Внутреннее представление чисел в библиотеке длинной арифметики для GPU

Верещак М.И., Неласая А.В

Запорожский национальный технический университет, ул. Жуковского 64, Запорожье, Украина

max@phoinic.com.ua, annanelasa@gmail.com

Abstract. *CUDA can be also used to improve perfomance of cryptographic algorithms. Most of cryptosystems are based on arbitrary-precision arithmetic, but arbitrary-precision arithmetic libraries are not available for CUDA now. This paper describes the first steps of developing universal arbitrary-precision arithmetic library for CUDA technology.*

1 Введение

Технологии неграфических вычислений на графических процессорах набирают все большую популярность. Концепция GPGPU позволила значительно снизить удельную стоимость высокопроизводительных вычислений, уменьшить габариты и энергопотребление параллельных вычислительных систем. На сегодня существует несколько реализаций данной концепции: CUDA, DirectCompute, OpenCL, AMD FireStream, C++ AMP. Различаются они набором поддерживаемого оборудования, особенностями реализации программного интерфейса, подходом к лицензированию. Одной из наиболее активно развивающихся в последнее время технологий является технология CUDA, разрабатываемая компанией Nvidia. Преимуществами данной технологии являются:

- качественная документация
- удобный программный интерфейс
- широкие возможности.

Главный же ее недостаток — совместимость исключительно с графическими ускорителями компании Nvidia, что приводит к необходимости закупки специального оборудования.

2 Применение CUDA в решении задач криптографии

Наибольшее распространение технология CUDA получила в решении задач, в основе которых лежит работа с вещественными числами. Однако обработкой вещественных чисел возможности данной технологии не исчерпываются: наравне с вещественной арифметикой в ней реализованы и операции с целочисленными данными. Существует множество отраслей знаний, базирующихся на операциях с целыми числами и при этом нуждающихся в применении высокопроизводительных вычислений. К ним, в частности, относятся задачи криптографии и криптоанализа. Еще одной важной особенностью задач в области криптографии является необходимость использования библиотек длинной арифметики. Для обеспечения достаточной безопасности криптосистем используются ключи, значения которых выходят за рамки разрядной сетки вычислительных систем, и обойти эти ограничения пока не представляется возможным.

В реализации различных криптографических примитивов используются такие математические структуры, как арифметика конечных полей, полиномиальная арифметика, арифметика простых и расширенных полей Галуа. Как показал анализ существующих разработок в области применения концепции GPGPU для решения задач криптографии и криптоанализа, данное направление развито довольно слабо. В ряде зарубежных публикаций исследуется производительность параллельных алгоритмов решения отдельных задач криптоанализа и криптографии, например: решение проблемы дискретного логарифмирования на эллиптических кривых методом ро Полларда [1], реализация методов целочисленной факторизации на эллиптических кривых [2]. Однако, анализ этих публикаций приводит к выводу об отсутствии какой-либо универсальной реализации библиотеки криптопримитивов и библиотеки длинной арифметики для GPU,

используемой исследователями в данной области. В поисках подходящих библиотек были обнаружены два интересных проекта:

- CUMP — библиотека вещественных чисел произвольной точности на основе технологии CUDA [3];
- GMP-ECM — реализация арифметики эллиптических кривых на основе проекта GMP [4], в исходных кодах которой найдены функции, использующие GPU, в случае его наличия, для повышения производительности вычислений.

Первый проект не был завершен: в нем реализованы лишь 3 базовые операции: сложение, вычитание и умножение; и судя по отсутствию обновлений в течении последнего года был заброшен своими создателями. Что касается второго проекта, то он по-прежнему не отвечает критерию универсальности и представляет собой лишь реализацию нескольких отдельно взятых задач. В результате было принято решение о необходимости и целесообразности создания собственной библиотеки длинной арифметики для GPU, и основанной на ней библиотеки криптопримитивов, для проведения разносторонних исследований по эффективности использования концепции GPGPU в области криптографии.

