Ein mehrfädiger, superskalarer Mikroprozessor

Ulrich Sigmund
VIONA Development GmbH
Karlstr. 27
D-76133 Karlsruhe
Telefon: 0721 913440
Fax: 07219134499
uli@viona.DE

Theo Ungerer
Universität Karlsruhe
Institut für Rechnerentwurf und Fehlertoleranz
D-76128 Karlsruhe
Telefon: 0721 608–6048
Fax 0721 370455
ungerer@Informatik.Uni-Karlsruhe.DE

Die Weiterentwicklung heutiger Mikroprozessoren durch Fortschreiben der Superskalartechnik läßt keine signifikante Leistungssteigerung erwarten. Da die feinkörnige Parallelität in einem konventionellen Befehlsstrom begrenzt ist, kann nur die zusätzliche Nutzung grobkörniger Parallelität zu einer weiteren Leistungssteigerung bei zukünftigen Mikroprozessoren führen. Die Kombination der Superskalar-Technik mit der mehrfädigen (multithreaded) Prozessortechnik erscheint dafür erfolgversprechend. Ein mehrfädiger, superskalarer Prozessor kann pro Takt Befehle aus mehreren Kontrollfäden den Ausführungseinheiten zuordnen und ausführen.

In der vorliegenden Arbeit wird ein mehrfädiger, superskalarer Mikroprozessor als Erweiterung des PowerPC 604 Prozessors vorgestellt. Pro Takt können Befehle aus 8 Kontrollfäden bis zu einer maximalen Zuordnungsbandbreite von 8 Befehlen den Ausführungseinheiten zugeordnet und von diesen parallel ausgeführt werden. Unsere Ergebnisse zeigen, daß ein Prozessor mit 8 Kontrollfäden und mit einer Zuordnungsbandbreite von 8 Befehlen einen Durchsatz von 4.2 Befehlen pro Takt erreicht.

1 Einführung

Heutige Mikroprozessoren nutzen feinkörnige Parallelität durch eine lange Befehlspipeline und durch die Superskalartechnik. DEC Alpha 21164, PowerPC 604 und 620, MIPS R10000, Sun UltraSparc I und HP PA-8000 sind vierfach superskalare Prozessoren, d.h., sie können pro Takt bis zu vier Befehle eines konventionellen Befehlsstroms ihren Ausführungseinheiten zuordnen.

Die VLSI-Technologie wird es zukünftigen Mikroprozessoren ermöglichen, die Zuordnungsbandbreite auf 8 oder mehr Befehle pro Takt zu erhöhen. Dies kann durch eine Weiterführung der Superskalartechnik geschehen (der IBM power2 Prozessor ist bereits 6-fach superskalar), durch die VLIW-Technik (Very long instruction word, Beispiel Philips TriMedia Prozessor), durch die SIMD-Technik (Single instruction, multiple data,

Beispiel HP PA-7100LC) oder durch einen CISC-Befehlssatz (Complex instruction set computer), bei dem komplexe Befehle dynamisch in ihre RISC-Partikel aufgespalten und diese dann durch die Superskalartechnik ausgeführt werden (Beispiele AMD K5 und Intel PentiumPro). Die feinkörnige Parallelität, die in einer konventionellen Befehlsfolge vorhanden ist und lokal durch eine Befehlszuordnungseinheit entdeckt werden kann, ist jedoch begrenzt [1]. Die Zuordnungsbandbreite kann deshalb schon von heutigen superskalaren Mikroprozessoren selten vollständig ausgenutzt werden. Unter Verwendung der SPEC92 Benchmarks zeigte sich, daß der PowerPC 620 im Mittel nur 0.96 bis 1.77 Befehle pro Takt ausführt [2], und sogar ein auf 8-fach superskalar erweiterter Alpha-Prozessor nicht mehr als durchschnittlich 1.5 Befehle pro Takt ausführen würde [3].

Als Lösung bietet sich die zusätzliche Nutzung auch grobkörniger Parallelität auf dem Prozessor-Chip an. Die zwei wesentlichen Ansätze hierfür sind der Multiprozessor-Chip und der mehrfädige Prozessor. Ein Multiprozessor-Chip integriert zwei oder mehr vollständige Prozessoren auf einem Chip. Jede Funktionseinheit eines Prozessors wird verdoppelt und unabhängig von den Funktionseinheiten der anderen Prozessoren auf dem Chip verwendet. Beispielsweise werden beim Texas Instruments TMS320C80 Multimedia Video Processor [4] vier Signalprozessoren und ein skalarer RISC-Prozessor auf einem Chip vereint.

