(19) **Bundesrepublik Deutschland**
Deutsches Patent- und Markenamt

(10) **DE 698 35 100 T2** 2007.02.01

(12) # Übersetzung der europäischen Patentschrift

(97) **EP 1 031 074 B1**
(21) Deutsches Aktenzeichen: **698 35 100.2**
(86) PCT-Aktenzeichen: **PCT/US98/22030**
(96) Europäisches Aktenzeichen: **98 953 695.8**
(87) PCT-Veröffentlichungs-Nr.: **WO 1999/026132**
(86) PCT-Anmeldetag: **19.10.1998**
(87) Veröffentlichungstag der PCT-Anmeldung: **27.05.1999**
(97) Erstveröffentlichung durch das EPA: **30.08.2000**
(97) Veröffentlichungstag der Patenterteilung beim EPA: **28.06.2006**
(47) Veröffentlichungstag im Patentblatt: **01.02.2007**

(51) Int Cl.[8]: ***G06F 9/38*** (2006.01)
G06F 7/50 (2006.01)

(30) Unionspriorität:

65878 P	**17.11.1997**	**US**
115123	**14.07.1998**	**US**

(73) Patentinhaber:
Advanced Micro Devices, Inc., Sunnyvale, Calif., US

(74) Vertreter:
Grünecker, Kinkeldey, Stockmair & Schwanhäusser, 80538 München

(84) Benannte Vertragsstaaten:
DE, GB

(72) Erfinder:
WITT, B., David, Austin, TX 78759, US

(54) Bezeichnung: **PROZESSOR KONFIGURIERT UM VORAUSSCHAUENDE RESULTATE VON ZUSAMMENGEFASSTEN ÜBERTRAGUNGS-, VERGLEICHS- UND EINFACHEN ARITHMETISCHEN BEFEHLEN ZU PRODUZIEREN**

Beschreibung

Hintergrund der Erfindung

1. Gebiet der Erfindung

[0001] Die vorliegende Erfindung betrifft das Gebiet der Prozessoren und betrifft insbesondere die parallele Bearbeitung von Befehlen in Prozessoren.

2. Beschreibung des Stands der Technik

[0002] In superskalaren Prozessoren wird versucht, ein hohes Leistungsvermögen zu erreichen, indem mehrere Befehle pro Taktzyklus ausgegeben und ausgeführt werden, und indem bei der kleinsten möglichen Taktzykluszeit, die mit dem Aufbau verträglich ist, gearbeitet wird. In dem Maße, wie es gelingt, in einem bestimmten Prozessor mehrere Befehle pro Taktzyklus auszugeben und/oder auszuführen, kann auch ein hohes Leistungsvermögen erreicht werden. Um die mittlere Anzahl der Befehle, die pro Taktzyklus ausgegeben werden, zu erhöhen, haben Entwurfsingenieure superskalare Prozessoren entworfen, die größere Ausgaberaten anwenden. Ein superskalarer Prozessor mit „breiterer Ausgabe" ist in der Lage, eine größere maximale Anzahl an Befehlen pro Taktzyklus auszugeben (oder zu verteilen), als ein superskalarer Prozessor mit „schmaler Ausgabe" dazu in der Lage ist. Während der Taktzyklen, in denen die Anzahl der ausgegebenen Befehle größer ist als von einem Prozessor mit schmaler Ausgabe verarbeitbar ist, kann der Prozessor mit breiterer Ausgabe mehr Befehle ausgeben, wodurch eine höhere mittlere Anzahl an ausgegebenen Befehlen pro Taktzyklus erreicht wird.

[0003] Jedoch erfordern höhere Ausgaberaten im Allgemeinen zur Ausführung einen größeren Anteil an Hardware in dem Prozessor. Wenn eine ausreichende Ausführungshardware nicht vorgesehen ist, kann der Befehlsdurchsatz des Prozessors darunter leiden, selbst wenn der Prozessor in der Lage ist, eine große Anzahl an Befehlen gleichzeitig auszugeben. Die Ausführungshardware kann einen wesentlichen Anteil der Halbleitersubstratfläche einnehmen, wodurch die Gesamtchipgröße des Prozessors und damit dessen Kosten ansteigen.

[0004] Ferner sind viele Befehle relativ einfache Befehle, die durch eine einfache Ausführungshardware gehandhabt werden könnten. Beispielsweise sind Verschiebungs- bzw. „Move"-Befehle, die lediglich Registeroperanden spezifizieren (d. h. eine Verschiebung von einem Quellenregister zu einem Zielregister) einfache Befehle, die im Wesentlichen keine Hardware zum Ausführen erfordern. Verschiebebefehle mit einem Speicheroperanden und einem Registeroperanden beinhalten eine Adressenerzeugung, erfordern jedoch relativ wenig zusätzliche Hardware. Des weiteren sind Additionsbefehle (d. h. Addieren/Subtrahieren/inkrementieren/Dekrementieren) mit Registeroperanden relativ einfache Befehle. Die einfacheren Befehle können relativ häufig in üblichen Codierungssequenzen auftreten. Jedoch muss die Ausführungshardware in der Lage sein, auch die komplexeren Befehle auszuführen. Es wurden superskalare Prozessoren vorgeschlagen, die eine weniger kostenintensive Ausführungshardware besitzen, indem sowohl eine komplexe als auch eine einfache Einheit vorgesehen wird und indem die Ausgabe von Befehlen zu den Ausführungseinheiten so gesteuert wird, dass die einfachere Ausführungseinheit nur einfache Befehle erhält, während die komplexeren Einheiten einfachere Befehle oder komplexe Befehle erhalten. Obwohl eine derartige Strategie die von der Ausführungshardware eingenommene Fläche verringern kann, wird die Ausgabelogik komplexer. Die komplexe Logik kann eine größere Fläche einnehmen oder kann eine Beschränkung der Taktzykluszeit hervorrufen. Folglich sind effizientere Verfahren zum Handhaben der Mischung aus einfachen und komplexen Befehlen wünschenswert.

[0005] Um höhere Taktfrequenzen (d. h. kürzere Taktzykluszeiten) zu unterstützen, werden in superskalaren Prozessoren längere Pipelines (d. h. Pipelines mit mehr Stufen) sowie breitere Ausgaberaten eingesetzt. Während längere Pipelines das Erreichen höherer Taktfrequenzen zulassen, führen diese längeren Pipelines auch zu weiteren Entwurfsherausforderungen. Insbesondere wenn eine größere Anzahl an Befehlen abgeholt und in die Pipeline eingespeist wird, bevor vorhergehende Anweisungen in ihrer Ausführung abgeschlossen sind, ist eine zusätzliche Weiterleitungshardware erforderlich, um die parallele Ausführung zu unterstützen. Beispielsweise können mehrere Anweisungen über die Operandenabholstufe hinaus weiterschreiten, bevor die vorhergehenden Befehle ausgeführt sind. Wenn diese Befehle von dem vorhergehenden Befehlen abhängig sind, sind die Operanden für diese Befehle unter Umständen nicht verfügbar, wenn die Befehle die Operandenabholstufe erreichen. Diesen Befehlen kann das Weiterschreiten zu nachfolgenden Pipeline-Stufen erlaubt werden, wenn eine Weiterleitungshardware vorgesehen ist, um die Operanden zu den Befehlen zu leiten, wenn diese während der Ausführung in der Pipeline weiterschreiten. Jedoch ist die Weiterleitungshardware im

Hinblick auf Chipfläche und Komplexität aufwendig. Eine effizientere Lösung zur Bereitstellung der Operanden für abhängige Befehle ist daher vorteilhaft.

[0006] Im hierin verwendeten Sinne bezeichnet der Begriff „Abhängigkeit" die Beziehung zwischen einem ersten Befehl und einem nachfolgenden zweiten Befehl, wobei der zweite Befehl das Ausführen des ersten Befehles vor dem Ausführen des zweiten Befehls erfordert. Beispielsweise kann der zweite Befehl einen Quellenoperanden aufweisen, der durch Ausführen des ersten Befehls erzeugt wird. Im Allgemeinen ist ein Operand ein Wert, mit dem während des Ausführens eines Befehls operiert wird. Die Operanden für einen speziellen Befehl werden mittels Operandenspezifizierungen aufgefunden, die in den Befehl codiert sind. Beispielsweise können gewisse Operanden in Registern gespeichert sein, die in dem Prozessor verwendet sind. Eine Registeroperandenspezifizierung, die in dem Befehl codiert ist, wählt das spezielle Register aus, in dem der Operand gespeichert ist. Die Registeroperandenspezifizierung kann auch als eine Registeradresse oder eine Registernummer bezeichnet werden. Andererseits können andere Befehle einen Speicheroperanden spezifizieren, der in einer Speicherstelle innerhalb eines Hauptspeichers abgelegt ist, mit dem der Prozessor gekoppelt ist. Die Speicheradresse wird ebenfalls über Operandenspezifizierungen spezifiziert. Beispielswiese kann der Befehl eine Verschiebung beinhalten, die die Speicherstelle kennzeichnet, in der der Speicheroperand abgelegt ist. Andere Befehle können Adressenoperandenspezifizierungen enthalten, die Registeroperanden spezifizieren, die zur Bildung der Speicheradresse verwendet werden. Ein Operand kann ein Quellenoperand sein, wenn der Operand ein Eingabewert für den Befehl ist. Ein Operand kann ein Zieloperand sein, wenn der Operand das Ergebnis des Befehls ist. Die Zieloperandenspezifizierung spezifiziert die Speicherstelle, in der das Ergebnis des gerade ausgeführten Befehls abzulegen ist.

[0007] WO 93/20505 beschreibt eine Befehlsdisponierung bzw. eine Ablauforganisation in einem superskalaren RISC-Prozessor, der eine Ausführung außer der Reihenfolge ermöglicht. Es wird darin eine Registerumbenennungsschaltung offenbart, die ein Teil der Disponierlogik ist und die die parallele Abarbeitung von Befehlen beschleunigt, indem Datenabhängigkeiten in einer Abhängigkeitsprüfeinheit so aufgelöst werden, dass das Ergebnis eines Befehls in einem Temporärpufferbefehlsfenster gespeichert wird, ohne dass dieses in die Registerdatei übertragen wird. Das Weiterleiten des Ergebnisses zu einem zweiten Befehl unter Anwendung des Ergebnisses als einen Eingangsoperanden wird somit möglich. Jedoch tritt diese Bearbeitung in der Befehlsausführungsstufe der Pipeline auf, nachdem Befehle ausgegeben wurden sind.

[0008] US-A-5,675,758 offenbart einen leistungsfähigen Prozessor, in welchem bestehende Ausführungseinheiten durch das Hinzufügen einer ergänzenden Ganzzahlausführungseinheit erweitert sind, die als Add/Move-Einheit (AMU) bezeichnet wird, die ausgewählte Additionen und Verschiebungen parallel und in Bezug auf die anderen Ausführungseinheiten außerhalb der Reihe ausführt. Die AMU entfernt Datenabhängigkeiten und vergrößert damit die verfügbare Befehlsebenenparallelität.

[0009] Die obigen dargestellten Probleme werden zum großen Teil durch einen Prozessor gemäß der vorliegenden Erfindung gelöst, wie er in Anspruch 1 der begleitenden Patentansprüche definiert ist. In einer Ausführungsform umfasst der Prozessor eine vorausschauende Adressen/Ergebnis-Berechnungseinheit, die ausgebildet ist, Operandeninformationen zu empfangen (entweder den Operanden oder eine Marke, die den Befehl kennzeichnet, der den Operandenwert erzeugen wird), die dem entsprechenden Quellenoperanden eines oder mehrerer Befehle entspricht. Wenn die Operanden verfügbar sind, erzeugt die vorausschauende Adressen/Ergebnis-Berechnungseinheit entweder eine Vorausschauadresse für einen Speicheroperanden des Befehls oder ein Vorausschauergebnis, das einer funktionellen Befehlsoperation des Befehls entspricht. Die Vorausschauadresse kann einer Lade/Schreib- bzw. Lade/Speichereinheit für eine frühe Initiierung eines Speichervorgangs gemäß dem Befehl zugeführt werden. Das vorausschauende Ergebnis kann einer spekulativen Operandenquelle (beispielsweise einer Zukunftsdatei) für die Aktualisierung zugeführt werden. Vorteilhafterweise wird ein vorausschauender Zustand für ein Register früh in der Pipeline vorgesehen. Nachfolgende Befehle erhalten den vorausschauenden Zustand und benutzen den vorausschauenden Zustand, um in einer frühen Phase einen zusätzlichen vorausschauenden Zustand zu erzeugen. Andererseits erhalten die nachfolgenden Befehle den vorausschauenden Zustand und sind somit für die Abarbeitung vorbereitet, wenn diese in ein Befehlsfenster ausgegeben werden (im Gegensatz zum Warten auf das Befehlsfenster zur Ausführung des vorhergehenden Befehls).

[0010] Der Prozessor kann eine effizientere Weiterleitung auf vielfältige Weisen erreichen. Beispielsweise können viele Befehle vorausschauende Ergebnisse beim Abholen von Operanden erhalten und müssen daher nicht auf die nachfolgende Weiterleitung von Operanden warten. Da ferner einige Befehlsoperanden in der vorausschauenden Adressen/Ergebnis-Berechnungseinheit abgeschlossen werden können, sind weniger Funktionseinheiten erforderlich, um eine breite Ausgaberate zu unterstützen. Folglich können weniger weiterleiten-

de Busse eingerichtet werden (entsprechend der geringeren Anzahl an Funktionseinheiten). Des weiteren kann das Leistungsvermögen gesteigert werden, indem vorausschauende Operanden für Befehle vor den Befehlen bereitgestellt werden, die die vorausschauenden Operanden erzeugen, die die Ausführungsstufe der Prozessorpipeline erreichen.

[0011] In einer Ausführungsform umfasst der Prozessor ferner eine Operandenzusammenfasseinheit, die ausgebildet ist, die vorausschauenden Ergebnisse in nachfolgende gleichzeitig decodierte Befehle (reiheninterne Abhängigkeiten) zusammenzufassen. Des weiteren ist die Operandenzusammenfasseinheit ausgebildet, einen Vergleichsbefehl in einen nachfolgenden Verzweigungsbefehl einzubinden, der von dem Ergebnis des Vergleichs abhängt. Vorteilhafterweise können die Vergleichs/Verzweigungskombinationen eine einzelne Funktionseinheit besitzen.

[0012] Allgemein gesagt, betrifft die vorliegende Erfindung einen Prozessor mit einer reiheninternen Abhängigkeitsprüfeinheit, einer Vorausschauberechnungseinheit, einer Operandenzusammenfass- bzw. zusammenführeinheit und einem oder mehreren Befehlsfenstern. Die reiheninterne Abhängigkeitsprüfeinheit, die angeschlossen ist, mehrere Operandenspezifizierungen entsprechend einer Linie bzw. Reihe aus Befehlen zu empfangen, ist ausgebildet, Abhängigkeiten zwischen Befehlen innerhalb der Reihe aus Befehlen zu bestimmen, indem die mehreren Operandenspezifizierungen verglichen werden. Die Vorausschauberechnungseinheit, die angeschlossen ist, einen oder mehrere Operanden, die durch die mehreren Operandenspezifizierungen gekennzeichnet sind, zu empfangen, ist ausgebildet, ein vorausschauendes Ergebnis entsprechend einem ersten Befehl innerhalb der Reihe aus Befehlen zu berechnen, wenn jeder Operand, der von dem ersten Befehl zur Erzeugung des vorausschauenden Ergebnisse verwendet wird, innerhalb des einen oder der mehreren Operanden enthalten ist. Die Operandenzusammenfasseinheit, die mit der Vorrauschauberechnungseinheit und der reiheninternen Abhängigkeitsprüfeinheit verbunden ist, ist ausgebildet, das vorrausschauende Ergebnis als einen Operanden eines zweiten Befehls aus der Reihe aus Befehlen bereitzustellen. Die Operandenzusammenführeinheit stellt das vorausschauende Ergebnis als den Operanden in Reaktion bereit auf: (i) eine Angabe aus der Vorrauschauberechnungseinheit, dass das vorrausschauende Ergebnis gültig ist, (ii) eine Angabe aus der reiheninternen Abhängigkeitsprüfeinheit, dass der zweite Befehl von dem ersten Befehl abhängig ist. Die Befehlsfenster, die mit der Operandenzusammenführeinheit gekoppelt sind, sind ausgebildet, Befehle zu speichern, bis entsprechende Operanden bereitgestellt werden und nachfolgend die Befehle zum Ausführen auszuwählen. Die Operandenzusammenführeinheit ist ausgebildet, den Befehlsfenstern anzuzeigen, die Ausführung mindestens einer ersten Befehlsoperation des ersten Befehls, der durch das vorrausschauende Ergebnis repräsentiert ist, zu verhindern, wenn das vorrausschauende Ergebnis gültig ist.

[0013] Die vorliegende Erfindung betrifft ferner einen Prozessor mit einer Operandenzusammenführeinheit und einem oder mehreren Befehlsfenstern. Die Operandenzusammenführeinheit ist angeschlossen, mehrere Operanden zu empfangen, die einer Reihe bzw. Linie aus Befehlen entsprechen. Die Operandenzusammenführeinheit ist ausgebildet, einen oder mehrere der mehreren Operanden entsprechend einem ersten Befehl in der Linie aus Befehlen als Operanden eines zweiten Befehls aus der Linie aus Befehlen bereitzustellen in Reaktion auf: (i) dass der erste Befehl ein Vergleichsbefehl ist, (ii) der zweite Befehl ein bedingter Verzweigungsbefehl ist, und (iii) der zweite Befehl auf den ersten Befehl folgt. Die Befehlsfenster, die mit der Operandenzusammenführeinheit verbunden sind, sind ausgebildet, Befehle zu speichern, bis entsprechende Operanden bereitgestellt werden und nachfolgend die Befehle für das Ausführen auszuwählen. Die Operandenzusammenführeinheit ist ausgebildet, den Befehlsfenstern anzuzeigen, den Empfang eines ersten Befehls zu verhindern, wenn: (i) der erste Befehl ein Vergleichsbefehl ist, (ii) der zweite Befehl ein bedingter Verzweigungsbefehl ist, und (iii) der zweite Befehl auf den ersten Befehl folgt.

[0014] Des weiteren betrifft die vorliegende Erfindung ein Verfahren zum Ausführen einer Reihe aus Befehlen in einem Prozessor. Die Reihe aus Befehlen ist decodiert, um mehrere Operandenspezifizierungen zu erkennen. Zieloperandenspezifizierungen jedes Befehls innerhalb der Reihe von Befehlen werden mit Quellenoperandenspezifizierungen jedes nachfolgenden Befehls aus der Reihe der Befehle verglichen, um reiheninterne Abhängigkeiten zu erkennen. Eine spekulative Operandenquelle wird ausgelesen, um Quellenoperanden zu erkennen, die durch Quellenoperandenspezifizierungen gekennzeichnet sind. Es wird ein vorrausschauendes Ergebnis für einen ersten Befehl aus der Reihe berechnet, wenn die Quellenoperanden in der spekulativen Operandenquelle verfügbar sind. Einem Befehlsfenster, das ausgebildet ist, den ersten Befehl aufzunehmen, wird angezeigt, die Ausführung mindestens einer ersten Befehlsoperation des ersten Befehls zu verhindern, der das vorrausschauende Ergebnis erzeugt, wenn das vorrausschauende Ergebnis erfolgreich berechnet ist. Das vorrausschauende Ergebnis wird einem zweiten Befehl aus der Reihe aus Befehlen bereitgestellt, wenn der Vergleich eine Abhängigkeit des zweiten Befehls von dem ersten Befehl angibt. Das Bereitstellen wird vor dem Speichern des zweiten Befehls in dem Befehlsfenster ausgeführt.

[0015] Die vorliegende Erfindung betrifft ferner ein Computersystem mit einem Prozessor und einer Eingabe/Ausgabe-(I/O) Einrichtung. Der Prozessor umfasst eine Ausrichteinheit, die ausgebildet ist, eine Linie bzw. Reihe aus Befehlen auf mehrere Ausgabepositionen auszurichten, eine spekulative Operandenquellen, die ausgebildet ist, spekulative Operanden zu speichern, ein Befehlsfenster, das ausgebildet ist, Befehle zu speichern, bis Operanden für diese Befehle bereitgestellt sind und eine Vorausschau/Zusammenführeinheit. Die Vorausschau/Zusammenführeinheit ist mit der spekulativen Operandenquelle, dem Befehlsfenster und der Ausrichteinheit verbunden und ist ausgebildet, Operanden von der spekulativen Operandenquelle in Reaktion darauf, dass die Reihe aus Befehlen von der Ausrichteinheit empfangen wird, auszulesen. Die Vorausschau/Zusammenführeinheit ist ausgebildet, ein vorrausschauendes Ergebnis entsprechend einem ersten Befehl innerhalb der Befehlsreihe in Reaktion auf die Operanden zu erzeugen. Die Vorausschau/Zusammenführeinheit ist ferner ausgebildet, die spekulative Operandenquelle mit dem vorrausschauenden Ergebnis zu aktualisieren und ist ferner ausgebildet, das vorrausschauende Ergebnis an einen zweiten Befehl innerhalb der Befehlsreihe weiterzuleiten, der vom dem ersten Befehl abhängig ist. Die Vorausschau/Zusammenführeinheit ist ausgebildet, dem Befehlsfenster anzuzeigen, das Ausführen mindestens einer ersten Befehlsoperation des ersten Befehls zu unterdrücken, der das vorausschauende Ergebnis erzeugt. Die I/O-Einrichtung, die mit dem Prozessor verbunden ist, ist ausgebildet, eine Kommunikation zwischen dem Computersystem und einem weiteren Computersystem, mit welchem die I/O-Einrichtung verbunden ist, herzustellen.

Kurze Beschreibung der Zeichnungen

[0016] Weitere Aufgaben und Vorteile der Erfindung gehen aus dem Studium der folgenden detaillierten Beschreibung und durch Bezugnahme zu den begleitenden Zeichnungen hervor, in denen:

[0017] Fig. 1 eine Blockansicht einer Ausführungsform eines Prozessors ist;

[0018] Fig. 2 eine Blockansicht einer Ausführungsform einer Abhol/Such- bzw. Abtasteinheit ist, die in Fig. 1 gezeigt ist;

[0019] Fig. 3 eine Blockansicht einer Ausführungsform einer Vorrausschau/Zusammenführeinheit ist, die in Fig. 1 gezeigt ist;

[0020] Fig. 4 eine Blockansicht von Bereichen der in Fig. 3 gezeigten Vorausschau/Zusammenführeinheit ist, wobei interne Verbindungen gezeigt sind;

[0021] Fig. 5 eine Ansicht ist, die einen Teil einer Ausführungsform einer vorrausschauenden Adressen/Ergebnis-Erzeugungseinheit, die in den Fig. 3 und Fig. 4 gezeigt ist, darstellt;

[0022] Fig. 6 eine Wahrheitstabelle gemäß einer Ausführungsform einer in Fig. 5 gezeigten Steuereinheit für die vorrausschauende Adressenerzeugung ist;

[0023] Fig. 7 eine Wahrheitstabelle gemäß einer Ausführungsform einer in Fig. 5 gezeigten Steuereinheit für die vorrausschauende Adressenerzeugung ist;

[0024] Fig. 8 eine Blockansicht eines Teils einer Ausführungsform der Operandenzusammenführeinheit ist, die in Fig. 3 dargestellt ist;

[0025] Fig. 9 eine Blockansicht eines Flussdiagramms ist, das eine Ausführungsform einer in Fig. 8 gezeigten Steuerungseinheit darstellt; und

[0026] Fig. 10 eine Blockansicht eines Computersystems ist, das den in Fig. 1 gezeigten Prozessor aufweist.

[0027] Obwohl die Erfindung diversen Modifizierungen und alternativen Formen unterliegen kann, sind dennoch spezielle Ausführungsformen beispielhaft in den Zeichnungen gezeigt und werden hierin detailliert beschrieben. Es sollte jedoch beachtet werden, dass die Zeichnungen und die detaillierte Beschreibung nicht beabsichtigen, die vorliegende Erfindung auf die speziellen offenbarten Formen einzuschränken, sondern die Erfindung soll vielmehr alle Modifizierungen, Äquivalente und Alternativen abdecken, die innerhalb des Grundgedankens und Schutzbereichs der vorliegenden Erfindung, wie sie durch die angefügten Patentansprüche definiert ist, liegen.

[0028] Fig. 1 zeigt eine Blockansicht einer Ausführungsform eines superskalaren Prozessors **10**. Andere Ausführungsformen sind möglich und hierin mit eingeschlossen. In der in Fig. 1 gezeigten Ausführungsform umfasst der Prozessor **10** eine Vordecadierungseinheit **12**, einen L1 I-Cache-Speicher **14**, einen L0 I-Cache-Speicher **16**, eine Abhol/Such- bzw. Abtasteinheit **18**, eine Befehlswarteschlange **20**, eine Ausrichteinheit **22**, eine Vorrausschau/Zusammenführeinheit **24**, eine Zukunftsdatei **26**, eine Umordnungspuffer/Registerdatei **28**, ein erstes Befehlsfenster **30a**, ein zweites Befehlesfenster **30b**, mehrere Funktionseinheiten **32a**, **32b**, **32c** und **32d**, mehrere Adressenerzeugungseinheiten **34a**, **34b**, **34c** und **34d**, eine Lade/Speicher- bzw. Schreibeinheit **36**, einen L1 D-Cache-Speicher **38**, eine FPU/Multimediaeinheit **40** und eine externe Schnittstelleneinheit **42**. Elemente, die hierin durch eine spezielle Bezugszahl gefolgt von einem bestimmten Buchstaben bezeichnet sind, werden gemeinsam auch unter Anwendung des Bezugszeichens alleine bezeichnet. Beispielsweise werden die Funktionseinheiten **32a**, **32b**, **32c** und **32d** gemeinsam auch als Funktionseinheiten **32** bezeichnet.

