Parallele Berechnung der Mandelbrotmenge mit grafischer Ausgabe auf einem FPGA

Meister Rados

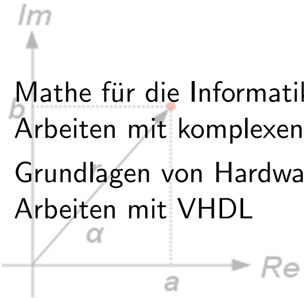
Goethe Universtität Frankfurt am Main

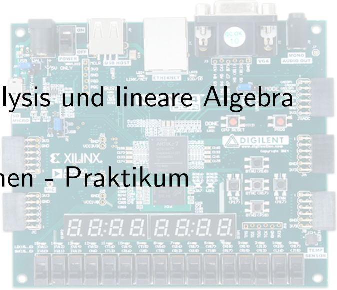
31. August 2017

Übersicht

1. Einleitung
   - Motivation
   - Arbeitsumgebung
   - Ziele
2. Die Mandelbrotmenge
   - Definition
   - Erzeugen der Bilder
3. Ansatz
4. Implementierung
5. Ergebnisse
   - Hardware
   - Software
   - Regression
   - Bottleneck
6. Ausblick

Motivation

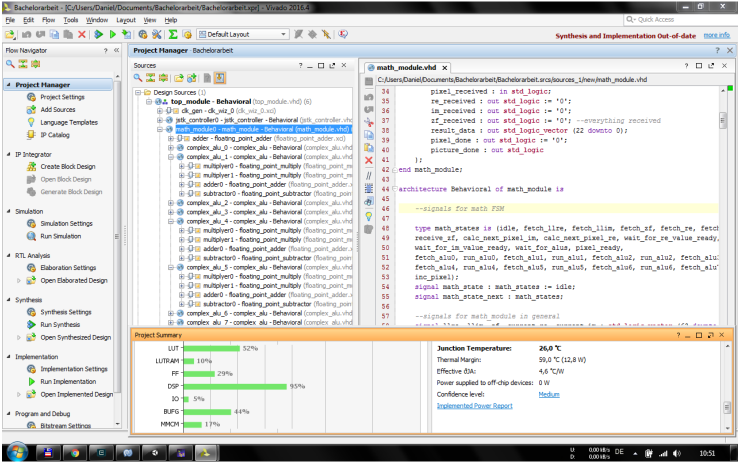




- Mathe für die Informatik 1: Analysis und lineare Algebra
  Arbeiten mit komplexen Zahlen
- Grundlagen von Hardwaresystemen - Praktikum
  Arbeiten mit VHDL

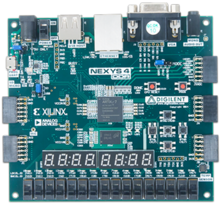
Arbeitsumgebung - Software

Vivado 2016.4



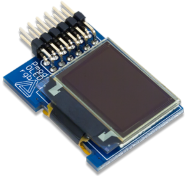
Arbeitsumgebung - Hardware

Nexys4 DDR mit Artix-7



Pmod Joystick

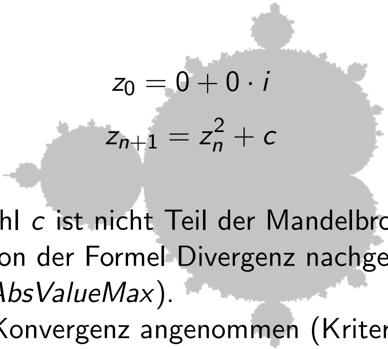
Pmod OLED Display



Ziele

- einen VHDL-Entwurf für FPGAs zu entwickeln, der darauf spezialisiert ist, die Mandelbrotmenge zu erkunden
- einen Joystick zur Eingabe einzubinden, um durch die Mandelbrotmenge navigieren zu können und diese auf einem Display auszugeben
- zu einer Beurteilung über Effizienz der Parallelisierbarkeit der Berechnung zu kommen