Im Gegensatz zum Multiprozessor-Chip speichert ein mehrfädiger (multithreaded) Prozessor die Kontexte mehrerer Kontrollfäden in verschiedenen Registersätzen auf dem Chip. Die Funktionseinheiten werden den Kontrollfäden, die auf den Chip geladen sind, im Multiplex-Verfahren zugeordnet. In Abhängigkeit vom speziellen Entwurf des mehrfädigen Prozessors wird nur eine einzige Befehlspipeline verwendet, oder eine Zuordnungseinheit ordnet Befehle verschiedener Befehlspuffer simultan den Ausführungseinheiten zu. Wegen des Vorhandenseins mehrerer Registersätze geschieht ein Kontextwechsel bei mehrfädigen Prozessoren sehr schnell. Mehrfädige Prozessoren können deshalb - im Gegensatz zum Multiprozessor-Chip - sowohl die Wartezeiten überdecken, die beim Speicherzugriff entstehen, als auch die Zeiten, die bei der Synchronisation mehrerer Kontrollfäden auftreten.

Trotz der einfacheren Implementierbarkeit des Multiprozessor-Chips ist die Technik des mehrfädigen Prozessors in Kombination mit der Superskalartechnik ein vielversprechender Ansatz. Derzeit lassen sich drei verschiedene mehrfädige Prozessortechniken unterscheiden:

- Die Cycle-by-Cycle-Interleaving-Technik: Mit jedem Prozessortakt wird ein Befehl eines anderen Kontrollfadens in die Prozessorpipeline eingespeist. Dieses bei den Rechnern HEP [5] und Tera [6] eingesetzte Verfahren besitzt den Nachteil einer geringen Leistung, falls nur wenige Kontrollfäden als Last zur Verfügung stehen. Denn im Regelfall kann ein Befehl desselben Kontrollfadens erst in die Prozessor-Pipeline eingespeist werden, nachdem der Vorgängerbefehl die Pipeline vollständig durchlaufen hat (die Anwendung der Dependence-look-ahead-Technik bei HEP

und Tera mildert diesen Nachteil). Die Pipeline ist dafür sehr einfach, da keine Pipeline-Konflikte auftreten können.

- Die Block-Multithreading-Technik [7]: Die Befehle eines Kontrollfadens werden solange aufeinanderfolgend ausgeführt, bis ein Ereignis eintritt, das zu Wartezeiten führt. Dann wird ein Kontextwechsel durchgeführt. Dieses Ereignis ist beim Sparcle-Prozessor [8] ein Cache-Fehlzugriff oder eine fehlgeschlagene Synchronisation. Der Nachteil dieser Technik ist, daß solche Ereignisse erst spät in der Pipeline erkannt werden und nachfolgende bereits in der Pipeline vorhandene Befehle nicht verwendet werden können. Daraus resultiert ein Kontextwechselaufwand von 14 Takten beim Sparcle-Prozessor. Der Rhamma-Prozessor [9, 10] nutzt ebenfalls die Block-Multithreading-Technik. Im Gegensatz zu dem Sparcle-Prozessor wird bei jedem Lade-, Speicher- oder Synchronisationsbefehl ein Kontextwechsel durchgeführt. Die Kontextwechsel sind somit im Befehlsstrom codiert und können bereits in der Decodierphase der Pipeline erkannt werden. Dadurch kann der Aufwand für einen Kontextwechsel auf wenige Prozessortakte reduziert werden. Allerdings werden Kontextwechsel häufiger als beim Sparcle-Prozessor durchgeführt.