[0029] In der Ausführungsform aus Fig. 1 ist die externe Schnittstelleneinheit **42** mit der Vordecodierungseinheit **12**, dem L1 D-Cache-Speicher **38**, einer L2-Schnittstelle **44** und einer Busschnittstelle **46** verbunden. Die Vordecodierungseinheit **12** ist ferner mit dem L1 I-Cache-Speicher **14** verbunden. Der L1 I-Cache-Speicher **14** ist mit dem L0 I-Cache-Speicher **16** und der Abhol/Sucheinheit **18** verbunden. Die Abhol/Sucheinheit **18** ist ferner mit dem L0 I-Cache-Speicher **16** und der Befehlswarteschlange **20** verbunden. Die Befehlswarteschlange **20** ist mit der Ausrichteinheit **22** verbunden, die wiederum mit der Vorausschau/Zusammenführeinheit **24** verbunden ist. Die Vorausschau/Zusammenführeinheit **24** ist ferner mit der Zukunftsdatei **26**, der Umordnungspuffer/Registerdatei **28**, der Lade/Schreibeinheit **36**, dem ersten Befehlsfenster **30a**, dem zweiten Befehlsfenster **30b** und der FPU-Multimediaeinheit **40** verbunden. Die FPU-Multimediaeinheit **40** ist mit der Lade/Schreibeinheit **36** und der Umordnungspuffer/Registerdatei **28** verbunden. Die Lade/Schreibeinheit **36** ist mit dem L1 D-Cache-Speicher **38** verbunden. Das erste Befehlsfenster **30a** ist mit den Funktionseinheiten **32a** bis **32b** und den Adressenerzeugungseinheiten **34a** bis **34b** verbunden. In ähnlicher Weise ist das zweite Befehlsfenster **30b** mit den Funktionseinheiten **32c** bis **32d** und den Adressenerzeugungseinheiten **34c** bis **34d** verbunden. Der L1 D-Cache-Speicher **38**, die Funktionseinheiten **32** und die Adressenerzeugungseinheiten **34** sind jeweils mit mehreren Ergebnisbussen **48** verbunden, die wiederum mit der Lade/Schreibeinheit **36**, dem ersten Befehlsfenster **30a**, dem zweiten Befehlsfenster **30b**, der Umordnungspuffer/Registerdatei **28** und der Zukunftsdatei **26** verbunden sind.

[0030] Allgemein gesagt, ist die Vorausschau/Zusammenführeinheit **24** ausgebildet, vorrausschauende Ergebnisse für gewisse Befehle vor der Ausgabe dieser Befehle zur Ausführung zu erzeugen. Ein vorrausschauendes bzw. ein Vorausschauergebnis wird für gewisse Befehle erzeugt, wenn die Operanden, die zum Erzeugen des Ergebnisses verwendet werden, innerhalb der Zukunftsdatei **26** beim Abholen von Operanden von dieser Datei verfügbar sind. Ein Operand ist verfügbar innerhalb der Zukunftsdatei **26**, wenn der Wert in der Zukunftsdatei **26** gültig ist (im Gegensatz dazu, dass dieser eine Ergebniswartenschlangenmarkierung ist, die den Befehl kennzeichnet, der den Operanden als Ergebnis beim Ausführen erzeugen soll). Wenn ein vorrausschauendes Ergebnis erfolgreich erzeugt ist, wird das Ergebnis der Zukunftsdatei **26** zur Speicherung zugeführt und wird abhängigen Befehlen innerhalb der gleichen „Linie bzw. Reihe" aus Befehlen zugeführt (d. h. gleichzeitig mit dem Befehl decodiert). Vorteilhafterweise werden spekulative Ergebnisse für nachfolgende Befehle früher in der Pipeline verfügbar. Somit sind die vorrausschauenden Ergebnisse für die nachfolgenden Befehle berechenbar. Ferner kann das Weiterleiten effizienter sein, da das Weiterleiten häufiger über die Zukunftsdatei **26** möglich ist. Folglich kann die Weiterleitungshardware reduziert werden. Des weiteren kann der Aufwand an Ausführungshardware, die in dem Prozessor **10** eingesetzt wird, reduziert werden, wobei dennoch eine breite Ausgaberate unterstützt wird, da Ergebnisse außerhalb der Ausführungshardware für gewisse Befehle erzeugt wird.

[0031] In einer Ausführungsform ist die Vorausschau/Zusammenführeinheit **24** ausgebildet, eine vorrausschauende Adresse für Befehle mit einem Speicheroperanden zu erzeugen. Für derartige Befehle wird die Adressenerzeugungsbefehlsoperation, die ansonsten von einer der Adressenerzeugungseinheiten **34** ausgeführt wird, von der Vorausschau/Zusammenführeinheit **24** ausgeführt. Folglich erhält die Lade/Schreibeinheit **36** die Adresse früher in der Pipeline als dies ansonsten möglich ist. Wenn der Befehl eine Additionsbefehlsoperation oder eine Register-Register-Verschiebung bzw. Übertragung spezifiziert und keinen Speicheroperanden enthält, kann die Vorausschau/Zusammenführeinheit **24** ein vorrausschauendes Ergebnis erzeugen. Folglich wird die funktionale Befehlsoperation, die ansonsten durch eine der Funktionseinheiten **32** ausgeführt würde, von der Vorausschau/Zusammenführeinheit **24** früher in der Pipeline ausgeführt.

[0032] Die Vordecodierungseinheit **12** empfängt Befehlsbytes, die von der externen Schnittstelleneinheit **42** abgeholt werden, und vordecodiert die Befehlsbytes vor ihrer Speicherung innerhalb des L1 I-Cache-Speichers **14**. Die vordecodierte Information, die von der Vordecodierungseinheit **12** erzeugt wird, wird in dem L1 I-Cache-Speicher **14** ebenso abgelegt. im Allgemeinen wird die vordecodierte Information vorgesehen, um die Erkennung von Befehlsmerkmalen zu unterstützen, was während des Abholens und der Ausgabe von Befehlen vorteilhaft sein kann, die jedoch während des Abhol- und Ausgabeoperationsvorgangs jedoch nur unter Schwierigkeiten rasch erzeugt werden kann. Der Begriff „Vordecodieren", wie er hierin verwendet ist, bezeichnet das Decodieren von Befehlen, um die Vordecodierungsinformation zu erzeugen, die später mit den Befehlsbytes gespeichert wird, die in einem Befehlscache-Speicher (beispielsweise L1 I-Cache-Speicher **14** und/oder L0 I-Cache-Speicher **16**) decodiert werden.

[0033] In einer Ausführungsform verwendet der Prozessor 2 Bits an Vordecodierungsinformation pro Befehlsbyte. Eines der Bits, das als das „Startbit" bezeichnet wird, gibt an, ob das Befehlsbyte das erste Byte eines Befehls ist. Wenn eine Gruppe aus Befehlsbytes abgeholt wird, kennzeichnet der entsprechende Satz aus Startbits die Grenzen zwischen den Befehlen innerhalb der Gruppe aus Befehlsbytes. Folglich können mehrere Befehle gleichzeitig aus der Gruppe aus Befehlsbytes ausgewählt werden, indem nach den entsprechenden Startbits gesucht wird. Während die Startbits verwendet werden, um Befehlsgrenzen zu erkennen, indem das Anfangsbytes jedes Befehls ermittelt wird, können Endbits alternativ verwendet werden, um Befehlsgrenzen zu erkennen, indem das letzte Byte jedes Befehls ermittelt wird.

[0034] Das zweite Vordecodierungsbit, das in dieser Ausführungsform verwendet wird, wird als das "Steuertransfer"-Bit bezeichnet und gibt an, welche Befehle Verzweigungsbefehle sind. Das Steuertransferbit entspricht dem Anfangsbyte eines Befehls, der angibt, ob der Befehl ein Verzweigungsbefehl ist oder nicht. Das Steuertransferbit, das nachfolgenden Bytes des Befehls entspricht, ist ein „beliebig" mit Ausnahme für relative Verzweigungsbefehle mit einem kleinen Verschiebungsfeld. Gemäß einer speziellen Ausführungsform ist das kleine Verschiebungsfeld ein 8-Bit-Feld. Im Allgemeinen bezeichnet ein „kleines Verschiebungsfeld" ein Verschiebungsfeld mit weniger Bits. als die Zieladresse, die von den Verzweigungsbefehlen erzeugt wird. Für relative Verzweigungsbefehle mit kleinen Verschiebungsfeldern wird das Steuertransferbit, das dem Verschiebungsbyte entspricht, in der Weise verwendet, wie dies nachfolgend beschrieben ist.

[0035] Zusätzlich zu dem Erzeugen von Vordecodierungsinformationen, die den Befehlsbytes entsprechen, ist die Vordecodierungseinheit **12** ausgebildet, das Verschiebungsfeld relativer Verzweigungsbefehle umzucodieren, um tatsächlich die Zieladresse in der vorliegenden Ausführungsform zu speichern. Anders ausgedrückt, die Vordecodierungseinheit **12** addiert die Verschiebung des relativen Verzweigungsbefehles, die dem relativen Verzweigungsbefehl entspricht, wie er durch den Befehlssatz definiert ist, der von dem Prozessor **10** verwendet wird. Die sich ergebende Zieladresse wird in das Verschiebungsfeld als ein Ersatz für die Verschiebung codiert, und das aktualisierte Verschiebungsfeld wird in dem L1 I-Cache-Speicher **14** anstelle des ursprünglichen Verschiebungsfeldes gespeichert. Die Zieladressenerzeugung wird vereinfacht, indem relative Zieladressen im Voraus berechnet werden, und somit kann der Verzweigungsvorhersagemechanismus effizienter arbeiten.

[0036] In einer Ausführungsform des Prozessors **10**, in der der x86-Befehlssatz verwendet ist, ist die Vordecodierungseinheit **12** ausgebildet, 8 Bit- und 32-Bit-Verschiebungsfelder neu zu codieren. Die 32-Bit-Verschiebungsfelder können alle Zieladressen aufnehmen. Andererseits wird das 8-Bit-Verschiebungsfeld codiert. Insbesondere werden das 8-Bit-Verschiebungsfeld und entsprechende Steuertransfervordecodierungsbits in einen Cache-Zeilenoffsetbereich und in einen relativen Cache-Zeilenbereich unterteilt. Der Cache-Zeilenoffsetbereich ist der Cache-Zeilenoffsetbereich der Zieladresse. Der relative Cache-Zeilenbereich definiert die Cache-Zeile, die von der Zieladresse angegeben ist (die „Zielcache-Zeile") in Bezug auf eine Anzahl an Cache-Zeilen über oder unter der Cache-Zeile, in der der relative Verzweigungsbefehl gespeichert ist. Eine erste Cache-Zeile ist über einer zweiten Cache-Zeile, wenn jedes Byte in der ersten Cache-Zeile an einer Adresse gespeichert ist, die numerisch größer ist als die Adressen, an denen die Bytes in der zweiten Cache-Zeile gespeichert sind. Andererseits ist eine erste Cache-Zeile unter der zweiten Cache-Zeile, wenn jedes Byte in der ersten Cache-Zeile an einer Adresse gespeichert ist, die numerisch kleiner ist als die Adressen, an denen die Bytes in der zweiten Cache-Zeile gespeichert sind. Eine 8-Bit-Verschiebung mit Vorzeichen spezifiziert eine Adresse, die innerhalb +/– 128 Bytes der Adresse liegt, die dem Verzweigungsbefehl entspricht. Folglich ist die Anzahl über und unter den Cache-Zeilen, die durch einen relativen Verzweigungsbefehl mit einer 8-Bit-Verschiebung erreicht werden kann, begrenzt. Der relative Cache-Zeilenbereich codiert diesen begrenzten Salz aus darüberliegenden und darunterliegenden Cache-Zeilen. Im Allgemeinen besitzen Verzweigungsbefehle, die ein kleines Verschiebungsfeld besitzen, Verschiebungen innerhalb eines vordefinierten Bereichs, wohin größere Verschiebungsfelder Werte außerhalb des vordefinierten Bereichs speichern können.

[0037] Die nachfolgenden Tabellen 1 und 2 zeigen eine beispielhafte Codierung der Vordecodierungsinformation, die einem Byte entspricht, gemäß einer Ausführungsform des Prozessors **10**.

Tabelle 1: Vordecodierungscodierung

Startbit	Steuertransferbit	Bedeutung
1	0	Startbyte eines Befehls, der keine Verzweigung ist
1	1	Startbyte eines Verzweigungsbefehls
0	x	Keine Befehlsgrenze. Steuertransferbit, das der Verschiebung entspricht, wird auf 8 Bit relative Verzweigungen angewendet, um die Zieladresse zu codieren, wie in der nachfolgenden Tabelle 2 gezeigt ist.

Tabelle 2: Zieladressencodierung

Steuer-transferbit	Verschiebungsbyte signifikanteste Bits (binär)	Bedeutung
0	00	innerhalb aktueller Cache-Zeile
0	01	eine Cache-Zeile darüber
0	10	zwei Cache-Zeilen darüber
1	01	eine Cache-Zeile darunter
1	10	zwei Cache-Zeilen darunter

Beachte: Verbleibende Verschiebungsbytes-Bits sind der Offset innerhalb der Ziel-Cache-Zeile. Das Steuertransferbit ist tatsächlich eine Richtung und die signifikantesten Bits des Verschiebungsbytes sind die Anzahl der Cache-Zeilen.

[0038] Die Vordecodierungseinheit **12** übertragt die empfangenen Befehlsbytes und die entsprechende Vordecodierungsinformation zur Speicherung an den L1 I-Cache-Speicher **14**. Der L1 I-Cache-Speicher **14** ist ein Hochgeschwindigkeitscache-Speicher zur Speicherung von Befehlsbytes und Vordecodierungsinformationen. Der L1 I-Cache-Speicher **14** kann eine beliebige geeignete Konfiguration aufweisen, zur der eine direkt abgebildete und eine teilassoziative Konfiguration gehört. In einer speziellen Ausführungsform ist der L1 I-Cache-Speicher **14** ein 128 KB zweiwegeteilassoziativer Cache, in welchem 64 Byte-Cache-Zeilen verwendet sind. Der L1 I-Cache-Speicher **14** umfasst einen zusätzlichen Speicherplatz für die Vordecodierungsinformation, die den darin gespeicherten Befehlsbytes entspricht. Der zusätzliche Speicherplatz ist in ähnlicher Weise wie der Speicherplatz für die Befehlsbytes organisiert. Im hierin verwendeten Sinne bezeichnet der Begriff „Cache-Zeile" die Einheit für die Reservierung von Speicherplatz in einem speziellen Cache-Speicher. Im Allgemeinen werden die Bytes innerhalb einer Cache-Zeile von dem Cache-Speicher als eine Einheit manipuliert (d. h. reserviert und freigegeben). In einer Ausführungsform wird der L1 I-Cache-Speicher **14** linear adressiert und physikalisch markiert. Ein Cache wird linear adressiert, wenn mindestens eines der Adressenbits, das zur Indizierung des Cache-Speichers verwendet wird, ein lineares Adressenbit ist, das nachfolgend in ein physikalisches Adressenbit übersetzt wird. Die Markierungen eines linear adressierten/physikalisch markierten Cache-Speichers enthalten jedes übersetzte Bit zusätzlich zu den Bits, die nicht zur Indizierung verwendet sind. Wie dies durch die x86-Architektur spezifiziert ist, werden Befehle definiert, um logische Adressen zu erzeugen, die über einen Segmentierungsübersetzungsmechanismus in eine lineare Adresse und ferner über einen Seitenübersetzungsmechanismus in eine physikalische Adresse übersetzt werden. Es ist zunehmend gebräuchlich, einen ausgeglichenen Adressierungsmodus einzusetzen, in welchem die logische Adresse und die entsprechende lineare Adresse gleich sind. Der Prozessor **10** kann so ausgebildet sein, dass dieser einen ausgeglichenen Adressierungsmodus ausübt. Folglich sind Abholadressen, Zieladressen, etc., wie sie beim Ausführen von Befehlen erzeugt werden, lineare Adressen. Um zu bestimmen, ob ein Treffer in L1 I-Cache-Spei-

cher **14** vorliegt, wird die durch die Abhol/Sucheinheit **18** repräsentierte lineare Adresse unter Anwendung eines Übersetzungsnebenordnungspuffers (TLB) in eine entsprechende physikalische Adresse übersetzt, die mit den physikalischen Marken der indizierten Cache-Zeilen verglichen wird, um einen Treffer/Nichttreffer zu erkennen. Wenn der ausgeglichene Adressierungsmodus nicht verwendet wird, kann der Prozessor dennoch eine Codierung ausführen, wobei jedoch zusätzliche Taktzyklen verwendet werden, um lineare Adressen aus den logischen Adressen zu erzeugen.

[0039] Der L0 I-Cache-Speicher **16** ist ebenfalls ein Hochgeschwindigkeitscache-Speicher zur Speicherung von Befehlsbytes. Da der L1 I-Cache-Speicher **14** groß ist, kann auch die Zugriffszeit für den L1 I-Cache-Speicher **14** groß sein. In einer speziellen Ausführungsform verwendet der L1 I-Cache-Speicher **14** zwei Taktzyklen an Zugriffszeit. Um eine Abholzugriffszeit innerhalb eines einzelnen Taktzyklus zu ermöglichen, wird der L0 I-Cache-Speicher **16** verwendet. Der L0 I-Cache-Speicher **16** ist deutlich kleiner als der L1 I-Cache-Speicher **14** und bietet damit eine deutlich geringere Zugriffszeit. In einer speziellen Ausführungsform ist der L0 I-Cache-Speicher **16** ein 512 Byte vollassoziativer Cache-Speicher. Ähnlich zu dem L1 I-Cache-Speicher **14** ist der L0 I-Cache-Speicher **16** ausgebildet, Cache-Zeilen von Befehlsbytes und entsprechende Vordecodierungsinformationen (beispielsweise 512 Bytes speichern 864 Cache-Zeilen und entsprechende Vordecodierungsdaten werden zusätzlichen Speicherplatz abgelegt) zu speichern. In einer Ausführungsform wird der L0 I-Cache-Speicher **16** linear adressiert und linear markiert.

[0040] Die Abhol/Sucheinheit **18** ist ausgebildet, Adressen für den L0 I-Cache-Speicher **16** zu erzeugen und Adressen für den L1 I-Cache-Speicher **14** abzuholen oder vorabzuholen. Von dem L0 I-Cache-Speicher **16** abgeholte Befehle werden von der Abhol/Sucheinheit **18** durchsucht, um Befehle für die Ausgabe zu erkennen sowie Verzweigungsbefehle zu ermitteln und Verzweigungsvorhersagen entsprechend den erkannten Verzweigungsbefehlen zu erzeugen. Befehlssuchinformationen und entsprechende Befehlsbytes werden in der Befehlswarteschlange **20** durch die Abhol/Sucheinheit **18** gespeichert. Zusätzlich werden die erkannten Verzweigungsbefehle und Verzweigungsvorhersagen verwendet, um nachfolgende Abholadressen für den L0 I-Cache-Speicher **16** zu erzeugen.

[0041] Die Abhol/Sucheinheit **18** verwendet einen Vorabholalgorithmus, um zu versuchen, Cache-Zeilen von dem L1 I-Cache-Speicher **14** in den L0 I-Cache-Speicher **16** einzuladen, bevor die vorabgeholten Cache-Zeilen von der Abhol/Sucheinheit **18** zur Ausgabe in den Prozessor **10** abgeholt werden. Es kann ein beliebiger geeigneter Vorabholalgorithmus eingesetzt werden. Eine Ausführungsform des Vorabholalgorithmus ist nachfolgend detaillierter erläutert.

[0042] Die Abhol/Sucheinheit **18** verwendet einen aggressiven Verzweigungsvorhersagemechanismus in dem Versuch, größere "Abschnitte" aus Befehlen während eines Taktzyklus abzuholen. Im hierin verwendeten Sinne bezeichnet "ein Abschnitt" aus Befehlen einen Satz aus einem oder mehreren Befehlen, von denen vorhergesagt wird, dass sie in der durch den Satz spezifizierten Reihenfolge ausgeführt werden. Beispielsweise kann die Abhol/Sucheinheit **18** Abschnitte aus 24 Befehlsbytes aus dem L0 I-Cache-Speicher **16** abholen. Jeder Abschnitt wird in mehrere Teile unterteilt, die die Abhol/Sucheinheit **18** parallel durchsucht, um Verzweigungsbefehle zu erkennen und um die Befehlssuchinformation für die Befehlswarteschlange **20** zu erzeugen. Gemäß einer Ausführungsform versucht die Abhol/Sucheinheit **18** bis zu zwei Verzweigungsbefehle pro Taktzyklus vorherzusagen, um große Befehlsabschnitte zu unterstützen.

[0043] Die Befehlswarteschlange **20** ist ausgebildet, Befehlsbytes, die von der Abhol/Sucheinheit **18** bereitgestellt werden, für das nachfolgende Ausgeben zu speichern. Die Befehlswarteschlange **20** kann als ein „zuerst herein, zuerst heraus" (FIFO) Puffer arbeiten. In einer Ausführungsform ist die Befehlswarteschlange **20** ausgebildet, mehrere Einträge zu speichern, wobei jeder Eintrag umfasst: einen Abschnitt aus Befehlen, Suchdaten, die bis zu fünf Befehle innerhalb jedes Teils des Abschnitts kennzeichnen, und Adressen entsprechend jedem Teil des Abschnitts. Ferner kann die Befehlswarteschlange **20** ausgebildet sein, bis zu 6 Befehle innerhalb bis zu vier aufeinanderfolgender Abschnitte zur Zuführung zu der Ausrichteinheit **22** auszuwählen. Die Befehlswarteschlange **20** kann beispielsweise zwei bis drei Einträge verwenden.

[0044] Die Ausrichteinheit **22** ist ausgebildet, Befehle, die von der Befehlswarteschlange **20** gesendet sind, zu einem Satz aus Ausgabepositionen innerhalb der Vorausschau/Zusammenführeinheit **24** weiterzuleiten. Anders ausgedrückt, die Ausrichteinheit **22** wählt die Bytes, die jeden Befehl bilden, aus den Abschnitten aus, die von der Befehlswarteschlange **20** bereitgestellt werden, in Reaktion auf die Suchinformation, die von der Befehlswartschlange **20** bereitgestellt wird. Die Befehle werden den Ausgabepositionen in der Programmreihenfolge zugeführt (d. h. der Befehl, der in der Programmreihenfolge zuerst erfolgt, wird der ersten Ausgabeposition zugeführt, der zweite Befehl in der Programmreihenfolge wird der zweiten Ausgabeposition zugeführt,

etc.).

[0045] Die Vorausschau/Zusammenführeinheit **24** decodiert die Befehle, die von der Ausrichteinheit **22** bereitgestellt werden. FPU-Multimedia-Befehle, die von der Vorausschau/Zusammenführeinheit **24** erkannt werden, werden der FPU-Multimediaeinheit **40** zugeführt. Andere Befehle werden dem ersten Befehlsfenster **30a**, dem zweiten Befehlsfenster **30b** und/oder der Lade/Schreibeinheit **36** zugeführt. In einer Ausführungsform wird ein spezieller Befehl dem ersten Befehlsfenster **30a** oder dem zweiten Befehlsfenster **30b** auf der Grundlage der Ausgabeposition zugeführt, die dem Befehl von der Ausrichteinheit **22** zugeordnet ist. Gemäß einer speziellen Ausführungsform werden Befehle von abwechselnden Ausgabepositionen abwechselnd den Befehlsfenster **30a** und **30b** zugeführt. Beispielsweise werden die Befehle aus den Ausgabepositionen 0, 2 und 4 dem ersten Befehlsfenster **30a** und Befehle von den Ausgabepositionen 1, 3 und 5 werden dem zweiten Befehlsfenster **30b** zugeführt. Befehle, die eine Speicheroperation beinhalten, werden ebenso zu der Lade/Schreibeinheit **36** für einen Zugriff auf den L1 D-Cache-Speicher **38** zugeführt.

[0046] Ferner versucht die Vorausschau/Zusammenführeinheit **24**, vorrausschauende Adressen oder Ausführungsergebnisse für gewisse Arten von Befehlen zu erzeugen. Die vorausschauende Adressen/Ergebniserzeugung kann insbesondere vorteilhaft sein in Ausführungsformen, in denen der x86-Befehlssatz verwendet ist. Auf Grund der Natur des x86-Befehlssatzes sind viele der Befehle in einer typischen Codierungssequenz einfache Verschiebe- bzw. Übertragungsbefehle. Ein Grund für diese Eigenschaft besteht darin, dass die x86-Befehle zwei Operanden beinhalten, wovon beide Quellenoperanden sind und wovon einer ein Zieloperand ist. Daher wird einer der Quellenoperanden jedes Befehls mit einem Ausführungsergebnis überschrieben. Ferner spezifiziert der x86-Befehlssatz sehr wenige Register zur Speicherung von Registeroperanden. Folglich sind viele Befehfe Übertragungen von Operanden zu und von einem Stapel, der in einem Speicher bewahrt wird. Des weiteren sind viele Befehlsabhängigkeiten Abhängigkeiten von ESP/EBP-Registern und somit sind viele der Aktualisierungen dieser Register Inkremente und Dekremente der zuvor gespeicherten Werte.

