HOCHSCHULE COBURG

Hochschule für angewandte Wissenschaften Coburg

Fakultät Elektrotechnik und Informatik

Studiengang: Informatik

Bachelorarbeit

Entwicklung einer hardwarebeschleunigten Berechnung der Mandelbrotmenge auf einem FPGA

von

Daniel Kirchner

Matrikelnummer: 02219415

Abgabe der Arbeit: 15.07.2019

Betreut durch: Prof. Oliver Engel, Hochschule Coburg

Zusammenfassung

Im Rahmen dieser Bachelorarbeit wurde eine hardwarebeschleunigte Visualisierung der Mandelbrotmenge auf einem FPGA realisiert. Hierfür werden diverse mathematische und designtechnische Performanceoptimierungen vorgestellt, welche dann in einparalleles FPGA-Design implementiert wurden. Weiterhin sollen einige Eigenschaften und Besonderheiten der Mandelbrotmenge und von Fraktalen im Allgemeinen aufgezeigt werden.
Das Projekt wurde für das Zybo Zynq-7000 Trainer Board entwickelt, welches über einen VGA-Output die Repräsentation des Fraktals in Form eines 800x600@60Hz Videosignals ausgibt. Zur optimalen Ausnutzung der auf diesem Board gegebenen Ressourcen (DSPs, BRAM) wurde die *Vivado Design Suite* mit dem integrierten IP-Katalog verwendet.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Codebeispielverzeichnis

Abkürzungsverzeichnis

JAX-RS	Java API for RESTful Web Services

1 Einleitung

1.1 Motivation

Die stets wachsende Zahl von Komponenten die auf einem mikroelektronischen Bauteil pro Zeiteinheit untergebracht wird, ist ein Phänomen, welcbes Gordon Moore schon im Jahr 1965 aufgefallen ist [4]. Die populäre, nach ihm benannte Beobachtung, dass die Anzahl der Transistoren pro integriertem Schaltkreis exponentiell mit der Zeit ansteigt ist allgemein als das Mooresche Gesetz bekannt.

Diese Gesetzlichkeit machte es möglich den stets wachsenden Leistungsanforderungen an moderne Hardware gerecht zu werden, indem immer mehr (und komplexere) identische Allzweck-Prozessoren (Kerne) pro CPU verbaut wurden.

Dieses Vorgehen kann jedoch nicht unbegrenzt lange betrieben werden, da die heute verwendeten MOS-Transistoren sich rapide ihren physikalischen Grenzen annähern. Ein besserer Umgang mit dem stetig steigenden Bedarf an Rechenleistung ist die Entwicklung von spezialisierter Hardware, welche zwar nur ein kleines Aufgabenspektrum abdeckt, dies jedoch mit hoher Performanz und Energieeffizienz tut.

Ein Beispiel hierfür ist die moderne Grafikkarte (GPU), welche dem Prozessor Darstellungsberchnungen abnimmt, wodurch dieser mehr Zeit hat, andere Aufgaben zu übernehmen. Die Grafikkarte führt diese Aufgaben mit enorm hohem Durchsatz und niedrigen Berechnungszeiten durch, welche ein herkömmlicher Prozessor alleine nicht erreichen könnte.

Auch andere Hardwarekomponenten, wie die Netzwerkkarte, kryptographische Beschleuniger, oder Soundkarten sind in fast allen Computersystemen verbaut und entlasten den Hauptprozessor. Man spricht auch von heterogenen Computersystemen.

Die im Rahmen dieser Arbeit vorgestellte Hardware soll ein Beispiel für eine derartige heterogene Komponente sein. Auf einem Field Programmable Gate Array (FPGA) soll eine performante und energieffiziente Visualisierung der sogenannten Mandelbrotmenge realisiert werden. Diese Problemstellung ist auch durch einen ordinären Prozessor lösbar, lastet diesen jedoch enorm aus und ist somit auch sehr energieineffizient.