- Die Simultaneous-Multithreading-Technik [3]: Die Ausführungseinheiten eines superskalaren Prozessors werden simultan aus mehreren Befehlspuffern bestückt. Jedem Kontrollfaden ist ein eigener Registersatz und ein eigener Befehlspuffer zugeordnet. Es wird die Superskalartechnik mit breiter Zuordnungsbandbreite mit der mehrfädigen Prozessortechnik kombiniert. Bei genügend Last kann die volle Zuordnungsbandbreite ausgenutzt werden, da Befehle aus mehreren Kontrollfäden zur Ausführung bereitstehen. Wartezeiten, die bei der Ausführung eines Kontrollfadens auftreten, werden durch die Ausführung von Befehlen eines anderen Kontrollfadens überbrückt. Dieses Verfahren wurde in eingeschränktem Maße erstmals beim Entwurf des mehrfädigen Prozessors des Media Research Laboratory der Matsushita Electric Industrial Co. [11] vorgestellt. In der Arbeit von Tullsen, Eggers und Levy [3] wird die Simultaneous-Multithreading-Technik als Erweiterung eines Alpha 21164 Prozessors untersucht.

Unser eigener Ansatz eines mehrfädigen, superskalaren Prozessors nutzt eine ähnliche Befehlszuordnungstechnik, wie der Ansatz von Tullsen, Eggers und Levy, basiert jedoch auf der Erweiterung eines PowerPC 604 Prozessors [12] im Hinblick auf die mehrfädige Prozessortechnik.

Im nächsten Abschnitt wird der mehrfädige, superskalare Mikroprozessor im Detail vorgestellt. Die Abschnitte 3 und 4 skizzieren den Simulator, die Lastprogramme und die wichtigsten Simulationsergebnisse, die in anderen Arbeiten [13, 14, 15] ausführlich beschrieben werden.

2 Der mehrfädige, superskalare Prozessor

Der mehrfädige, superskalare Prozessor basiert auf dem PowerPC 604 Prozessor. Die meisten dort angewandten Implementierungstechniken moderner Mikroprozessoren werden übernommen. Das betrifft die Superskalartechnik mit einer Zuordnung in Programmreihenfolge, einer Ausführung auch außerhalb der Programmreihenfolge und einer Vervollständigungsphase, welche die ursprüngliche Programmreihenfolge wiederherstellt. Weiterhin werden separate Code- und Daten-Cache-Speicher, ein Sprungzieladreß-Cache, Sprungvorhersage, unabhängige Ausführungseinheiten mit Reservation Stations und Umbenennungspufferregister eingesetzt. Wir verwenden die Pipeline des PowerPC 604, aber erweitern diese um die Fähigkeit einer mehrfädigen Ausführung.

Als Basis des Befehlssatzes des mehrfädigen, superskalaren Prozessors wurde nicht der komplexe Befehlssatz der PowerPC-Architektur sondern derjenige des einfacheren DLX-Prozessors [16] gewählt. Dieser wurde wie beim Rhamma-Prozessor [10] um Synchronisationsbefehle, Datentransferbefehle zwischen verschiedenen Registersätzen und um Befehle zum Management von Kontrollfäden erweitert. Vereinfachungen gegenüber dem PowerPC 604 sind insbesondere die Verwendung einer statischen Sprungvorhersage und der Verzicht auf eine Gleitpunkteinheit.

Der Prozessor ist im Entwurf skalierbar gehalten: die Anzahl paralleler Kontrollfäden, die Größe der internen Puffer, der Registersätze und Cache-Speicher sowie die Anzahl und Typen der Ausführungseinheiten werden nicht festgelegt. Dies ermöglicht es, mit verschiedenen Konfigurationen zu experimentieren, um eine der aktuellen Chipgröße angepaßte Konfiguration zu ermitteln.

Der Prozessor (siehe Abb. 1) kann in vier Teile aufgeteilt werden:

- die Steuerpipeline, die aus Befehlslade- (Fetch), Decodier- (Decode), Zuordnungs- (Dispatch) und Vervollständigungseinheit (Completion) mit den zugehörigen Steuerpipeline-Puffern (Fetch Buffer, Issue Reservation Queue, Completion Queue) besteht,
- mehrere, voneinander unabhängige Ausführungseinheiten, wie z.B einfache und komplexe Integer-Einheiten, Lade-/Speicher-, Verzweigungs- und Kontrollfaden-Steuereinheit (Thread Control Unit),
- Befehls- und Daten-Cache-Speicher mit zugehöriger Cachesteuerung und Speicherzugriffseinheiten sowie der Sprungzieladreß-Cache (Branch Target Address Cache),
- Registersätze (Register Files), Umbenennungspufferregister (Register Rename) und ein Aktivitätsrahmen-Cache (Activation Frame Cache).

