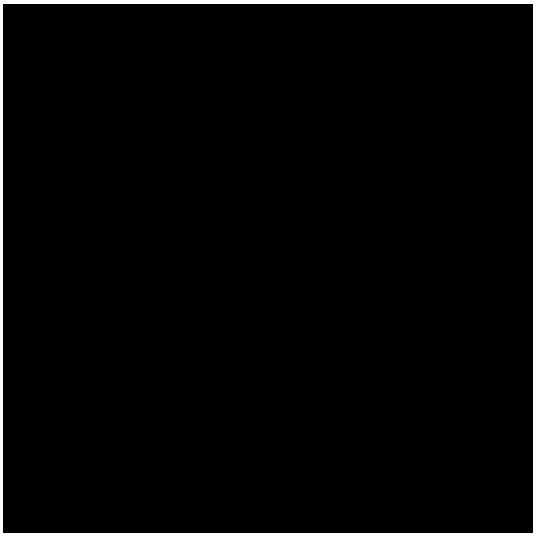


Diplomarbeit

Entwicklung eines kon gurierbaren Vektorprozessors

Harald Manske
Fachhochschule Augsburg

19. Dezember 2007

Erstellungserklärung

Hiermit erkl•

Abstract

When it comes to personal computers, processors with SIMD-extensions are already widely spread. With their help it is possible to considerably increase the performance of processing multimedia-based data compared to common, scalar-only architectures. With a processor like that, one single command VHDL,]d 2[(5A0 -134)-o.e134134p ri2(m)28a3484l-c

1 Einleitung

1.1 Überblick

1.1.1 Multimediale Anwendungen

Die Möglichkeit, Bilder und Filme überall aufnehmen zu können, wird von immer mehr Menschen genutzt. Digitale Kameras dominieren mittlerweile den Markt und selbst Mobiltelefone sind in der Lage, Bilder mit bis zu fünf Megapixeln einzufangen. Dabei ist die Aufnahme allerdings nur der erste Schritt, die entstehenden Mediendaten müssen auch komprimiert und abgespeichert werden. Oft findet auch eine visuelle Verbesserung des Bildes

Moglichkeit, fertige IP-Cores zu verwenden, die unter einer Open-Source-Lizenz vero entlicht wurden. Hierfur gibt es im Internet bereits etablierte Webseiten, auf denen eine Vielzahl verschiedener Bausteine zum Download angeboten wird. Das Spektrum reicht von Multiplizierern uber Ethernet-

welche Komponenten bei der Implementierung Probleme bereitet haben und wie diese gel•

2 Grundlagen

und deren Ausgangssignale per Assert-Anweisung überprüft. Bei einer Abweichung des Überprüften von dem Referenzwert, arbeitet die Komponente nicht nach Spezifikation.

2.3 SIMD- und Vektorprozessoren

Die Rechnerklassifikation nach Flynn [Fly72] unterteilt Architekturen anhand ihrer Anzahl an parallelen Befehls- und Datenströmen in vier Gruppen:

SISD - Single instruction stream, single data stream
SIMD - Single instruction stream, multiple data streams
MISD - Multiple instruction streams, single data stream
MISM - Multiple instruction streams, multiple data streams

Vetreter von SISD-Systemen sind Architekturen, bei denen ein einzelnes Leit- bzw. Steuerwerk genau ein Rechen- bzw. Operationswerk verwaltet. [Mar94] Die Operationen werden sequentiell abgearbeitet; in jedem Befehl

parallel aufgebaute Operationswerk wirkt, anstatt auf einzelne Operanden, auf einen kompletten Vektor von Daten.

Jeder dieser Vektoren umfasst mehrere Daten-Wörter, deren Länge in vielen der heute üblichen SIMD-Prozessoren im Befehl kodierbar ist. Ein 128-Bit-

werden, indem die Mittelwerte für mehrere Pixel gleichzeitig berechnet werden. Der Leistungsgewinn wird hierbei über die Länge des Vektorregisters limitiert. Je mehr Wörter in eines der Register passen, umso schneller wird die Verarbeitung.

MMX-Einheit keine zusatzliche Unterstutzung durch das Betriebssystem. Leider ist es so aber auch nicht moglich, Flie komma- und MMX-Operationen zur selben Zeit abzuarbeiten.

Es wird eine einfache Variante von bedingter Ausfuhrung unterstutzt. Vergleiche durch MMX-Anweisungen resultieren in einer Bitmaske, die der Lange der Operatoren entspricht. Diese Bitmaske kann dann mit Hilfe von logischen Verknupfungen dazu verwendet werden, das Ergebnis zu bilden.