1.2 Aufgabenstellung

Die Aufgabe dieses Projektes ist es, eine komplett in Hardware stattfindende Berechnung der Mandelbrotmenge durchzuführen und die Ergebnisse über eine VGA-Schnittstelle darzustellen

(ähnlich Bild)

Weiterhin soll die Hardware durch externe Peripherie konfigurierbar hinsichtlich der angestellten Berechnungen sein. So soll etwa aktuell abgebildete Bereich der Mandelbrotmenge oder auch die Farbgebung der Darstellung im laufenden Betrieb geändert werden können.

Hierfür wurde das FPGA-Trainer Board *Zybo Zynq-7000* zur Verfügung gestellt, welches in genauer vorgestellt wird.

1.3 Mitgelieferte Skripte

Im Git-Repository ... sind sämtliche Python-Skripts, die zur Erstellung von selbsterstellten Bildern verwendet wurden, enthalten. Des weiteren gibt die dort enhaltene Datei *readme.md* Aufschluss über nützliche Skripts, die im Rahmen dieser Arbeit verwendet wurden.

2 Technische Grundlagen

Zum besseren Verständnis des Gesamtprojektes sollen in diesem Kapitel einige technische Konzepte erläutert werden.

2.1 FPGAs

Ein *Field Programmable Gate Array* (kurz FPGA) ist ein Schaltkreis, welcher mit Hilfe von Hardwarebeschreibungssprachen konfiguriert werden kann, um beliebig komplexe logische Schaltungen zu realisieren.
Das Grundelement eines solchen Bausteines bilden die sogenannten *Lookup Tables* (kurz LUTs), welche zu einem beliebigen n-bit Input ein 1-bit Output Signal produzieren. Die zugrundeliegende Logiktabelle einer LUT ist hierbei frei programmierbar.

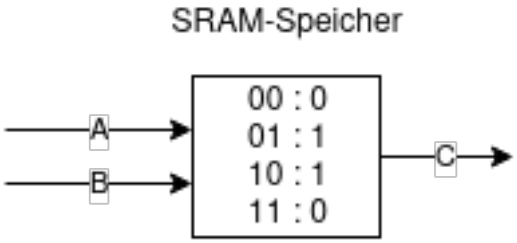

Abbildung 1: Beispielhafter Aufbau einer 2-Input LUT

Eine LUT, welche die Operation $C = A \oplus B$ implementiert ist in Abbildung 1 zu sehen. In dieser wird in einem SRAM-Speicher für jede Inputkombination ein Outputwert hinterlegt, wodurch jede 2-Bit Funktion abgebildet werden kann. In Verbindung mit einem Flipflop [1] bildet eine LUT dann einen sogenannten Logikblock (s. Abbildung 2). [1]

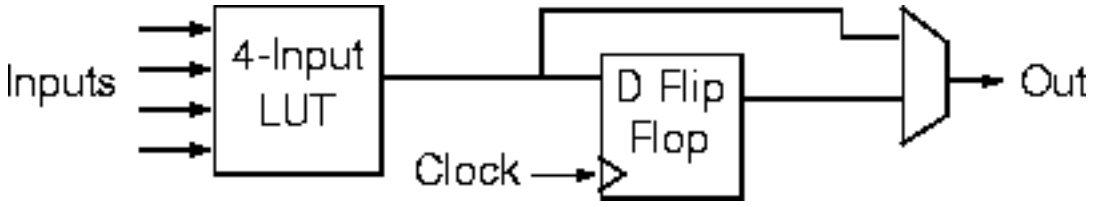

Abbildung 2: Logikblock, aus [1]

Ein FPGA verbindet nun durch ebenfalls konfigurierbare Bussysteme viele solcher Logikblöcke um komplexere Schaltkreise abzubilden.

[1] Ein Flipflop ist ein Speicherelement, welches einen einzigen Bit Daten halten kann.

Weiterhin verfügen FPGA-Boards oft noch über ergänzende Hardwarekomponenten, von denen die im Falle dieses Projektes vorhandenen im Folgenden gezeigt werden sollen.

