Vorlesung Architekturen und Entwurf von Rechnersystemen

TECHNISCHE UNIVERSITÄT DARMSTADT

Wintersemester 2018/2019

Übungsblatt 7

Prof. Andreas Koch, Jaco Hofmann

Alle Teilkomponenten sind nun entworfen. Es fehlt nur noch das Topmodul, das alle miteinander verbindet.

Aufgabe 7.1 Topmodul

Machen Sie sich bewusst, wie sie den `RowBuffer` und den `Filter` zusammenschließen können. Den 7x7 Ausschnitt aus dem Bild können Sie in 49 Registern speichern, wenn sie nicht mit den `mkBuffer` arbeiten möchten. Parametriesieren Sie das Topmodul auch über die Bildbreite und über den Wert, der dem Filter übergeben wird, wenn der eigentliche Pixel nicht verfügbar ist (Randbehandlung).

Aufgabe 7.2 Testen der Implementierung

Wir stellen Ihnen eine C-Implementierung und ein Kompilierungsskript, sodass Sie ihre Bluespecimplementierung mit einem richtigen .png-Bild testen können. Darüberhinaus stellen wir Ihnen eine Beispeilimplementierung, damit Sie sehen, wie die C-Funktionen zu verwenden sind.

Aufgabe 7.3 Erweiterung: Durchsatz, Latenz und kritischer Pfad

Die bereitgestellte Lösung ist auf die Taktrate getrimmt. Wenn ein Bild eingelesen wird, dann dauert es genau 3*Bildbreite + 5 Takte, bis der erste Pixel aus dem Modul herauskommt. Woher kommt dieser eine Takt? Es dauert 3*Bildbreite + 4 Takte, bis der erste Pixel in der Mitte des Filters angekommen ist und alle Module brauchen 0 Takte zum Berechnen des Ergebnisses. Können Sie auch diesen letzten Takt wegoptimieren? Analysieren Sie zudem den kritischen Pfad Ihrer gesamten Implementierung und der gegebenen Lösung. Notieren Sie sich dazu jede Aktion, die ausgeführt wird. Schätzen Sie den kritischen Pfad ab, nutzen Sie dabei folgende Werte für die Dauer der einzelnen Aktionen:

- `Addition`: 3 ns
- `Multiplikation`: 6 ns
- `AND, OR, XOR, ...`: 1 ns
- `Vergleich`: 4 ns
- `Dequeue, Enqueue, reg.write, reg.read`: ↘ 0 ns

Machen Sie sich klar, für welche Teile Ihres Codes überhaupt Hardware erzeugt wird. Nehmen Sie an, bei Tuplen z.B. werden nur Kabel “zusammengelegt” oder “auseinander gezogen”. Ebenso sind Berechnungen wie n % 16 kaum rechenintensiv. Machen Sie sich auch klar, welche Aktionen parallel ausgeführt werden können. Vernachlässigen Sie auch die die Zeit, die für den Wege zwischen den verschiedenen Hardwarebausteinen gebraucht wird. Mit welcher Frequenz kann Ihre Implementierung maximal betrieben werden? Was ist die maximale Taktrate, mit der der Algorithmus laufen kann? Führen Sie in Ihrer Implementierung Pipelinestufen ein, sodass Sie eine Taktrate >= 100 MHz erreichen.