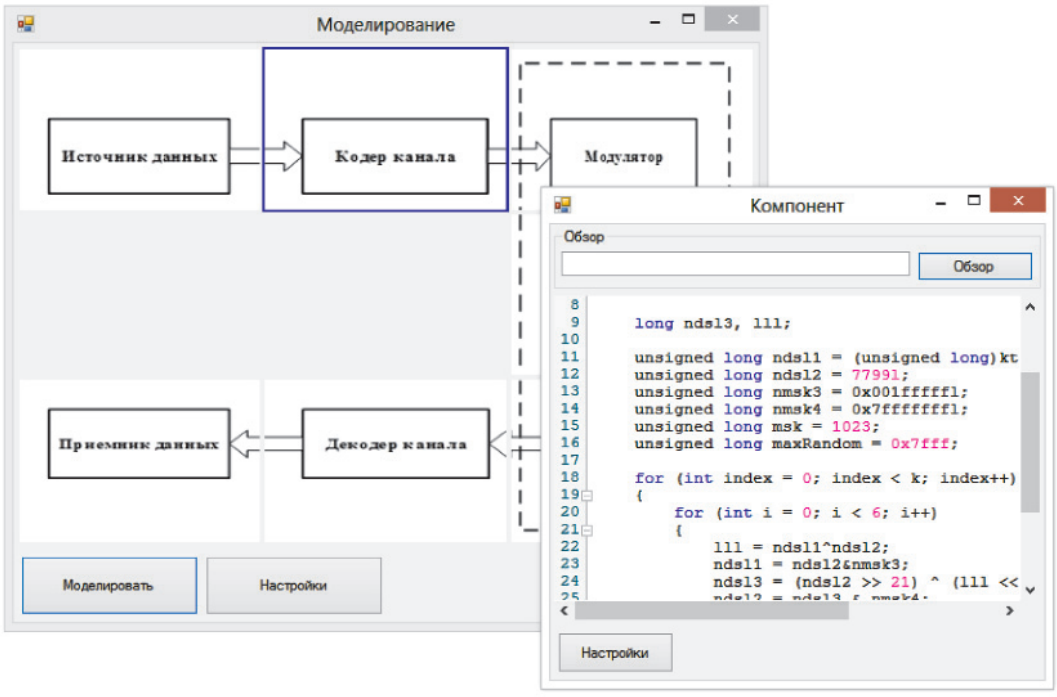

Рис. 5. Интерфейс десктоп-приложения для моделирования

Данный генератор позволяет получить случайные числа с равномерным законом распределения в диапазоне [0; 1).

Описанная выше структура для хранения данных и генератор случайных чисел были использованы в десктоп-приложении, разработанного для моделирования работы МПД с использованием GPU. Одна из форм приложения представлена на рис. 5.

Данное приложение максимально эффективно использовало ресурсы GPU и позволило увеличить скорость моделирования передачи данных по каналу связи с МПД до 11 мбит/с. Таким образом, по сравнению с реализацией через CPU скорость моделирования была увеличена в 34 раза. При такой скорости моделирование передачи 10^{12} бит через канал связи с кодированием и декодированием займет всего около 25 часов, что значительно меньше времени, затраченного при моделировании передачи данных с использованием CPU. В данном эксперименте использовался GPU *AMD Radeon HD 6300M Series* с 128 ядрами в процессоре.

Вывод

Полученные результаты дают основание считать перспективным использование предложенного подхода к решению задачи уменьшения временных затрат на ресурсоемкое моделирование характеристик МПД. Существующие результаты имеют широкий потенциал для дальнейшего алгоритмического улучшения. Так же очевиден легко извлекаемый аппаратный потенциал, который позволит повысить скорость моделирования за счет использования более мощных GPU, которые в настоящее время широко доступны.

Работа выполнена при поддержке РФФИ (грант № 13-07-00391) и гранта Президента РФ (грант МД-639.2014.9).

Список литературы

1. Боресков А.В., Харламов А.А., Марковский Д.А., Микушин Д.Н., Мортиков Е.В., Мыльцев А.А., Сахарных Н.А., Фролов В.А. Параллельные вычисления на GPU. Архитектура и программная модель CUDA: учебное пособие. – М.: Изд-во Московского университета, 2012. – 336 с.

2. Золотарев В.В., Зубарев Ю.Б., Овечкин Г.В. Многопороговые декодеры и оптимизационная теория кодирования – М.: Горячая линия – Телеком, 2012. – 239с.

3. Золотарев В.В., Овечкин Г.В. Помехоустойчивое кодирование. Методы и алгоритмы: справочник. – М.: Горячая линия–Телеком, 2004. – 126 с.

4. Подвальный С.Л., Холопкина Л.В., Попов Д.В., Численные методы и вычислительный эксперимент. – Уфа: Уфимский государственный авиационный технический университет, 2005. – 224 с.

5. The OpenCL Specification. Version: 2.0. Document Revision: 22 / Khronos OpenCL Working Group, Editor: AaftabMunshi. – 2014. – 483 p.

References

1. Boreskov A.V., KHarlamov A.A., Markovskiy D.A., Mikushin D.N., Mortikov E.V., Myltsev A.A., Sakharnykh N.A., Frolov V.A. Parallelnye vychisleniya na GPU. Arkhitektura i programmnaya model CUDA: Uchebnoe posobie [Parallel computing on the GPU. Architecture and programming model CUDA: Textbook], 2012, 336 p.

2. Zolotarev V.V., Zubarev U.B., Ovechkin G.V. Mnogoporogovye dekodery i optimizatsionnaya teoriya kodirovaniya [Multithreshold decoders and theory of optimization coding]. Moscow, Goryachayaliniya. Telekom Publ, 2012, 239 p.

3. Zolotarev V.V., Ovechkin G.V. Pomekhoustoychivoe kodirovanie. Metody i algoritmy. Spravochnik [Noiseless coding.Methods and Algorithms. Directory] –Moscow, Goryachayaliniya. Telekom Publ, 2004, 126 p.

4. Podvalnyy S.L., Kholopkina L.V., Popov D.V. Chislennye metody i vychislitelnyy eksperiment [Numerical methods and computer experiment], Ufa State Aviation Technical University Publ, 2005, 224 p.

5. The OpenCL Specification. Version: 2.0. Document Revision: 22,Khronos OpenCL Working Group, Editor, Aaftab Munshi, 2014, 483 p.