Inhaltsverzeichnis

1 Einführung und Grundlagen

1.1 Aufgabenstellung

1.2 Grundlagen

1.2.1 AES im Kurzüberblick

Der Advanced Encryption Standard (AES) ist ein symmetrisches Kryptosystem. Es wurde von Joan Daemen und Vincent Rijmen im Rahmen eines international ausgeschriebenen Wettbewerbes des National Institute of Standards and Technology (NIST) entwickelt. Als Nachfolger von DES und 3DES, gilt AES seit 2000 als De-facto Verschlüsselungsstandard, welcher Dank seiner starken Verschlüsselung selbst höchsten Sicherheitsansprüchen genügt.

Bei AES handelt es sich um ein Blockverschlüsselungssystem, auch Blockchiffre genannt, also ein Verschlüsselungsverfahren, bei dem der Klartext in eine Folge gleichgroßer Blöcke zerlegt wird. Diese Blöcke werden anschließend unabhängig voneinander mit einem aus einem Schlüsselwort berechneten Blockschlüssel chiffriert. Somit werden auch Chiffretextblöcke mit einer festen Länge erzeugt und letzendlich zum endgültigen Chiffretext aneinandergereiht.

AES schränkt die Blocklänge auf 128 Bit ein. Die Schlüssellänge kann jedoch zwischen 128, 192 und 256 Bit gewählt werden, weshalb zwischen den drei AES-Varianten AES-128, AES-192 und AES-256 unterschieden wird. AES bietet ein sehr hohes Maß an Sicherheit und ist in den USA sogar für staatliche Dokumente mit höchster Geheimhaltungsstufe zugelassen. Der Algorithmus ist frei verfügbar und darf ohne Lizenzgebühren eingesetzt sowie in Software bzw. Hardware implementiert werden.

1.2.2 CUDA Framework

Das „Compute Unified Device Architecture Software Developer Kit“ (CUDA SDK) wurde von NVIDA am 15. Februar 2007 erstmals der Öffentlichkeit vorgestellt. Intention dieses SDKs ist, die Programmierung aktueller Grafikkarten unter einer einheitlichen und standardisierten Schnittstelle zu ermöglichen.

Die Architektur moderner GPUs ist aufgrund ihrer Geschichte als reine Berechnungseinheit für Bildschirmausgaben für den Zweck ausgelegt, Operationen parallel auszuführen. Diese Parallelität kann dazu genutzt werden, Algorithmen erheblich schneller auszuführen, sofern sie Teile enthält, deren Berechnungen unabhängig voneinander durchgeführt werden können.

CUDA basiert auf einer angepassten Variante von C und ist damit weitestgehend plattformunabhängig. So ist es möglich, „CUDA-Programme“ unter Windows, Linux und Mac OS X auszuführen - eine kompatible Grafikkarte vorausgesetzt.

2 GPU Architektur

2.1 Historie

Ursprünglich war es die einzige Aufgabe einer Grafikkarte, ein Bild auf einem Anzeigegerät (wie z.B. einem Monitor) darzustellen. Im Laufe der Geschichte haben sie sich in programmierbare Prozessoren entwickelt. Aufgrund ihrer Geschichte aus der Grafikberechnung, sind Grafikarten in der Lage, sehr viele Operationen parallel zu berechnen. Wie genau das von statten geht, wir im Kapitel der Architektur erwähnt. Durch Ausnutzen der Parallelität der Grafikkarte in Berechnungen kann gegenüber der Berechnung mit der CPU schneller berechnet werden.

2.2 Architektur

2.2.1 Prozessoraufbau

Moderne Grafikarten sind über die PCI-E Schnittstelle an die CPU angebunden. Über diesen Bus werden die Daten und Prozesse an die berechnenden Einheiten der GPU übertragen.

Im Folgenden ist die Architektur der GPU abgebildet. Hierbei ist zu beachten, dass die Anzahl der Multiprozessoren auf der GPU, sowie die Anzahl der Streaming-Prozessoren (SP) je nach Modell unterschiedlich sind.

2.2.2 Multithreaded Instruction Unit (MT IU)

Die MT IU verwaltet die Ausführung von Threads auf dem Multiprocessor. Hierbei werden einzelne Threads zu einem Block, mehrere Blöcke zu einem Grid zusammengefasst.