Die Mandelbrotmenge - Definition



Formel:

$$z_0 = 0 + 0 \cdot i$$

$$z_{n+1} = z_n^2 + c$$

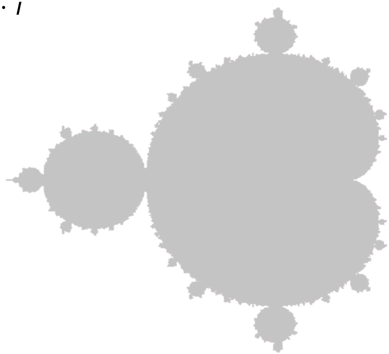
Eine komplexe Zahl c ist nicht Teil der Mandelbrotmenge, wenn für c unter Iteration der Formel Divergenz nachgewiesen werden kann (Kriterium *AbsValueMax*).
Andernfalls wird Konvergenz angenommen (Kriterium *IterationMax*).

Die Mandelbrotmenge - Beispiel-Iteration

Sei $c := -0,3 + 1 \cdot i$

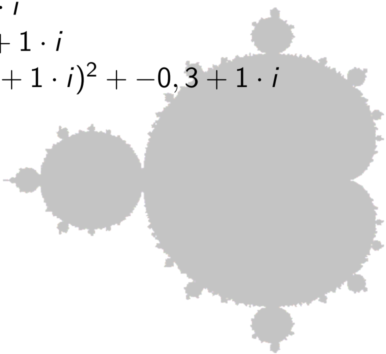
So ist $z_0 = 0 + 0 \cdot i$

$\implies z_1 =$



Die Mandelbrotmenge - Beispiel-Iteration

Sei $c := -0,3 + 1 \cdot i$
So ist $z_0 = 0 + 0 \cdot i$
$\implies z_1 = -0,3 + 1 \cdot i$
$\implies z_2 = (-0,3 + 1 \cdot i)^2 + -0,3 + 1 \cdot i$



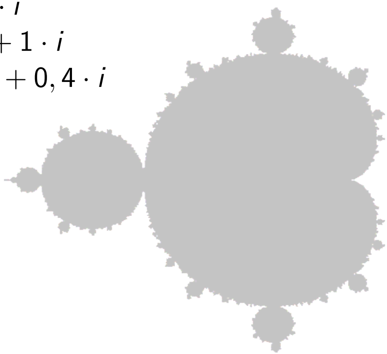
Die Mandelbrotmenge - Beispiel-Iteration

