(19) Bundesrepublik Deutschland
Deutsches Patent- und Markenamt

(10) **DE 60 2004 010 265 T2** 2009.05.07

(12) # Übersetzung der europäischen Patentschrift

(97) **EP 1 644 823 B1**
(21) Deutsches Aktenzeichen: **60 2004 010 265.2**
(86) PCT-Aktenzeichen: **PCT/US2004/017096**
(96) Europäisches Aktenzeichen: **04 753 838.4**
(87) PCT-Veröffentlichungs-Nr.: **WO 2004/111839**
(86) PCT-Anmeldetag: **02.06.2004**
(87) Veröffentlichungstag der PCT-Anmeldung: **23.12.2004**
(97) Erstveröffentlichung durch das EPA: **12.04.2006**
(97) Veröffentlichungstag der Patenterteilung beim EPA: **21.11.2007**
(47) Veröffentlichungstag im Patentblatt: **07.05.2009**

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

(30) Unionspriorität:
458457 **10.06.2003** **US**

(84) Benannte Vertragsstaaten:
DE, FR, GB

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

(74) Vertreter:
Grünecker, Kinkeldey, Stockmair & Schwanhäusser, 80802 München

(72) Erfinder:
FILIPPO, Michael A., Manchaca, TX 78652, US;
PICKETT, James K., Austin, TX 78733, US;
SANDER, Benjamin T., Austin, TX 78735, US;
GOPAL, Rama S., Austin, TX 78759, US

(54) Bezeichnung: **LOAD-STORE-EINHEIT MIT WIEDERHOLUNGSMECHANISMUS**

Beschreibung

[0001] Diese Erfindung betrifft das Gebiet der Mikroprozessoren und betrifft insbesondere das Ausführen einer Datenspekulation in einem Mikroprozessor.

[0002] Superskalare Mikroprozessoren erreichen ein hohes Leistungsvermögen, indem mehrere Befehle gleichzeitig ausgeführt werden, und indem die kleinste mögliche Taktdauer oder der kürzeste Taktzyklus angewendet wird, der mit dem Aufbau verträglich ist. Jedoch begrenzen Datenabhängigkeit und Steuerablaufabhängigkeiten zwischen Befehlen die Möglichkeit, wie viele Befehle gleichzeitig ausgegeben werden können. Als Folge davon unterstützen einige Mikroprozessoren die spekulative Ausführung, um damit zusätzliche Vorteile im Leistungsverhalten zu erreichen.

[0003] Eine Art der Spekulation ist die Steuerablaufspekulation. Die Steuerablaufspekulation sagt die Richtung vorher, in der Lithogrammsteuerung voranschreiten wird. Beispielsweise kann eine Verzweigungsvorhersage angewendet werden, um vorherzusagen, ob eine Verzweigung genommen wird. Es sind viele Arten von Verzweigungsvorhersage verfügbar, die Verfahren beinhalten, die einfach jedes mal die gleiche Vorhersage treffen, und wozu auch Verfahren gehören, die anspruchsvolle geschichtliche Entwicklungen der vorhergehenden Verzweigungen in dem Programm bewahren, um damit eine Vorhersage auf der Grundlage der höheren Entwicklung zu treffen. Die Verzweigungsvorhersage kann durch Hardwareoptimierungen, Compiler-Optimierungen oder beides vereinfacht werden. Auf der Grundlage der Vorhersage, die von dem Verzweigungsvorhersagemechanismus geliefert wird, können Befehle spekulativ abgeholt und ausgeführt werden. Wenn der Verzweigungsbefehl schließlich bewertet wird, kann die Verzweigungsvorhersage verifiziert werden. Wenn die Vorhersage nicht korrekt war, können Befehle, die spekulativ auf der Grundlage der falschen Vorhersage ausgeführt werden, verworfen werden.

[0004] Eine weitere Art der Spekulation ist die Datenspekulation, die Datenwerte vorhersagt. Vorgeschlagene Arten an Datenspekulation beinhalten die spekulative Erzeugung von Adressen für Speicheroperationen und das spekulative Erzeugen von Datenwerten zur Verwendung in Rechenoperationen. Wie bei der Steuerungsspekulation werden die zu Grunde liegenden Bedingungen, die für das spekulative Erzeugen eines Wertes angewendet wurden, schließlich bewertet, wodurch es möglich ist, die Spekulation zu verifizieren oder zu verwerfen.

[0005] Da die Spekulation es ermöglicht, mit der Ausführung fortzufahren, ohne auf den Abschluss einer Abhängigkeitsprüfung zu warten, können deutliche Leistungssteigerungen erreicht werden, wenn die durch die korrekten Spekulationen gewonnenen Leistungsverbesserungen die Leistungseinbuße übertreffen, die durch nicht korrekte Spekulationen auftritt. Das Reduzieren der Leistungsbeeinträchtigungen auf Grund nicht korrekter Spekulationen ist daher wünschenswert.

