РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ ВИЗУАЛИЗАЦИИ СИСТЕМ ЧАСТИЦ НА ВЫСОКОПРОИЗВОДИТЕЛЬНЫХ GPU СИСТЕМАХ

С.А. Копылов

Технологический институт Южного Федерального университета, Таганрог

Для визуализации сложных эффектов необходимо устройство с параллельной архитектурой. Поэтому для этих целей были выбраны именно графические процессоры. В разрабатываемом приложении для получения более эффективных результатов интегрирована поддержка технологии для параллельных вычислений Nvidia CUDA, так же разработана возможность работы приложения на гибридных системах, для одновременного использования CPU и GPU.

Графические процессоры (GPU) являются параллельными архитектурами. GPU обладает своей памятью (DRAM), объем которой уже достигает 3 Гбайта для некоторых моделей. Также GPU содержит ряд потоковых мультипроцессоров (SM, Streaming Multiprocessor), каждый из которых способен одновременно выполнять 768 (1024 для более поздних моделей) нитей. При этом количество потоковых мультипроцессоров зависит от модели GPU. Как можно увидеть на рис. 1, в состав GTX 550 Ti входит 192 потоковых процессора, собранных в четыре мультипроцессора по 48 штук в каждом. Каждый мультипроцессор работает независимо от остальных.

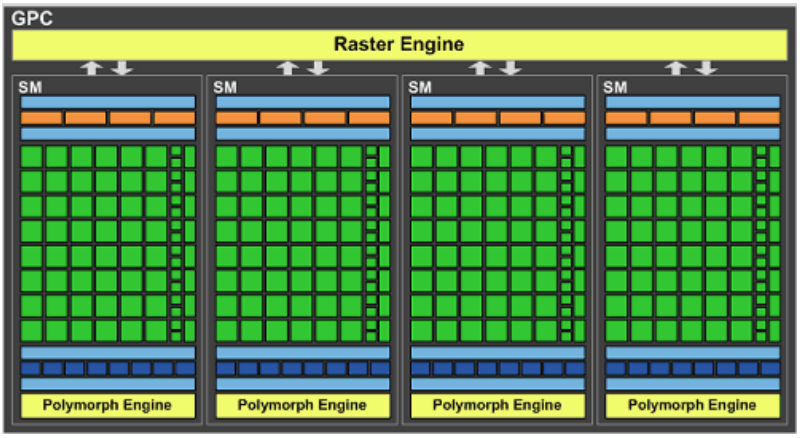

Рис 1. Схема GPU в GF106

Одним из серьезных отличий между GPU и CPU является организация памяти и работа с ней. Обычно большую часть CPU занимают кеши различных уровней. Основная часть GPU отведена на вычисления. Как следствие, в отличие от CPU, где есть всего один тип памяти с несколькими уровнями кеширования в самом CPU, GPU обладает более сложной, но в то же время гораздо более документированной структурой памяти [1].

Разделяемая память, размещенная непосредственно в самом мультипроцессоре и доступная на чтение и запись всем нитям блока, является одним из важнейших отличий CUDA от традиционного (то есть основанного на использовании графических API) GPGPU. Правильное использование разделяемой памяти играет огромную роль в создании эффективных программ для GPU. Когда на CUDA запускается ядро на выпол-

нение, то блоки сетки выполняются на имеющихся мультипроцессорах. При этом каждый блок целиком выполняется на одном из мультипроцессоров, мультипроцессор же способен одновременно выполнять до восьми блоков. По мере того как отдельные блоки завершают свое выполнение, на их место становятся новые блоки. Таким образом, можно даже на довольно небольшом числе потоковых мультипроцессоров запустить на выполнение сетку с очень большим числом блоков [2].

Основной целью разработки является воссоздание физической сцены. Она будет содержать большое количество визуализируемых элементов. После проведенных испытаний можно будет получить результаты производительности, которые должны отображать преобладание в эффективности CPU или GPU. Реализация алгоритмов компьютерной симуляции базовых физических законов – очень трудоемкий процесс. Были проведены опыты, с помощью которых было доказано, что, какой бы сложный рендеринг ни выполнял GPU, часть его ядер все равно простаивает. Специфика моделирования поведения и взаимодействия большого количества сложных объектов, а также сложных систем частиц, использующихся для создания воды, дыма и т.п., требует применения параллельных вычислений, в противном случае есть риск получить неудовлетворительную производительность. Для реализации в 3D-приложениях физической модели, максимально приближенной к реальной, необходимо вычислительное устройство с параллельной архитектурой, которое в состоянии быстро выполнить ряд довольно сложных расчетов. Для полной нагрузки ядер будет смоделирована сцена с использование систем частиц.

В настоящее время системы частиц очень популярны для получения реалистичных эффектов, но они очень трудоемки при просчете, поэтому для этих целей были использованы GPU.

Для создания частицы нужно установить четыре вершины, образующие два полигона (в целях оптимизации обычно используется полоса треугольников). Координаты вершин определяют исходный размер частицы, который в дальнейшем будет масштабирован. У каждой частицы есть собственные уникальные свойства, включая ее собственный цвет (который реализуется с помощью материала). Затем используется эта структура в комбинации с единым буфером вершин, содержащим два полигона (образующие квадрат), чтобы визуализировать полигоны на устройстве трехмерной графики. Перед началом рисования каждая частица ориентируется посредством собственной мировой матрицы. В завершении для рендеринга нужно скомбинировать матрицу мирового преобразования частицы с ее же матрицей преобразования масштабирования [3].

Предполагается провести испытания с разным количеством частиц, после чего можно собрать результаты и представить общую картину. Последним этапом на многопроцессорной GPU-системе будет рассчитываться сцена с большим количеством частиц, что покажет эффективность алгоритмов распараллеливания.

Основная концепция, предлагаемая в разработке, заключается в том, что для достижения максимальной степени реализма приложение должно состоять из трех основных компонентов, каждый из которых берет на себя обработку соответствующего 3D-приложения. Этими компонентами непосредственно являются CPU, GPU+PPU (на современных графических адаптерах это является одним целым). Центральный процессор в такой связке занимается процессом взаимодействия с пользователем и задачей графического процессора является рендеринг и отображение 3D-сцены, а на плечи PPU (Physical Processing Unit) ложится вся нагрузка по просчету физической модели 3D-мира. Другими словами, PPU в этой концепции отвечает за движение и взаимодействие всех объектов в сцене, например за поведение жидкостей и обломков, а в данной разработке за расчет систем частиц.

Литература

1. CUDA Programming Guide 1.1.
2. Боресков А.В. Харламов А.А. Основы работы с технологией CUDA. М.: Изд-во ДМК Пресс, 2010. 232 с. ISBN 978-5-94074-578-5.э
3 Адамс Д. DirectX: продвинутая анимация. Комплект. – «КУДИЦ-ПРЕСС», 2004. – 480 с. – ISBN 5-9579-0025-7.