Das Blockschaltbild in Abbildung 1 zeigt eine Ausprägung des mehrfädigen, superskalaren Prozessors mit einer einzelnen Steuerpipeline, die mit dreifachen Zwischenpuffern ausgestattet ist, so daß drei Kontrollfäden gleichzeitig ausgeführt werden können.

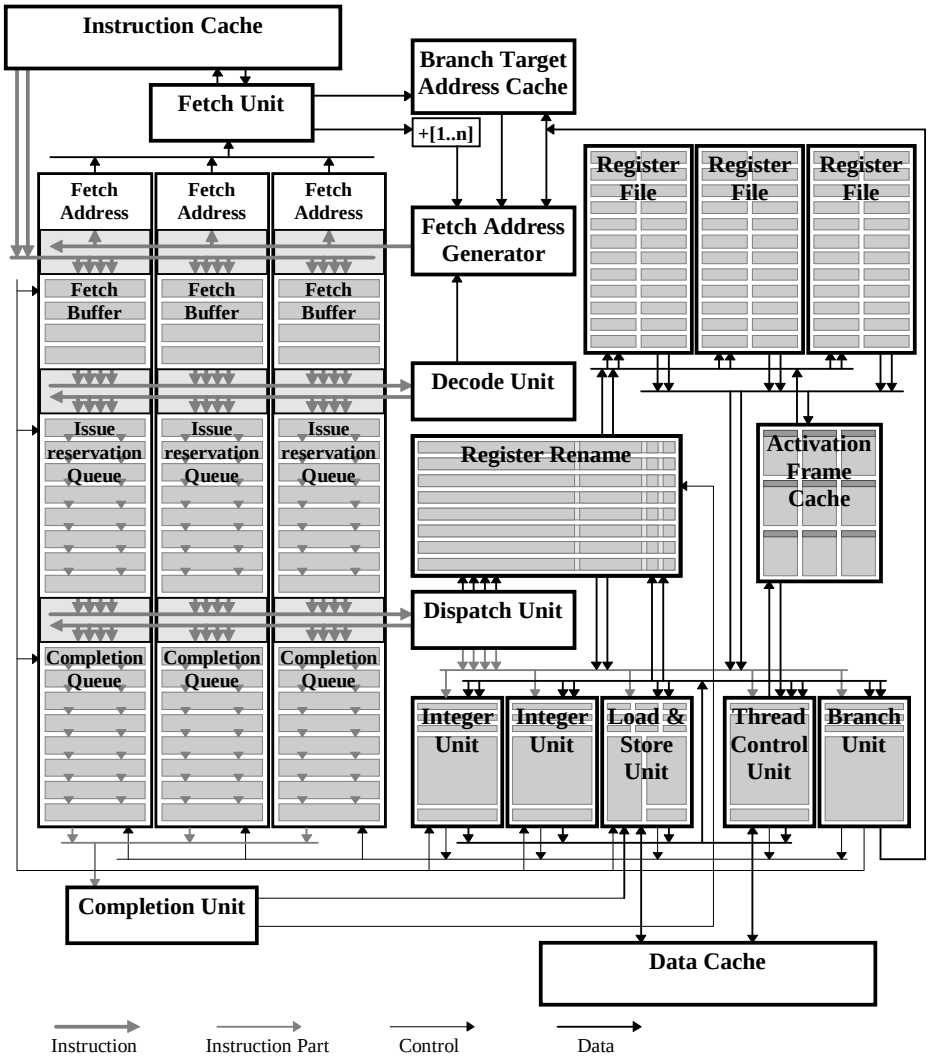

Abbildung 1: Der mehrfädige, superskalare Prozessor

Entsprechend zu der dreifachen Auslegung der Steuerpipeline-Puffer werden auch drei Registersätze verwendet. In der Abbildung sind weiterhin zwei Integer-Einheiten, eine Lade-/Speichereinheit, eine Kontrollfaden-Steuer- und eine Verzweigungseinheit eingezeichnet.

In der Steuerpipeline können auch mehrere Befehlslade- und Decodiereinheiten verwendet werden, so daß Anweisungen von mehreren Kontrollfäden gleichzeitig geladen und decodiert werden können. Die Befehlslade- und Decodiereinheiten können je nach Bedarf dynamisch den einzelnen Kontrollfäden zugeordnet werden.