[0006] WO 02/42902 offenbart das erneute Disponieren mehrerer Mikrooperationen in einem Prozessor unter Anwendung einer Wiederholwarteschlange. Der Prozessor umfasst eine Wiederholwarteschlange, um mehrere Befehle zu empfangen, und beinhaltet eine Ausführungseinheit, um die mehreren Befehle auszuführen. Eine Disponiereinheit bzw. eine Einrichtung zur Ablauforganisation ist zwischen der Wiederholwarteschlange und der Ausführungseinheit vorgesehen. Die Disponiereinheit disponiert Spekulativbefehle für das Ausführen und gibt jeden Befehl an die Ausführungseinheit aus. Eine Prüfeinheit ist mit der Ausführungseinheit verbunden, um zu bestimmen, ob jeder Befehl erfolgreich ausgeführt wurde. Die Prüfeinheit ist ferner mit der Wiederholwarteschlange gekoppelt, um der Wiederholwarteschlange jedem Befehl zuzuleiten, der nicht erfolgreich ausgeführt wurde.

[0007] WO 01/50253 offenbart eine Disponiereinheit, die Befehlsoperationen zum Ausführen ausgibt, die jedoch die Befehlsoperationen bewahrt. Wenn eine spezielle Befehlsoperation im Nachhinein erkannt wird, dass diese in nicht spekulativer Weise auszuführen ist, ist diese spezielle Befehlsoperation noch in der Disponiereinheit gespeichert. Im Anschluss an das Bestimmen, dass die spezielle Befehlsoperation nicht-spekulativ wurde (durch das Ausgeben und Ausführen von Befehlsoperationen vor der speziellen Befehlsoperation), kann die spezielle Befehlsoperation von der Disponiereinheit erneut ausgegeben werden. Die Einbuße für das nicht korrekte Disponieren von Befehlsoperationen, die in in-spekulativer Weise auszuführen sind, kann im Vergleich dazu verringert werden, dass die spezielle Befehlsoperation und jüngere Befehlsoperationen aus der Verarbeitungs-Pipeline bzw. aus der Vektorverarbeitung entfernt werden und die spezielle Befehlsoperation erneut abgeholt wird. Zusätzlich kann die Disponiereinheit die Abhängigkeitsangaben für jede Befehlsoperation bewahren, die ausgegeben wurde. Wenn die spezielle Befehlsoperation erneut ausgegeben wurde, können die Befehlsoperationen, die von der speziellen Befehlsoperation (direkt oder indirekt) abhängig sind, mittels der Abhängigkeitsangaben erkannt werden. Die Disponiereinheit gibt dann auch die abhängigen Befehlsoperationen erneut aus. Befehlsoperationen, die auf die spezielle Befehlsoperation in der Programmreihenfolge erfolgen, die aber nicht von der speziellen Befehlsoperation abhängig sind, werden nicht erneut ausgegeben. Deutlich kann die Einbuße für das nicht korrekte Disponieren von Befehlsoperationen, die in nicht-spekulativer Weise

auszuführen sind, weiter verringert werden gegenüber dem Verwerfen der speziellen Befehlsoperation und aller jüngeren Befehlsoperationen und dem erneuten Abrufen der speziellen Befehlsoperation.

Überblick über die Erfindung

[0008] Es sind diverse Ausführungsformen von Verfahren und Systemen zum Wiederholen von Operationen in einer Lade/Schreibeinheit eines Daten-spekulativen Mikroprozessors offenbart.

[0009] Die vorliegende Erfindung stellt gemäß einem ersten Aspekt einen Mikroprozessor bereit, der umfasst: eine Disponiereinheit, die ausgebildet ist, Operationen auszugeben; und eine Lade/Schreibeinheit, die ausgebildet ist, Speicheroperationen zu empfangen, die von der Disponiereinheit ausgebildet ist, wobei die Lade/Schreibeinheit ferner ausgebildet ist, die Speicheroperationen auszuführen; wobei die Lade/Schreibeinheit auch ausgebildet ist, Information zu speichern, die mehrere Speicheroperationen bezeichnet, die an die Lade/Schreibeinheit ausgegeben wurden, wobei in Reaktion auf das Erkennen einer nicht korrekten Datenspekulation für eine der mehreren Speicheroperationen die Lade/Schreibeinheit ausgebildet ist, eine Wiederholangabe zu der Disponiereinheit zu senden, die angibt, das mindestens eine der mehreren Speicheroperationen in der Lade/Schreibeinheit erneut ausgegeben werden sollte; wobei die Disponiereinheit ausgebildet ist, die mindestens eine der mehreren Speicheroperationen in Reaktion auf die Angabe aus der Lade/Schreibeinheit erneut auszugeben.