Sei $c := -0,3 + 1 \cdot i$

So ist $z_0 = 0 + 0 \cdot i$

$\implies z_1 = -0,3 + 1 \cdot i$

$\implies z_2 = -1,21 + 0,4 \cdot i$



Die Mandelbrotmenge - Beispiel-Iteration

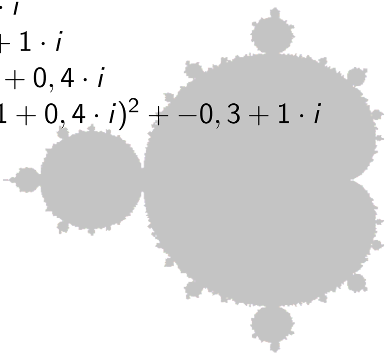
Sei $c := -0,3 + 1 \cdot i$

So ist $z_0 = 0 + 0 \cdot i$

$\implies z_1 = -0,3 + 1 \cdot i$

$\implies z_2 = -1,21 + 0,4 \cdot i$

$\implies z_3 = (-1,21 + 0,4 \cdot i)^2 + -0,3 + 1 \cdot i$



Die Mandelbrotmenge - Beispiel-Iteration

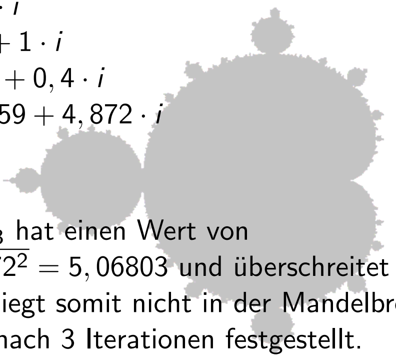
Sei $c := -0,3 + 1 \cdot i$

So ist $z_0 = 0 + 0 \cdot i$

$\implies z_1 = -0,3 + 1 \cdot i$

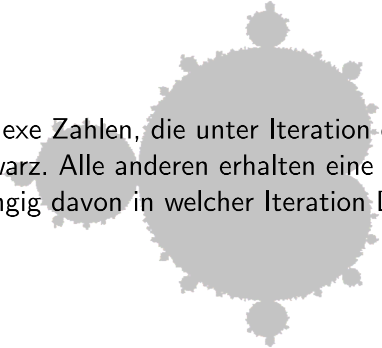
$\implies z_2 = -1,21 + 0,4 \cdot i$

$\implies z_3 = -1,3959 + 4,872 \cdot i$



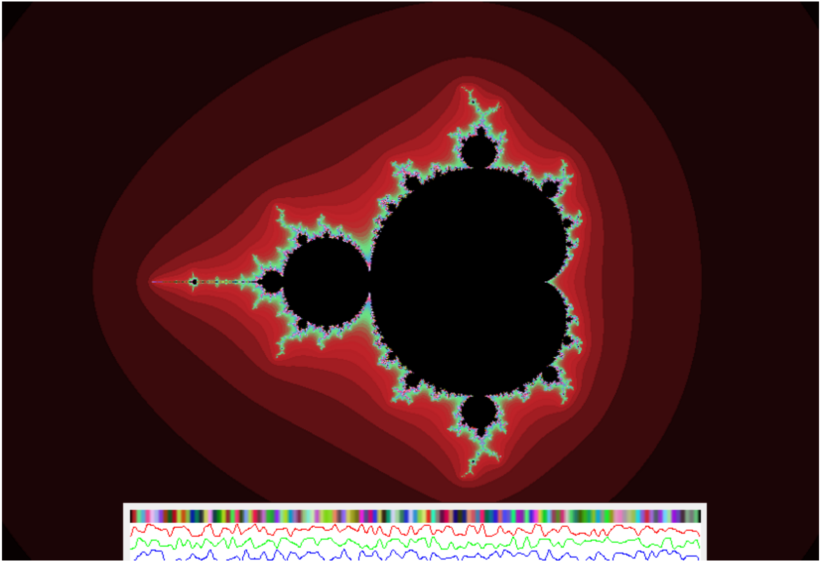
Der Betrag von z_3 hat einen Wert von $\sqrt{1,3959^2 + 4,872^2} = 5,06803$ und überschreitet somit *AbsValueMax*. c liegt somit nicht in der Mandelbrotmenge. Divergenz wurde nach 3 Iterationen festgestellt.

Die Mandelbrotmenge - Erzeugen der Bilder



Farben: Für komplexe Zahlen, die unter Iteration der Formel konvergieren: schwarz. Alle anderen erhalten eine Farbe aus einer Farbpalette, abhängig davon in welcher Iteration Divergenz festgestellt wurde.

