Institut für Informatik
Henning Wenke M.Sc.
Sascha Kolodzey B.Sc.
Nils Vollmer B.Sc.

Universität Osnabrück
http://www-lehre.inf.uos.de/~pa/

Übung: Parallele Algorithmen mit OpenCL

Sommersemester 2013

Blatt4

Laden Sie sich den in der Übung vorgestellten OpenCL Wrapper für Aufgabe 4.2 und 4.3 unter folgender Adresse http://www-lehre.inf.uos.de/~pa/Uebungen/Blatt4/PA_Blatt4.zip herunter.

Aufgabe 4.1 (20 Punkte) OpenCL Fehlerkontrolle

Erklären Sie Ihrem Tutor die Funktionsweise der in OpenCL integrierten Fehlerkontrolle.

Aufgabe 4.2 (30 Punkte) OpenCL Programbuild Info

Implementieren Sie die Methode `printProgramInfo` in der Klasse `CLUtil` dahingehend, dass Sie Informationen des OpenCL Programs und eventuelle Fehlerlogs auf der Konsole ausgibt. Benutzen Sie dazu die Funktion `clGetProgramBuildInfo` mit den jeweiligen Parametern, um an die Informationen zu gelangen. Informationen zu der Funktion und wie sie zu benutzen ist sind in der OpenCL Dokumentation zu finden.
Die Ausgabe soll nach folgender Struktur geschehen:

```
‘‘ProgramInfo:’’  {’’Status=’’status}, {’’Options=’’options]
[‘‘ErrorLog=’’errorlog]
```

wobei `status`, `options` und gegebenenfalls `errorlog` durch die jeweiligen Werte ersetzt werden.

Aufgabe 4.3 (50 Punkte) Matrixmultiplikation

Implementieren Sie die in der Vorlesung vorgestellte Matrixmultiplikation unter Verwendung von OpenCL. Legen Sie dazu eine neue Java Klasse `MatrixMultiplication` im package `main` des installierten Wrappers an. Sie können sich an folgender Programmstruktur orientieren:

- Platform/Device auswählen
- Context erstellen
- CommandQueue erstellen
- Program erstellen
- Kernel erstellen
- Memory Objects anlegen
- Kernel vorbereiten
- Kernel ausführen
- Daten zurück lesen und ausgeben
- Speicher aufräumen

Erstellen Sie im Ordner `programs` eine neue Datei `MatrixMul.cl`, in die Sie den Sourcecode des Kernels zur Matrixmultiplikation schreiben. Lesen Sie den Sourcode mit der Hilfsmethode `readFileContent` ein und übergeben Sie diesen beim Erstellen des OpenCL Programs.

Der zu schreibende Kernel soll jeweils einen Eintrag C_{ij} der Matrixmultiplikation $C = A \cdot B \; mit \; A = (a_{ij})_{i=1...l,j=1...m}$, $B = (b_{ij})_{i=1...m,j=1...n}$, $C = (c_{ij})_{i=1...l,j=1...n}$ berechnen. Die Dimensionen der Matrizen A und B sollen im Programm über Variablen festgelegt werden können. Achten Sie darauf, dass ein Fehler geworfen wird, sobald eine Multiplikation nicht möglich ist. Füllen Sie die zu multiplizierenden Matrizen mit zufällig erzeugten Integer Werten. Zum Erzeugen von reproduzierbaren Zufallszahlen benutzen Sie die entsprechende `next*` Methode der Klasse `MathUtil`. Geben Sie nach der Multiplikation den Inhalt der Matrix C auf der Konsole aus. Benutzen Sie für die Ausgabe auf der Konsole die Methode `printBuffer` der Klasse `BufferHelper`.

Es ist Ihnen freigestellt, ob Sie eine eindimensionale oder zweidimensionale Indizierung der Kernel verwenden.