[0047] Um das Ausführen dieser Befehle zu beschleunigen, erzeugt die Vorausschau/Zusammenführeinheit **24** vorausschauende Kopien der ESP und EBP-Register für jeden der Befehle, die während eines Taktzyklus decodiert werden. Ferner greift die Vorausschau/Zusammenführeinheit **24** auf die Zukunftsdatei **26** für Registeroperanden, die durch den jeweiligen Befehl ausgewählt sind, zu. Für jeden Registeroperanden kann die Zukunftsdatei **26** ein Ausführungsergebnis oder eine Marke enthalten, die einen Umordnungspuffereregebniswarteschlangeneintrag kennzeichnet, der dem jüngsten Befehl entspricht, der dieses Register als einen Zieloperanden aufweist.

[0048] In einer Ausführungsform versucht die Vorausschau/Zusammenführeinheit **24**, eine Adressenberechnung für jeden Befehl auszuführen, der: (i) einen Speicheroperanden enthält; und wenn (ii) Registeroperanden, die zur Erzeugung der Adresse des Speicheroperanden verwendet werden, aus der Zukunftsdatei **26** oder vorrausschauenden Kopien von ESP/EBP verfügbar sind. Des weiteren versucht die Vorausschau/Zusammenführeinheit **24** eine Ergebnisberechnung für jeden Befehl auszuführen, der: (i) keinen Speicheroperanden enthält; (ii) eine Addier/Subtrahieroperation einschließlich Inkrement oder Dekrement angibt; und wenn (iii) ein Registeroperanden aus der Zukunftsdatei **26** oder Vorausschaukopien von ESP/EBP verfügbar sind. Auf diese Weise können viele einfache Operationen abgeschlossen werden, bevor Befehle zu den Befehlsfenster **30a** bis **30b** gesendet werden.

[0049] Die Vorausschau/Zusammenführeinheit **24** erkennt Abhängigkeiten zwischen einer Gruppe aus Befehlen, die ausgegeben wurden, und führt Ausführungsergebnisse, die dann erzeugt werden, in Befehle zusammen, die von diesen Befehlsergebnissen abhängen. Ferner aktualisiert die Vorausschau/Zusammenführeinheit **24** die Zukunftsdatei **26** mit den vorrausschauenden Ausführungsergebnissen. Befehlsoperationen, die von der Vorausschau/Zusammenführeinheit **24** abgeschlossen werden (d. h. Adressenerzeugungen und/oder Befehlsergebnisse werden erzeugt und die Lade/Schreibeinheit **36** oder die Zukunftsdatei **26** und die Ergebniswarteschlange werden aktualisiert) werden nicht an die Befehlsfenster **30a** bis **30b** ausgegeben.

[0050] Die Vorausschau/Zusammenführeinheit **24** reserviert einen Ergebniswarteschlangeneintrag in der Umordnungspuffer/Registerdatei **28** für jeden ausgegebenen Befehl. In einer speziellen Ausführungsform umfasst die Umordnungspuffer/Registerdatei **28** eine Ergebniswarteschlange, die in einer zeilenorientierten Weise organisiert ist, wobei die Speicherplätze für die Ausführungsergebnisse in Zeilen mit ausreichend Speicherplatz für Ausführungsergebnisse, die einer maximalen Zahl an gleichzeitig ausgebbaren Befehlen entspricht, reserviert und freigegeben werden. Wenn weniger als die maximale Anzahl an Befehlen ausgegeben werden, dann bleiben gewisse Speicherplätze innerhalb der Zeile leer. Nachfolgend ausgegebene Befehle verwenden die nächste verfügbare Zeile, wobei gewisse Speicherplätze leer bleiben. In einer Ausführungsform umfasst

die Ergebniswarteschlange **40** Zeilen, wovon jede bis zu 6 Ausführungsergebnisse entsprechend den gleichzeitig ausgegebenen Befehlen speichern kann. Ausführungsergebnisse werden aus der Ergebniswarteschlange der Reihenfolge nach in die Registerdatei, die innerhalb der Umordnungspuffer/Registerdatei **28** enthalten ist, zurückgeführt. Des weiteren handhabt der Umordnungspuffer Verzweigungsfehlvorhersagen und übermittelt die korrigierte Abholadresse, die von der Ausführung des Verzweigungsbefehls erzeugt wird, an die Abhol/Sucheinheit **18**. In ähnlicher Weise werden Befehle, die andere Sonder- bzw. Ausnahmeereignisse erzeugen, innerhalb des Umordnungspuffers gehandhabt. Ergebnisse, die auf den das Ausnahmeereignis erzeugenden Befehl folgen, werden von dem Umordnungspuffer verworfen. Die Registerdatei umfasst einen Speicherplatz für jedes aufgebaute Register. Beispielsweise definiert der x86-Befehlssatz 8 speziell aufgebaute Register. Die Registerdatei für eine derartige Ausführungsform umfasst 8 Speicherplätze. Die Registerdatei kann ferner Speicherplätze enthalten, die als temporäre Register durch eine Mikrocodierungseinheit in Ausführungsformen verwendet werden, in denen Mikrocodierungseinheiten eingesetzt sind.

[0051] Die Zukunftsdatei **26** bewahrt den spekulativen Zustand jedes speziell gestalteten Registers, wenn Befehle von der Vorausschau/Zusammenführeinheit **24** ausgegeben werden. Wenn ein Befehl mit einem Registerzieloperanden von der Vorausschau/Zusammenführeinheit **24** decodiert wird, wird die Markierung, die den Speicherplatz innerhalb des Ergebniswarteschlangenbereichs der Umordnungspuffer/Registerdatei **28**, der dem Befehl zugeordnet ist, in dem Speicherplatz der Zukunftsdatei **26** abgelegt, der diesem Register entspricht. Wenn das entsprechende Ausführungsergebnis bereitgestellt wird, wird das Ausführungsergebnis in dem entsprechenden Speicherplatz abgelegt (wobei angenommen wird, dass ein nachfolgender Befehl, der das Register aktualisiert, nicht ausgegeben ist).

[0052] Es ist zu beachten, dass in einer Ausführungsform eine Gruppe bis zu 6 Befehlen aus der Befehlswarteschlange **20** ausgewählt wird und durch die Pipeline innerhalb der Vorausschau/Zusammenführeinheit **24** als eine Einheit wandert. Wenn einer oder mehrere Befehle innerhalb der Gruppe eine Abbruchbedingung erzeugt, wird die ganze Gruppe beendet. Eine Ausnahme dieser Regel besteht darin, wenn die Vorausschau/Zusammenführeinheit **24** eine Bedingung zur Aufteilung der Reihe auf Grund der Anzahl von ESP-Aktualisierungen innerhalb der Gruppe erzeugt. Eine derartige Gruppe aus Befehlen wird als eine „Reihe oder Linie" aus Befehlen hierin bezeichnet.

[0053] Die Befehlsfenster **30** empfangen Befehle von der Vorausschau/Zusammenführeinheit **24**. Die Befehlsfenster **30** speichern die Befehle bis die Operanden, die den Befehlen entsprechen, empfangen werden, und wählen dann die Befehle für die Ausführung aus. Sobald die Adressenoperanden eines Befehls, der eine Speicheroperation enthält, empfangen sind, wird der Befehl zu einer der Adressenerzeugungseinheiten **34** übermittelt. Die Adressenerzeugungseinheiten **34** erzeugen eine Adresse aus den Adressenoperanden und leiten die Adresse zu der Lade/Schreibeinheit **36** weiter. Wenn andererseits die Ausführungsoperanden eines Befehls empfangen werden, wird der Befehl zu einer der Funktionseinheiten **32** für die Ausführung weitergeleitet. In einer Ausführungsform enthält jedes Befehlsfenster **30a** bis **30b** 25 Speicherplätze für Befehle. Jedes Befehlsfenster **30a** bis **30b** ist ausgebildet, bis zu zwei Adressenerzeugungen und zwei Funktionseinheitenoperationen für die Ausführung in jedem Taktzyklus in den Adressenerzeugungseinheiten **34** und in den Funktionseinheiten **32**, die damit verbunden sind, auszuwählen. In einer Ausführungsform bleiben die Befehle, die aus dem L0 I-Cache-Speicher **16** abgeholt werden, in der gleichen Reihenfolge, wie sie abgeholt werden, bis sie in einem der Befehlsfenster **30** gespeichert werden, und ab diesem Zeitpunkt können die Befehle außer der Reihenfolge ausgeführt werden.

[0054] In einigen Ausführungsformen des Prozesses **10**, wobei der x86-Befehlssatz eingesetzt ist, kann ein Befehl implizierte Speicheroperationen für die Lade/Schreibeinheit **36** sowie explizite Funktionsoperationen für die Funktionseinheiten **32** enthalten. Befehle mit keinem Speicheroperanden beinhalten keine Speicheroperationen und werden von den Funktionseinheiten **32** behandelt. Befehle mit einem Quellenspeicheroperanden und einem Registerzieloperanden enthalten eine implizite Ladespeicheroperation, die von der Lade/Schreibeeinheit **36** behandelt wird, und eine explizite Funktionsoperation, die von den Funktionseinheiten **32** behandelt wird. Befehle mit einem Speicherquellen/Zieloperanden enthalten implizite Lade- und Schreibspeicheroperationen, die von der Lade/Schreibeinheit **36** behandelt werden, und besitzen explizite Funktionsoperationen, die von den Funktionseinheiten **32** behandelt werden. Schließlich werden Befehle, die keine explizite Funktionsoperation aufweisen, von der Lade/Schreibeinheit **36** behandelt. Jede Speicheroperation führt zu einer Adressenerzeugung, die von der Vorausschau/Zusammenführeinheit **24** oder den Adressenerzeugungseinheiten **34** behandelt wird. Speicheroperationen und Befehle (d. h. Funktionsoperationen) werden hierin separat beschrieben, können jedoch von einem einzelnen Befehl stammen.

[0055] Die Adressenerzeugungseinheiten **34** sind ausgebildet, Adressenerzeugungsoperationen auszufüh-

ren, wodurch Adressen für Speicheroperationen in der Lade/Schreibeinheit **36** erzeugt werden. Die erzeugten Adressen werden zu der Lade/Schreibeinheit **36** über Ergebnisbusse **48** weitergeleitet. Die Funktionseinheiten **32** sind ausgebildet, arithmetische/logische Ganzzahloperationen und Verzweigungsbefehle auszuführen. Die Ausführungsergebnisse werden zu der Zukunftsdatei **26**, der Umordnungspuffer/Registerdatei **28** und dem Befehlsfenster **30a** bis **30b** über die Ergebnisbusse **48** weitergeleitet. Die Adressenerzeugungseinheiten **34** und die Funktionseinheiten **32** übermitteln die Ergebniswarteschlangenmarke, die dem Befehl zugeordnet ist, der ausgeführt wird, wenn die Ergebnisbusse **48** den gerade ausgeführten Befehl kennzeichnen. Auf diese Weise können die Zukunftsdatei **26**, die Umordnungspuffer/Registerdatei **28**, die Befehlsfenster **30a** bis **30b** und die Lade/Schreibeinheit **36** Ausführungsergebnisse des entsprechenden Befehls erkennen. Die FPU-Multimediaeinheit **40** ist ausgebildet, Flieskomma- und Multimediabefehle auszuführen.

[0056] Die Lade/Schreibeinheit **36** ist ausgebildet, mit dem L1 D-Cache-Speicher **38** zu kommunizieren, um Speicheroperationen auszuführen. Eine Speicheroperation ist ein Datentransfer zwischen dem Prozessor **10** und einem externen Speicher. Die Speicheroperation kann ein expliziter Befehl sein oder kann ein impliziter Bereich eines Befehls sein, der auch Operationen enthält, die von den Funktionseinheiten **32** auszuführen sind. Die Ladespeicheroperationen geben einen Datentransfer von einem externen Speicher zu dem Prozessor **10** an, und Schreibspeicheroperationen spezifizieren einen Datentransfer von dem Prozessor **10** zu dem externen Speicher. Wenn ein Treffer für eine Speicheroperation in den L1 D-Cache-Speicher **38** erkannt wird, wird die Speicheroperation dann abgeschlossen, ohne dass auf den externen Speicher zugegriffen wird. Die Lade/Schreibeinheit **36** empfängt Adressen für Speicheroperationen von der Vorausschau/Zusammenführeinheit **24** (über eine vorrausschauende Adressenberechnung) oder von den Adressenerzeugungseinheiten **34**. In einer Ausführungsform ist die Lade/Schreibeinheit **36** ausgebildet, bis zu drei Speicheroperationen pro Taktzyklus an dem L1 D-Cache-Speicher **38** auszuführen. Für diese Ausführungsform ist die Lade/Schreibeinheit **36** ausgebildet, bis zu 30 Lade/Schreibspeicheroperationen zu speichern, die noch nicht auf dem D-Cache-Speicher **38** zugegriffen haben. Die Ausführungsform kann ferner ausgebildet sein, um einen 96 Einträgenichttrefferpuffer zur Speicherung von Ladespeicheroperationen zu speichern, die nicht den D-CacheSpeicher **38** treffen, und kann ferner einen 32 Einträgeschreibdatenpuffer aufweisen. Die Lade/Schreibeinheit **36** ist ausgebildet, eine Speicherabhängigkeitsprüfung zwischen den Lade- und Schreibspeicheroperationen auszuführen.

[0057] Der L1 D-Cache-Speicher **38** ist ein Hochgeschwindigkeitscachespeicher zur Datenspeicherung. Es kann eine beliebige Konfiguration für den L1 D-Cache-Speicher **38** verwendet werden, einschließlich einer teilassoziativen Konfiguration und einer direkt abgebildeten Konfiguration. In einer speziellen Ausführungsform ist der L1 D-Cache-Speicher **38** ein 128 KB Zweiwege-teilassoziativer Cache-Speicher mit 64 Bytezeilen. Der L1 D-Cache-Speicher **38** kann beispielsweise in 32 Bänken aus Cache-Speichern pro Weg eingeteilt sein. Zusätzlich kann der L1 D-Cache-Speicher **38** ein linear adressierter, physikalisch markierter Cache-Speicher sein, in welchem ein TLB ähnlich zu dem L1 I-Cache-Speicher **14** verwendet ist.

[0058] Die externe Schnittstelleneinheit **42** ist ausgebildet, Cache-Zeilen aus Befehlsbytes und Datenbytes in den Prozessor **10** in Reaktion auf Nichttreffer im Cache zu übertragen. Befehlscache-Zeilen werden zu der Vordecodierungseinheit **12** geleitet, und Datencache-Zeilen werden zu dem L1 D-Cache-Speicher **38** geführt. Ferner ist die externe Schnittstelleneinheit **42** ausgebildet, Cache-Zeilen, die von dem L1 D-Cache-Speicher **38** in dem Speicher abgelegt werden, wenn die abgegebenen Cache-Zeilen modifiziert werden, an den Prozessor **10** zu übertragen. Wein **Fig. 1** gezeigt ist, ist die externe Schnittstelleneinheit **42** ausgebildet, eine Verbindung mit einem externen L2 Cache-Speicher über die L2-Schnittstelle **44** sowie als Schnittstelle mit einem Computersystem über die Busschnittstelle **46** zu dienen. In einer Ausführungsform umfasst die Busschnittstelleneinheit **46** eine EV/6-Schnittstelle.

[0059] **Fig. 2** ist eine Blockansicht einer Ausführungsform einer Abhol/Sucheinheit **18**. Andere Ausführungsformen sind möglich und hierin mit eingeschlossen. Wie in **Fig. 2** gezeigt ist, umfasst die Abhol/Sucheinheit **18** eine Abholsteuereinheit **50**, mehrere "nächste Wahl"-Blöcke **52a** bis **52c**, einen Befehlsauswahlmultiplexer (mux) **54**, einen Befehlsabtaster bzw. Sucher **56**, einen Verzweigungsabtaster bzw. Sucher **58**, eine Verzweigungshistorientabelle **60**, einen Verzweigungsauswahlmux **62**, einen Rücksprungstapel **64**, einen indirekten Adressencache-Speicher **66** und eine Vorwärtszusammenführeinheit **68**. Die Abholsteuereinheit **50** ist mit dem L1 I-Cache-Speicher **14**, dem L0 I-Cache-Speicher **16**, dem indirekten Adressencachespeicher **66**, den Rücksprungstapel **64**, der Verzweigungshistorientabelle **60**, dem Verzweigungssucher **58** und dem Befehlsauswahlmux **54** verbunden. Der „nächste Wahl"-Block **52a** ist mit dem L1 I-Cache-Speicher **14** verbunden, während die „nächste Wahl"-Blöcke **52b** bis **52c** mit dem L0 I-Cache-Speicher **16** verbunden sind. Jeder „nächste Wahl"-Block **52** ist mit einem Befehlsauswahlmux **54** verbunden, der ferner mit dem Verzweigungssucher **58** und dem Befehlssucher **56** verbunden ist. Der Befehlssucher **56** ist mit der Befehlswarteschlange **20** verbun-

den. Der Verzweigungssucher **58** ist mit der Verzweigungshistorientabelle **60**, dem Rückkehrstapel **64** und dem Verzweigungsauswahlmux **68** verbunden. Der Verzweigungsauswahlmux **62** ist mit dem indirekten Adressencache-Speicher **66** verbunden. Die Verzweigungshistorientabelle **60** und der Verzweigungssucher **58** sind mit der Weiterleitungszusammenführeinheit **68** verbunden, die mit der Befehlswartschlange **20** verbunden ist.

[0060] Die Abholsteuereinheit **50** empfängt eine Verzweigungsvorhersageinformation (einschließlich der Zieladressen und der Vorhersage über durchgeführt/nicht durchgeführt) von dem Verzweigungssucher **58**, der Verzweigungshistorientabelle **60**, dem Rücksprungstapel **64** und dem indirekten Cache-Speicher **66**. In Reaktion auf die Verzweigungsvorhersageinformation erzeugt die Abholsteuereinheit **50** Abholadressen für dien L1 I-Cache-Speicher **16** und eine Abhol- oder Vorabholadresse für den L1 I-Cache-Speicher **14**. In einer Ausführungsform erzeugt die Abholsteuereinheit **50** zwei Abholadressen für den L0 I-Cache-Speicher **16**. Die erste Abholadresse wird als die Zieladresse entsprechend dem ersten Verzweigungsbefehl, der von dem Verzweigungssucher **58** erkannt wird (wenn es welche gibt), ausgewählt. Die zweite Abholadresse ist die sequenzielle Adresse zu der Abholadresse, die in dem vorhergehenden Taktzyklus ausgewählt wurde, d. h. die Abholadresse, die dem Abschnitt entspricht, der von dem Befehlsauswahlmux **54** ausgewählt wurde.

[0061] Der L0 I-Cache-Speicher **14** liefert die Cache-Zeilen (und die Vordecodierungsinformation), die den beiden Abholadressen entspricht, sowie die Cache-Zeilen (und die Vordecodierungsinformation), die sequenziell zu jeder dieser Cache-Zeilen sind, um die „nächste Wahl"-Blöcke **52b** bis **52c** auszuwählen. Insbesondere empfängt der „nächste Wahl"-Block **52b** die sequenzielle Cache-Zeile, die der sequenziellen Adresse entspricht, und die nächste inkrementierte Cache-Zeile zu der sequenziellen Cache-Zeile. Der „nächste Wahl"-Block **52c** empfängt die Zielcache-Zeile entsprechend der Zieladresse sowie die Cache-Zeile, die der Zielcachezeile folgt. Zusätzlich empfangen die „nächste Wahl"-Blöcke **52b** bis **52c** den Offsetbereich der entsprechenden Abholadresse. Die „nächste Wahl"-Blöcke **52b** bis **52c** wählen jeweils einen Abschnitt aus Befehlsbytes (und entsprechende Vordecodierungsinformation) aus dem empfangenen Cache-Zeilen aus, wobei mit den Abschnittsteil begonnen wird, der den Offsetbereich der entsprechenden Abholadresse beinhaltet. Da der Offsetbereich jeder Abholadresse irgendwo innerhalb der Cache-Zeile beginnen kann, kann der ausgewählte Abschnitt Teile der abgeholten Cache-Zeile und der sequenziellen Cache-Zeile in Bezug auf die abgeholte Cache-Zeile enthalten. Somit werden sowohl die abgeholte Cache-Zeile als auch die nachfolgende Cache-Zeile von den „nächste Wahl"-Blöcken **52b** bis **52c** empfangen.

[0062] In ähnlicher Weise erhält der „nächste Wahl"-Block **52a** eine vorabgeholte Cache-Zeile (und entsprechende Vordecodierungsinformation) aus dem L1 I-Cache-Speicher **14** und wählt daraus einen Befehlsabschnitt aus. Da eine einzelne Cache-Zeile von dem L1 I-Cache-Speicher **14** vorabgeholt wird, kann der daraus ausgewählte Abschnitt mehr als einen vollständigen Abschnitt enthalten, wenn der Offsetbereich der Vorabholungsadresse in der Nähe des Endes der Cache-Zeile liegt. Zu beachten ist, dass die Abholcache-Zeilen aus dem L0 I-Cache-Speicher **16** in dem gleichen Taktzyklus mit der entsprechenden Adresse von der Abholsteuereinheit **50** erzeugt werden, jedoch kann die Vorabholungscachezeile um einen Taktzyklus auf Grund der größeren Größe und der geringeren Zugriffszeit des L1 I-Cache-Speichers **14** verzögert werden. Zusätzlich zu dem Bereistellen der vorabgeholten Cache-Zeile für den „nächste Wahl"-Block **52a** liefert der L1 I-Cache-Speicher **14** die vorabgeholte Cache-Zeile an den L0 I-Cache-Speicher **16**. Wenn die vorabgeholte Cache-Zeile bereits in dem L0 I-Cache-Speicher **16** gespeichert ist, kann der L0 I-Cache-Speicher **16** die vorabgeholte Cache-Zeile verwerfen. Wenn jedoch die vorabgeholte Cache-Zeile noch nicht in dem L0 I-Cache-Speicher **14** abgelegt ist, wird die vorabgeholte Cache-Zeile in dem L0 I-Cache-Speicher **16** gespeichert. Auf diese Weise werden Cache-Zeilen, auf die aktuell zugegriffen wird, in den L0 I-Cache-Speicher **16** gebracht, um darin einen schnellen Zugriff zu ermöglichen. Gemäß einer beispielhaften Ausführungsform umfasst der L0 I-Cache-Speicher **16** eine vollständig assoziative Cache-Struktur mit 8 Einträgen. Ein vollständig assoziative Struktur kann auf Grund der relativ geringen Anzahl an Cache-Zeilen, die in dem L0 I-Cache-Speicher **16** enthalten sind, eingesetzt werden. In anderen Ausführungsformen werden andere Organisationsformen (beispielsweise teilassoziativ oder direkt abgebildet) eingesetzt.

[0063] Die Abholsteuereinheit **50** wählt den Befehlsabschnitt, der von einem der „nächste Wahl"-Blöcke **52** ausgewählt wird, in Reaktion auf die Verzweigungsvorhersageinformation aus, indem der Befehlsauswahlmux **54** angesteuert wird. Wie nachfolgend detaillierter beschrieben ist, empfängt die Abholsteuereinheit **50** (in der vorliegenden Ausführungsform) Zieladressen von dem Verzweigungssucher **58**, dem Rücksprungstapel **64** und dem indirekten Adressencache-Speicher **66** früh in dem Taktzyklus und zumindest einen Teil des Operationscodierungsbytes des ersten Verzweigungsbefehls, der von dem Verzweigungssucher **58** erkannt wird. Die Abholsteuereinheit **50** decodiert den Bereich des Operationscodierungsbytes, um die Zieladresse, die aus dem L0 I-Cache-Speicher **16** abzuholen ist, von den diversen Zieladressquellen auszuwählen, und liefert die ausgewählte Zieladresse zu dem L0 I-Cache-Speicher **16**. Parallel dazu wird die sequenzielle Adresse für die Ab-

holadresse, die in dem vorhergehenden Taktzyklus ausgewählt wurde (entweder die Zieladresse oder die nachfolgende Adresse aus dem vorhergehenden Taktzyklus, abhängig von der Verzweigungsvorhersage aus dem vorhergehenden Taktzyklus) berechnet und dem L0 I-Cache-Speicher **16** zugeführt. Die Verzweigungsvorhersageinformation (d. h. ausgeführt oder nicht ausgeführt) wird von der Verzweigungshistorientabelle **60** spät in dem Taktzyklus bereitgestellt. Wenn der Verzweigungsbefehl, der der aus dem L0 I-Cache-Speicher **16** abgeholten Zieladresse entspricht, als durchgeführt vorhergesagt wird, dann wählt die Abholsteuereinheit **50** den Befehlsabschnitt aus, der von dem „nächste Wahl"-Block **52c** bereitgestellt wird. Wenn andererseits der Verzweigungsbefehl als nicht durchgeführt vorhergesagt wird, dann wird der Befehlsabschnitt, der von dem „nächste Wahl"-Block **52b** ausgewählt wurde, ausgewählt. Der Befehlsabschnitt, der von dem „nächste Wahl"-Block **52a** bereitgestellt wird, wird ausgewählt, wenn eine vorhergesagte Abholadresse nicht den L0 I-Cache-Speicher **16** in einem vorhergehenden Taktzyklus betraf und aus dem L1 I-Cache-Speicher **14** abgeholt wurde. Ferner wird der Befehlsabschnitt aus dem L1 I-Cache-Speicher **14** ausgewählt, wenn der Befehl in Reaktion auf einen Verzweigungsbefehl mit einer 32 Bit-Verschiebung oder einer indirekten Zieladressenerzeugung ausgeführt wurde oder ein L0 I-Cache-Speichernichttreffer abgeholt wurde.