Die Mandelbrotmenge - Visualisiert

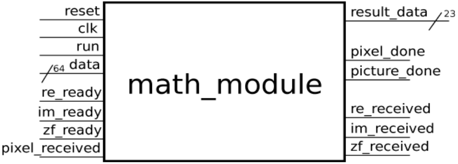


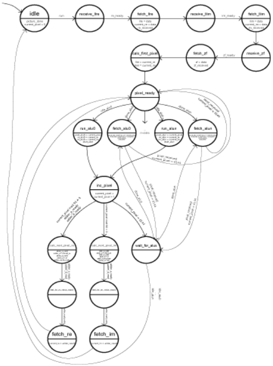
Ansatz - Systemeigenschaften

- Mensch-Computer-Interaktion via Joystick und Display
- Komplexe Ebene auf dem Display anzeigen
- Sinnvolle Werte für *AbsValueMax* und *IterationMax*
- Beliebig viele ALUs, die die Formel iterieren können, müssen mit komplexen Zahlen des aktuell sichtbaren Bereiches versorgt werden
- Dies soll parallel laufen
- Ergebnisse sichern und Farbe zuweisen

Ansatz

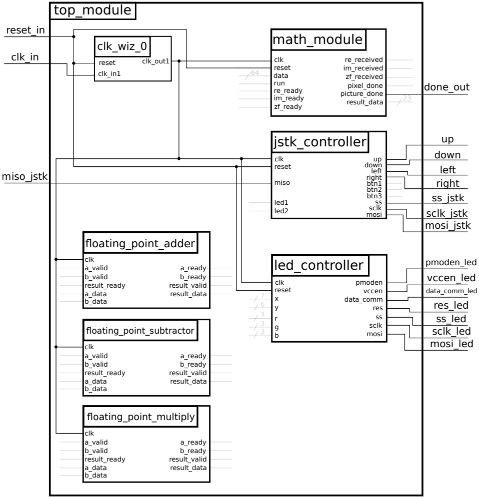
- Modularisierung
- Moore-Automaten
- IP-Cores





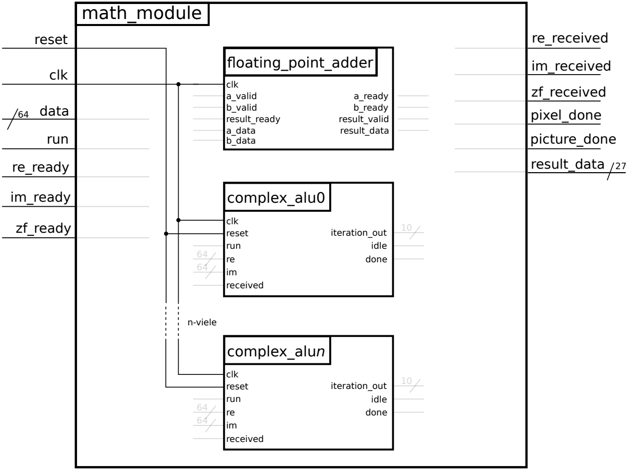
Implementierung - Topmodul

Das Topmodul:

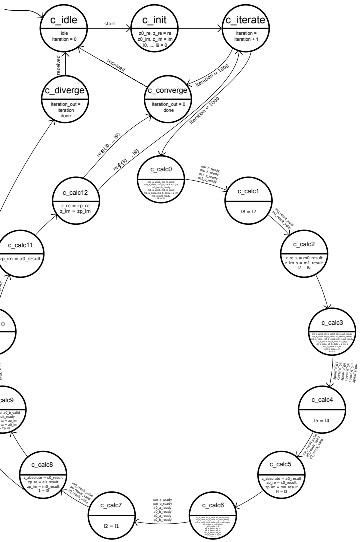


Implementierung - Math-Modul

Das Math-Modul:

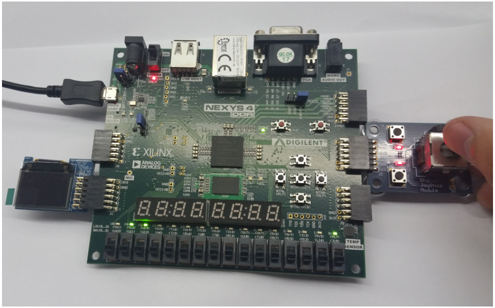


Implementierung - Komplexe ALU



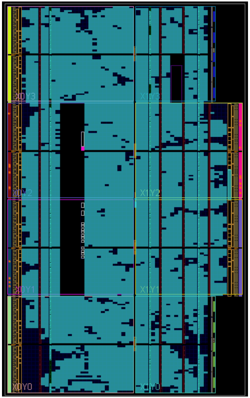
Ergebnisse - Hardware zur Interaktion

Joystick-Ansteuerung erfolgreich. Eine Implementation des Display-Controllers war zeitlich nicht möglich.