DSP Ein *Digitaler Signalprozessor* (DSP) ist ein fest integrierter Baustein, welcher durch Multiplizierer und Akkumulatoren binäre Algorithmen beschleunigt. So übernimmt dieser etwa grundlegende mathematische Operationen, was dazu führt, dass diese flächen- und energiesparender durchgeführt werden, als bei der Verwendung von LUTs [5, S. 52]. Typische Anwendungsgebiete dieser Bausteine sind Fließkommamultiplikationen, Schnelle Fourier-Transformationen oder einfache Zähler [5, S. 14]. In diesem Projekt wurden die DSPs hauptsächlich aufgrund ihrer 25x18 Bit Multiplizierer verwendet, welche kaskadiert werden können um beliebig Breite Multiplikationen durchzuführen.

Block RAM FPGAs verfügen meist über *Block RAM* (BRAM), welcher zur Speicherung binärer Daten dient. Dieser Speicher ist lese-/schreibsynchron, wodurch Inkonsistenzen beim Speicherprozess ausgeschlossen sind [6, S. 11]. Um auf diese Speicherblöcke Zugriff zu erlangen muss der von Xilinx mitgelieferte Baustein "Block RAM Generator"verwendet werden.

IO-Komponenten Zur Kommunikation mit der Außenwelt verfügen FPGAs über diverse Schnittstellen wie z.B. Knöpfe, Schalter, aber auch komplexere Anbindungen wie etwa VGA (s. hierzu Abschnitt 2.2) oder PMOD-Anschlüsse. Diese sind so angebunden, dass ihre Signale direkt in Logikschaltungen von LUTs integriert werden können.

2.2 VGA-Schnittstelle

Eine *Video Graphics Array* (VGA) Schnittstelle wird durch einen Videoübertragungsstandard, welcher erstmals von IBM in ihrer *IBM Personal System/2* Modellreihe verbaut wurde, spezifiziert [7]. Das Darstellungsverfahren verwendet einen 15-poligen Anschluss, um Videosignale in variabler Auflösung und Bildwiederholungsrate zu übertragen.
Hierbei liegen die RGB-Werte eines jeden Pixels als analoge Spannungen an und werden zu bestimmten Zeitpunkten vom Bildschirm ausgelesen. Da der VGA-Standard im Jahre 1987 aufkam, war er ursprünglich noch für Kathodenstrahlröhrenbildschirme (auch Röhrenbildschirme genannt) ausgelegt. Die Elektronenstrahlen dieser Bildschirme konnten sich nicht ohne kurze Verzögerungen über die Anzeigefläche bewegen, was bedeutet, dass der VGA-Standard dies berücksichtigt und dem Bildschirm einige μs für größere Sprünge des Strahls einräumen muss.

Die größten solchen Sprünge finden statt, wenn eine Pixelreihe übertragen wurde und der Strahl an die erste Pixelposition der nächsten Reihe bewegt werden muss oder wenn das gesamte Bild übertragen wurde und wieder zum oberen linken Pixel gesprungen werden muss. Während dieser Pausen werden keine RGB-Werte übertragen, man nennt diese zeitlich gedachten Bereiche auch "Porch"(engl. Vorbau).