[0064] Der ausgewählte Befehlsabschnitt wird dem Befehlssucher **56** und dem Verzweigungssucher **58** zugeführt. Der Befehlssucher **56** sondiert die Vordecodierungsinformation entsprechend den ausgewählten Befehlsabschnitt, um Befehle innerhalb des Befehlsabschnitts zu erkennen. Insbesondere sondiert in einer Ausführungsform der Befehlssucher **56** die Startbits, die jedem Abschnittsteil entsprechen, parallel ab und erkennt bis zu 5 Befehle innerhalb jedes Teilabschnitts. Es werden Zeiger zu den erkannten Befehlen (Offsetwerte innerhalb des Teilabschnitts) erzeugt. Die Zeiger, Befehlsbytes und Adressen (eine pro Teilabschnitt) werden von dem Befehlssucher **56** zu der Befehlswarteschlange **20** übermittelt. Wenn ein spezieller Teilabschnitt mehr als 5 Befehle enthält, wird die Information, die den Teilabschnitten entspricht, die auf den speziellen Teilabschnitt folgt, als unzulässig erklärt, und der spezielle Teilabschnitt und die nachfolgenden Teilabschnitte werden während des nächsten Taktzykluss erneut abgesucht.

[0065] Der Verzweigungssucher **58** sondiert den Befehlsabschnitt parallel mit dem Befehlssucher **56**. Der Verzweigunggssucher **58** sucht die Startbits und die Steuertransferbits des Befehlsabschnitts, um die ersten beiden Verzweigungsbefehle innerhalb des Befehlsabschnitts zu erkennen. Wie zuvor beschrieben ist, wird ein Verzweigungsbefehl durch das gesetzte Steuertransferbit, das dem Startbyte eines Befehls entspricht (wie dies durch das Startbit gekennzeichnet ist) erkannt. Beim Auffinden des ersten der beiden Verzweigungsbefehle nimmt der Verzweigungsasucher **58** an, dass die Befehle relative Verzweigungsbefehle sind und wählt die entsprechenden codierten Zieladressen aus den Befehlsbytes aus, die auf das Startbyte des Verzweigungsbefehls folgen. Bei Ausführungsformen, in denen der x86-Befehlssatz verwendet ist, wird eine 9-Bit-Zieladresse (das Verschiebungsbyte sowie das entsprechende Steuertransferbit) ausgewählt, und es wird auch eine 32-Bit-Zieladresse ausgewählt. Ferner wird mindestens ein Teil des Operationscodierungsbytes, das von den Start- und Steuertransferbits gekennzeichnet ist, ausgewählt. Die Zieladressen und die Operationscodierungsbytes werden der Abholsteuereinheit **50** zugeführt, um beim Auswählen einer Zieladresse zum Abholen aus dem L0 I-Cache-Speicher **16** zu dienen. Die Abholadressen jedes Verzweigungsbefehls (die aus der Abholadresse des Teilabschnitts, der jeden Verzweigungsbefehl enthält und aus der Position des Verzweigungsbefehls innerhalb des Teilabschnitts bestimmt) werden zu der Verzweigungshistorientabelle **60** geleitet, um eine „durchgeführt/nicht durchgeführt" Vorhersage entsprechend jedem Verzweigungsbefehl auszuwählen. Ferner werden die Abholadressen, die jedem Verzweigungsbefehl entsprechen, zu dem Verzweigungsauswahlmux **62** geführt, und dann weiter zu dem indirekten Adressencache-Speicher **66** geführt. Die Zieladresse jedes Verzweigungsbefehls wird zu der Weiterleitungszusammenführeinheit **68** geführt. Gemäß einer Ausführungsform ist der Verzweigungssucher **58** ausgebildet, jeden Teilabschnitt parallel nach den ersten beiden Verzweigungsbefehlen abzusuchen und dann das Suchergebnis zu kombinieren, um die ersten beiden Verzweigungsbefehle innerhalb des Abschnitts auszuwählen.

[0066] Der Verzweigungssucher **58** kann ferner ausgebildet sein, zu bestimmen, ob ein Unterroutinenaufrufbefehl während eines Taktzyklus erkannt wird. Der Verzweigungssucher **58** kann die Abholadresse des nächsten Befehls, der auf den erkannten Unterroutinenaufrufbefehl folgt, zum Rücksprungstapel **64** zur Speicherung weiterleiten.

[0067] Wenn in einer Ausführungsform mehr als zwei Verzweigungsbefehle innerhalb eines Abschnitts vorhanden sind, wird der Abschnitt während eines nachfolgenden Taktzyklus erneut abgesucht, um den nachfolgenden Verzweigungsbefehl zu ermitteln.

[0068] Die Abholadressen der erkannten Verzweigungsbefehle werden der Verzweigungshistorientabelle **60** zugeführt, um eine „durchgeführt/nicht durchgeführt" Vorhersage für jeden Befehl zu bestimmen. Die Verzwei-

gungshistorientabelle **60** umfasst mehrere „durchgeführt/nicht durchgeführt" Vorhersageelemente, die dem zuvor erkannten Verhalten der Verzweigungsbefehle entsprechen. Eines der Vorhersageelemente wird durch Beibehalten einer Historie der jüngsten Vorhersagen ausgewählt und diese jüngsten Vorhersagen werden mit einem Teil der Abholadressen, die dem Verzweigungsbefehlen entsprechen, durch eine exklusiv-Oder-Funktion verknüpft. Die am wenigsten aktuelle (älteste) Vorhersage wird mit dem signifikantesten Bit innerhalb des Bereichs der Abholadresse durch die exklusiv-Oder-Funktion verknüpft, usw., bis zu der aktuellsten Vorhersage, die mit dem wenigsten signifikanten Bit innerhalb des Bereichs der Abholadresse mittels der exklusiv-Oder-Funktion verknüpft wird. Da zwei Vorhersageelemente pro Taktzyklus ausgewählt werden, hängt das Vorhersageelement, das dem zweiten Verzweigungsbefehl entspricht, von der Vorhersage des ersten Verzweigungsbefehls ab (für die exklusiv-Oder-Verknüpfung mit dem am wenigsten signifikanten Bit der entsprechenden Abholadresse). Die Verzweigunghistorientabelle **60** liefert das zweite Verzweigungselement durch Auswählen sowohl der Verzweigungselemente, die ausgewählt werden könnten (d. h. das Vorhersageelement, das ausgewählt würde, wenn der erste Verzweigungsbefehl als nicht genommen vorhergesagt wird und das Verzweigungselement, das ausgewählt würde, wenn der erste Verzweigungsbefehl als durchgeführt vorhergesagt wird), und durch Auswählen eines der beiden Vorhersageelemente auf der Grundlage der tatsächlichen Vorhersage, die für den ersten Verzweigungsbefehl ausgewählt wurde.

[0069] Die Verzweigungshistorientabelle **60** empfängt die Information hinsichtlich des Ausführens der Verzweigungsbefehle von den Funktionseinheiten **32a** bis **32d**. Die Historie der jüngsten Vorhersagen, die dem ausgeführten Verzweigungsbefehl entsprechen, sowie die Abholadresse des ausgeführten Verzweigungsbefehls werden bereitgestellt, um ein Verzweigungselement für die Aktualisierung auszuwählen, und es wird das „durchgeführt/nicht durchgeführt" Ergebnis des ausgeführten Verzweigungsbefehls bereitgestellt. Die Verzweigungshistorientabelle **60** wählt das entsprechende Vorhersageelement aus und aktualisiert das Vorhersageelement auf der Grundlage des „durchgeführt/nicht durchgeführt" Ergebnisses. In einer Ausführungsform speichert die Verzweigungshistorientabelle einen bimodalen Zähler. Der bimodale Zähler ist ein Sättigungszähler, der bei einem minimalen und maximalen Wert die Sättigungsgrenze erreicht (d. h. nachfolgende Dekrementierung des minimalen Wertes oder Erhöhungen des maximalen Wertes verursachen keine Änderung in dem Zähler). Jedes mal, wenn ein Verzweigungsbefehl genommen wird, wird der entsprechende Zähler erhöht und jedes mal wenn ein Verzweigungsbefehl nicht genommen wird, wird der entsprechende Zähler erniedrigt. Das signifikanteste Bit des Zählers gibt die „durchgeführt/nicht durchgeführt" Vorhersage an (beispielsweise ist es gesetzt, wenn die Vorhersage durchgeführt wurde, oder nicht gesetzt, wenn dieser nicht durchgeführt wurde). In einer Ausführungsform speichert die Verzweigungshistorientabelle **60** 64K an Vorhersageelementen und bewahrt eine Geschichte der 16 jüngsten Vorhersagen. Bei jedem Taktzyklus werden die während des Taktzyklus ausgewählten Vorhersagen in die Geschichtstabelle geschoben und die ältesten Vorhersagen werden aus der Geschichtstabelle herausgeschoben.

[0070] Der Rücksprungstapel **64** wird verwendet, um die Rücksprungadressen entsprechend den erkannten Unterroutinenaufrufbefehlen zu speichern. Der Rücksprungstapel **64** empfängt die Abholadresse eines Unterroutinenaufrufbefehls von dem Verzweigungssucher **58**. Die Adresse des Bytes, das auf dem Aufrufbefehl folgt (die aus der Abholadresse berechnet wird, die von dem Rücksprungstapel **64** geliefert wird) wird oben auf dem Rücksprungstapel **64** angeordnet. Der Rücksprungstapel **64** liefert die Adresse, die oben auf dem Rücksprungstapel abgelegt ist, an die Abholsteuereinheit **50** zur Auswahl als eine Zieladresse, wenn ein Rücksprungbefehl von dem Verzweigungssucher **58** und der Abholsteuereinheit **50** erkannt wird. Auf diese Weise erhält jeder Rücksprungbefehl die Adresse als Zieladresse, die dem jüngsten erkannten Aufrufbefehl entspricht. Im Allgemeinen ist in dem x86-Befehlssatz ein Aufrufbefehl ein Steuertransferbefehl, der angibt, dass die sequenzielle Adresse nach dem Aufrufbefehl auf dem Stapel anzuordnen ist, der durch die x86-Architektur definiert ist. Ein Rücksprungbefehl ist ein Befehl, der die Zieladresse an dem Stapel auswählt. Im Allgemeinen werden Aufruf- und Rücksprungbefehle verwendet, um innerhalb einer Codierungssequenz in eine Unterroutine zu springen und diese zu verlassen. Durch Anordnen der Adressen, die den Aufrufbefehlen entsprechen, in dem Rücksprungstapel **64** und durch Verwenden der Adresse an der Oberseite des Rücksprungstapels **64** als die Zieladresse von Rücksprungbefehlen kann die Zieladresse des Rücksprungbefehls in korrekter Weise vorhergesagt werden. In einer Ausführungsform umfasst der Rücksprungstapel **64** 16 Einträge.

[0071] Der indirekte Adressencachespeicher **66** speichert die Zieladressen, die den vorhergehenden Ausführungen indirekter Verzweigungsbefehle entsprechen. Die Abholadresse, die einem indirekten Verzweigungsbefehl entspricht, und die Zieladresse, die dem Ausführen des indirekten Verzweigungsbefehls entspricht, werden von den Funktionseinheiten **32a** bis **32d** an dem indirekten Adressencache-Speicher **66** geliefert. Der indirekte Adressencache-Speicher **66** speichert die Zieladressen, die von den entsprechenden Abholadressen angegeben werden. Der indirekte Adressencache-Speicher **66** empfängt die Abholadresse, die von dem Verzweigungsauswahlmux **62** ausgewählt wird (in Reaktion auf das Erkennen eines indirekten Verzweigungsbe-

fehls), und wenn die Abholadresse ein Treffer in dem indirekten Adressencache-Speicher **66** ist, liefert dieser die entsprechende Zieladresse an die Abholsteuereinheit **50**. In einer Ausführungsform umfasst der indirekte Adressencache-Speicher **66** 32 Einträge.

[0072] Gemäß einer Ausführungsform kann, wenn der indirekte Adressencache-Speicher **66** einen Nichttreffer für eine Abholadresse erkennt, dieser ausgebildet sein, eine Zieladresse auszuwählen, um einen der Einträge bereitzustellen. Auf diese Weise wird ein „Vorschlag" für ein Verzweigungsziel in dem Falle bereitgestellt, dass ein indirekter Verzweigungsbefehl decodiert wird. Das Abholen gemäß dem Vorschlag wird ausgeführt, anstatt auf die Adresse über das Ausführen des indirekten Verzweigungsbefehls zu warten. Alternativ wird in einer weiteren Ausführungsform auf die Adresse gewartet, die mittels des Ausführens des indirekten Verzweigungsbefehls bereitgestellt wird.

[0073] Es ist zu beachten, dass wenn eine codierte Zieladresse ausgewählt wird, die tatsächliche Zieladresse den L0 I-Cache-Speicher **16** zugeführt werden kann. Die Abholsteuereinheit **50** kann ausgebildet sein, jede der möglichen darüber/darunter liegenden Zieladressen im Voraus zu berechnen und die korrekte Adresse auf der Grundlage der codierten Zieladresse auszuwählen. Alternativ kann die Abholsteuereinheit **50** aufzeichnen, welche L0 I-Cache-Speicherplätze die darüber liegenden und darunter liegenden Cache-Zeilen beinhalten, und kann die Speicherplätze direkt ohne einen Markierungsvergleich auswählen.

[0074] Die Weiterleitungs/Zusammenführeinheit **68** empfängt die Zieladressen und die Positionen innerhalb des Befehlsabschnitts jedes ausgewählten Verzweigungsbefehls sowie die „durchgeführt/nicht durchgeführt" -Vorhersagen. Die Weiterleitungszusammenführeinheit **68** bestimmt, welche Befehle innerhalb des Abschnitts gelöscht werden sollten auf der Grundlage der empfangenen Vorhersagen. Wenn der erste Verzweigungsbefehl als durchgeführt vorhergesagt wird und nach hinten gerichtet ist (d. h. die Verschiebung ist negativ), werden alle auf dem ersten Verzweigungsbefehl folgenden Befehle gelöscht. Wenn der erste Verzweigungsbefehl als durchgeführt vorhergesagt wird und in Vorwärtsrichtung stattfindet, aber die Verschiebung klein ist (beispielsweise innerhalb des Abschnitts aus Befehlen), werden die Befehle, zwischen dem ersten Verzweigungsbefehl und der Zieladresse gelöscht. Der zweite Verzweigungsbefehl, wenn dieser weiterhin innerhalb des Abschnitts gemäß der Vorhersage des ersten Verzweigungsbefehls ist, wird in ähnlicher Weise behandelt. Löschangaben für die Befehle innerhalb des Abschnitts werden der Befehlswarteschlange **20** zugeführt.

[0075] Fig. 3 ist eine Blockansicht einer Ausführungsform einer Vorauschau/Zusammenführeinheit **24**. Andere Ausführungsformen sind möglich und hierin mit berücksichtigt. Wie in Fig. 3 gezeigt ist, umfasst die Vorausschau/Zusammenführeinheit **24** mehrere Decodiereinheiten **70a** bis **70f**, eine ESP/EBP-Vorausschaueinheit **72**, eine Erzeugungseinheit **73** für ein unmittelbares Feld, eine reiheninterne Abhängigkeitsprüfeinheit **75**, eine Vorausschauadressen/Ergebnisberechnungseinheit **74**, eine Ausgabesteuereinheit **76** und eine Operandenzusammenführeinheit **78**. Die Decodiereinheiten **70a** bis **70f** sind angeschlossen, um Befehle aus der Ausrichteinheit **22** zu empfangen. Die Decodiereinheiten **70a** bis **70f** sind angeschlossen, um decodierte Befehle und/oder Befehlsinformation an die FPU-Multimediaeinheit **40**, die ESP/EBP-Vorausschaueinheit **73**, die Erzeugungseinheit für das unmittelbare Feld **73**, die reiheninterne Abhängigkeitsprüfeinheit **75**, die Zukunftsdatei **26** und die Vorausschauadressen/Ergebnisberechnungseinheit **74** zu liefern. Die ESP/EBP-Vorausschaueinheit **72** ist mit der Vorausschauadressen/Ergebnisberechnungseinheit **74** verbunden, ebenso wie die Zukunftsdatei **26**, die Erzeugungseinheit für das unmittelbare Feld **73**, und die reiheninterne Abhängigkeitsprüfeinheit **75**. Die Vorausschauadressen/Ergebnisberechnungseinheit **74** ist ferner mit der Lade/Schreibeinheit **36** und der Ausgabesteuereinheit **76** verbunden. Die Ausgabeeinheit **76** ist ferner mit der Operandenzusammenführeinheit **78**, der Zukunftsdatei **26**, der Lade/Schreibeinheit **36** und dem Umordnungspuffer **28** verbunden. Die Operandenzusammenführeinheit **78** ist mit den Befehlsfenstern **30** verbunden.

[0076] Jede Decodierungseinheit **70a** bis **70f** bildet eine Ausgabeposition, an der die Ausrichteinheit **22** einen Befehl geeignet bereit stellt. Obwohl dies in der Fig. 3 der Einfachheit halber nicht explizit gezeigt ist, bleibt ein spezieller Befehl innerhalb seiner Ausgabeposition, wenn der Befehl sich durch die Vorausschau/Zusammenführeinheit **24** bewegt und wird zu einem der Befehlsfenster **30a** bis **30b** geleitet, wenn er nicht innerhalb der Vorausschau/Zusammenführeinheit **24** abgeschlossen wird.

[0077] Die Decodiereinheiten **70a** bis **70f** leiten FPU-Multimediabefehle zu der FPU-Multimediaeinheit **40**. Wenn jedoch die FPU-Multimediabefehle Speicheroperanden enthalten, werden Speicheroperationen ebenso an die Lade/Schreibeinheit **36** in Reaktion auf den Befehl durch die Vorausschauadressen/Ergebnisberechnungseinheit **74** ausgegeben. Wenn ferner die Adresse für die Speicheroperationen nicht von der Vorausschauadressen/Ergebnisberechnungseinheit **74** erzeugt werden kann, wird eine Adressenerzeugungsoperation an eine der Adressenerzeugungseinheiten **34a** bis **34d** über die Befehlsfenster **30a** bis **30b** ausgegeben.

Ferner werden Einträge in den Umordnungspuffer **28** für die FPU-Multimediabefehle zur Aufrechterhaltung der Programmreihenfolge reserviert. Im Allgemeinen werden Einträge innerhalb des Umordnungspuffers **28** von den Decodierungseinheiten **70a** bis **70f** für jeden darin empfangenen Befehl reserviert.

[0078] Jede der Decodierungseinheiten **70a** bis **70f** ist ausgebildet, die Operationscodierung und die mod r/m Bytes des Befehls, der gerade darin decodiert wird, zu der ESP/EBP-Vorausschaueinheit **72** zu liefern. Die ESP/EBP-Vorausschaueinheit **72** kann bestimmen: (i) ob der Befehl die ESP oder EBP-Register als einen Quellenoperanden verwendet oder nicht; und (ii) ob der Befehl die ESP/EBP-Register modifiziert oder nicht (d. h. ob der Befehl die ESP oder EBP-Register als Zieloperanden besitzt). Die ESP/EBP-Vorausschaueinheit **72** erzeugt die Vorausschauinformation für jeden Befehl, der die ESP oder EBP-Register als Quellenoperanden verwendet. Die Vorausschauinformation kann eine Konstante enthalten, die dem aktuellen Vorausschauwert des entsprechenden Registers hinzuzufügen ist und/oder kann eine Angabe über eine Abhängigkeit von einem Befehl in einer vorhergehenden Ausgabeposition enthalten. In einer Ausführungsform ist die ESP/EBP-Vorausschaueinheit **72** ausgebildet, Vorausschauinformation bereitzustellen, solange die Reihe aus Befehlen, die von dem Decodiereinheiten **70a** bis **70f** bereitgestellt wird, nicht mehr enthalten als: (i) zwei Schiebe- bzw. „Push"-Operationen (die das ESP-Register um einen konstanten Wert dekrementieren); (ii) zwei „Pop"-Operatinen (die das ESP-Register um einen konstanten Wert erhöhen); (iii) eine Übertragungsoperation zu dem ESP-Register; (iv) einen arithmetischen/logischen Befehl mit dem ESP als Ziel; oder (v) drei Befehle, die das ESP aktualisieren. Wenn eine dieser Einschränkungen überschritten wird, ist die ESP/EBP-Vorausschaueinheit **71** ausgebildet, Befehle, die über jene, die nicht die Einschränkungen überschreiten, hinausgehen, bis zum nachfolgenden Taktzyklus zu unterdrücken (im Fall von „aufgeteilter Linie bzw. Reihe"). Für jene Befehle, die in dem gleichen Taktzyklus aber in früheren Ausgabepositionen nach Befehlen folgen, die das ESP-Register inkrementieren oder dekrementieren, erzeugt die ESP/EBP-Vorausschaueinheit **72** eine Konstante, die die kombinierte gesamte Modifizierung des ESP-Registers der vorhergehenden Befehle angibt. Für Befehle, die nach einer Übertragungsoperation oder einer arithmetischen Operation an den ESP oder EBP-Registern folgen, erzeugt die ESP/EBP-Vorausschaueinheit **72** einen Wert, der die Ausgabeposition kennzeichnet, die den Übertragungs- oder Arithmetikbefehl enthält.

[0079] Die Vorausschauwerte können von der Vorausschauadressen/Ergebnisberechnungseinheit **74** verwendet werden, um eine vorausschauende Adresse zu erzeugen, die dem Befehl innerhalb der Ausgabeposition entspricht (wodurch eine Adressenerzeugungsoperation verhindert wird, die ansonsten durch eine der Adressenerzeugungseinheit von **34a** bis **34b** ausgeführt werden könnte), oder um ein vorausschauendes Ergebnis zu erzeugen, das dem Befehl entspricht (wodurch der Vorausschaustatus der Zukunftsdatei **26** früher in der Pipeline zugeführt wird). Das Leistungsvermögen kann gesteigert werden, indem die Adressenerzeugungsoperationen entfernt werden und/oder indem der vorrausschauende Zustand vor dem Ausführen der Befehlsoperationen in den Funktionseinheiten **32a** bis **32d** und den Adressenerzeugungseinheiten **34a** bis **34d** bereitgestellt wird. Viele x86-Codierungssequenzen enthalten eine große Anzahl relativ einfacher Operationen, etwa Übertragungsoperationen von Werten von einer Quelle zu einem Ziel ohne eine arithmetische/logische Operation oder einfache arithmetische Operationen, etwa Addieren/Subtrahieren mit einer kleinen Konstante oder Inkrementierung/Dekrementierung eines Registeroperanden. Folglich können die Funktionseinheiten **32a** bis **32d** typischerweise die komplexeren arithmetischen/logischen Operationen und Verzeigungsbefehle ausführen, und die Adressenerzeugungseinheiten **34a** bis **34d** können typischerweise die komplexeren Adressenerzeugungen durchführen. Somit kann der Befehlsdurchsatz dadurch gesteigert werden.

[0080] Die Erzeugungseinheit für das unmittelbare Feld **73** ist ausgebildet, unmittelbare Datenfelder aus der Reihe aus Befehlen, die dann decodiert sind, zu ermitteln (im hierin verwendeten Sinne können die unmittelbaren Daten eine Verschiebung zur Verwendung in der Adressenerzeugung oder unmittelbare Daten zur Verwendung in funktionalen Befehlsoperationen sein). Die unmittelbaren Daten werden zu der Vorausschau/Ergebnisberechnungseinheit **74** geleitet. Des Weiteren sind die Decodiereinheiten **70a** bis **70f** ausgebildet, Registeroperandenspezifizierungen zu erkennen, die von den Befehlen verwendet werden, und die Registeroperandenanforderungen zu der Zukunftsdatei **26** weiterzuleiten. Die Zukunftsdatei **26** gibt entsprechende spekulative Registerwerte oder Ergebniswarteschlangenmarken für jeden Registeroperanden zurück. Die reiheninterne Abhängigkeitsprüfeinheit **75** liefert eine Abhängigkeitsprüfung in der Reihe aus Befehlen. Die Operandenzusammenführeinheit **78** empfängt die Abhängigkeitsinformation, die von der reiheninternen Abhängigkeitsprüfeinheit **75** erzeugt wurde, um geeignete Operanden für jeden Befehl weiterzuleiten.

[0081] Die Vorausschauadressen/Ergebnisberechnungseinheit **74** empfängt die vorrausschauenden Werte von den ESP/EBP-Vorausschaueinheiten **72**, die unmittelbaren Daten von der Erzeugungseinheit für das unmittelbare Feld **73** und die spekulativen Registerwerte oder Ergebniswarteschlangenmarken von der Zukunftsdatei **26**. Die Vorausschauadressen/Ergebnisberechnungseinheit **74** versucht, eine vorrausschauende Adres-

se entsprechend einem Speicheroperanden des Befehls zu erzeugen, oder ein vorausschauendes Ergebnis, wenn der Befehl keinen Speicheroperanden enthält. Beispielsweise können einfache Register-zu-Register-Übertragungsoperationen abgeschlossen werden (in Bezug auf die Funktionseinheiten **32** und die Adresseerzeugungseinheiten **34**), indem der Quellenoperand als der Zieloperand weitergeleitet wird. Übertragungsoperationen verwenden eine Speicheroperation und ein Registerziel kann abgeschlossen werden (in Bezug auf die Funktionseinheiten **32** und die Adresserzeugungseinheiten **34**), wenn eine Adressenerzeugung von der Vorausschauadressen/Ergebnisberechnungseinheit **74** ausgeführt werden kann. In einer Ausführungsform ist die Vorausschauadressen/Ergebnisberechnungseinheit **74** ausgebildet, Adressen unter lediglich einer Verschiebung, Register plus Verschiebung, ESP/EBP plus Verschiebung und Skalierungs-Index-Basisadressierungsmodi mit Ausnahme für Index- oder Basisregister, die die ESP/EPD-Register sind, zu berechnen. Die Lade/Schreibeinheit **36** führt die Speicheroperation aus und gibt die Speicheroperationsergebnisse über die Ergebnisbusse **48** zurück. Selbst wenn keine Adresse für eine Speicheroperation von der Vorausschauadressen/Ergebnisberechnungseinheit **74** erzeugt wird, gibt diese die Speicheroperation und die entsprechende Ergebniswarteschlangenmarke an die Lade/Schreibeinheit **36** aus, um Speicherplatz innerhalb der Lade/Schreibeinheit **36** für die Speicheroperation zu reservieren.