Folgendes Beispiel soll das verdeutlichen:

```
if (bedingung = wahr) Ra = Rb else Ra = Rc;
```

Die Ergebnismaske der Vergleichsoperation ist in *Rx* gespeichert. Diese enthalt fur alle Bits eines Elements den icalt alt w,68 -0.di16(eines=)-525(wah(Elemr0.475 3ges02

mit ernsthafter SIMD-Unterstützung. Stattdessen sind dort eher Nachbauten oder Eigenentwicklungen von einfachen RISC-Prozessoren beziehbar.

Es gibt aber Ans•

Die Anzahl der Datenregister ergibt sich daraus, dass im Befehl die Quell- und Zielregister mit je zwei Bit kodiert sind. Dadurch kann insgesamt eine Auswahl aus vier Quellen getro en werden. Drei davon beziehen sich auf die Register, der vierte auf den Wert Null oder den Direktwert.

SUB	reg_{axy0}, reg_{axy0}, reg_{axyi}	Subtraktion Beispiel: SUB 0, Y, $FF

VMOL	$vreg_l$, $vreg_l$	Verschieben von allen Datenwortern im Vektorregister nach links Beispiel: VMOL R2, R1
VMOR	$vreg_l$, $vreg_l$	Verschieben von allen Datenwortern im Vektorregister nach rechts Beispiel: VMOR R0, R3
VADD	.breite$_{voll}$ $vreg_l$, $vreg_l$, $vreg_l$	Vektor-Addition Beispiel: VADD.DW R2, R1, R0
VSUB	.breite$_{voll}$ $vreg_l$, $vreg_l$, $vreg_l$	Vektor-Subtrak.398 w 0 0,m 0 10.959 l SQBT.f.901 cm[]0 d 0 J 0.398 w 0 0

Abbildung 4: Aufteilung eines 32 Bit Befehlswortes

Beispiele f•

Von anderen Architekturen ist man als Softwareentwickler gewohnt, dass

Diesen Transferbefehlen sehr ähnlich sind die Anweisungen „VMOL\ und „VMOR\ . Es wird aber nicht, wie das bei „VMOV\ der Fall ist, das Wort an Stelle *i* aus dem Quellregister auch an Stelle *i* sondern an *i+1* bzw. *i-1* im Zielregister geschrieben. Damit können die Datenwörter eines Vektorregisters um jeweils ein Wort nach links oder rechts verschoben werden. Das Datenwort, das am Anfang bzw. Ende des Registers herausfällt, wird am anderen Ende wieder eingefügt. Die gewünschte Wortbreite für diese Operationen ist konfigurierbar, muss aber in den Quellen des Prozessors vorge-

3.2.5 Kooperationsbefehle

Die Vektor- und Skalareinheit des Prozessors sind weitgehend autonom, es gibt jedoch auch Befehle, die nur in Kooperation ausgefuhrt werden konnen. Dies dann ist dann der Fall, wenn Daten von oder zur Vektoreinheit transportiert werden mussen.

Um mit dem skalaren Teil des Prozessors Daten austauschen zu konnen,

Abbildung 10: Skalareinheit mit Speicherschnittstelle (Vereinfacht)

Abbildung 13: Befehlszahler und Adressleitung

3.3.2 Mikroarchitektur der Vektoreinheit

Die Vektoreinheit (Abbildung 14) ist einfacher aufgebaut als die Skalareinheit, da diese nur Daten verarbeiten muss und sich nicht um den Programmablauf und die Speicheradressierung kummern muss. Das zentrale Element dieser Einheit ist der Vektorregistersatz, da jeder Befehl auf diese Ressource zugreift.

Der Vektorregistersatz beinhaltet mehrere Vektorregister, deren Anzahl (N) in der Kon guration vorgegeben wird. Jedes dieser Vektorregister besteht aus beliebig vielen 32-Bit-Wortern. Auch die Anzahl der Worter (K

Abbildung 14: Vektor-Operationswerk

aufgeteilt wird.

