Institut für Informatik — TU München WS 2010/11
Scientific **C**omputing in **C**omputer **S**cience
Dipl.Inf. T. Auckenthaler, Dipl.Inf. W. Eckhardt

Masterpraktikum Scientific Computing (High Performance Computing) Übungsblatt 4: GPGPU - General Purpose GPU Programmierung mit NVidia Cuda

Zur Übung am 21.12.2009

Aufgabe 11 „Einarbeitung in CUDA I“(1 Punkte)

Machen Sie sich mit den CUDA zugrundeliegenden Konzepten vertraut. Verschaffen Sie sich hierzu einen Überblick über die Kapitel 1, 2, 3 bis einschliesslich 3.2.3, und 4 des `NVidia CUDA Programming Guide (Version 2.3)`!

Beantworten Sie stichpunktartig die folgenden Fragen:

- Wie läuft der Übersetzungsvorgang eines CUDA-Programmes ab?
- Wie ist der grundlegende, schematische Ablauf eines CUDA-Programms?
- Was sind die laut dem Handbuch markanten Architekturunterschiede zwischen CPU und GPU?
- Welche Unterschiede besteht zwischen CPU- und GPU-Threads?
- Wie ist der Zusammenhang zwischen der Thread-Hierarchie und der Speicherhierarchie durch die Graphikkaren-Hardware begründet?
- Was versteht man unter der Compute Capability einer Nvidia Graphikkarte? Welche Compute Capability steht Ihnen auf dem LRZ-Cluster zur Verfügung? Welche Bedeutung hat dies für Fliesskommaberechnungen?

Aufgabe 12 „Einarbeitung in CUDA II“(0,5 Punkte)

Lösen Sie die Einführungsaufgaben von NVidia. Beachten Sie die Anleitung zu den Aufgaben, sowie den „CUDA Programming Guide “und das „CUDA Reference Manual “!

`Anmerkung` Übung 5: Je nach verwendeter Version heissen die Flags des Cuda-Profilers `gld uncoalesced` statt `GLD_INCOHERENT`, etc...

Aufgabe 13 „Gauss-Filter I“(1,5 Punkte)

Von Übungsblatt 3 ist der Gauss-Filter bekannt. Entwickeln Sie ein einfaches CUDA-Programm für die Graphikkarte. Verwenden Sie einfache single-precision-floating-point-Arithmetik. Es genügt, wenn Sie ausschliesslich globalen Speicher verwenden. Messen Sie die erreichbare FLOP-Rate und vergleichen Sie sie mit der Performance der simplen Implementierung für die CPU und der parallelisierten Fassung aus `Aufgabe 10`.

Aufgabe 4 „Gauss-Filter II“(3 Punkte)

Das Programm aus Aufgabe 3 kann noch verbessert werden. Analysieren Sie das Programm mit dem Profiler, überlegen Sie sich eine geeignete Implementierung und realisieren Sie diese!
Analysieren Sie das Verhalten Ihres Programmes nochmals mit dem Profiler, und messen Sie die erreichbare FLOPS-Rate.
Hierzu gehen Sie am besten in Schritten vor:

- Beschreiben Sie das Problem der Implementierung aus Aufgabe 3.
- Welche 2 Möglichkeiten fallen Ihnen ein, um die Daten auf Threads aufzuteilen? Welche davon wird effizienter sein? Berücksichtigen Sie Branch-Divergenz und Coalesced Memory Access.
- Überlegen Sie sich nun eine geeignete Anordnung der Daten im Speicher. Implementieren dieses Muster und verifizieren Sie es mit Hilfe des Profilers (z.b. indem Sie jeden Thread seine Id in ein Element eines Arrays schreiben lassen).
- Implementieren Sie nun den Gauss-Filter.

Hinweise:

- Sie können die Programmieraufgaben auch ohne CUDA-fähige Hardware lösen. Dazu verwenden Sie die Option `--device-emulation` des NVidia Compilers.
- Im Emulationsmodus stehen Ihnen sämtliche Debugging-Möglichkeiten ihres Systems zur Verfügung (siehe Abschnitt 3.2.10 des Programming Guide).
- Darüberhinaus steht Ihnen der CUDA-GDB zur Verfügung!

Viel Erfolg beim Bearbeiten!

Die Abgabe ist bis 18.01.2010, 9.00 Uhr möglich.

Alle Programme finden Sie zum Herunterladen auf der Praktikumsseite unter:
`http://www5.in.tum.de/wiki/index.php/Masterpraktikum_Scientific_Computing_-_High`