Der Befehls-Cache ist als nicht blockierender Cache entworfen, d.h. im Falle eines Cache-Fehlzugriffs kann die Befehlsladeeinheit trotzdem weiter Befehle für einen anderen Kontrollfaden laden. Dies entkoppelt die Befehlsladeeinheit vom aktuellen Speicherzugriff.

Die Befehlslade- und die Dekodiereinheiten arbeiten jeweils an kontinuierlichen Anweisungsblöcken. Die aktuelle Größe der geladenen Befehlsblöcke ist aufgrund von Beschränkungen der Cache-Zeilengröße (ein Befehlsladezugriff darf keine Cache-Zeilen überlappen) und durch falsch vorhergesagte Sprünge nicht immer von maximaler Größe. Die maximale Größe entspricht der Größe der Steuerpipeline-Puffer und ist entsprechend der maximalen Zuordnungsbandbreite der aktuellen Prozessorauslegung gewählt.

Es wird nur jeweils eine einzelne Zuordnungs- und eine Vervollständigungseinheit verwendet. Die Zuordnungseinheit kann aufgrund ihrer Aufgabe nicht vervielfältigt werden. Die Vervollständigungseinheit kann gleichzeitig Anweisungen beliebiger Kontrollfäden (bis zu einer Maximalzahl) abschließen.

Die Zuordnung von Anweisungen ist keinen Beschränkungen bezüglich der Verteilung auf mehrere Kontrollfäden unterworfen, sondern lediglich durch die maximale Zuordnungsbandbreite begrenzt. Diese wird vor allem durch die eingeschränkte Anzahl an Bussen zu den Umbennungspufferregistern und den Ausführungseinheiten eingeschränkt. Es existiert keine feste Zuordnung zwischen Kontrollfäden und Ausführungseinheiten. Befehle werden immer in Programmordnung den Reservation Stations der Ausführungseinheiten zugeordnet. Sie werden dann in beliebiger Reihenfolge ausgeführt. Die Verwendung von Reservation Stations, die an die Ausführungseinheiten gebunden sind, vereinfacht die Aufgabe der Zuordnungseinheit, da nur ein Mangel an Befehlen, ein Mangel an Ressourcen und die maximale Zuordnungsbandbreite die gleichzeitige Befehlszuordnung einschränken.

Scheinbare Datenabhängigkeiten werden durch die Umbenennungspufferregister aufgelöst. Echte Datenabhängigkeiten werden von den Reservation Stations beachtet, die dafür sorgen, daß ein Befehl erst dann ausgeführt wird, wenn alle Quelloperanden in den Registern oder den Umbenennungspufferregistern bereit stehen. Befehle können somit von der Zuordnungseinheit ohne Test auf Daten- und Kontrollabhängigkeiten den Reservation Stations zugeordnet werden.

Die Umbenennungspufferregister, der Sprungzieladreß-Cache, der Daten- und der Befehls-Cache werden von allen aktiven Kontrollfäden gemeinsam benutzt. Dies ermöglicht die maximale Nutzung durch eine beliebige Anzahl von Kontrollfäden, kann aber im Falle der Cache-Speicher zum Flattern von Cache-Zeilen führen.

Die Inhalte der Register eines Registersatzes beschreiben den kompletten Zustand eines Kontrollfadens, und bilden somit den Aktivitätsrahmen. Ein Aktivitätsrahmen wird als aktiv bezeichnet, wenn der zugehörige Kontrollfaden gerade durch den Prozessor ausgeführt wird, also nur dann, wenn der Aktivitätsrahmen auch durch einen Registersatz im Prozessor repräsentiert wird. Neben den aktiven Aktivitätsrahmen enthält der Prozessor auch einen Cache, der die Aktivitätsrahmen von gerade nicht ausgeführten Kontrollfäden speichert. Die Inhalte der Registersätze und des Aktivitätsrahmen-Caches können ohne signifikante Verzögerung ausgetauscht werden.

Der Aktivitätsrahmen-Cache wird auch verwendet, um eine Registerfenstertechnik für die Parameterübergabe bei Unterprogrammaufrufen zu implementieren. Für jeden Un-

terprogrammaufruf wird ein neuer Aktivitätsrahmen erzeugt. Dies reduziert die Anzahl an Zugriffen auf den Daten-Cache.