Ergebnisse - Hardwareentwurf

Auf dem Artix-7 lassen sich bis zu 8 ALUs unterbringen. Die Korrektheit des Entwurfes muss noch gezeigt werden.



Ergebnisse - Software

ALUs	Simulationszeit	Speed-Up Faktor	Rechenzeit
1	2,980427328 s	100,0000%	87h 14m 59s
2	1,499020169 s	198,8825%	39h 33m 52s
4	0,749917795 s	397,9743%	20h 19m 45s
8	0,3789402895 s	786,8651%	12h 42m 21s
16	0,195372699 s	1525,5086%	7h 32m 02s
32	0,992802226 s	3002,0353%	4h 01m 29s
64	0,051410618 s	5797,2991%	3h 04m 51s
128	0,027015148 s	11032,4301%	2h 44m 52s

Tabelle: Ergebnisse der Simulation

Ergebnisse - Regression

Eine Exponenten-Regression über die ermittelten Werte des Speed-Up Faktors ergibt folgende Funktion:

$$SpeedUpFaktor \approx 102.1063968 \cdot AnzahlALUs^{0.9710699028}$$

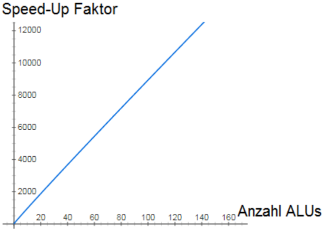
Ergebnisse - Fehler der Regression

ALUs	Speed-Up Faktor	Regressionswert	∅ Fehler
1	100%	102,1063968	2,106396819
2	198,8825%	200,1585377	2,132467248
4	397,9743%	392,3695425	-0,121841053
8	786,8651%	769,1595855	-4,3014843815
16	1525,5086%	1507,77877	-6,8854325912
32	3002,0353%	2955,689399	-13,4562939776
64	5797,2991%	5794,0196348	-11,959732807
128	11032,4301%	11357,98083	30,282837006375

Tabelle: Exponenten-Regression über den Speed-Up Faktor

Bottleneck - Math-Modul

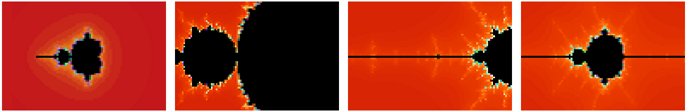
Was ist bei Hinzunahme von weiteren ALUs zu erwarten?
Bereits 3 ALUs können das Math-Modul auslasten.
Das Holen von Ergebnisse und Erzeugen neuer komplexen Zahlen dauert eine Zeit $t > 0s$.



Die ermittelte Regressionsfunktion grafisch dargestellt

Ausblick

Die Lehrveranstaltungen wurden erfolgreich verbunden.
Berechnung anderer Fraktale möglich.
Gar komplett andere Berechnungen möglich.
Andere Displaygrößen
Aufbauend: Implementation des LED-Controllers, sowie zeigen der Korrektheit steht noch aus.



Quellen

Inhalt: Bachelorarbeit
Hintergrund Folie 1:
http://de.numberempire.com/images/complex-numbers.png
Folie 5:
https://reference.digilentinc.com/reference/pmod/pmodoledrgb/reference-manual
http://store.digilentinc.com/pmod-jstk-2-axis-joystick-limited-time/
http://store.digilentinc.com/pmod-oledrgb-96-x-64-rgb-oled-display-with-16-bit-color-resolution/