Vorlesung Architekturen und Entwurf von Rechnersystemen

TECHNISCHE UNIVERSITÄT DARMSTADT

Wintersemester 2018/2019

Übungsblatt 5

Prof. Andreas Koch, Jaco Hofmann

Nachdem nun in der letzten Übung ein Modul entwickelt wurde, welches eine Menge von zusammenhängenden Pixeln und dem mittleren Pixel vergleicht, liegt in dieser Übung der Fokus auf dem Entwurf eines Moduls, welches als Eingabe den ganzen Bresenhamkreis und den mittleren Pixel als Eingabe erhält und damit die nötigen einzelnen Vergleicher füllt.

Aufgabe 5.1 Vergleich für FAST

In dieser Teilaufgabe soll der komplette Filter implementiert werden, welcher zwischen 1 und 16 Vergleicher aus Übung 4 benutzt. Entscheiden Sie sich zuerst für ein passendes Interface, hier kann ein bereits bekanntes wiederverwendet werden. Wie bei der letzten Aufgabe muss hier eine Entscheidung gefällt werden. Sollen 16 Vergleicher eingesetzt werden oder einer, der 16 verschiedene Datensätze seriell verarbeitet? Auch hier stellt sich die Frage, welche Art von Zwischenspeicherung für die Daten genutzt werden soll.

Aufgabe 5.2 Testen der Implementierung

Testen Sie Ihre Implementierung mit einer herkömmlichen Testbench. Der Filter soll `True` zurückliefern, falls alle 12 aneinanderliegende Pixel größer als der mittlere sind.

Aufgabe 5.3 Testen der Implementierung mit Bluecheck

Testen Sie auch Ihre Implementierung mit Hilfe vom Bluecheck. Um einen verlässlichen Wert zu bekommen, können Sie hier `List#(t)` zusammen mit `toList()` und `sort()` verwenden. Diese Funktionen sind allerdings nicht sehr performant, weswegen sie nicht in normalen Modulen eingesetzt werden sollten.

Aufgabe 5.4 Parametrisieren der Implementierung

So wie auch in der letzten Aufgabe, soll der Filter über die Anzahl der zusammenhängenden Pixel und den Threshold parametrisierbar sein. Erweitern Sie dafür Ihren Code mit den in Übung 4 eingeführten Hilfsmitteln.