[0082] Einfache arithmetische Operationen, die einen Quellenoperanden inkrementieren oder dekrementieren, einen kleinen unmittelbaren Wert zu einem Quellenoperanden addieren oder subtrahieren, oder zwei Registerquellenoperanden addieren/subtrahieren können ebenso mittels der Vorausschauadressen/Ergebnisberechnungseinheit **74** abgeschlossen werden, wenn die Quellenoperanden aus der Zukunftsdatei **26** verfügbar sind (d. h. es wird ein spekulativer Registerwert anstelle einer Ergebniswarteschlangenmarke empfangen). Befehle, die von der Vorausschauadressen/Ergebnisberechnungseinheit **74** abgeschlossen werden, werden als vollständig angegeben und sind reservierte Einträge in dem Umordnungspuffer **28**, werden jedoch nicht an die Befehlsfenster **30** ausgegeben. Die Vorausschauadressen/Ergebnisberechnungseinheit **74** kann beispielsweise einen Addierer für jede Ausgabeposition zusammen mit der entsprechenden Steuerungslogik zum Auswählen aus den Vorausschauwerten, den unmittelbaren Daten und den spekulativen Registerwerten aufweisen. Zu beachten ist, dass einfache arithmetische Operationen dennoch an die Befehlsfenster zur Erzeugung von Bedingungsmarken bzw. Flaggen gemäß der vorliegenden Ausführungsform weitergeleitet werden können. Jedoch liefert das Erzeugen des Funktionsergebnisses in der Vorausschauadressen/Ergebnisberechnungseinheit **74** den Vorausschauzustand früher, wodurch nachfolgende Adressenerzeugungen/Befehle früher ausgeführt werden können.

[0083] Die Vorausschauadressen/Ergebnisberechnungseinheit **74** ist ausgebildet, separate vorrausschauende Kopien der ESP/EBP-Register zusätzlich zu den Zukunftsdateikopien zu halten. Wenn jedoch Aktualisierungen der ESP/EBP erkannt werden, die nicht von der Vorausschauadressen/Ergebnisberechnungseinheit **74** berechnet werden können, werden nachfolgende Befehle angehalten, bis eine neue vorrausschauende Kopie des ESP/EBP von der Zukunftsdatei **26** bereitgestellt werden kann (nach dem Ausführen des Befehls, der das ESP/EBP in unmittelbarer Weise aktualisiert).

[0084] Die Ausgabesteuereinheit **76** bestimmt, ob eine Gruppe aus Befehlen ausgegeben wird oder nicht, um eine Pipelineablaufsteuerung bereitzustellen. Die Ausgabesteuereinheit **76** empfängt Befehlsanzahlen von den Befehlsfenstern **30** und Lade/Schreibanzahlen von der Lade/Schreibeinheit **36** und bestimmt – unter der Annahme, dass die maximal mögliche Anzahl an Befehlen gerade in den Pipeline-Stufen zwischen den Ausgabesteuereinheiten **76** und den Befehlsfenstern **30** und der Lade/Schreibeinheit **36** sind, – ob Platz für das Speichern der auszugebenden Befehle in den Befehlsfenstern **30** und/oder der Lade/Schreibeinheit **36** verfügbar ist oder nicht, wenn die Befehle darin eintreffen. Wenn die Ausgabesteuereinheit **76** bestimmt, dass nicht ausreichend Platz in der Lade/Schreibeinheit **36** und in keinem der Befehlsfenster **30** vorhanden ist, wird die Ausgabe unterbrochen, bis die Befehlsanzahl, die von der Ausgabesteuereinheit **76** empfangen wird, auf einen ausreichend geringen Wert abgefallen ist.

[0085] Bei der Freigabe von Befehlen für die Ausgabe über die Ausgabesteuereinheit **76** werden die Zukunftsdatei **26** und der Umordnungspuffer **28** mit spekulativ erzeugten vorausschauenden Ergebnissen aktualisiert. In einer Ausführungsform ist die Anzahl der nicht ESP/EBP-Aktualisierungen, die unterstützt werden, beispielsweise auf zwei begrenzt, um damit die Anzahl an Anschlüssen bzw. Ports in der Zukunftsdatei **26** zu begrenzen. Ferner führt die Operandenzusammenführeinheit **78** spekulativ erzeugte vorrausschauende Ergebnisse in nachfolgenden gleichzeitig decodierten Befehlen zusammen, die von diesen Ergebnissen abhängen, wie dies durch die zuvor bestimmten reiheninternen Abhängigkeiten angegeben ist. Auf diese Weise erhalten die abhängigen Befehle die spekulativ erzeugten vorrausschauenden Ergebnisse, da diese Ergebnisse nicht nachfolgend von den Funktionseinheiten **32a** bis **32d** weitergeleitet werden. Diese Befehle, die von der Vorrausschauadressen/Ergebnisberechnungseinheit **74** nicht abgeschlossen werden, werden dann zu einem

der Befehlsfenster **30a** bis **30b** auf der Grundlage der Ausgabeposition ausgegeben, die diesen Befehlen von der Ausrichteinheit **22** zugewiesen wurde.

[0086] Zu beachten ist, dass in gewissen Ausführungsformen des Prozessors **10** eine Mikrocodierungseinheit (nicht gezeigt) zum Ausführen komplexer Befehle vorgesehen ist, um mehrere einfache Befehle, die als Mikrocodierungsroutinen bezeichnet werden, auszuführen. Die Decodiereinheiten **70a** bis **70f** können ausgebildet sein, zu erkennen, welche Befehle Mikrocodierungsbefehle sind und können die Mikrocodierungsbefehle zu der Mikrocodierungseinheit weiterführen. Beispielsweise kann das Fehlen eines direkt decodierten Befehls, der von einer Decodierungseinheit **70** ausgegeben wird, die einen gültigen Befehl empfangen hat, eine Angabe für die Mikrocodierungseinheit sein, die Ausführung des entsprechenden gültigen Befehls zu beginnen. Ferner ist zu beachten, dass diverse Speichereinrichtungen in den Fig. 2 und Fig. 3 (beispielsweise die Einrichtungen **79a**, **79b** und ähnliche Einrichtungen in Fig. 2 und die Einrichtungen **79c**, **79d** und ähnliche Einrichtungen in **Fig.** 3d) gezeigt sind. Die Speichereinrichtungen repräsentieren Signalzwischenspeicher, Register, Flip-Flops, und dergleichen, die zur Trennung von Pipeline-Stufen verwendet werden können. Jedoch sind die speziellen in den Fig. 2 und Fig. 3 gezeigten Pipeline-Stufen nur eine einzelne Ausführungsform geeigneter Pipeline-Stufen für eine einzelne Ausführungsform des Prozessors **10**. Andere Pipeline-Stufen können in anderen Ausführungsformen ebenso verwendet werden.

[0087] Zu beachten ist, dass obwohl der x86-Befehlssatz und Architektur als ein Beispiel zuvor verwendet ist und auch als Beispiel nachfolgend verwendet ist, eine beliebige Architektur und ein beliebiger Befehlssatz verwendet werden können. Ferner können Verschiebungen mit einer beliebigen gewünschten Größe (zusätzlich zu 8 Bit und 32 Bit-Größen, die hierin als Beispiel verwendet sind) eingesetzt werden. Obwohl ferner die Cache-Zeilenabholung hierin beschrieben ist, ist zu beachten, dass Cache-Zeilen Sektoren sein können, und Sektoren abgeholt werden können, wenn dies wünschenswert ist, auf der Grundlage einer Cache-Zeilengröße und der Anzahl von Bytes, die abzuholen sind.

[0088] Fig. 4 zeigt eine Blockansicht der Decodierungseinheit **70a**, der ESP/EBP-Vorausschaueinheit **72**, der reiheninternen Abhängigkeitsprüfeinheit **75**, der Erzeugungseinheit **73** für das unmittelbare Feld und eines Befehlsbytespeichers **80**. Eine Verbindung zwischen den dargestellten Blöcken gemäß einer Ausführungsform des Prozessors **10** ist in Fig. 4 gezeigt. Andere Ausführungsformen mit zusätzlichen, alternativen und/oder unterschiedlichen Verbindungsschemata sind ebenso mit eingeschlossen. In der Ausführungsform aus Fig. 4 ist die Decodiereinheit **70a** mit einem Operationscodierungs/mod R/M-Bus **82a**, einem Quellen- und Zielregisterspezifizierungsbus **84a** und einem Startpositionsbus **86a** verbunden. Der Startpositionsbus **86a** und der Operationscodierungs/mod-R/M-Bus **82a** sind mit der Erzeugungseinheit **73** für das unmittelbare Feld verbunden, das ferner mit dem Befehlsbytespeicher **80** verbunden ist. Der Operationscodierungs/mod-R/M-Bus **82a** ist ferner mit der ESP/EBP-Vorausschaueinheit **72** verbunden. Der Quellen- und Zielregisterspezifizieurungsbus **84a** ist mit der reiheninternen Abhängigkeitsprüfeinheit **75** verbunden, die wiederum mit ähnlichen Quellen- und Zielregisterspezifizierungsbussen **84b** bis **84f** von anderen Decodiereinheiten **70** verbunden ist. Die ESP/EBP-Vorausschaueinheit **72** ist mit einem Vorausschau-ESP/EBP-Bus **88** und einem Konstantbus **90a** verbunden, die wiederum mit der Vorausschauadressen/Ergebnisberechnungseinheit **74** verbunden sind. Die reiheninterne Abhängigkeitsprüfeinheit **75** ist mit einem Abhängigkeitsbus **92** verbunden, der ferner mit der Operandenzusammenführeinheit **78** verbunden ist. Die Erzeugungseinheit **73** für das unmittelbare Feld ist mit einem Bus **94a** für das unmittelbare Feld und einer Vergleichsleitung **86a** verbunden. Die Vergleichsleitung **86a** ist mit der Operandenzusammenführeinheit **78** verbunden, und der Bus **94a** für das unmittelbare Feld ist mit der Vorrausschauadressen/Ergebnisberechnungseinheit **74** verbunden. Busse und Leitungen, die mit einer Bezugszahl mit anschließenden Buchstaben bezeichnet sind, entsprechen dem Befehl, der von der Decodiereinheit **70a** gerade decodiert wird. Eine ähnliche Verbindungsstruktur kann bereitgestellt werden, die den Befehlen entspricht, die gleichzeitig von den Decodiereinheiten **70b** bis **70f** decodiert werden. Der Einfachheit halber sind in Fig. 4 die Verbindungen, die den anderen Befehlen, die gerade gleichzeitig decodiert werden, nicht gezeigt (und auch in den nachfolgenden Figuren sind diese nicht gezeigt). Jedoch können ähnliche Verbindungsstrukturen, wie sie gezeigt sind, auch für die anderen Befehle innerhalb der Reihe vorgesehen werden.

[0089] Die Decodiereinheit **70a** decodiert einen Befehl, der von der Befehlsausrichteinheit **22** bereitgestellt wird, und erkennt die Startposition des Befehls innerhalb der Befehlsbytes, die der Reihe aus Befehlen entspricht, die gleichzeitig von der Befehlsausrichteinheit **22** an die Decodiereinheiten **70a** bis **70f** geliefert werden. Die Befehlsbytes, die der Reihe entsprechen, sind in dem Befehlsspeicher **80** abgelegt (der beispielsweise ein Register sein kann), und die Decodiereinheit **70a** empfängt den Teil des Befehls, der von der Decodiereinheit **70a** verwendet wird, um die Decodieraufgabe auszuführen. In einer Ausführungsform empfängt die Decodiereinheit **70a** das Präfix, den Operationscode und mod-R/M-Bytes, die dem zu decodierenden Befehl ent-

sprechen, sowie die Startposition des Befehls innerhalb der Befehlsbytes, die der Reihe entsprechen. In anderen Ausführungsformen können andere Bereiche des Befehls empfangen werden, abhängig von der Befehlssatzarchitektur, die in der entsprechenden Ausführungsform verwendet ist.

[0090] Die Decodiereinheit **70a** erkennt die Quellen- und Zielregisteroperandenspezifizierungen für den Befehl und überträgt die Spezifizierungen auf dem Quellen- und Zielregisterspezifizierungsbus **84a**. In der vorliegenden Ausführungsform kann ein Befehl bis zu 12 Quellenoperanden und einen Zieloperanden aufweisen. Wenn der Befehl einen Speicheroperanden beinhaltet, können die Quellenoperanden Adressenoperanden enthalten. Zusätzlich überträgt die Decodiereinheit **70a** die Startposition des Befehls auf den Startpositionsbus **86a**.

[0091] Die ESP/EBP-Vorausschaueinheit **72** empfängt die Operationscodierungs- und Mode-R/M-Bytes und bestimmt, ob der entsprechende Befehl die ESP oder EBP-Register als Quellenoperanden besitzt, sowie ob der Befehl eine Aktualisierung dieser Register spezifiziert oder nicht. Die ESP/EBP-Vorausschaueinheit **72** liefert einen Vorausschau-ESP-Wert und einen Vorausschau-EBP-Wert auf dem Vorausschau-ESP/EBP-Bus **88**. Die Vorausschauregisterwerte entsprechen der Gesamtheit der Ergebnisse von Befehlen, die vor der Reihe aus Befehlen, die gerade von den Decodiereinheiten **70a** bis **70f** decodiert werden, ausgegeben wurden. Zusätzlich erzeugt die ESP/EBP-Vorausschaueinheit **72** eine Konstante, die dem Vorausschau-ESP- oder EBP-Wert hinzugefügt wird, um einen Quellenoperanden für den Befehl zu erzeugen, der gerade von der Decodiereinheit **70a** decodiert wird. Ähnliche Konstante werden für andere Befehle innerhalb der Reihe erzeugt. Die Konstante für jede Ausgabeposition repräsentiert die gesamte Wirkung der Befehle vor und einschließlich des Befehls, der diese Ausgabeposition innerhalb der Reihe entspricht. In dem Falle der Decodiereinheit **70a** gibt es keine Befehle vor dem Befehl innerhalb der Reihe, und somit entspricht die Konstante einer Modifizierung des entsprechenden Registers, die vor dem Verwenden des Wertes als einen Quellenoperanden für den Befehl ausgeführt wird. Beispielsweise spezifiziert der „Push"-Befehl, der in der x86-Befehlssatzarchitektur definiert ist, dass der ESP-Registerwert dekrementiert wird, bevor der Wert als ein Adressenoperand für den Befehl verwendet wird.

[0092] Die reiheninterne Abhängigkeitsprüfeinheit **75** empfängt die Quellen- und Zielregisterspezifizierungen von der Decodiereinheit **70a** sowie von anderen Decodiereinheiten **70b** bis **70f**. Die reiheninterne Abhängigkeitsprüfeinheit **75** führt eine Abhängigkeitsprüfung für jeden Quellenoperanden, der einem Befehl innerhalb einer speziellen Ausgabeposition entspricht, gegenüber den Zieloperanden jedes früheren Befehls innerhalb der Reihe aus. Wenn eine Abhängigkeit erkannt wird, wird eine entsprechende Angabe über den Abhängigkeitsbus **92** bereitgestellt. Folglich enthält der Abhängigkeitsbus **92** eine Angabe für jeden möglichen Quellenoperanden, die angibt, ob eine Abhängigkeit für diesen Quellenoperanden erkannt wurde oder nicht, sowie über die Ausgabeposition, von der der Quellenoperand abhängt. Beispielsweise kann der Abhängigkeitsbus **92** ein Abhängigkeitssignal entsprechend jedem Quellenoperanden sowie einer Ausgabepositionsnummer entsprechend jedem Quellenoperanden enthalten.

[0093] Die Erzeugungseinheit **73** für das unmittelbare Feld decodiert die Operationscodierungs- und mod-R/M-Bytes des Befehls, um zu bestimmen, ob ein unmittelbares Feld in dem Befehl enthalten ist. Die Erzeugungseinheit **73** für das unmittelbare Feld extrahiert das unmittelbare Feld aus dem Befehlsbytespeicher **80** und liefert das unmittelbare Feld auf dem Bus **94a** für das unmittelbare Feld. In einer Ausführungsform werden diverse Größen für unmittelbare Felder unterstützt (8 Bit und 32 Bit). Die Erzeugungseinheit **73** für das unmittelbare Feld kann ausgebildet sein, eine Vorzeichenerweiterung oder eine Erweiterung Null für kleinere unmittelbare Felder in Bezug auf die Größe des größten unterstützten unmittelbaren Felds auszuführen, wie dies geeignet ist. Wenn ferner der Befehl ein Inkrement/Dekrement-Befehl ist, ist die Erzeugungseinheit **73** für das unmittelbare Feld ausgebildet, eine Konstante für das unmittelbare Feld zu erzeugen, die die Größe der Inkrementierung/Dekrementierung wiedergibt. Wenn der Befehl kein unmittelbares Feld enthält und kein Inkrement/Dekrement-Befehl ist, ist die Erzeugungseinheit **73** für das unmittelbare Feld ausgebildet, einen Wert von 0 auf dem Bus **94a** für das unmittelbare Feld zu erzeugen. Die Erzeugungseinheit **73** für das unmittelbare Feld ist ferner ausgebildet, ähnliche Operationen auszuführen, die den Befehlen innerhalb anderer Decodiereinheiten **70** entsprechen. Die Erzeugungseinheit **73** für das unmittelbare Feld wählt das unmittelbare Feld aus dem Befehlsbytespeicher **80** gemäß der Startposition aus, die von der Decodiereinheit **70a** und einer Decodierung der Operationscodienangs- und mod-R/M-Bytes bestimmt sind.

[0094] In der vorliegenden Ausführungsform enthält der Befehlsbytespeicher **80** die Befehlsbytes, die der Reihe aus Befehlen entsprechen, die gleichzeitig decodiert werden. Die Befehlsbytes werden entsprechend der Reihe aus Befehlen durch die Pipeline verarbeitet, wodurch diverse Felder des Befehls nach Bedarf herausgelöst werde können. In anderen Ausführungsformen werden die Befehle vollständig innerhalb jeder Aus-

gabeposition weitergeleitet (anstatt, dass eine Teilmenge der Befehle und Zeiger zu den Befehlen innerhalb des Befehlsbytespeichers **80** weitergeleitet werden).

[0095] In der vorliegenden Ausführungsform ist die Erzeugungseinheit **73** für das unmittelbare Feld ferner ausgebildet, einen Vergleichsbefehl innerhalb einer speziellen Ausgabeposition zu erkennen, dem ein Verzweigungsbefehl in der unmittelbaren nachfolgenden Ausgabeposition folgt, der von dem Vergleichsergebnis abhängt (d. h. ein bedingter Verzweigungsbefehl). Beim Erkennen einer derartigen Kombination erzeugt die Erzeugungseinheit **73** für das unmittelbare Feld eine Vergleichsindikation für die Operandenzusammenführeinheit **78**. In anderen Ausführungsformen wird eine separate Steuereinheit vorgesehen, um die Vergleich/Verzweigungskombination zu erkennen, oder eine der anderen Einheiten ist ausgebildet, diese Funktion zu ermöglichen. Ferner sind Ausführungsformen berücksichtigt, in denen die Vergleich/Verzweigungskombination nicht erkannt wird. Die Operandenzusammenführeinheit **78** kann ausgebildet sein, die Vergleichsoperation und die Verzweigungsoperation in einer einzelnen Ausgabeposition zusammenzuführen (d. h. eine Funktionseinheit kann sowohl die Vergleichsoperation als auch die abhängige Verzweigungsoperation gleichzeitig ausführen).

[0096] Zu beachten ist, dass in Fig. 4 gewisse Ausgangssignale der gezeigten Einheiten so beschrieben sind, dass diese in Pipeline-Struktur bis hinab zu beispielsweise der Operandenzusammenführeinheit **78** angeordnet sind. Dies ist beabsichtigt, um anzuzeigen, dass diese Ausgänge mit dem entsprechenden Befehl durch die Pipeline zu der empfangenden Einheit erzeugt und transportiert werden. Alternativ könnten diese Ausgangssignale durch die empfangene Einheit aufgenommen und mit dem entsprechenden Befehl beim Eintreffen des Befehls darin verknüpft werden.

[0097] Fig. 5 zeigt einen Teil einer Ausführungsform einer Vorausschauadressen/Ergebnisberechnungseinheit **74**. Andere Ausführungsformen sind möglich und hierin mit eingeschlossen. In der Ausführungsform aus Fig. 5 ist ein Teil der Vorausschauadressen/Ergebnisberechnungseinheit **74** gezeigt, der der Ausgabeposition 0 entspricht (d. h. der der Decodiereinheit **70a** entspricht). Ähnliche Hardware kann für die anderen Ausgabepositionen vorgesehen sein. Eine Steuereinheit **100** ist so gezeigt, dass diese mit dem Operationscodierungs/mod-R/M-Bus **82a** in Pipeline-Art abwärts von der Decodiereinheit **70a** verbunden ist. Die Steuereinheit **100** ist ferner angeschlossen, gültige Angaben zu empfangen, die den Quellenoperanden des Befehls entsprechen, wie sie von der Zukunftsdatei **26** über einen Bus **102** bereitgestellt werden. Ein Quellenoperand ist gültig in der Zukunftsdatei **26**, wenn der Quellenoperand anstelle einer Ergebniswarteschlangenmarke bereitgestellt wird, die den Befehl angibt, der den Quellenoperanden beim Ausführen erzeugt. Die Steuereinheit **100** ist angeschlossen, um Auswahlsteuerungen für Operandenauswahlmultiplexer (mux) **104a**, **104b** bereitzustellen. Der Operandenauswahlmux **104a** ist mit einem Vorausschau-ESP-Bus **88-1** (einem Teil des Vorausschau-ESP/EBP-Busses **88**, der den Vorausschau-ESP-Wert liefert), dem Konstanten-Bus **90a** und einem Bus **112** für eine Quelle 1 aus der Zukunftsdatei **26** verbunden. Der Operandenauswahlmux **104b** ist mit einem Vorausschau-EBP-Bus **88-2** (ein Teil des Voraausschau-ESP/EBP-Busses **88**, der den Vorausschau-EBP-Wert liefert), dem Konstanten-Bus **90a** und einem Bus **114** für eine Quelle 2 aus der Zukunftsdatei **26** verbunden. Ein Addierer **106** ist so gezeigt, dass dieser mit den Operandenauswahlmultiplexern **104** sowie dem unmittelbaren Feldbus **94a** und einem Segmentbus **108** verbunden ist. Der Addierer **106** liefert ein Ergebnis auf einem Ergebnisbus **110a**, das der Lade/Schreibeinheit **36**, der Zukunftsdatei **26** und der Operandenzusammenführeinheit **78** zugeführt ist. Die Steuereinheit **100** erzeugt ein Adressengültigkeitssignal auf einer Adressengültigkeitsleitung **116a** für die Lade/Schreibeinheit **36** und die Operandenzusammenführeinheit **78**, und erzeugt ein Ergebnisgültigkeitssignal auf einer Ergebnisgültigkeitsleitung **118a** für die Zukunftsdatei **26** und die Operandenzusammenführeinheit **78**. Zu beachten ist, dass, wie zuvor erläutert ist, Verbindungen, die mit einer Bezugszahl mit anschließenden Buchstaben bezeichnet sind, für Befehle in anderen Ausgabepositionen erneut verwendet werden können (in Fig. 5 nicht gezeigt).

[0098] Die Steuereinheit **100** ist ausgebildet, die Operationscodierungs- und mod-R/M-Bytes des Befehls zu decodieren, um zu bestimmen, ob eine Vorausschauadressenerzeugung (für Befehle mit einem Speicheroperanden), eine Vorausschauergebniserzeugung (für Befehle ohne einen Speicheroperanden und zur Ausführung einer Additionsbefehlsoperation und/oder einer Übertragungsbefehlsoperation) auszuführen ist oder nicht. Die Steuereinheit **100** kann Operanden über die Operandenauswahlmultiplexer **104** gemäß einer Decodierung des Befehls auswählen. Der Addierer **106** ist ausgebildet, um zu addieren: (i) die Werte, die von dem Operandenauswahlmultiplexer **104a**, **104b** bereitgestellt werden; (ii) den unmittelbaren Wert, der von dem Bus **94a** für das unmittelbare Feld bereitgestellt wird; (iii) und die Segmentbasisadresse, die auf dem Segmentbus **108** bereitgestellt wird, um das Ergebnis zu erzeugen. Für Ergebniserzeugungen (im Gegensatz zu Adressenerzeugungen) kann der Segmentwert 0 sein oder kann von dem Addierer **106** ignoriert werden in Reaktion auf ein Steuersignal von der Steuereinheit **100** (nicht gezeigt).