3.4 Speicherschnittstelle

Die Speicherschnittstelle abstrahiert den Zugri auf den Arbeitsspeicher. Dem Prozessor muss nicht bekannt sein, welche technische Umsetzung von Speicher (Distributed RAM, Block-RAM, externer SRAM, externer DRAM, etc.) tatsachlich verwendet wird. Dadurch kann die Art des RAM leicht ge•

Abbildung 17: Externe Sicht der Speicherschnittstele

des Programmes umzusetzen. Das bedeutet, es ist zusätzlich verantwort-

Das Steuerwerk der Skalareinheit hat neben diesen Signalen nur noch wenige Eingänge. Dabei handelt es sich um das Befehlswort, das *ready*-Signal der Speicherschnittstelle, das *Reset*-Signal und um die beiden Flags *Carry* und *Zero* des Statusregisters. Das skalare Operationswerk ist so ausgelegt, dass es nur mit einzelnen Steuersignalen auskommt und viele Multiplexer direkt aus Teilen des Befehlswortes geschaltet werden.

Der Zustandsautomat im skalaren Steuerwerk wurde mit den Phasen *Instruction Fetch*, *Instruction Decode* und *Execute*

4 Implementierung

Eine Voraussetzung an den Prozessor ist, dass dieser einfach erweiterbar sein soll. Das wird durch drei Ansätze erreicht:

1. Die Befehlskodierung ist so ausgelegt, dass noch Platz für neue Befehle frei ist bzw. leicht geschaffen werden kann.
2. Der Prozessor wird als Soft-Core in der Hardwarebeschreibungssprache VHDL (siehe Abschnitt 2.2) implementiert und kann damit auch von demjenigen modifiziert werden, der den Prozessor einsetzen will.
3. Der Quellcode ist in mehreren, klar voneinander getrennten Kompo-

Scheiben, welche aus Vektor-ALUs, Multiplexern und einem Registersatz aufgebaut sind.

4.2 Kon gurierbarkeit

Was am Prozessor kon guriert werden kann, wurde bereits in Kapitel 3.1 aufgef•

```
        slice: vector_slice
            port map (
                ...
            );
    end generate ;
```

Konstante *use_skalar_mult* für *generate*-Bedingung

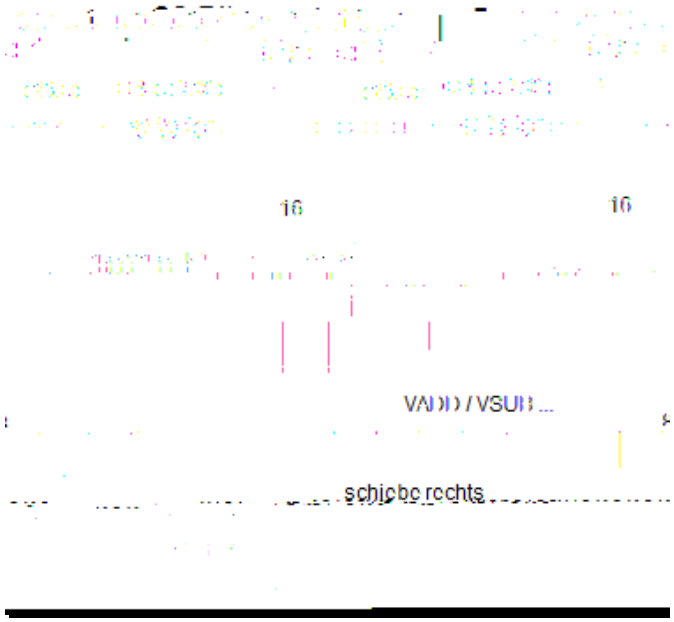
```
    mult_gen: if use_scalar_mult generate
```

Abbildung 20: Carry-Select-Prinzip

Ein anderer, allgemeiner Ansatz zur Erhöhung der Taktfrequenz ist, eine Berechnung auf mehrere Takte zu verteilen und dabei in jedem Takt nur ein Teilproblem zu lösen.

Dieser Ansatz wurde umgesetzt, indem die 32-Bit-ALUs komplett neu und als Schaltwerk (Abbildung 21) statt wie bishhdct-Pra.mpat-P0 1b7(u)1gebautneu undDie1)8(die)

Dadurch entfallen zwar in jeder 32-Bit-ALU drei der vier 8-Bit-ALUs, allerdings ist auch zusätzliche Logik für das Register, die Multiplexer und das Steuerwerk der Vektor-ALUs notwendig. Da die Steuerung der Vektor-ALU-Befehle unabh•


16
16
VADD / VSUB ...
schiebe rechts

Die De nition des Mischvorgangs (Abbildung 23) erfolgt mit Hilfe der 14-Bit-Perumtationsbeschreibung des „VSHUF\-Befehls. Von Links gesehen geben deren erste zwei Bits (*ww*) die Wortlange an. Anhand der nachsten vier Bits (*ssss*

Durch die Permutation wird die Reihenfolge der Daten im Quellregister dann so ver•

und das eigentliche Mischen nur eine einzige Wortbreite verwendet werden muss. Dadurch kommt diese Implementierung auf weniger benötigte Chip fläche (1135 Slices und 2383 LUTs auf dem Test-FPGA XC3S700A bei $K = 16$) als alle anderen getesteten Verfahren und führt den Mischvorgang unabhängig von der Wortbreite in vier Takten durch.