2.2.3 Streaming Multiprocessor

Auf jeden Streaming Multiprocessor wird genau ein Block abgebildet. Die in dem Block befindlichen Threads werden -soweit möglich- parallel abgearbeitet. Hierbei hat jeder Thread in dem Block eine eindeutige ID, auf welche auch in dem Thread zugegriffen werden kann.

2.2.4 Streaming Processor (SP)

Jeder SP führt genau einen Tread aus.

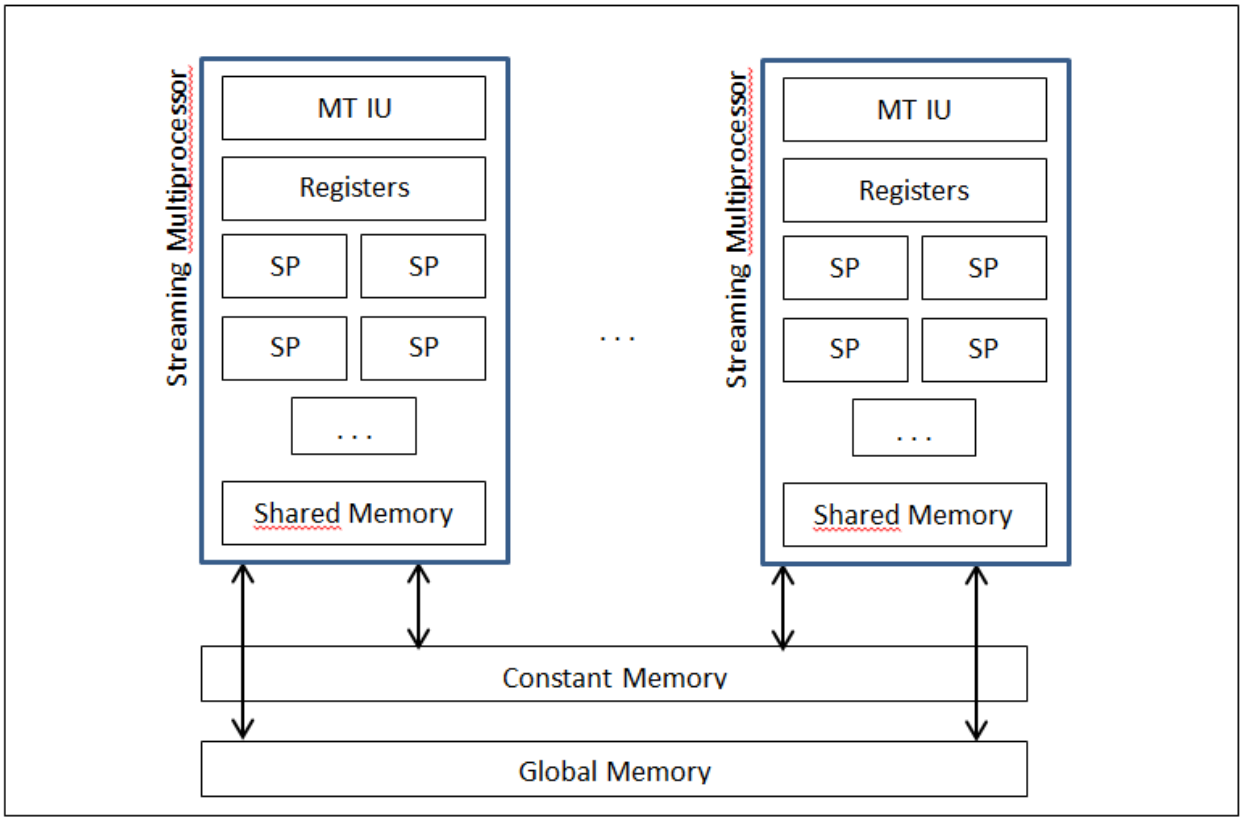

Abbildung 1: GPU-Architecture

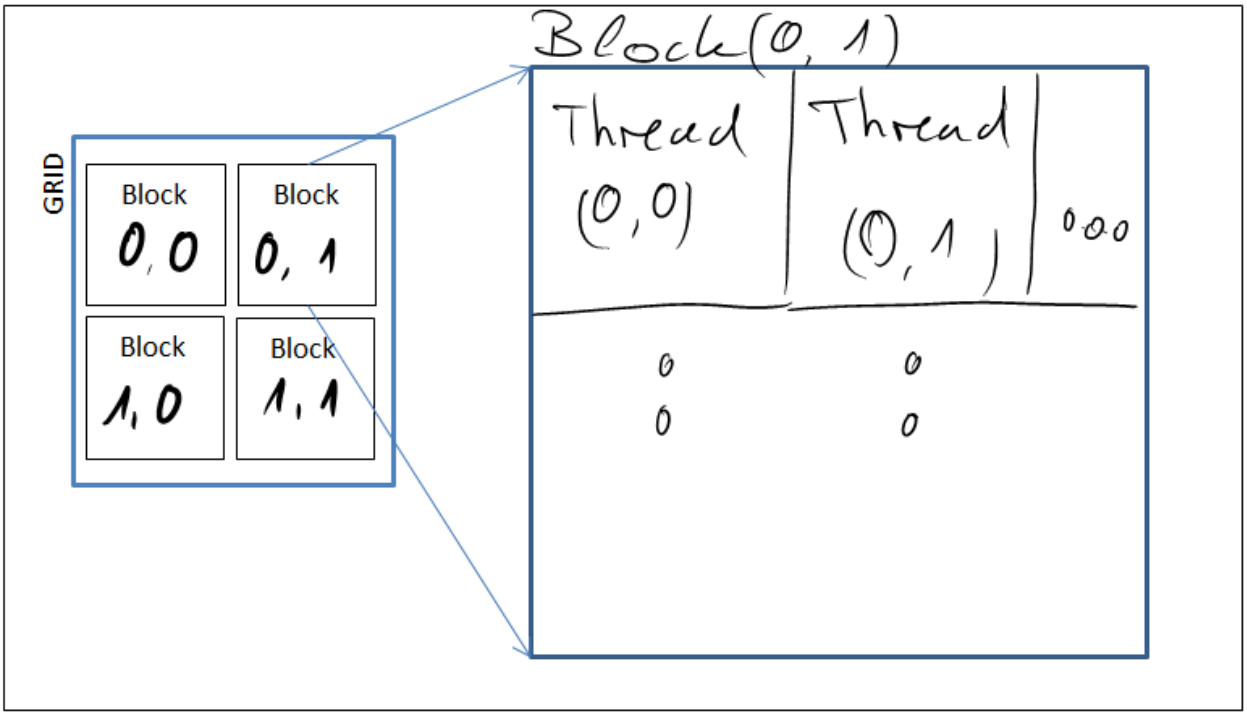

Abbildung 2: Streaming Processor

2.3 Speicherhierarchie

2.3.1 Global Memory

Der Global Memory (RAM) ist der Größte Bereich und sowohl von CPU als auch von der GPU schreib- und lesbar. Dieser Speicher ermöglicht den Austausch von Daten zwischen GPU und CPU. Dieser Speicher hat die größte Kapazität, ist jedoch der langsamste der Speicherhierarchie bezüglich der GPU.

2.3.2 Constant Memory

Der Constant Memory ist physikalisch auf 64KB beschränkt.

2.3.3 Register

Threads eines Bocks teilen sich gemeinsame Register.

Shared Memory

Für jeden Streaming-Multiprozessor-Prozessor ist ein Shared Memory vorgesehen, welches der schnellste -mit 16 KB jedoch auch der kleinste- Speicher in der Hierarchie der GPU ist. Der Multiprozessor teilt diesen verfügbaren Speicher und seinen Streamingprozessoren auf. Dieser Speicher kann nur von der GPU gelesen und geschrieben werden.

3 Implementierung

3.1 Funktionen im Detail

3.1.1 Galois-Feld-Theorie etc.

3.1.2 MixColumn-Funktion

3.1.3 Substitutionsbox

3.1.4 ShiftRow-Funktion

3.2 CUDA-spezifische Veränderungen

3.2.1 Prozessaufteilung

3.2.2 Speichernutzung

4 Tests und Benchmarks

4.1 Testumgebung

4.2 Ergebnisse

5 Ausblick