[0099] Beispielsweise kann die Wahrheitstabelle **120**, die in Fig. 6 gezeigt ist, für eine Ausführungsform der Steuereinheit **100** verwendet werden, um Werte der Operandenauswahlmultiplexer **104** auszuwählen, wenn der Befehl eine Adressenerzeugungsbefehlsoperation für einen Speicheroperanden enthält. Die Wahrheitstabelle **120** umfasst eine Adressiermodusspalte, die den Adressiermodus kennzeichnet, der von dem Befehl verwendet wird, um die Adresse des Speicheroperanden zu erzeugen, eine Multiplexer 1 Spalte, die den von dem Operandenauswahlmultiplexer **104a** in Reaktion auf die Steuerung durch die Steuereinheit **100** ausgewählten Wert angibt, und einen Multiplexer 2 Spalte, die den Wert angibt, der von dem Operandenauswahlmultiplexer **104b** in Reaktion auf die Ansteuerung durch die Steuereinheit **100** ausgewählt wird. Andere Ausführungsformen sind möglich und hierin mit eingeschlossen. Die in Fig. 6 gezeigte Ausführungsform unterstützt die folgenden Addressiermodi: (i) nur Verschiebung; (ii) Kombinationen aus einer optionalen Verschiebung und einem oder mehreren Adressenoperanden (ausschließlich der ESP/EBP-Register); oder (iii) die ESP- oder EBP-Register als Quellenoperanden zusammen mit einer optionalen Verschiebung. Zu beachten ist, dass ein Maßstabsfaktor in den Skalierungs-Index-Basisadressiermodi enthalten sein kann, wie sie in der x86-Befehlssatzarchitektur spezifizierbar sind. Der Multiplexer **104a** kann ausgebildet sein, den Operanden der Quelle 1 durch Auswählen des Operanden der Quelle 1, der um ein oder zwei Bits nach links verschoben ist, zu skalieren. Ferner ist zu beachten, dass die in Fig. 6 gezeigte Verschiebung auf dem unmittelbaren Bus **94a** in der vorliegenden Ausführungsform bereitgestellt wird.

[0100] Abhängig von dem Adressiermodus, der durch den Befehl spezifiziert ist, untersucht die Steuereinheit **100** die gültigen Indikationen aus der Zukunftsdatei **26** und bestimmt, ob der Addierer **106** die Adresse des Speicheroperanden erfolgreich erzeugt. Anders ausgedrückt, wenn jeder Quellenoperand, der ein Adressenoperand ist, gültig ist, kann der Addierer **106** erfolgreich die Adresse erzeugen. Wenn die Adresse erfolgreich erzeugt ist, setzt die Steuereinheit **100** das Adressengültigkeitssignal auf der Adressengültigkeitsleitung **116a**, um der Lade/Schreibeinheit **36** anzuzeigen, dass die auf den Ergebnisbus **110a** bereitgestellte Adresse aufzunehmen und dass die Adresse für den Speicheroperanden für den entsprechenden Befehl zu verwenden ist. Wenn die Adresse nicht erfolgreich erzeugt wird, können die Adressenerzeugungseinheiten **34** verwendet werden, um die Adresse nachfolgend zu erzeugen und die Adresse dann zu der Lade/Schreibeinheit **36** weiterzuleiten.

[0101] Der Segmentbus **108** liefert eine Segmentbasisadresse gemäß dem Segmentierungsübersetzungsmechanismus, der durch x86-Befehlssatzarchitektur definiert ist. Andere Ausführungsformen für Prozessoren verwenden andere Befehlssatzarchitekturen und können den Segmentbus **108** nicht aufweisen. Die Segmentbasisadresse, die auf dem Segmentbus **108** bereitgestellt wird, ist die Segmentbasisadresse des ausgewählten Segments, das den Befehl innerhalb der aktuellen Ausgabeposition entspricht. Alternativ kann jede verfügbare Segmentbasisadresse bereitgestellt und gemäß dem entsprechenden Befehl ausgewählt werden. Segmentinformationen können in einer geeigneten Übersetzungseinheit oder einer speziellen Registereinheit aufbewahrt werden, wie dies im Stand der Technik bekannt ist.

[0102] Wenn der Befehl in der aktuellen Ausgabeposition keinen Speicheroperanden enthält, kann die Steuereinheit **100** versuchen, Operanden auszuwählen, um ein vorrausschauendes Ergebnis für den Befehl zu erzeugen. In der vorliegenden Ausführungsform kann die Steuereinheit **100** eine Addier/Subtrahier- oder Inkrement- oder Dekrement-Operation mit einem oder zwei Quellenoperanden und/oder einem unmittelbaren Wert zu unterstützen. Die Steuereinheit **100** kann ferner Register-zu-Register-Übertragungen durch Bereitstellen des zweiten Quellenoperanden als den einzigen Eingang für den Addierer **106** zu unterstützen.

[0103] Beispielsweise operiert in einer Ausführungsform die Steuereinheit **100** gemäß der Wahrheitstabelle **122**, die in Fig. 7 gezeigt ist. Andere Ausführungsformen sind ebenso möglich. Die in Fig. 7 gezeigte Wahrheitstabelle **122** enthält eine arithmetische Operationsspalte, die die arithmetischen Operationen angibt, die von der Vorausschauadressen/Ergebnisberechnungseinheit **74** gemäß der vorliegenden Ausführungsform unterstützt werden. Ähnlich zu der in Fig. 6 gezeigten Wahrheitstabelle **120** enthält die Tabelle **122** eine Multiplexer 1 Spalte, die den von dem Operandenauswahlmultiplexer **104** ausgewählten Operanden angibt, und enthält einen Multiplexer 2 Spalte, die den von dem Operandenauswahlmultiplexer **104** in Reaktion auf entsprechende Ansteuerungen aus der Steuereinheit **100** angibt. Gemäß der Tabelle wird eine Addition oder Subtraktion von 1 (wenn 0 durch den Multiplexer **104b** ausgewählt ist) oder zwei Quellenoperanden und ein optionales unmittelbares Feld sowie Inkrementierung oder Dekrementierung und ein Register-Register-Transfer unterstützt.

[0104] Die Steuereinheit **100** ist ausgebildet zu bestimmen, ob der Addierer **106** erfolgreich ein vorausschauendes Ergebnis auf dem Ergebnisbus **110a** erzeugt. Die Steuereinheit **100** bestimmt, dass ein erfolgreiches Ergebnis erzeugt ist, wenn jeder der Quellenoperanden, der zum Erzeugen eines Ergebnisses verwendet wur-

de, durch die Zukunftsdatei **26** gültig ist und wenn die Funktionsbefehlsoperation, die von dem Befehl spezifiziert ist, eine ist, die von der Vorausschauadressen/Ergebnisberechnungseinheit **74** unterstützt wird. Wenn das Ergebnis als erfolgreich erzeugt bestimmt wird, setzt die Steuereinheit **100** ein Ergebnisgültigkeitssignal auf der Ergebnisgültigkeitsleitung **118** für die Zukunftsdatei **26** und die Operandenzusammenführeinheit **78**. Die Zukunftsdatei **26** kann ausgebildet sein, das Ergebnis gemäß dem Zielregister des Befehls zu speichern (das auch auf dem Ergebnisbus **110a** bereitgestellt werden kann). Die Operandenzusammenführeinheit **78** ist ausgebildet, das Ergebnis in einen Quellenoperanden eines abhängigen Befehls einzubinden (gemäß den Abhängigkeitsangaben, die von der reiheninternen Abhängigkeitsprüfeinheit **75** bereitgestellt werden).

[0105] Fig. 8 ist eine Blockansicht, die einen Teil einer Ausführungsform einer Operandenzusammenführeinheit **78** zeigt. Andere Ausführungsformen sind möglich und hierin berücksichtigt. In dem in Fig. 8 gezeigten Teil ist eine Steuereinheit **130** zusammen mit Operandenzusammenführ-Multiplexern **132a** bis **132d** gezeigt. Die Steuereinheit **130** ist mit mehreren Ergebnisgültigkeitsleitungen **118** (einschließlich der Ergebnisgültigkeitsleitung **118a**, die in Fig. 5 gezeigt ist), mehreren Adressengültigkeitsleitungen **116** (einschließlich der Adressengültigkeitsleitung **116a**, die in Fig. 5 gezeigt ist), dem Abhängigkeitsbus **92** aus der reiheninternen Abhängigkeitsprüfeinheit **75**, mehreren Vergleichsleitungen **96** (einschließlich der Vergleichsleitung **96a**, die in Fig. 4 gezeigt ist), und einem Verhinderungsbus **134** verbunden. Die Steuereinheit **130** ist ferner angeschlossen, um Auswahlsteuersignale für jeden der Zusammenführ- Multiplexer **132** bereitzustellen (sowie für die Zusammenführmultiplexer für andere Ausgabepositionen, die in Fig. 8 nicht gezeigt sind). Jeder der Zusammenführmultiplexer **132a** bis **132d** entspricht einem Quellenoperanden innerhalb einer einzelnen Ausgabeposition. Jede der Zusammenführmultiplexer **132a** bis **132d** ist angeschlossen, die Zielmarken zu empfangen, die den vorhergehenden Ausgabepositionen innerhalb der Reihe entsprechen, sowie die vorrausschauenden Ergebnisse zu empfangen, die diesen Ausgabepositionen entsprechen. Des weiteren sind die Zusammenführmultiplexer **132a** bis **132d** angeschlossen, die Ausgabe der Zukunftsdatei für den Quellenoperanden des Befehls innerhalb der Ausgabeposition, die unmittelbar vor der Ausgabeposition ist, die diesen Zusammenführmultiplexern **132a** bis **132d** entspricht, und dem Quellenoperanden des Befehls innerhalb der Ausgabeposition, die diesen Zusammenführmultiplexern **132a** bis **132d** entspricht, zu empfangen. Beispielsweise stellt der Zusammenführmultiplexer **132a** den Operanden der Quelle 1 für die Ausgabeposition 1 bereit und ist angeschlossen, um zu empfangen: (i) die Zielmarkierung für den Befehl innerhalb der Ausgabeposition 0; (ii) das vorrausschauende Ergebnis, das auf dem Ergebnisbus **110a** entsprechend der Ausgabeposition 0 bereitgestellt wird; und (iii) die Operanden der Quelle 1 aus der Zukunftsdatei **26** entsprechend den Operanden der Quelle 1 der Ausgabeposition 0 und dem Operanden der Quelle 1 der Ausgabeposition 1. Wie in Fig. 8 gezeigt ist, gibt der Großbuchstabe P im Anschluss an das Bezugszeichen die Ausgabeposition entsprechend dem angegebenen Wert an (d. h. P0 ist die Ausgabeposition 0, die 1 ist die Ausgabeposition 1, etc.).

[0106] Wie zuvor beschrieben ist, führt die Operandenzusammenführeinheit **78** die vorausschauenden Ergebnisse, die von der Vorausschauadressen/Ergebnisberechnungseinheit **74** erzeugt werden, in dem Quellenoperanden der abhängigen Befehle innerhalb der Reihe zusammen. In der in Fig. 8 gezeigten Ausführungsform ist ein Operandenzusammenführmultiplexer **132** für jeden Quellenoperanden und jede Ausgabeposition, für die eine Zusammenführung ausgeführt werden kann, vorgesehen. Folglich sind keine Operandenzusammenführmultiplexer für die Ausgabeposition 0 gezeigt, da die Ausgabeposition 0 die erste Ausgabeposition innerhalb der Reihe ist und somit keine reiheninterne Abhängigkeit aufweist. Die Operandenzusammenführmultiplexer **132a** und **132b** liefern die Quellenoperanden für die Ausgabeposition 1, während die Operandenzusammenführmultiplexer **132c** und **132b** die Quellenoperanden für die Ausgabeposition 2 liefern. Ähnliche Operandenklassenmultiplexer liefern die einzelnen Quellenoperanden für die Ausgabepositionen 3, 4 und 5 (in Fig. 8 nicht gezeigt).

[0107] Die Steuereinheit **130** empfängt Angaben, welche Ergebnisse gültig sind, über die Ergebnisgültigkeitsleitungen **118** und empfängt Angaben der reiheninternen Abhängigkeiten auf dem Abhängigkeitsbus **92**. Wenn eine Abhängigkeit für einen speziellen Quellenoperanden in einer speziellen Ausgabeposition über den Abhängigkeitsbus **92** angegeben wird und das Ergebnis gültig ist, wie dies durch die Ergebnisgültigkeitsleitung **118** für die Ausgabeposition angegeben wird, von der der Quellenoperand abhängig ist, steuert die Steuereinheit **130** einen entsprechenden Operandenzusammenführmultiplexer **132** an, um das Ergebnis von dem entsprechenden Vorausschauergebnisbus auszuwählen. Wenn andererseits eine Abhängigkeit angegeben ist, dass das entsprechende Ergebnis nicht gültig ist, dann steuert die Steuereinheit **130** einen entsprechenden Operandenzusammenführmultiplexer **132** an, um die Zielmarke der Ausgabeposition, von der der Quellenoperand abhängig ist, auszuwählen. Die Steuereinheit **130** empfängt ferner die Vergleichssignale auf den Vergleichsleitungen **96**, die anzeigen, dass ein Vergleich/Verzweigung erkannt wurde. Wenn eine Vergleich/Verzweigungskombination erkannt wurde, wählt die Steuereinheit **130** die Zukunftsdateiausgabe für die Quellenoperanden der vorhergehenden Ausgabeposition für die Operandenzusammenführmultiplexer **132** der Ausgabe-

position, die den Verzweigungsbefehl enthält, aus. Auf diese Weise können die Quellenoperanden des Vergleichsbefehls in der Ausgabeposition, die den Verzweigungsbefehl enthält, bereitgestellt werden. Nachfolgend kann die empfangende Funktionseinheit sowohl den Vergleich (unter Anwendung der Vergleichsquellenoperanden) ausführen und den Verzweigungsbefehl als genommen oder nicht ausgeführt bestimmen, in Abhängigkeit des Ergebnisses des Befehls. Wenn keine Abhängigkeit für einen speziellen Quellenoperanden einer speziellen Ausgabeposition angegeben ist und die Ausgabeposition nicht der Verzweigungsbereich einer Vergleichs/Verzweigungskombination ist, kann die Steuereinheit **130** den entsprechenden Operandenzusammenführmultiplexer **132** ansteuern, die Ausgabe der Zukunftsdatei für diesen Quellenoperanden in dieser Ausgabeposition auszuwählen (der eine Ergebniswarteschlangenmarke oder ein gültiger Quellenoperand sein kann).

[0108] Eine zusätzliche Prüfung, die die Steuereinheit **130** ausführen kann, besteht darin zu bestimmen, ob ein spezieller Quellenoperand, der in einer speziellen Ausgabeposition als von einer vorhergehenden Ausgabeposition abhängig angegeben ist, für die ein Ergebnis als von der Vorausschauadressen/Ergebnisberechnungseinheit **74** als gültig angegeben ist. Wenn die vorhergehende Ausgabeposition eine reiheninterne Quellenabhängigkeit aufweist, die angegeben ist, dann ist das Ergebnis, das von der Vorausschauadressen/Ergebnisberechnungseinheit **74** geliefert wird, tatsächlich unzulässig (da dieses auf einem nicht korrekten Quellenoperanden beruht). In derartigen Fällen kann die Steuereinheit **130** die Auswahl des vorausschauenden Ergebnisses, das von der Vorauschauadressen/Ergebnisberechnungseinheit **74** geliefert wird, verhindern und kann stattdessen die Zielmarke, die der vorhergehenden Ausgabeposition entspricht, auswählen. In einer Ausführungsform kann die Steuereinheit **130** die Ergebnisgültigkeitssignale mit den entsprechenden Abhängigkeitsangaben auf dem Abhängigkeitsbus **92** so maskieren bzw. verknüpfen, dass die maskierten Ergebnisgültigkeitssignale zurückgesetzt werden, wenn eine Abhängigkeit in der entsprechenden Ausgabeposition angegeben ist. Zu beachten ist, dass die Zukunftsdatei **26** und die Lade/Schreibeinheit **36** die vorausschauenden Ergebnisse/Adressen in gleicher Weise als unzulässig erklären können.

[0109] Die Steuereinheit **130** kann ferner ausgebildet sein, die Befehlsfenster **30** über den Verhinderungsbus **134** anzusprechen. Für jede Befehlsoperation, die von der Vorausschauadressen/Ergebnisberechnungseinheit **74** abgeschlossen wird (entweder Adressenerzeugung oder Funktionsoperation), kann die Steuereinheit **130** die entsprechende Operation in den Befehlsfenstern **30** verhindern, so dass die Befehlsoperation nicht für das Auswählen mittels der Funktionseinheiten **32** oder den Adressenerzeugungseinheiten **34** ausgewählt wird. Beispielsweise kann der Verhinderungsbus **134** ein Adressenerzeugungsverhinderungssignal und ein Funktionsoperationsverhinderungssignal für jede Ausgabeposition enthalten. Die Steuereinheit **130** kann das Adressenerzeugungsverhinderungssignal für eine spezielle Ausgabeposition aktivieren, wenn die Vorausschauadressen/Ergebnisberechnungseinheit **74** erfolgreich die vorrausschauende Adresse für diese speziellen Ausgabepositionen erzeugt hat (einschließlich der Auswirkungen von reiheninternen Abhängigkeiten). Andererseits kann die Steuereinheit **130** das Funktionsoperationsverhinderungssignal aktivieren, wenn die Vorausschauadressen/Ergebnisberechnungseinheit **74** erfolgreich das vorrausschauende Ergebnis entsprechend jener speziellen Ausgabeposition (einschließlich der Auswirkungen von reiheninternen Abhängigkeiten) erzeugt hat. Ferner kann die Steuereinheit **130** das Funktionsoperationsverhinderungssignal aktivieren, wenn die spezielle Ausgabeposition den Vergleichsbereich einer Vergleichs/Verzweigungskombination enthält. Wie zuvor erläutert ist, können arithmetische Operationen eine Flaggen- bzw. Markierungserzeugung zusätzlich zu der funktionalen Operation des arithmetischen Befehls enthalten. Derartige Befehle werden nicht verhindert, um damit die Erzeugung des Markierungsergebnisses zu ermöglichen, obwohl das funktionale Ergebnis bereits erzeugt ist. Alternativ wird das funktionale Ergebnis nicht verhindert, um das Weiterleiten zu Befehlen zu ermöglichen, die die Zukunftsdatei **26** ausgelesen haben, bevor die Zukunftsdatei **26** mit dem vorausschauenden Ergebnis aktualisiert wurde. Ansonsten kann eine lokale Weiterleitung innerhalb der Pipeline-Stufen zwischen dem Lesen der Zukunftsdatei und dem Bereitstellen der vorausschauenden Ergebnisse ausgeführt werden. Die Steuereinheit **130** empfängt die Adressengültigkeitssignale, um die Adressenerzeugungsverhinderungssignale zu erzeugen. Ferner Kann die Steuereinheit **130** ausgebildet sein, eine funktionale Operation nicht zu verhindern, wenn der entsprechende Befehl die Bedingungscodierungen sowie ein Ergebnisregister aktualisiert. Die Funktionsoperation kann dabei auf eine Funktionseinheit zur Berechnung der Bedingungscodierungen übertragen werden.

[0110] Fig. 9 ist ein Flussdiagramm, das die Arbeitsweise einer Ausführungsform einer Steuereinheit **130** zum Auswählen eines speziellen Quellenoperanden für eine spezielle Ausgabeposition zeigt. Andere Ausführungsformen sind möglich und hierin mit berücksichtigt. Die in Fig. 9 gezeigten Schritte können parallel für jeden Quellenoperanden ausgeführt werden. Die in Fig. 9 gezeigten Schritte sind in einer speziellen Reihefolge zum einfacheren Verständnis dargestellt, können jedoch in einer anderen geeigneten Reihenfolge ausgeführt werden. Ferner können die Schritte nach Bedarf in der Steuereinheit **130** parallel ausgeführt werden.

[0111] Die Steuereinheit **130** bestimmt, ob es eine reiheninterne Abhängigkeit gibt, die für den speziellen Quellenoperanden angegeben ist (Entscheidungsblock **140**). Wenn eine reiheninterne Abhängigkeit angegeben ist, bestimmt die Steuereinheit **130**, ob die Vorausschauadressen/Ergebnisberechnungseinheit **74** ein vorausschauendes Ergebnis für die frühere Ausgabeposition, von der der spezielle Quellenoperand abhängig ist, erfolgreich erzeugt wurde (Entscheidungsblock **142**). Wenn ein vorrausschauendes Ergebnis nicht erfolgreich erzeugt wurde, wählt die Steuereinheit **130** die Zielmarke entsprechend der vorhergehenden Ausgabeposition aus (Schritt **144**). Wenn ferner das Ergebnis in der vorhergehenden Ausgabeposition gültig ist, aber eine reiheninterne Abhängigkeit für einen oder mehreren der Quellenoperanden für die vorhergehende Ausgabeposition angegeben ist (Entscheidungsblock **146**), wählt die Steuereinheit **130** die Zielmarkierung entsprechend der vorhergehenden Ausgabeposition ebenso aus (Schritt **144**). Wenn keine Abhängigkeit für die Quellenoperanden der vorhergehenden Ausgabeposition angegeben ist und das Ergebnis gültig ist, wählt die Steuereinheit **130** das Ergebnis, das von der Vorausschauadressen/Ergebnisberechnungseinheit **74** bereitgestellt wird (Schritt **148**).

[0112] Wenn keine reiheninterne Abhängigkeit für den speziellen Quellenoperanden angegeben ist (Entscheidungsblock **140**), bestimmt die Steuereinheit **130**, ob eine Vergleichs/Verzweigungskombination erkannt wurde, für die die spezielle Ausgabeposition der Verzweigungsbereich ist (Entscheidungsblock **150**). Wenn die Vergleichs/Verzweigungskombination erkannt wurde, wird der Quellenoperand von der vorhergehenden Ausgabeposition (d. h. der Ausgabeposition, die den Vergleichsbefehl enthält) durch die Steuereinheit **130** ausgewählt (Schritt **152**). Wenn andererseits die Vergleichs/Verzweigungskombination nicht erkannt wurde, wählt die Steuereinheit **130** die Ausgabe der Zukunftsdatei für den speziellen Quellenoperanden (Schritt **154**).

[0113] Zu beachten ist, obwohl diverse Merkmale als Multiplexer dargestellt sind, eine beliebige parallele oder serielle Kombination der Auswahllogik verwendet werden kann, um die beschriebenen Auswahlvorgänge auszuführen. Ferner ist zu beachten, dass, obwohl die vorliegende Erfindung das Zusammenführen von reiheninternen Ergebnissen und einer Vergleichs/Verzweigungskombination liefert, auch Ausführungsformen betrachtet werden, die lediglich eines dieser Merkmale bereitstellen. Ferner ist zu beachten, dass obwohl eine Zukunftsdatei als eine Quelle für Operanden in der vorliegenden Ausführungsform verwendet wird, in anderen Ausführungsformen anders spekulative Speicherelemente (etwa Umordnungspuffer, Umbenennungsregisterdateien, etc. als Quellen für Operanden verwendet werden können.

[0114] Fig. 10 ist eine Blockansicht einer Ausführungsform eines Computersystems **200** mit einem Prozessor **10**, der mit einer Vielzahl von Systemkomponenten über eine Busbrücke **202** verbunden ist. Andere Ausführungsformen sind möglich und hierin mit eingeschlossen. In dem dargestellten System ist ein Hauptspeicher **204** mit der Busbrücke **202** über einen Speicherbus **206** verbunden, und eine Graphiksteuerung **208** ist mit der Busbrücke **202** über einen AGP-Bus **210** verbunden. Schließlich sind mehrere PCI-Einrichtungen **212a** bis **212b** mit der Busbrücke **202** über einen PCI-Bus **214** verbunden. Eine zweite Busbrücke **216** kann ferner vorgesehen sein, um eine elektrische Schnittstelle zu einer oder mehreren ELSA- oder ISA-Einrichtung **218** über einen EISA/ISA-Bus **220** bereitzustellen. Der Prozessor **10** ist mit der Busbrücke **202** über die Busschnittstelle **46** verbunden.

[0115] Die Busbrücke **202** stellt eine Schnittstelle zwischen dem Prozessor **10**, dem Hauptspeicher **204**, der Graphiksteuerung **208** und Einrichtungen bereit, die mit PCI-Bus **214** verbunden sind. Wenn eine Operation von einer der Einrichtungen, die mit Busbrücke **202** verbunden sind, empfangen sind, erkennt die Busbrücke **202** das Ziel der Operation (beispielsweise eine spezielle Einrichtung oder im Falle des PCI-Busses **214**, dass das Ziel auf dem PCI-Bus **214** liegt). Die Busbrücke leitet die Operation an die Zieleinrichtung weiter. Die Busbrücke **202** übersetzt im Wesentlichen eine Operation aus dem Protokoll, das von der Quelleneinrichtung oder dem Bus verwendet wird, in ein Protokoll, das von der Zieleinrchtung oder dem Zielbus verwendet wird.

[0116] Zusätzlich zu dem Bereitstellen einer Schnittstelle mit einem ISA/EISA-Bus für den PCI-Bus **214** kann ferner die zweite Busbrücke **216** nach Bedarf weitere Funktionen beinhalten. Beispielsweise umfasst in einer Ausführungsform die zweite Busbrücke **216** einen Master-PCI-Verteiler (nicht gezeigt) zum Verteilen der „Besitzrechte" an dem PCI-Bus **214**. Eine Eingabe/Ausgabe-Steuenang (nicht gezeigt), die extern zu der zweiten Busbrücke **216** ist oder integriert in dieser vorgesehen sein kann, kann ebenso in dem Computersystem **200** enthalten sein, um eine Funktionsunterstützung für eine Tastatur oder Maus **222** und für diverse serielle und parallele Anschlüsse nach Bedarf bereitzustellen. Eine externe Cache-Speichereinheit (nicht gezeigt), kann ferner mit der Busschnittstelle **46** und zwischen dem Prozessor **10** und der Busbrücke **202** in werteren Ausführungsformen angeschlossen sein. Alternativ kann der externe Cache-Speicher mit der Busbrücke **202** verbunden sein und die Cache-Steuerungslogik für den externen Cache-Speicher kann in der Busbrücke **202** integriert sein.