2.1 Библиотеки длинной арифметики

Существует множество различных библиотек длинной арифметики и арифметики произвольной точности для CPU: NTL, CLN, Miracl, GMP и пр. Некоторые из них являются также и библиотеками криптопримитивов (к примеру библиотека Miracl). Анализу и сравнению быстродействия этих библиотек была посвящена отдельная статья [5]. В этих библиотеках присутствует реализация всего необходимого для криптографических исследований математического аппарата. Изучение опыта и особенностей реализации таких библиотек, а также теории построения алгоритмов длинной арифметики в целом было необходимым условием для создания эффективной библиотеки длинной арифметики для GPU.

2.2 Внутреннее представление длинных чисел

В основе любой арифметической библиотеки лежит внутренний формат представления чисел. От него зависит как эффективность работы с памятью при выборке и изменении этих данных, так и эффективность применения различных алгоритмов обработки этих данных. В разных библиотеках этот вопрос решался по-разному. Основные три параметра, определяющих способ хранения данных, это:

- емкость разряда числа;
- порядок следования разрядов;
- способ хранения знака числа

Для начала сделаем краткий обзор способа хранения данных в различных существующих библиотеках. Для того, чтобы проиллюстрировать озвученные методы и подходы, будет использоваться синтаксис языка C++.

2.3 Библиотека VeryLong

Первым и самым естественным с точки зрения человека вариантом хранения длинных числе в памяти является их хранение в десятичной системе счисления, с порядком следования разрядов от старшего к младшему, а знак числа сохранять в виде отдельного флага. В таком случае емкость разряда такого числа — 10 значений, и для его хранения в памяти может быть использован тип char (1 байт). Такой подход к хранению длинных чисел применяется в библиотеке VeryLong:

```
class Verylong {
        ...
        char* Mass;
        bool sign;
        int length;
        …
}
```

К преимуществам такого метода можно отнести:

- данный способ интуитивно понятен и напрашивается сам собой;

• простота реализации операций ввода/вывода;

Недостатки:

• неполное использование аппаратных ресурсов машины (каждый байт в представлении числа может вмещать 256 различных значений, из которых используется лишь 10);

• при таком порядке следования разрядов (от старшего к младшему) при увеличении числа необходимо выполнение операции сдвига — копирование значений всех разрядов в соседние ячейки памяти;

• трудности реализации побитовых операций (сдвиг, логическое и, или, не и т. д.);

В сумме все недостатки данного метода приводят к крайне неэффективному использованию памяти и снижению быстродействия алгоритмов, использующих такую библиотеку. А поскольку на операции с памятью приходится основная масса накладных расходов, связанных с использованием библиотек длинной арифметики, данный способ представления длинных чисел не имеет никакой практической ценности. Очевидно, что простота конструкций и удобство восприятия не оправдывают многократное снижении эффективности алгоритмов.

2.4 Библиотека FLINT

Альтернативный подход к представлению длинных чисел можно увидеть в библиотеке FLINT, которая детально описана в книге «Криптография на Си и C++ в действии» [6]. Во-первых в этой библиотеке выбран обратный порядок следования разрядов: от младшего к старшему. Такой подход значительно упрощает процесс дополнения числа старшими разрядами или выделения дополнительной памяти для его хранения, так как отсутствует необходимость в смещении значений младших разрядов в памяти. Таким образом минимизируется количество обращений к памяти при выполнении ряда операций над числами. Во-вторых в этой библиотеке принят иной способ работы с разрядами чисел, который можно назвать универсальным для архитектуры современных вычислительных систем. Для наиболее эффективного использования памяти и аппаратных возможностей процессора нельзя обходить стороной понятие машинного слова. В любом процессоре, будь то CPU и потоковый мультипроцессор GPU, все операции выполняются над машинными словами, размер которых зависит от разрядности системы. Для достижения максимальной эффективности необходимо чтобы разряд внутреннего представления числа в библиотеке наиболее полно использовал это машинное слово. Проиллюстрируем это для случая 32-разрядной системы. Основными операциями, которые будут выполняться над разрядами чисел, являются сложение, вычитание и умножение. При сложении и вычитании мы получаем число той же разрядности, что и исходные операнды, плюс флаг переноса. При умножении двух чисел результат занимает в два раза больше памяти, чем исходные операнды. Исходя из этого для 32-разрядной системы целесообразно использовать 16-битные разряды длинных чисел. Пример из библиотеки FLINT:

```
…
typedef unsigned short clint;
typedef clint* CLINT;
…
```