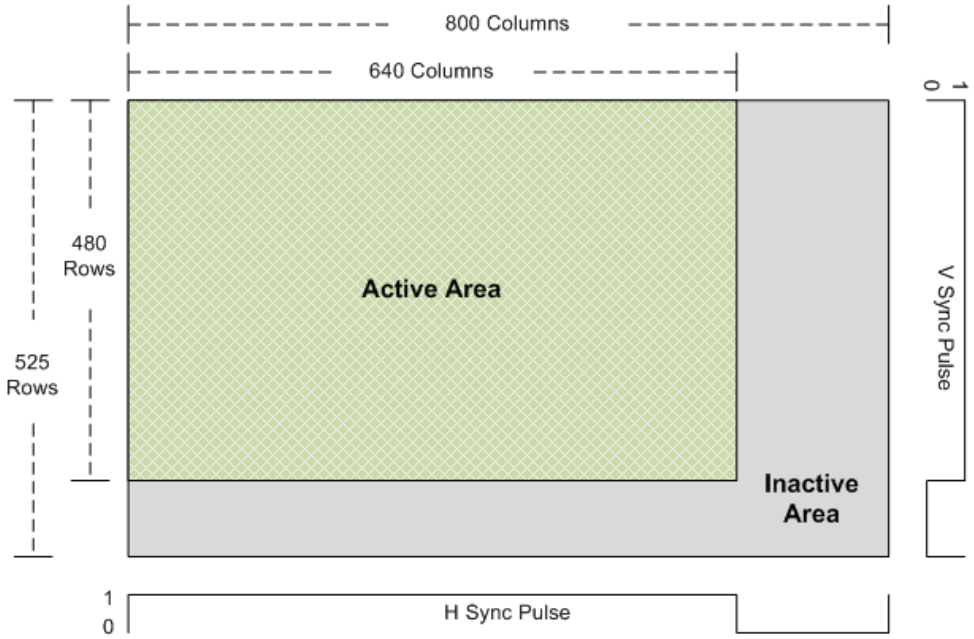

Abbildung 3: VGA-Timing/Aufbau für eine 640x480 Auflösung, aus [2]

Eine 640x480 Pixel Bild baut sich dann wie in Abbildung 3 gezeigt auf: Zuerst werden 640 Pixel RGB Daten für die erste Reihe empfangen, danach kommen 160 Pixel inaktiver Bereich (⇒ Porch). Das Ende dieses Bereiches wird durch das Ansteigen des low-aktiven Signal HSYNC signalisiert. Dieser Vorgang wird nun 480 mal wiederholt, bis das gesamte Bild übertragen wurde. Draufhin wird analog zur horizonalen Synchronisation das VSYNC Signal für 45 Pixel auf auf 0 gesetzt, um den vertikalen inaktiven zu signalisieren. Die Länge des inaktiven Bereiches setzt sich aus Front-, bzw. Backporch und der Länge des Sync-Pulses zusammen. Die Bedeutungen der einzelnen Signale können fügen jedoch dem nötigen Verständnis nicht mehr hinzu, weswegen die Summe dieser Bereiche als Ganzes angesehen werden kann. Wie schon erwähnt ist 640x480 jedoch nicht die einzige Auflösung, die mit einem VGA-Anschluss realisierbar ist. Andere Auflösungen (mit anderen Widerholungsraten) müssen vom verwendeten Bildschirm unterstützt sein, und werden durch verschieden schnelle Pixelclocks und Porches realisiert.
Die in dieser Arbeit verwendete Auflösung von 800x600 Pixeln bei 60Hz benötigt die in Tabelle 1 dargestellten Werte.

aktiver Bereich (horizontal)	Pixel	800
aktiver Bereich (vertikal)	Pixel	600
inaktiver Bereich (horizontal)	Pixel	256
inaktiver Bereich (vertikal)	Pixel	28
Pixelfrequenz	MHz	40

Tabelle 1: VGA-Werte für ein 800x600@60Hz Signal

Die in Tabelle 1 gezeigte Pixelfrequenz von 40 MHz bedeutet, dass alle 25 ns ein neuer RGB-Wert anliegen muss. Dies ist ein wichtiger Aspekt für weitere grundliegende Designentscheidungen.

3 Theoretische Grundlagen

Die dieser Arbeit zugrundelegenden mathematischen Grundlagen und Definitionen sollen in diesem Kapitel näher erläutert werden.

3.1 Fraktale

Der Begriff *Fraktal* wurde vom französischen Mathematiker Benoît Mandelbrot geprägt und leitet sich vom lateinischen Adjektiv *fractus* ab, was „in Stücke gebrochen“ oder „irregulär“ bedeutet. Allgmein ist hiermit entweder eine natürlich vorkommende Strukur mit gewissen Eigenschaften oder eine genau definierte mathematische Menge gemeint. [8, S. 16]
Für ein intuitives Verständnis des Begriffes sollen im folgenden zuerst einige natürlich vorkommende Fraktale gezeigt werden, woraufhin im nächsten Abschnitt eine formale Definition des Begriffs *Fraktal* folgen soll.