Interessant wä

5 Testumgebung und Entwicklungswerkzeuge

5.1 Testumgebung

Da der Prozessor alleine nicht lau ahig ist, musste zusatzlich Speicher bereit gestellt werden, in dem Programme, Zwischen- und Endergebnisse abgelegt werden konnen. Der erzeugte Speicher wird von der Speicherschnittstelle angesteuert und liegt, aus Grunden der Zeitersparnis, nur als SRAM Imple-

Die Anweisung

Schnittstelle dar. Über diese Schnittstelle können vom einem anderen Com-

A	Register A	Gibt den Wert des Registers A zurück.

den Wert des Registers Azur ück.

7	Anzahl Takte •	Gebe die Anzahl an gezählten Takten zurück.

Tabelle 3: Befehle f•

Debuggers konnten verschiedene Programme ausgefuhrt und die Arbeitsweise nachvollzogen werden.

Um den Prozessor nach Anderungen schnell und einfach validieren zu konnen, wurde ein Programm erstellt, welches alle moglichen Befehle ausfuhrt und Ergebnisse der Operationen auf G•

6 Ergebnisse

6.1 Synthese-Ergebnisse in Abhängigkeit der Konfiguration

Vernachlässigt man den nicht von K abhängigen Teil der Vektoreinheit (z. B. deren Steuerwerk), steigt die Chip äche damit zumindest im getesteten Bereich ($K = 4 \ldots 36$) in etwa linear mit K an. Zu beachten ist, dass für K wegen dem 64-Bit-Modus des Prozessors nur gerade Werte verwendet werden dürfen. Eine Erhöhung um den Wert Eins ist

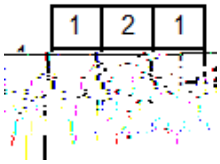


Abbildung 29: 3x3 Filterkern

7 Zusammenfassung und Ausblicke

7.1 Ergebnis dieser Arbeit

Literaturverzeichnis

[Fly72] M.J. Flynn, Some Computer Organizations and Their E ectiveness, IEEE Transactions on Computers, Vol. c-21, No. 9, pp. 948-960, 1972

[Mar94] C. Martin: Rechnerarchtitektur, Carl Hanser Verlag, 1994

[Mar03] C. Martin: Einfuhrung in die Rechnerarchitektur, Carl Hanser Verlag, 2003

[Men05] M. Menge: Moderne Prozessorarchitekturen, Springer-Verlag, 2005

[Gon02] R.C. Gonzalez, R.E. Woods: Digital Image Processing, Prentice Hall, 2002

[Sie01] C. Siemers: Hardwarem

http://softwarecommunity.intel.com/isn/Downloads/Intel SSE4 Programming Reference.pdf, zuletzt aufgerufen am 17.12.2007

[Bro05] J. Brown, Application-customized CPU design - The Microsoft Xbox 360 CPU story,
http://www-128.ibm.com/developerworks/power/library/pa-fpfxbox/?ca=dgr-lnxw09XBoxDesign, zuletzt aufgerufen am 17.12.2007

[Ful98] S. Fuller, Motorola's AltiVec Technology,
http://www.freescale.com/ les/32bit/doc/fact_sheet/ALTIVECWP.pdf, zuletzt aufgerufen am 17.12.2007

[Neon] ARM NEON Technology, http://www.arm.com/miscPDFs/6629.pdf, zuletzt aufgerufen am 10.12.2007

[Gui07] P. Guironnet de Massas, P. Amblard, F. Petrot, On SPARC LEON-2 ISA Extensions Experiments for MPEG Encoding Acceleration, VLSI Design, Vol. 2007, Hindawi Publishing Corporation

[Sha04] J. Shafer, Embedded Vector Processor Architecture for Real-Time Wavelet Video Compression,
http://www.je shafer.com/publications/shafer-masters_thesis.pdf, 2004, zuletzt aufgerufen am 10.12.2007