В этом случае объявить указатель на область памяти для хранения длинного числа можно как

```
…
CLINT n_l;
…
```

Вопрос хранения знака числа можно решить, зарезервировав нулевой разряд внутреннего представления числа для хранения служебной информации об этом числе. Так, к примеру, в библиотеке GMP в этом разряде хранится размер памяти, выделенный для числа. При этом знак этого значения указывает на знак всего числа. Описанный таким образом подход к представлению длинных чисел является наиболее эффективным с точки зрения архитектуры современных вычислительных систем, что подтверждается использованием его во всех современных библиотеках длинной арифметики, в частности в библиотеке GMP.

2.5 Длинные числа в CUDA

Все размышления относительно представления длинных чисел в памяти в равной мере справедливы как для CPU так и для GPU ввиду идентичности их архитектур на базовом уровне. При разработке библиотеки длинной арифметики под технологию CUDA необходимо лишь учитывать особенности работы с памятью, а вернее с различными типами памяти, присутствующими в CUDA. Однако это в большей степени относится уже к алгоритмам, используемым в библиотеке, нежели к особенностям внутреннего представления чисел. Как было

упомянуто, внутреннее представление чисел библиотеки GMP полностью удовлетворяет критерию эффективности и может быть взято в качестве основы нашей библиотеки длинной арифметики для GPU, обеспечив тем самым ее совместимость на уровне формата данных с GMP. Этот путь развития был подсказан создателями проекта CUMP. Помимо эффективности он предоставляет возможность использования множества вспомогательных функций, реализованных в GMP, таких как функции ввода/вывода, конвертация между различным системами счисления, управления памятью на стороне CPU.

3 Заключение

Выбор и реализация внутреннего представления длинных чисел, это лишь первый шаг в процессе создания библиотеки длинной арифметики для GPU. Тем не менее он в значительной мере определят весь дальнейший путь развития проекта, выбор алгоритмов обработки данных, способы взаимодействия с другими библиотеками. Эта задача требует тщательного и продуманного решения, поскольку в последствии пересмотреть его будет трудно. Использование внутреннего представления чисел библиотеки GMP, являющейся на сегодняшний день лидером по скорости работы с длинными числами, представляется наиболее эффективным решением данной задачи.

Литература

[1] ECDLP on GPU [Электронный ресурс]. — Электрон. дан. — Режим доступа: http://eprint.iacr.org/2011/146.pdf, свободный — Название с титул. экрана.

[2] ECM on Graphics Cards [Электронный ресурс]. — Электрон. дан. — Режим доступа: http://hgpu.org/?p=1191, свободный. — Название с титул. экрана.

[3] The CUDA Multiple Precision Arithmetic Library [Электронный ресурс]. — Электрон. дан. — Режим доступа: http://www.hpcs.cs.tsukuba.ac.jp/~nakayama/cump/ свободный. — Название с титул. экрана.

[4] GMP «Arithmetic without limitation» The GNU Multiple Precision Arithmetic Library [Электронный ресурс]. – Электрон. дан. – Режим доступа: http://gmplib.org/, свободный. – Название с титул. экрана.

[5] Оценка эффекивности использования библиотек длинной арифметики // Неласая А.В, Верещак М.И. // Радіоелектроніка. Інформатика. Управління. — 2010. — №23 — С. 67-73

[6] Вельшенбах, М. Криптография на Си и C++ в действии. Учебное пособие [Текст] / М. Вельшенбах. – М.: Издательство Триумф, 2004. – 464 с.