Перспективы применения архитектуры CUDA для решения задач реального времени в системах подвижной связи

Александр Тумачек (Москва)

Предложено использование процессоров с массовым параллелизмом для решения задач пространственно-временной обработки сигналов режиме реального времени в системах подвижной связи. Показаны преимущества и целесообразность применения архитектуры CUDA.

В настоящее время в научной литературе можно отметить значительный интерес к системам подвижной радиосвязи с применением пространственно-временной обработки сигналов (ПВО). Это объясняется, во-первых, возрастающими требованиями к качественным характеристикам систем, стремлением повысить их помехоустойчивость, быстродействие и ёмкость, во-вторых – увеличением сложности систем передач, развитием вычислительных средств и теории ПВО, открывающим новые пути для улучшения характеристик [6].

Основной задачей пространственно-временной обработки сигналов в системах подвижной связи является анализ результирующего поля с целью определения положения наблюдаемого объекта и его скорости. Алгоритмы пространственно-временной (или поляризационно-временной) обработки являются дополнениями к стандартным алгоритмам приёма сигналов и могут решать целый ряд практически важных задач [1–4].

Помехоустойчивые системы подвижной связи работают на основе анализа информации о волновых полях, создаваемых источниками излучения. Структура и параметры волнового поля, создаваемого удалённым объектом в той области, где осуществляется анализ этого поля, зависят от положения и скорости движения объекта относительно этой области и от характеристик самого объекта (его размеров, формы, колебаний относительно центра масс и т.д.). Поэтому такое волновое поле несёт информацию об источнике поля – абоненте – и в этом смысле может рассматриваться как пространственно-временной сигнал.

Однако для извлечения информации в системе связи используется не всё поле, излучаемое мобильной станцией, а лишь ограниченный его участок, попадающий в апертуру приёмной антенны. Воздействуя на элементы антенны, это поле образует пространственно-временной сигнал, обрабатываемый радиосистемой. В области наблюдения, кроме поля, несущего информацию об источнике сигнала, могут находиться и поля, создаваемые другими объектами и внешними источниками помех.

Системы подвижной связи (СПС) приобретают всё большее развитие как по масштабу, так и по количеству услуг связи. Электромагнитная обстановка (ЭМО) в СПС обладает существенными особенностями и характеризуется большой динамикой, нестационарностью и неравномерностью параметров. В условиях увеличивающейся плотности абонентских мобильных станций СПС возможно возникновение такой ЭМО, при которой наблюдается значительное ухудшение условий работы абонентов и качества приёма. Такая ситуация характерна для большого скопления источников сигналов: городских районов, территорий со сложной сигнально-помеховой обстановкой (электростанции, ЛЭП) и т.д. Попытки решения этой проблемы традиционными средствами не всегда успешны. Обычно применяются частотно-временные, кодовые и энергетические способы, однако тенденция постоянного усложнения ЭМО предполагает изыскание дополнительных методов помехозащиты систем и линий связи [1–3].

С появлением высокопроизводительных средств цифровой обработки сигналов (ЦОС), практическая реализация алгоритмов ПВО, действующих в СПС, значительно упрощается. Перспективной является реализация неграфических вычислений на графических процессорах. С развитием подхода GP GPU и появлением новой программно-аппаратной архитектуры CUDA появилась возможность быстро реализовывать сложные, ресурсоёмкие алгоритмы обработки в СПС.

Стоимость программно-аппаратного решения, основанного на графическом вычислителе, намного ниже по сравнению со специализированными инструментальными средствами. Вычислительные мощности процессора с массовым параллелизмом и платформа CUDA дают возможность реализовать многие наработанные идеи, которые ранее были недоступны в практической области пространственно-временной обработки сигналов. Обработка сигналов в СПС представляет собой задачу реального времени и требует не только больших вычислительных возможностей, но и детальной оптимизации алгоритмов ЦОС. Для решения задач обработки сигналов с заданной точностью необходимо осуществить выбор, оценку и реализацию подходящих математических методов.

Технология CUDA – это программно-аппаратная вычислительная архитектура фирмы NVIDIA, основанная на расширении языка Си, которая даёт возможность организации доступа к набору инструкций графического ускорителя и управления его памятью при организации параллельных вычислений. Технология CUDA помогает реализовывать алгоритмы, выполнимые на графических процессорах видеоускорителей GeForce восьмого поколения и старше (серии GeForce 8, GeForce 9, GeForce 200), а также Quadro и Tesla.

Трудоёмкость программирования GPU при помощи CUDA ниже, чем в ранних решениях GP GPU. Такие программы требуют разбиения приложения между несколькими мультипроцессорами, подобно MPI-программированию, но без разделения данных, которые хранятся в общей видеопамяти. CUDA-программирование для каждого мультипроцессора, подобно OpenMP-программированию, требует хорошего понимания организации памяти. Кроме того, сложность разработ-

ки и переноса на платформу CUDA сильно зависит от приложения.

