

**Laborprotokoll**

**GPGPU**

**Dezentale Systeme**

**5BHITT 2015/16**

**Michael Borko**

|  |  |
| --- | --- |
|  | **Version 0.2** |
| **Note:** | **Begonnen am 8. Jänner 2016** |
| **Betreuer: Borko** | **Beendet am** |

Inhaltsverzeichnis

[1Einführung 3](#__RefHeading___Toc105_1963609346)

[1.1Ziele 3](#__RefHeading___Toc107_1963609346)

[1.2Voraussetzungen 3](#__RefHeading___Toc109_1963609346)

[1.3Aufgabenstellung 3](#__RefHeading___Toc111_1963609346)

[2Ergebnisse 4](#__RefHeading___Toc113_1963609346)

# Einführung

## Ziele

Die Aufgabe beinhaltet eine Recherche über grundsätzliche Einsatzmöglichkeiten für GPGPU. Dabei soll die Sinnhaftigkeit der Technologie unterstrichen werden. Die Fragestellungen sollen entsprechend mit Argumenten untermauert werden.

Im zweiten Teil der Arbeit soll der praktische Einsatz von OpenCL trainiert werden. Diese können anhand von bestehenden Codeexamples durchgeführt werden. Dabei wird auf eine sprechende Gegenüberstellung (Benchmark) Wert gelegt.

Die Aufgabenstellung soll in einer Zweiergruppe bearbeitet werden.

## Aufgabenstellung

* Informieren Sie sich über die Möglichkeiten der Nutzung von GPUs in normalen Desktop-Anwendungen. Zeigen Sie dazu im Gegensatz den Vorteil der GPUs in rechenintensiven Implementierungen auf [1Pkt].
* Gibt es Entwicklungsumgebungen und in welchen Programmiersprachen kann man diese nutzen [1Pkt]?
* Können bestehende Programme (C/C++ und Java) auf GPUs genutzt werden und was sind dabei die Grundvoraussetzungen dafür [1Pkt]?
* Gibt es transcompiler und wie kommen diese zum Einsatz [1Pkt]?

Präsentieren Sie an einem praktischen Beispiel den Nutzen dieser Technologie. Wählen Sie zwei rechenintensive Algorithmen (z.B. Faktorisierung) und zeigen Sie in einem aussagekräftigen Benchmark welche Vorteile der Einsatz der vorhandenen GPU Hardware gegenüber dem Ausführen auf einer CPU bringt (OpenCL).

## Punkteschlüssel

* Auswahl und Argumentation der zwei rechenintensiven Algorithmen (Speicher, Zugriff, Rechenoperationen) [0..4Pkt]
* Sinnvolle Gegenüberstellung von CPU und GPU im Benchmark [0..2Pkt]
* Anzahl der Durchläufe [0..2Pkt]
* Informationen bei Benchmark [0..2Pkt]
* Beschreibung und Bereitstellung des Beispiels (Ausführbarkeit) [0..2Pkt]

# Ergebnisse

## Transcompiler

Um normalen Code in OpenCl–Code umwanden zu können, wird ein entsprechender Transcompiler Compiler benötigt.

### Aparapi

Eine Möglichkeit ist das von AMD entwickelte Projekt Aparapi [1]. Aparapi verfügt über eine ausführliche Dokumentation und bringt viele Code-Examples mit sich.

# Literaturverzeichnis

[1] GPGPU-Transcompiler; Apirapi [online]

verfügbar unter: https://github.com/aparapi/aparapi/

[zuletzt abgerufen am 15.01.2015]