Mit Ausnahme der Lade-/Speicher- und der Kontrollfaden-Steuereinheit kann jede der Ausführungseinheiten problemlos vervielfach werden. Die Kontrollfaden-Steuereinheit ist für die Erzeugung und die Beendigung von Kontrollfäden sowie für die Synchronisation und Kommunikation zwischen den Kontrollfäden verantwortlich. Es wäre sehr schwierig diese Einheit zu duplizieren, zudem wäre der Gewinn aufgrund des seltenen Auftretens von Kontrollfaden-Steuerbefehlen nur gering. Ein Duplizieren der Lade-/Speichereinheit ist im Prinzip möglich, jedoch wird die Daten-Cache-Speicherverwaltung erheblich komplizierter (im Falle des Befehls-Cache-Speichers gilt dies nicht in gleichem Maße, da aus dem Befehls-Cache nur geladen wird, keine Cache-Zeilen in den Speicher oder den Sekundär-Cache zurückgeschrieben werden und auch die Cache-Kohärenz nicht die gleiche Rolle wie beim Daten-Cache spielt).

Alle Standard-Integer-Operationen werden in einem einzelnen Takt von den einfachen Integer-Einheiten ausgeführt. Nur die Multiplikations- und Divisionsbefehle werden in einer komplexen Integer-Einheit ausgeführt und benötigen 5 bzw. 17 Takte. Das Multiplikationswerk ist im Gegensatz zum Divisionswerk als Pipeline realisiert.

Die Verzeigungseinheit kann pro Takt einen bedingten Sprung bearbeiten. Die Sprungvorhersage beginnt bereits in der Befehlsladeeinheit mit dem Sprungzieladreß-Cache. Sie wird in der Decodiereinheit mit einer statischen Vorhersage fortgeführt. Ein Sprung wird als genommen vorhergesagt, falls er rückwärts verzweigt, ein Sprung nach vorne wird als nicht genommen angenommen. Eine statische Sprungvorhersage vereinfacht den Prozessorentwurf, aber vermindert die Rate an korrekt vorhergesagten Sprüngen. In einem mehrfädigen Prozessor können die Latenzen, die durch falsch vorhergesagte Sprünge entstehen, durch die Ausführung anderen Kontrollfäden überdeckt werden, so daß diese sich in der Prozessorleistung nicht so stark wie bei heutigen Superskalar-Prozessoren bemerkbar machen.

Die Speicherschnittstelle ist in ihrer Breite und Geschwindigkeit frei konfigurierbar, so daß typsiche RAMs wie Static-Column, SDRAM oder EDO verwendet werden können. Alle Cache-Speicher sind ebenfalls frei als direkt-abgebildete oder n-fach satzassoziative Caches konfigurierbar, so daß das Flattern von Cache-Zeilen, das durch die Verwendung mehreren Kontrollfäden entstehen kann, durch eine adäquate Cache-Speicherorganisation verringert werden kann.

3 Simulator und Lastprogramme

Um den mehrfädigen, superskalaren Prozessor zu optimieren und zu bewerten wurde eine Softwaresimulation [13] durchgeführt, bei der alle im vorherigen Abschnitt beschriebenen Funktionseinheiten mit korrektem Zyklenverhalten simuliert werden. Der befehlsgesteuerte Simulator ermöglich es, alle bei der Ausführung auftretenden Zustände der Funktionseinheiten und der Puffer zu erfassen.

Neben einigen kleineren, von Hand erstellten Testprogrammen wurden synthetische, von einem Lastgenerator erzeugte Lastprogramme eingesetzt. Diese Lastprogramme wurden so erzeugt, daß eine Befehlszusammensetzung eingehalten wurde, wie sie von einem Compiler einer imperativen Hochsprache erzeugt wird. Compileroptimierungen werden dabei nicht berücksichtigt. Als Befehlszusammensetzung wurden folgende Werte gewählt:

Befehlsart	Durchschnittliches Vorkommen
Integer	63.8%
komplexe Integer	1.1%
Lade	13.2%
Speicher	7.0%
Sprung	10.8%
Kontrollfaden-Steuerung	4.1%

Die Befehlszusammensetzung enthält keine Gleitpunktbefehle. Durch die relative Häufigkeit von 20.2% Lade-/Speicherbefehlen ist bei Verwendung nur einer Lade-/Speichereinheit ein maximaler Durchsatz von ca. 5 Befehlen pro Takt möglich.