[0117] Der Hauptspeicher **204** ist ein Speicher, in welchem Anwendungsprogramme abgelegt sind und aus dem der Prozessor **10** im Wesentlichen heraus operiert. Ein geeigneter Hauptspeicher **204** umfasst einen DRAM (dynamischen Speicher mit wahlfreiem Zugriff) und vorzugsweise mehrere Bänke aus SDRAM's (synchrone DRAM).

[0118] Die PCI-Einrichtungen **212a** bis **212b** repräsentieren eine Vielzahl peripherer Einrichtungen, etwa beispielsweise Netzwerkschnittstellenkarten, Videobeschleuniger, Klangkarten, Festplatten oder Laufwerkssteuerungen, SCSI-(Kleincomputersystemschnittstellen-) Adapter und Telefonkarten. In ähnliche Weise ist die ISA-Einrichtung **218** anschaulich für diverse Arten von peripheren Einrichtungen, etwa ein Modem, eine Klangkarte, und eine Vielzahl von Datennahmekarten, etwa GPIB- oder Feldbusschnittstellenkarten.

[0119] Die Graphiksteuerung **208** ist vorgesehen, um das Darstellen von Text und Bildern auf einer Anzeige **226** zu steuern. Die Graphiksteuerung **208** kann einen typischen Graphikbeschleuniger repräsentieren, wie er im Allgemeinen im Stand der Technik bekannt ist, um dreidimensionale Datenstrukturen zu erzeugen, die in effektiver Weise in den Hauptspeicher **204** verschoben und aus diesem ausgelesen werden können. Die Graphiksteuerung **208** kann daher eine Befehlseinheit bzw. ein Master das AGP-Busses **210** sein, indem diese Zugriff auf eine Zielschnittstelle innerhalb der Busbrücke **202** anfordert und erhält, um damit Zugriff auf den Hauptspeicher **204** zu erhalten. Ein zugeordneter Graphikbus übernimmt die schnelle Abrufung von Daten aus dem Hauptspeicher **204**. Für gewisse Operationen kann die Graphiksteuerung **208** so gestaltet sein, dass PCI-Protokolltransaktionen auf dem AGP-Bus **210** erzeugt werden. Die AGP-Schnittstelle der Busbrücke **202** kann damit Funktionen enthalten, um sowohl AGP-Protokolltransaktionen sowie PCI-Protokollziel- und Initiatortransaktionen zu unterstützen. Die Anzeige **226** ist eine beliebige elektronische Anzeige, auf der Bild oder Text dargestellt werden kann. Zu einer geeigneten Anzeige **226** gehört eine Kathodenstrahlröhre („CRT"), eine Flüssigkristallanzeige („LCD"), etc.

[0120] Zu beachten ist, dass obwohl die AGP-, PCI- und ISA- oder EISA-Busse als Beispiele in der obigen Beschreibung angeführt sind, andere Busarchitekturen bei Bedarf verwendet werden können. Ferner ist zu beachten, dass das Computersystem **200** ein Multiprozessorcomputersystem mit mehreren Prozessoren (beispielsweise mit dem Prozessor **10a**, der in dem Computersystem **200** als optionale Komponente gezeigt ist) sein kann. Der Prozessor **10a** kann ähnlich zu dem Prozessor **10** ausgebildet sein. Insbesondere kann der Prozessor **10a** eine identische Kopie des Prozessors **10** sein. Der Prozessor **10a** kann die Busschnittstelle **46** mit dem Prozessor **10** gemeinsam nutzen (wie dies in Fig. 10 gezeigt ist) oder kann mit der Busbrücke **202** über einen unabhängigen Bus verbunden sein.

[0121] Gemäß der obigen Darstellung wird ein Prozessor gezeigt, der versucht, vorrausschauende Adressen und/oder Ergebnisse vor dem Ausführen der Befehle zu erzeugen. Reiheninterne Abhängigkeiten werden angegeben und die vorrausschauenden Ergebnisse werden in die Quellenoperanden der abhängigen Befehle zusammengeführt. Das Weiterleiten kann effizienter sein, indem die vorrausschauenden Ergebnisse früher bereitgestellt werden (wodurch eine Weiterleitung über eine Zukunftsdatei und den Zusammenführungsmechanismus bereit gestellt wird). Des weiteren ist es möglich, weniger Funktionseinheiten vorzusehen, da weniger Befehlsoperationen in den Funktionseinheiten ausführen sind.

[0122] Diverse Variationen und Modifizierungen werden für den Fachmann bei Würdigung der obigen Beschreibung offenkundig. Es ist daher beabsichtigt, dass die folgenden Patentansprüche so interpretiert werden, dass alle derartigen Variationen und Modifizierungen enthalten sind.

Patentansprüche

1. Prozessor mit:

einer reiheninternen Abhängigkeitsprüfeinheit (**75**), die angeschlossen ist, mehrere Operandenspezifizierungen entsprechend einer Reihe aus Instruktionen zu empfangen, wobei die reiheninterne Abhängigkeitsprüfeinheit ausgebildet ist, Abhängigkeiten zwischen Instruktionen in der Reihe aus Instruktionen durch Vergleichen der mehreren Operandenspezifizierungen zu bestimmen;

einer Vorschauberechnungseinheit (**74**), die angeschlossen ist, einen oder mehrere Operanden zu empfangen, die von den mehreren Operandenspezifizierungen spezifiziert sind, wobei die Vorschauberechnungseinheit ausgebildet ist, ein vorrausschauendes Ergebnis entsprechend einer Instruktion in der Reihe aus Instruktionen zu berechnen, wenn jeder von der ersten Instruktion zum Erzeugen des vorausschauenden Ergebnisses verwendete Operand in dem einen oder den mehreren Operanden enthalten ist;

einer Operandenzusammenfasseinheit (**78**), die mit der Vorschauberechnungseinheit und der reiheninternen Abhängigkeitsprüfeinheit verbunden ist, wobei die Operandenzusammenfasseinheit ausgebildet ist, das vor-

ausschauende Ergebnis als einen Operanden einer zweiten Instruktion in der Reihe aus Instruktionen bereitzustellen in Reaktion auf: (i) eine Angabe von der Vorausschauberechnungseinheit, dass das vorausschauende Ergebnis gültig ist, und (ii) eine Angabe von der reiheninternen Abhängigkeitsprüfeinheit, dass die zweite Instruktion von der ersten Instruktion abhängig ist; und
einem oder mehreren Instruktionsfenstern (**30**), die mit der Operandenzusammenfasseinheit gekoppelt sind, wobei die ein oder die mehreren Instruktionsfenster ausgebildet sind, Instruktionen zu speichern bis entsprechende Operanden bereitgestellt sind und nachfolgend die Instruktion zum Ausführen auszuwählen, wobei die Operandenzusammenfasseinheit ausgebildet ist, dem einen oder den mehreren Instruktionsfenstern anzuzeigen, die Ausführung mindestens einer ersten Instruktionsoperation der ersten Instruktion, die von dem vorausschauenden Ergebnis repräsentiert ist, zu unterbinden, wenn das vorausschauende Ergebnis gültig ist.

2. Prozessor nach Anspruch 1, wobei die Vorausschauberechnungseinheit (**74**) ausgebildet ist, das vorausschauende Ergebnis zu berechnen, wenn die Instruktion eine Verschiebungsinstruktion mit nur Registeroperanden ist, und wobei die Verschiebungsinstruktion die erste Instruktionsoperation umfasst, die von dem einen oder den mehreren Instruktionsfenstern (**30**) unterbunden wurde.

3. Prozessor nach Anspruch 1, wobei die Vorausschauberechnungseinheit (**74**) ausgebildet ist, das vorausschauende Ergebnis zu berechnen, wenn die Instruktion eine Additionsinstruktion mit nur Registerquellenoperanden ist.

4. Prozessor nach Anspruch 3, wobei die Additionsinstruktion dem einen oder den mehreren Instruktionsfenstern (**30**) für die nachfolgende Ausführung bereitgestellt wird, um ein Markierungsergebnis zu erzeugen, und wobei die erste Instruktionsoperation die Additionsoperation umfasst.

5. Prozessor nach Anspruch 3, wobei die Vorausschauberechnungseinheit (**74**) ausgebildet ist, eine Vorausschauadresse zu berechnen, wenn die erste Instruktion einen Speicheroperanden aufweist und der eine oder die mehreren Operanden eine oder mehrere Adressenoperanden entsprechend der ersten Instruktion aufweisen, und wobei die erste Instruktionsoperation die Adressenerzeugung umfasst.

6. Prozessor nach Anspruch 1, der ferner eine Funktionseinheit (**32**) aufweist, die mit dem einen oder den mehreren Instruktionsfenstern verbunden ist, wobei die Funktionseinheit ausgebildet ist, die von dem einen oder den mehreren Instruktionsfenstern zugeleitete Instruktionen auszuführen.

7. Prozessor nach Anspruch 6, wobei die Funktionseinheit einen Teil der ersten Instruktion, der durch das vorausschauende Ergebnis repräsentiert ist, nicht ausführt, wenn das vorausschauende Ergebnis gültig ist.

8. Prozessor nach Anspruch 7, der ferner umfasst: eine Zukunftsdatei (**26**), die mit der Vorausschauberechnungseinheit verbunden ist, wobei die Zukunftsdatei ausgebildet ist, den einen oder die mehreren Operanden der Vorausschauberechnungseinheit in Reaktion auf die mehreren Operandenspezifzierungen zuzuführen, wobei die Zukunftsdatei angeschlossen ist, das vorausschauende Ergebnis zu empfangen und einen Zieloperanden entsprechend dem vorausschauenden Ergebnis in Reaktion darauf zu aktualisieren.

9. Prozessor nach Anspruch 1, wobei die Operandenzusammenfasseinheit (**78**) ausgebildet ist, Operanden entsprechend der ersten Instruktion als Operanden der zweiten Instruktion bereitzustellen, wenn die erste Instruktion eine Vergleichsinstruktion ist, wobei die zweite Instruktion eine bedingte Verzweigungsinstruktion ist und die zweite Instruktion auf die erste Instruktion folgt.

10. Prozessor nach Anspruch 1, wobei die Operandenzusammenfasseinheit ferner ausgebildet ist, ein Empfangen der Vergleichsinstruktion in dem einen oder den mehreren Instruktionsfenstern zu verhindern.

11. Prozessor mit:
einer Operandenzusammenfasseinheit (**78**), die angeschlossen ist, mehrere Operanden entsprechend einer Reihe aus Instruktionen zu empfangen, wobei die Operandenzusammenfasseinheit ausgebildet ist, einen oder mehrere der mehreren Operanden entsprechend einer ersten Instruktion in der Reihe aus Instruktionen als Operanden einer zweiten Instruktion in der Reihe aus Instruktionen bereitzustellen in Reaktion darauf, dass: (i) die erste Instruktion eine Vergleichsinstruktion ist, (ii) die zweite Instruktion eine bedingte Verzweigungsinstruktion ist, und (iii) die zweite Instruktion auf die erste Instruktion folgt; und
einem oder mehreren Instruktionsfenstern (**30**), die mit der Operandenzusammenfasseinheit verbunden sind, wobei das eine oder die mehreren Instruktionsfenster ausgebildet sind, Instruktionen zu speichern, bis entsprechende Operanden bereitgestellt sind, und nachfolgend die Instruktionen für die Abarbeitung auszuwählen;

wobei die Operandenzusammenfasseinheit ausgebildet ist, dem einen oder den mehreren Instruktionsfenstern anzuzeigen, den Empfang der ersten Instruktion zu verhindern, wenn: (i) die erste Instruktion eine Vergleichsinstruktion ist, (ii) die zweite Instruktion eine bedingte Verzweigungsinstruktion ist und (iii) die zweite Instruktion auf die erste Instruktion folgt.

12. Verfahren zum Ausführen einer Reihe aus Instruktionen in einem Prozessor, wobei das Verfahren umfasst:
Decodieren der Reihe aus Instruktionen, um mehrere Operandenspezifizierungen zu erkennen;
Vergleichen von Zieloperandenspezifizierungen jeder Instruktion in der Reihe aus Instruktionen mit Quellenoperandenspezifizierungen jeder nachfolgenden Instruktion in der Reihe aus Instruktionen, um reiheninterne Abhängigkeiten zu erkennen;
Auslesen einer spekulativen Operandenquelle, um Quellenoperanden davon zu erfassen, die von den Quellenoperandenspezifizierungen spezifiziert sind;
Berechnen eines vorausschauenden Ergebnisses für eine erste Instruktion in der Reihe, wenn die Quellenoperanden in der spekulativen Operandenquelle verfügbar sind;
einem Instruktionsfenster, das ausgebildet ist, die erste Instruktion zu empfangen, anzeigen, das Ausführen zumindest einer ersten Instruktionsoperation der ersten Instruktion zu verhindern, die das vorausschauende Ergebnis erzeugt, wenn das vorausschauende Ergebnis erfolgreich berechnet ist; und
Bereitstellen des vorausschauenden Ergebnisses für eine zweite Instruktion in der Reihe aus Instruktionen, wenn das Vergleichen eine Abhängigkeit der zweiten Instruktion von der ersten Instruktion angibt, wobei das Bereitstellen vor dem Speichern der zweiten Instruktion in dem Instruktionsfenster ausgeführt wird.

13. Verfahren nach Anspruch 12, das ferner Auswählen der zweiten Instruktion aus dem Instruktionsfenster zur Ausführung umfasst.

14. Verfahren nach Anspruch 12, wobei das Berechnen erfolgreich ist, wenn die Instruktion eine Additionsinstruktion mit nur Registeroperanden ist und jeder der Registeroperanden in dem spekulativen Operandenspeicher beim Lesen verfügbar ist.

15. Verfahren nach Anspruch 14, das ferner umfasst: Berechnen einer Vorausschauadresse, wenn die erste Instruktion einen Speicheroperanden enthält und Adressenoperanden entsprechend der ersten Instruktion in der spekulativen Operandenquelle beim Lesen verfügbar sind.

Es folgen 9 Blatt Zeichnungen

Anhängende Zeichnungen

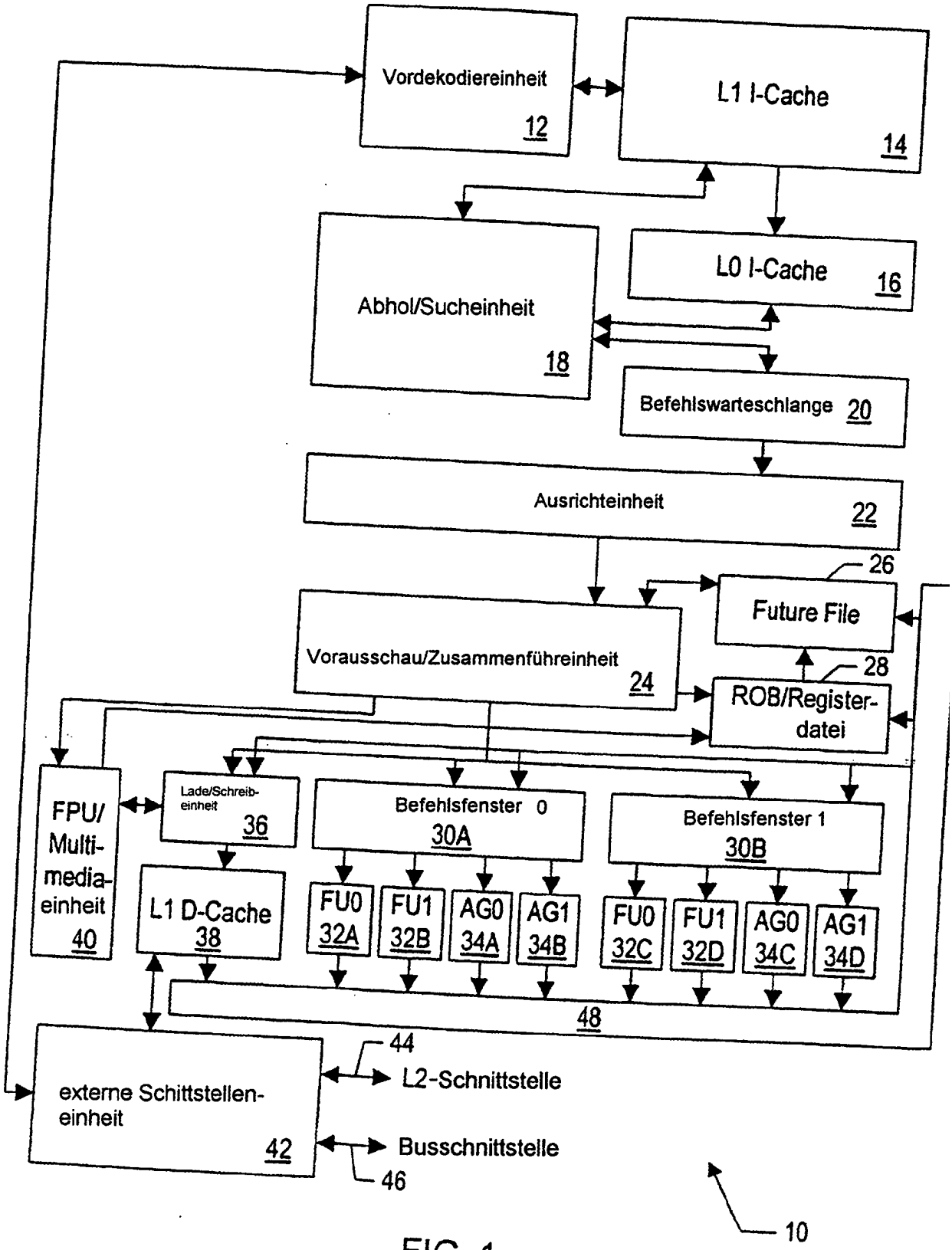

FIG. 1

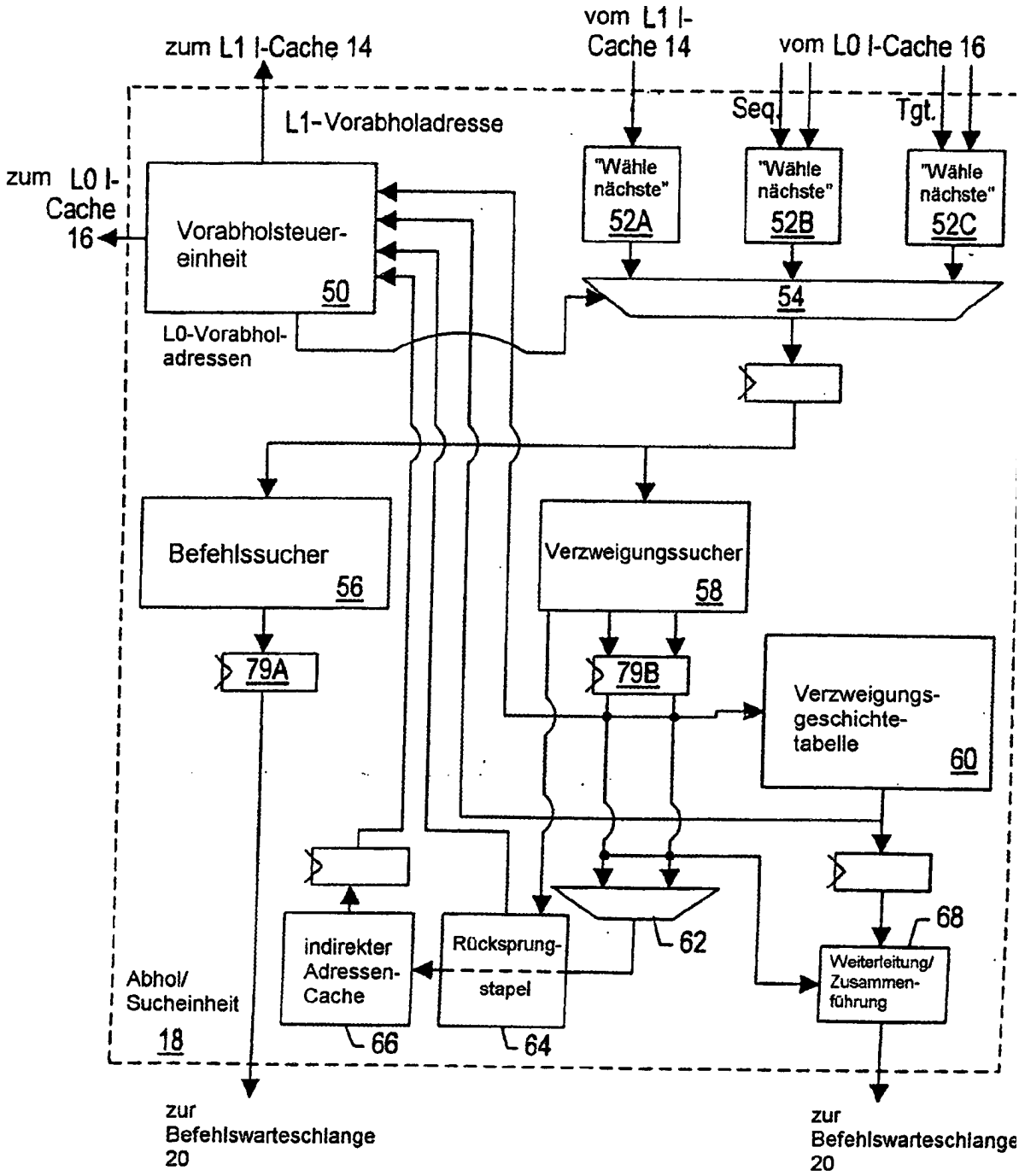
zum L1 I-Cache 14
vom L1 I-Cache 14
vom L0 I-Cache 16
L1-Vorabholadresse
Seq.
Tgt.
zum L0 I-Cache 16
Vorabholsteuer-einheit
50
"Wähle nächste"
52A
"Wähle nächste"
52B
"Wähle nächste"
52C
54
L0-Vorabhol-adressen
Befehlssucher
56
Verzweigungssucher
58
79A
79B
Verzweigungs-geschichte-tabelle
60
indirekter Adressen-Cache
66
Rücksprung-stapel
64
62
68
Weiterleitung/ Zusammen-führung
Abhol/ Sucheinheit
18
zur Befehlswarteschlange 20
zur Befehlswarteschlange 20

FIG. 2

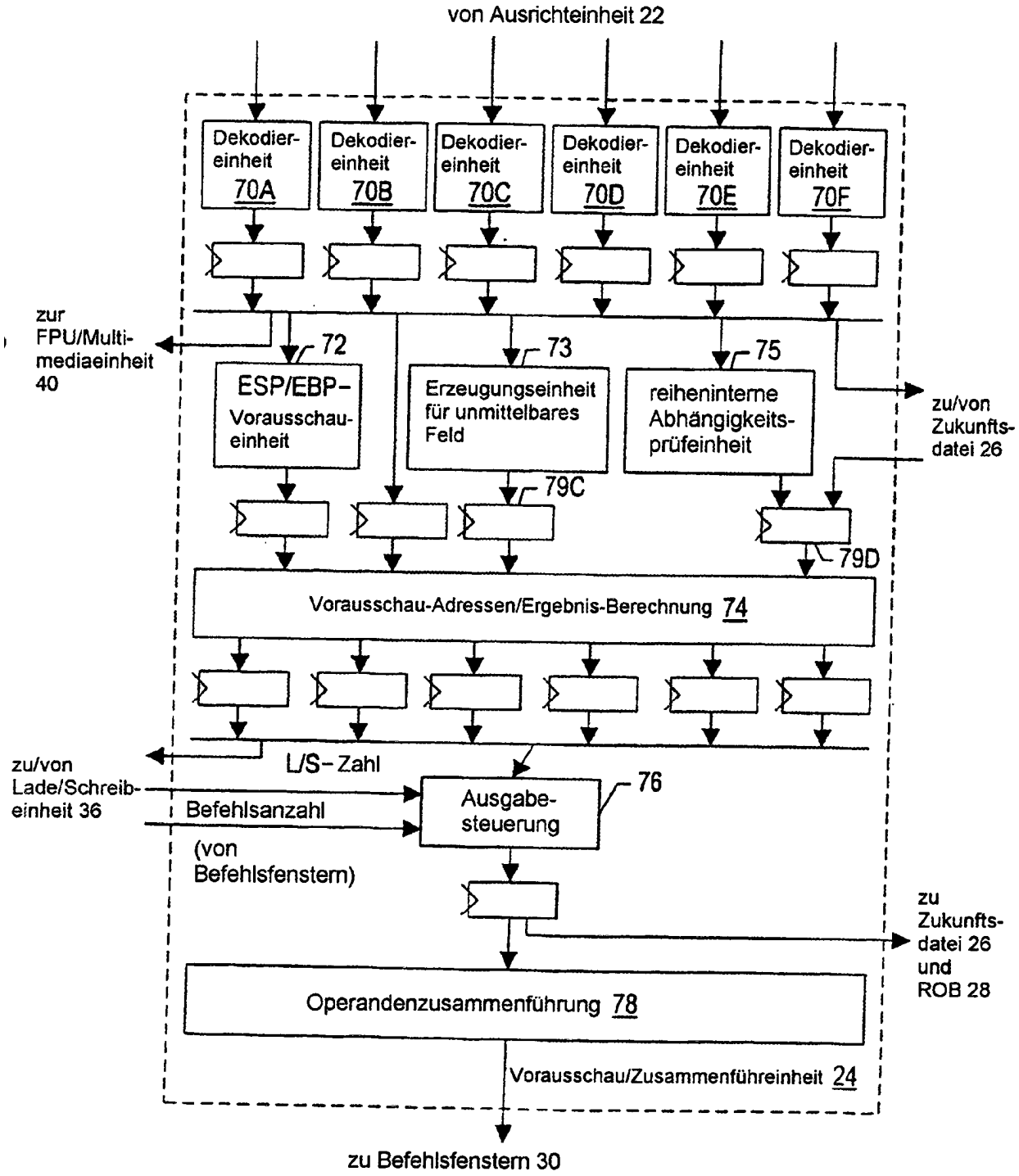
von Ausrichteinheit 22
Dekodier-einheit 70A
Dekodier-einheit 70B
Dekodier-einheit 70C
Dekodier-einheit 70D
Dekodier-einheit 70E
Dekodier-einheit 70F
zur FPU/Multi-mediaeinheit 40
72
ESP/EBP-Vorausschau-einheit
73
Erzeugungseinheit für unmittelbares Feld
75
reiheninterne Abhängigkeits-prüfeinheit
zu/von Zukunfts-datei 26
79C
79D
Vorausschau-Adressen/Ergebnis-Berechnung 74
L/S-Zahl
zu/von Lade/Schreib-einheit 36
Befehlsanzahl (von Befehlsfenstern)
Ausgabe-steuerung
76
zu Zukunfts-datei 26 und ROB 28
Operandenzusammenführung 78
Vorausschau/Zusammenführeinheit 24
zu Befehlsfenstern 30