Первым шагом при переносе существующего приложения на CUDA является его профилирование и определение участков кода, тормозящих работу. Если среди таких участков есть подходящие для быстрого параллельного исполнения, эти функции переносятся на расширения CUDA для выполнения на GPU. Программа компилируется при помощи поставляемого фирмой NVIDIA компилятора, который генерирует код и для CPU, и для GPU. При исполнении программы центральный процессор выполняет свои фрагменты кода, а GPU выполняет CUDA-код с наиболее «тяжёлыми», параллельными вычислениями. Эта часть, предназначенная для GPU, называется ядром. В ядре определяются операции, которые будут исполнены над данными.

Каждый мультипроцессор в составе GPU состоит из восьми ядер – потоковых процессоров, которые выполняют одну инструкцию умножения с накоплением (MAC) за один такт. Для исполнения одного 32-поточного блока требуется четыре такта работы мультипроцессора (на частоте 1,5 ГГц и выше).

Мультипроцессор не является традиционным многоядерным процессором; он приспособлен для многопоточности, поддерживая до 32 потоков одновременно. На каждом такте аппаратное обеспечение выбирает, какой из потоков исполнять, и переключается между ними без потерь на ожидание. Если проводить аналогию с центральным процессором, это похоже на одновременное исполнение 32 программ и переключение между ними без потери контекста. Реально ядра CPU поддерживают единовременное выполнение одной программы и переключаются на другие программы с задержкой в сотни тактов.

Естественно, в рамках обзорной статьи невозможно рассмотреть серьёзные вопросы оптимизации в CUDA-программировании. Поэтому мы расскажем о базовых принципах. Для эффективного использования возможностей CUDA следует забыть о традиционных методах написания программ для CPU и использовать алгоритмы, которые хорошо распараллеливаются на тысячи потоков. Важно найти оптимальное место для хранения данных (регистры, разделяемая память и т.п.), минимизировать передачу данных между CPU и GPU и широко использовать буферизацию [5].

Оценки производительности

Размерность, *N*	*t*, мс, на CPU	*t*, мс, на GPU
5	0,83968	0,112389
10	0,144613	0,10871
50	0,101279	0,113772
256	0,479118	0,147543
394	0,927336	0,183049
512	1,539904	0,223042
768	3,225049	0,285549
1024	5,792056	0,355544
1280	9,184576	0,424613
1792	17,657446	0,268173
2048	23,069201	0,679513
10 000	548,533386	10,7951
11 000	686,119507	15,733009
12 735	917,503418	26,757336

Для осуществления расчётов в реальном времени имеется инструментарий библиотечных процедур: QR- и SVD-разложения, алгоритмы Холецкого и LU; также существуют процедуры решения СЛАУ и пакет векторно-матричных операций. В проекте Komrade и Thrust разработаны библиотеки CUDA с STL-интерфейсом [9]. Всё это позволяет сократить затраты на разработку быстродействующего ПО реального времени.

В ходе эксперимента была реализована базовая процедура перемножения векторов с накоплением и получены оценки производительности (см. таблицу).

В заключение следует отметить, что современные тенденции увеличения количества и качества передаваемой информации требует от СПС высоких вычислительных возможностей. Одним из перспективных направлений является применение графических процессоров для целей неграфических вычислений (GP GPU). Эта идея не нова, но ранее имела ограничения, обусловленные сложностью реализации аппаратного обеспечения.

Разработка приложений была исключительно узконаправленной и требовала больших усилий. С развитием подходов к программированию на GP, появлению новых программно-аппаратных архитектур, таких как CUDA, это направление становится более актуальным. Технология CUDA предоставляет возможность в кратчайшие сроки осуществлять разработку программного обеспечения, отвечающего высоким требованиям и стандартам обработки информации в реальном масштабе времени.

Следует отметить массовость производства подобных решений для промышленности, игровой индустрии и научных исследований. Это обязывает производителя осуществлять постоянную поддержку программно-аппаратных решений. На подобных платформах эффективно решаются задачи пост-обработки и обработки в реальном масштабе времени. Это возможно за счёт высокой пропускной способности шин и поддержки SLI-режима микросхемами видеокарты. Решения Tesla 1060 и 1070 обладают достаточным объёмом оперативной памяти для обработки данных внутри устройства, что исключает потери производительности алгоритмов ПВО на промежуточных операциях ввода/вывода.

Литература

1. *Уидроу Б., Стирнз С.* Адаптивная обработка сигналов. Радио и связь, 1989.
2. *Пистолькорс А.А., Литвинов О.С.* Введение в теорию адаптивных антенн. Наука, 1991.
3. *Родимов А.П., Поповский В.В.* Статистическая теория поляризационно-временной обработки сигналов и помех. Радио и связь, 1984.
4. *Монзинго Р.А., Миллер Т.У.* Адаптивные антенные решётки: Введение в теорию. Радио и связь, 1986.
5. *Берило А.* NVIDIA CUDA неграфические вычисления на GPU. М., 22.10.2008.
6. *Кремер И.Я., Кремер А.И., Петров В.М.* и др. Пространственно-временная обработка сигналов. Под ред. *И.Я. Кремера*. Радио и связь, 1984.
7. *Кублановская В.Н.* Первая публикация по QR-алгоритму в Дополнении к изданию 1960 г. монографии *Д.К. и В.Н. Фадеевых* «Вычислительные методы и линейная алгебра».
8. *Парлет Б.* Симметричная проблема собственных значений: Численные методы. Мир, 1983.
9. http://code.google.com/p/komrade/wiki/Tutorial.