Die Simulationsergebnisse entstanden unter der Voraussetzung separater, 8 KByte großer, 4-fach satzassoziativer Daten- und Befehls-Cache-Speicher mit 32 Byte großen Cache-Zeilen und einer Cache-Füll-Rate von 4-2-2-2 Takten. Weiterhin wurden ein vollassoziativer, 32 Einträge fassender Sprungzieladreß-Cache und pro Kontrollfaden ein Registersatz mit 32 allgemeinen Registern gewählt. Es wurden bis zu 4 einfache und eine komplexe Integer-Einheit, eine Lade-/Speichereinheit, eine Verzweigungseinheit und eine Kontrollfaden-Steuereinheit eingesetzt, wobei jede Einheit mit je einer 4 Einträge fassenden Reservation Station ausgerüstet ist.

4 Simulationsergebnisse

Die Simulationen zeigten, daß 64 Umbenennungspuffer-Register und ein 12 Einträge fassender Rückordnungspuffer für bis zu 8-fach superskalare Zuordnung und bis zu 8 Kontrollfäden ausreichen (siehe auch [15]). Es zeigte sich weiterhin, daß der mehrfädige, superskalare Prozessor unter der Annahme nur einer Befehlslade- und nur einer Decodiereinheit nicht genügend Befehle laden und decodieren kann. Deshalb wurden zwei Befehlslade- und zwei Decodiereinheiten eingesetzt, welche die geladenen Kontrollfäden im Multiplex-Verfahren bearbeiten [15]. Die maximale Zuordnungsbandbreite und die Anzahl der geladenen Kontrollfäden wurden bei der Simulation jeweils zwischen 1 und 8 variiert. Die Anzahl der pro Takt von einer Befehlsladeeinheit geholten und von einer Decodiereinheit decodierten Befehle wurde wie auch die Größen der Steuerpipeline-Puffer an die gewählte Zuordnungsbandbreite angepaßt.

Unter diesen Voraussetzungen zeigten die Simulationen weiterhin, daß ein 8-fach superskalarer Prozessor mit einem Kontrollfaden nur einen Durchsatz von 1.14 Befehlen

erreicht, die pro Takt im Mittel ausgeführt werden. Ein vierfach superskalarer Prozessor schneidet mit 1.28 durchschnittlich pro Takt ausgeführten Befehlen sogar noch etwas besser ab [14], da das Flattern von Befehls-Cache-Zeilen gegenüber dem 5- bis 8-fach superskalaren Fall verringert wird.

Bis zu einer Zuordnungsbandbreite von 4 Befehlen läßt sich ein linearer Anstieg erreichen, danach wird ein Maximum bei etwa 4.2 Befehlen pro Takt nicht mehr überschritten. Das gilt auch falls die Anzahl der Kontrollfäden auf bis zu 8 erhöht wird [14]. Wie weitere Simulationen zeigen, ist dies gleichzeitig der maximale Durchsatz, der mit nur einer Lade-/Speichereinheit unter der Voraussetzung einer Befehlszusammensetzung mit 20% Lade-/Speicherbefehlen erreicht werden kann. Der Unterschied zwischen den erwarteten 5 Befehlen pro Takt und den erreichten 4.2 Befehlen ergibt sich aus Leertakten beim Zugriff in der Lade-/Speicher-Pipeline, die durch Daten-Cache-Fehlzugriffe hervorgerufen werden [15].

Für einen realistischen Vergleich verschiedener Prozessorkonfigurationen müssen die Hardwarekosten in Betracht gezogen werden. Wir messen die Hardwarekosten auf der Basis einer einfachen Formel [14], welche die Flächen der verschiedenen Einheiten auf dem PowerPC 604 Chip zur Grundlage hat. Unter Voraussetzung der so bewerteten Chip-Kosten zeigt ein Prozessor mit 4 Kontrollfäden und mit einer Zuordnungsbandbreite von 4 das beste Leistungs-/Kosten-Verhältnis [14].