FIG. 3

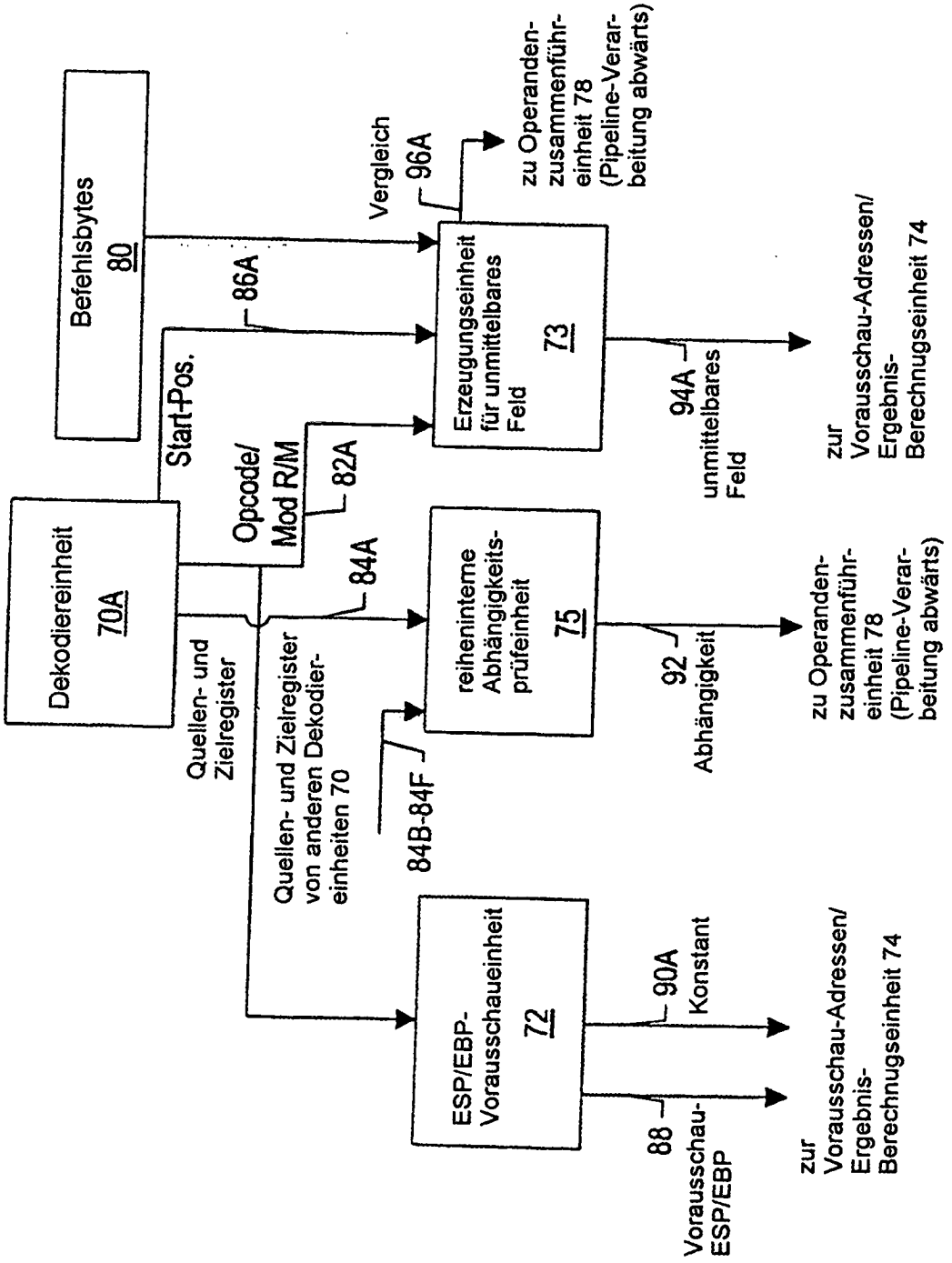
Befehlsbytes
80
Vergleich
96A
zu Operanden-zusammenführ-einheit 78 (Pipeline-Verar-beitung abwärts)
86A
Start-Pos.
Erzeugungseinheit für unmittelbares Feld
73
Opcode/ Mod R/M
82A
94A unmittelbares Feld
zur Vorausschau-Adressen/ Ergebnis-Berechnungseinheit 74
Dekodiereinheit
70A
84A
reihenineterne Abhängigkeits-prüfeinheit
75
Quellen- und Zielregister
Quellen- und Zielregister von anderen Dekodier-einheiten 70
84B-84F
92 Abhängigkeit
zu Operanden-zusammenführ-einheit 78 (Pipeline-Verar-beitung abwärts)
ESP/EBP-Vorausschaueinheit
72
90A Konstant
88 Vorausschau-ESP/EBP
zur Vorausschau-Adressen/ Ergebnis-Berechnungseinheit 74

FIG. 4

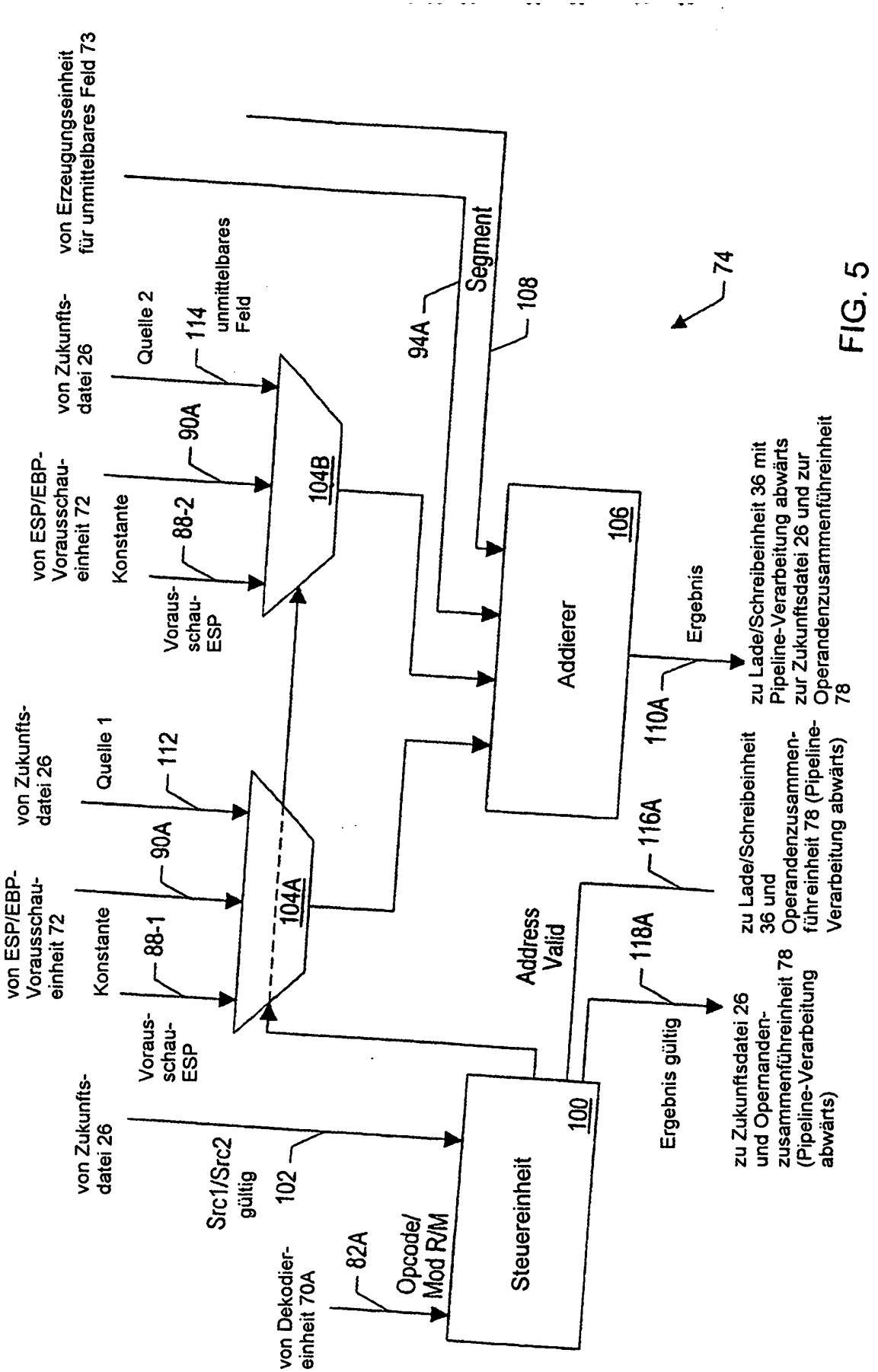
von Erzeugungseinheit für unmittelbares Feld 73
von Zukunfts-datei 26
Quelle 2
114
unmittelbares Feld
von ESP/EBP-Vorausschau-einheit 72
Konstante
90A
88-2
Voraus-schau-ESP
104B
94A
Segment
108
74
106
Addierer
Ergebnis
110A
zu Lade/Schreibeinheit 36 mit Pipeline-Verarbeitung abwärts zur Zukunftsdatei 26 und zur Operandenzusammenführeinheit 78
von Zukunfts-datei 26
Quelle 1
112
von ESP/EBP-Vorausschau-einheit 72
Konstante
90A
88-1
Voraus-schau-ESP
104A
Address Valid
116A
zu Lade/Schreibeinheit 36 und Operandenzusammen-führeinheit 78 (Pipeline-Verarbeitung abwärts)
118A
Ergebnis gültig
zu Zukunftsdatei 26 und Opernanden-zusammenführeinheit 78 (Pipeline-Verarbeitung abwärts)
von Zukunfts-datei 26
Src1/Src2 gültig
102
100
Steuereinheit
von Dekodier-einheit 70A
82A
Opcode/ Mod R/M

FIG. 5

Adressiermodus	Mux 1	Mux 2
Nur Verschiebung	0	0
Quelle 1 + optionale Verschiebung	Quelle 1	0
Quelle 1 + Quelle 2 +optionale Verschiebung	Quelle 2	Quelle 2
ESP + optionale Verschiebung	Vorausschau-ESP	Konstante
EBP + optionale Verschiebung	Konstante	Vorausschau-ESP

120

FIG. 6

Arithmetische Operation	Mux 1	Mux 2
Addieren/Subtra-hieren	Quelle 1	Quelle 2/0
Inkrement/ Dekrement	Quelle 1	0
Verschieben in Register	0	Quelle 2

122

FIG. 7

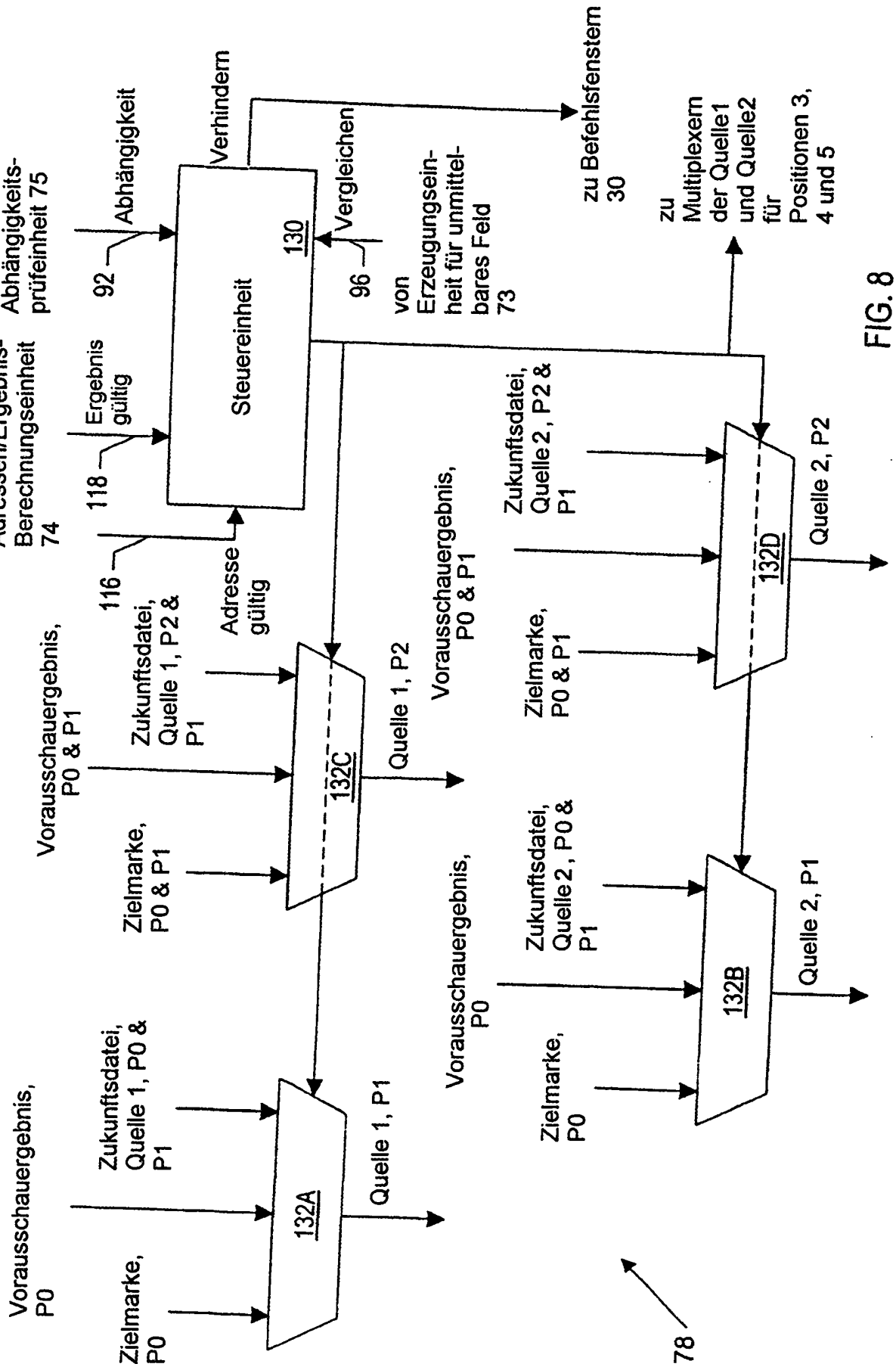
von reiheninterner Abhängigkeits-prüfeinheit 75
Abhängigkeit
92
Verhindern
zu Befehlsfenstern 30
Vergleichen
96
von Erzeugungsein-heit für unmittel-bares Feld 73
zu Multiplexern der Quelle1 und Quelle2 für Positionen 3, 4 und 5
von Vorausschau-Adressen/Ergebnis-Berechnungseinheit 74
Ergebnis gültig
118
Steuereinheit
130
116
Adresse gültig
Vorausschauergebnis, P0 & P1
Zukunftsdatei, Quelle 1, P2 & P1
Zielmarke, P0 & P1
132C
Quelle 1, P2
Vorausschauergebnis, P0 & P1
Zukunftsdatei, Quelle 2, P2 & P1
Zielmarke, P0 & P1
132D
Quelle 2, P2
Vorausschauergebnis, P0
Zukunftsdatei, Quelle 1, P0 & P1
Zielmarke, P0
132A
Quelle 1, P1
Vorausschauergebnis, P0
Zukunftsdatei, Quelle 2, P0 & P1
Zielmarke, P0
132B
Quelle 2, P1
78

FIG. 8

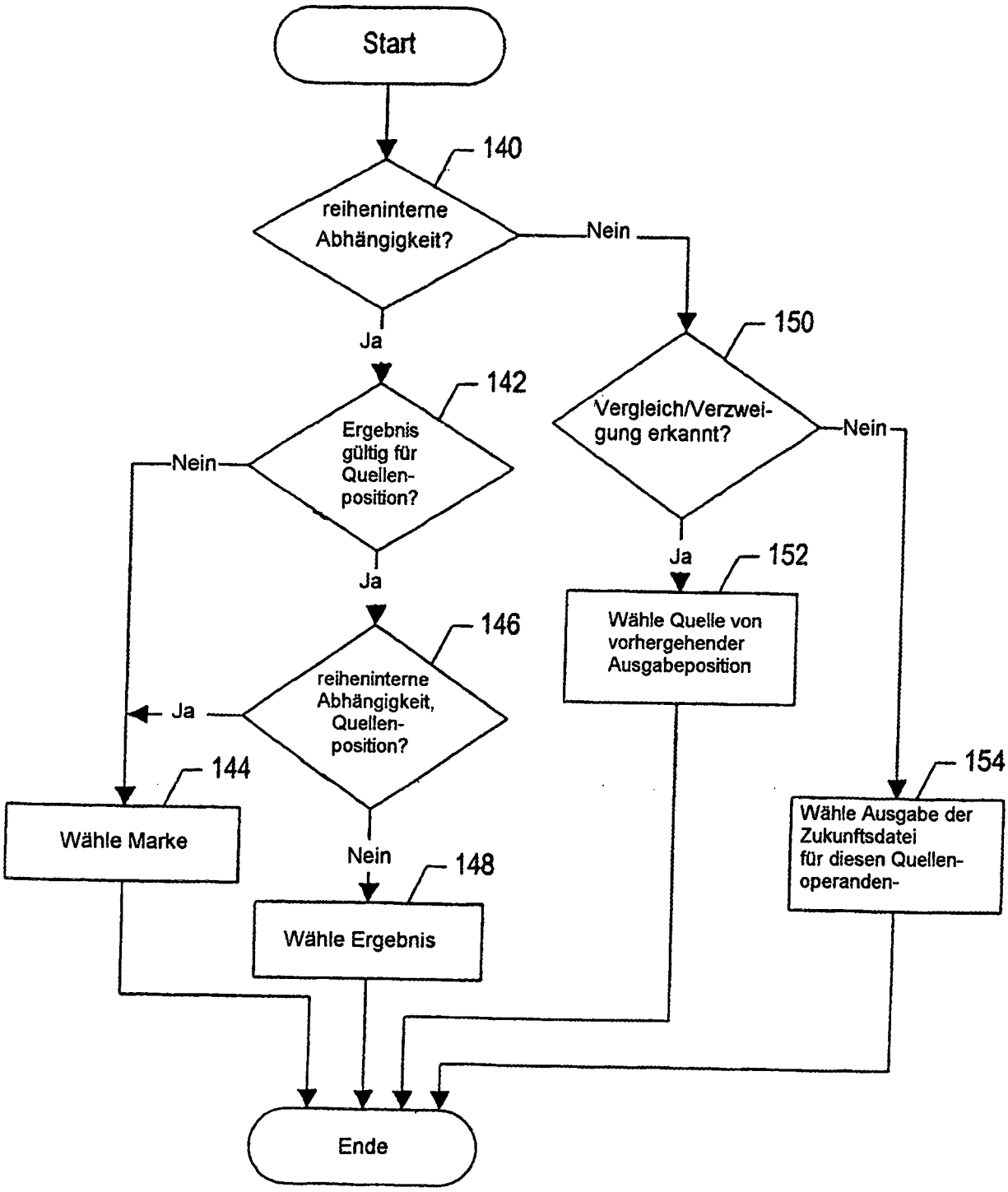
Start
140
reiheninterne Abhängigkeit?
Nein
Ja
150
Vergleich/Verzweigung erkannt?
Nein
Ja
142
Ergebnis gültig für Quellenposition?
Nein
Ja
152
Wähle Quelle von vorhergehender Ausgabeposition
146
reiheninterne Abhängigkeit, Quellenposition?
Ja
144
Wähle Marke
Nein
148
Wähle Ergebnis
154
Wähle Ausgabe der Zukunftsdatei für diesen Quellenoperanden-
Ende

FIG. 9

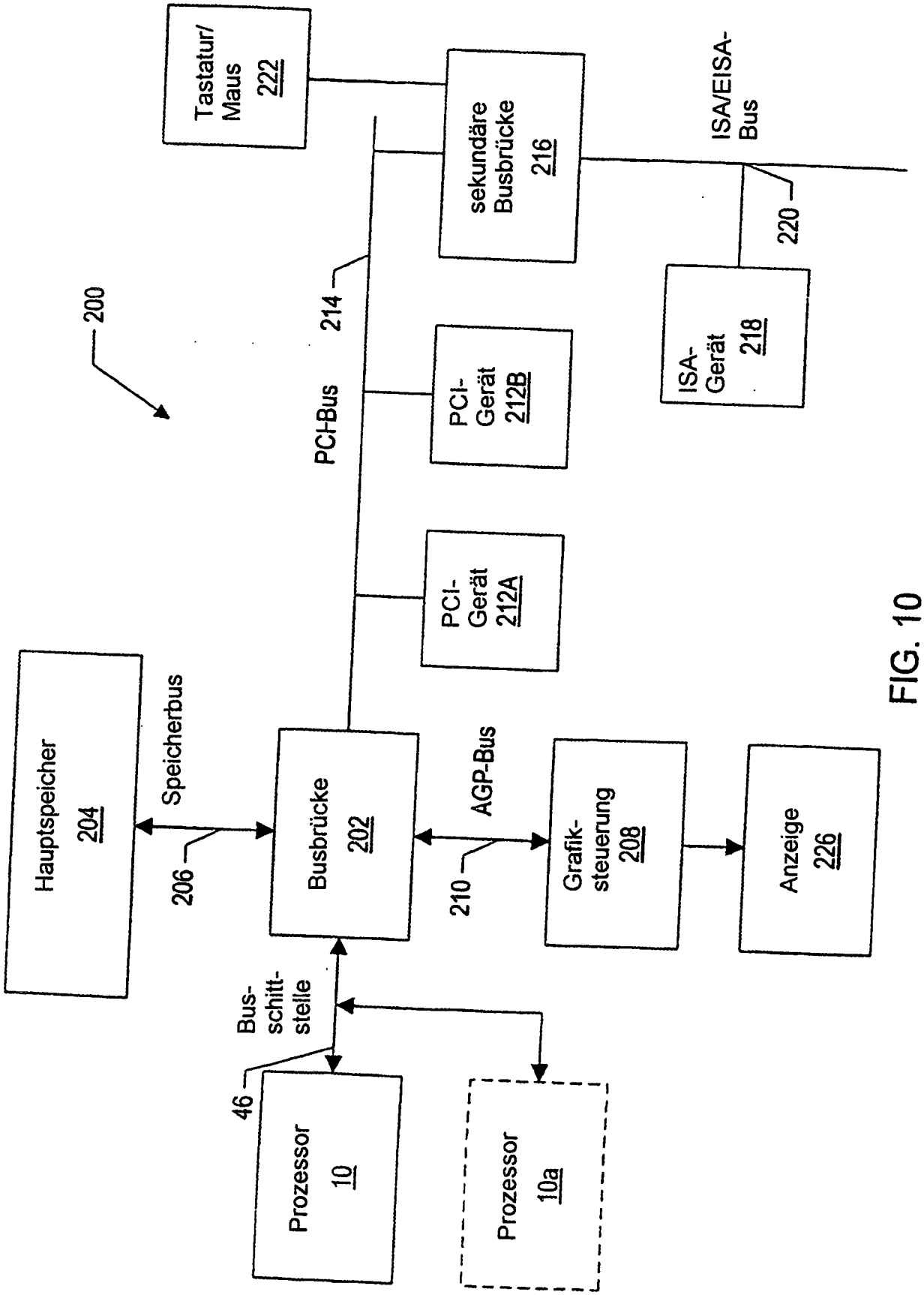
200
Tastatur/
Maus
222
sekundäre
Busbrücke
216
ISA/EISA-
Bus
220
214
PCI-Bus
PCI-
Gerät
212B
ISA-
Gerät
218
PCI-
Gerät
212A
Hauptspeicher
204
Speicherbus
206
Busbrücke
202
AGP-Bus
210
Grafik-
steuerung
208
Anzeige
226
Bus-
schnitt-
stelle
46
Prozessor
10
Prozessor
10a

FIG. 10