3.1.1 Natürlich vorkommende Fraktale

Fraktale besitzen oft selbstähnliche Strukturen, d.h. dass sich die Gesamtstruktur eines Objektes in kleinerem Maßstab immer wieder findet. Ein Beispiel hierfür ist ein fraktal definierter Baum wie er in Abbildung 4 abgebildet ist. Der Baum wird hierbei über ein einfaches rekursives Verfahren aufgebaut, bei dem immer wieder von jedem Teilbaum aus mit einem festen Winkel abgebogen wird.

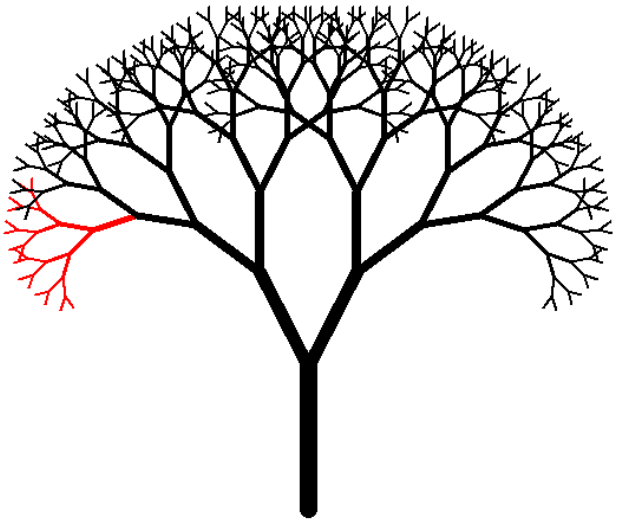

Abbildung 4: Fraktal definierter Baum, Skript: *baum.py*, abgewandelte Version von [3]

Der in Abbildung 4 rot markierte Teilbaum ist in seiner Struktur mit dem Gesamtbaum identisch und besitzt lediglich eine niedrigere rekursive Tiefe.
Der Blumenkohl (Abbildung 5) ist ein weiteres Beispiel für ein natürlich vorkommendes Fraktal.

Abbildung 5: Blumenkohl, von Rainer Renz

3.2 Die Mandelbrotmenge

4 Test und Umsetzung

4.1 Gesamtsystem

4.2 Komponentenbeschreibung

4.2.1 Mandelbrot-Core

4.2.2 Mandelbrot-Koordinator

4.2.3 Dual Port Block RAM

4.2.4 Lookup Tables

4.3 Peripherie und funktionale Beschreibung

5 Optimierungen

5.1 Algebraische Optimierung

5.2 Designoptimierungen

Literaturverzeichnis

[1] FPGA Architecture for the Challenge. http://www.eecg.toronto.edu/~vaughn/challenge/fpga_arch.html. Zugriff am: 12.06.2019.

[2] Go Board - VGA Introduction. https://www.nandland.com/goboard/vga-introduction-test-patterns.html. Zugriff am: 12.06.2019.

[3] Stackoverflow Antwort: Drawing a Fractal Tree in Python, User sheldonzy. https://stackoverflow.com/a/46754459. Zugriff am: 12.06.2019.

[4] Gordon E Moore et al. Cramming more components onto integrated circuits, 1965.

[5] Xilinx, Inc. *7 Series DSP48E1 Slice*, März 2018. v1.10.

[6] Xilinx, Inc. *7 Series FPGAs Memory Resources*, Februar 2019. v1.13.

[7] Chronology of IBM Personal Computers. https://web.archive.org/web/20150221071923/http://pctimeline.info/ibmpc/ibm1987.htm. Zugriff am: 12.06.2019.

[8] Benoît Mandelbrot. *Die fraktale Geometrie der Natur*. Springer-Verlag, 2013.