5 Zusammenfassung

In der Arbeit wurden Entwurf, Simulator, Lastprogramme und Simulationsergebnisse eines mehrfädigen, superskalaren Prozessors vorgestellt. Pro Takt können Befehle aus 8 Kontrollfäden bis zu einer maximalen Zuordnungsbandbreite von 8 Befehlen den Ausführungseinheiten zugeordnet und von diesen parallel ausgeführt werden.

Unsere Ergebnisse zeigen, daß ein Prozessor mit 8 Kontrollfäden und mit einer Zuordnungsbandbreite von 8 Befehlen einen Durchsatz von 4.2 Befehlen pro Takt erreicht. Dies ist gleichzeitig der maximale Durchsatz der mit nur einer Lade-/Speichereinheit unter der Voraussetzung einer Befehlszusammensetzung mit 20% Lade-/Speicherbefehlen erreicht werden kann. Falls die Chip-Kosten in Betracht gezogen werden, zeigt ein Prozessor mit 4 Kontrollfäden und mit einer Zuordnungsbandbreite von 4 das beste Leistungs-/Kosten-Verhältnis.

6 Literaturhinweise

[1] W. Wall: Limits of Instruction-Level Parallelism. Fourth International Conference on Architectural Support for Programming Languages and Operating Systems, 1991, 176 - 188.

[2] T.A. Diep, C. Nelson, J.P. Shen: Performance Evaluation of the PowerPC 620 Microprocessor. The 22nd Annual International Symposium on Computer Architecture, Santa Margherita Ligure, 22.-24. Juni 1995, 163 - 174.

[3] D. E. Tullsen, S. J. Eggers, H. M. Levy: Simultaneous Multithreading: Maximizing On-Chip Parallelism. The 22nd Annual International Symposium on Computer Architecture, Santa Margherita Ligure, 22.-24. Juni 1995, 392 - 403.

[4] Texas Instruments: TMS320C80 Technical Brief. Multimedia Video Processor (MVP). Texas Instruments 1994.

[5] B. J. Smith: The Architecture of HEP. In: J. S. Kowalik (Ed.): Parallel MIMD Computation: The HEP Supercomputer and Its Applications. The MIT Press, Cambridge 1985.

[6] R. Alverson et al.: The Tera Computer System. 4th International Conference on Supercomputing, Amsterdam, 11.-15. Juni 1990, 1- 6.

[7] A. Agarwal: Performance Tradeoffs in Multithreaded Processors. IEEE Transactions on Parallel and Distributed Systems, Band 3, Heft 5, September 1992, 525 - 539.

[8] A. Agarwal et al.: The MIT Alewife Machine: Architecture and Performance. The 22nd Annual International Symposium on Computer Architecture, Santa Margherita Ligure, 22.-24. Juni 1995, 2 - 13.

[9] W. Grünewald, Th. Ungerer: Simulation einer vielfädigen Prozessorarchitektur. PARS-Workshop, Stuttgart, 9. - 11. Oktober 1995, PARS-Mitteilungen, Band 14, Dezember 1995, 219 - 226.

[10] W. Grünewald, Th. Ungerer: Towards Extremely Fast Context Switching in a Block-multithreaded Processor. Erscheint in: 22nd Euromicro Conference, Prague, 2.-5. September 1996.

[11] H. Hirata et al.: An Elementary Processor Architecture with Simultaneous Instruction Issuing from Multiple Threads. The 19nd Annual International Symposium on Computer Architecture, 1992, 136 - 145.

[12] S. P. Song, M. Denman, J. Chang: The PowerPC 604 RISC Microprocessor. IEEE Micro, Band 14, Heft 5, Oktober 1994, 8 - 17.

[13] U. Sigmund: Design of a Multithreaded Superscalar Processor. Diplomarbeit, Universität Karlsruhe 1995.

[14] U. Sigmund, Th. Ungerer: Evaluating a Multithreaded Superscalar Microprocessor versus a Multiprocessor Chip. 4th PASA Workshop - Parallel Systems and Algorithms, Forschungszentrum Jülich, 10.-12. April 1996.

[15] U. Sigmund, Th. Ungerer: Identifying Bottlenecks in a Multithreaded Superscalar Microprocessor. Erscheint in: EuroPar 96: Workshop 20 Instruction-Level Parallelism, Lyon 1996.

[16] J. L. Hennessy, D. A. Patterson: Computer Architecture a Quantitative Approach, San Mateo 1996.