[0010] Die vorliegende Erfindung umfasst ferner ein Computersystem, das umfasst: einen Speicher, und einen Prozessor, der mit dem Speicher verbunden ist, wobei der Prozessor aufweist: eine Disponiereinheit, die ausgebildet ist, Speicheroperationen auszugeben; und eine Lade/Schreibeinheit, die ausgebildet ist, Speicheroperationen, die von der Disponiereinheit ausgegeben werden, zu empfangen und die Speicheroperationen auszuführen; wobei die Lade/Schreibeinheit ausgebildet ist, Information zu speichern, die mehrere Speicheroperationen kennzeichnet, die von der Lade/Schreibeinheit ausgegeben wurden, wobei in Reaktion auf das Erkennen einer nicht korrekten Datenspekulation für eine der mehreren Speicheroperationen die Lade/Schreibeinheit ausgebildet ist, eine Wiederholangabe für die Disponiereinheit bereitzustellen, die angibt, dass mindestens eine der mehreren Speicheroperationen in der Lade/Schreibeinheit erneut ausgegeben werden soll; wobei die Disponiereinheit ausgebildet ist, die mindestens eine der mehreren Speicheroperationen in Reaktion auf die Angabe von der Lade/Schreibeinheit erneut auszugeben.

[0011] Gemäß einem weiteren Aspekte der vorliegenden Erfindung wird ein Verfahren bereitgestellt, das umfasst:
Ausgeben einer Operation durch eine Disponiereinheit an eine Lade/Schreibeinheit zum Ausführen;
Speichern von Information, die mehrere Speicheroperationen, die von der Lade/Schreibeinheit ausgegeben werden, kennzeichnet, in der Lade/Schreibeinheit;
Erkennen durch die Lade/Schreibeinheit, dass die an der Operation ausgeführte Datenspekulation nicht korrekt ist;
in Reaktion auf dieses Erkennen, Erzeugen einer Wiederholangabe durch die Lade/Schreibeinheit, die mindestens eine Operation kennzeichnet, die aktuell an die Lade/Schreibeinheit ausgegeben wird und Bereitstellen der Wiederholangabe für die Disponiereinheit;
in Reaktion auf das Empfangen der Wiederholangabe erneutes Ausgeben der mindestens einen Operation an die Lade/Schreibeinheit durch die Disponiereinheit.

[0012] Somit umfasst in einigen Ausführungsformen ein Mikroprozessor eine Disponiereinheit, die ausgebildet ist, Operationen auszugeben, und umfasst ferner eine Lade/Schreibeinheit, die ausgebildet ist, von der Disponiereinheit ausgegebene Speicheroperationen auszuführen. Die Lade/Schreibeinheit ist ausgebildet, Information zu speichern, die Speicheroperationen kennzeichnet, die von der Lade/Schreibeinheit ausgegeben wurden. In Reaktion auf das Erkennen einer nicht korrekten Datenspekulation für eine der ausgegebenen Speicheroperationen ist die Lade/Schreibeinheit in der Lage, eine Wiederholangabe an die Disponiereinheit auszugeben, die angibt, dass mindestens eine der ausgegebenen Speicheroperationen in der Lade/Schreibeinheit erneut ausgegeben werden sollte. Die Disponiereinheit ist ausgebildet, in Reaktion darauf die Speicheroperationen, die von der Lade/Schreibeinheit gekennzeichnet sind, erneut auszugeben.

[0013] In einer Ausführungsform ist die Lade/Schreibeinheit ausgebildet, jede der Speicheroperationen, die eine mit der Adresse der nicht korrekt spekulierten Speicheroperation übereinstimmende Adresse aufweist, zu wiederholen. Die Lade/Schreibeinheit ist auch (oder alternativ) ausgebildet, jede der Speicheroperationen mit einer Adresse, die mit einer spekulativen Adresse der nicht korrekt spekulierten Speicheroperation übereinstimmt, zu wiederholen. In einer Ausführungsform wiederholt die Lade/Schreibeinheit lediglich Ladeoperationen, deren Adresse mit der Adresse der nicht korrekt spekulierten Speicheroperation übereinstimmt. In einigen Ausführungsformen überwacht die Lade/Schreibeinheit, welche Ladeoperationen Daten von Schreiboperationen weitergegeben haben, und wenn die Adresse einer Schreiboperation nicht spekulativ vorhergesagt wird, kann die Lade/Schreibeinheit jüngere Ladeope-

rationen wiederholen, die Daten von dieser Schreiboperation benutzt haben. Die Lade/Schreibeinheit kann ferner ausgebildet sein, die Speicheroperation zu wiederholen, für die eine nicht korrekte Datenspekulation in einigen Situationen erkannt wurde.

[0014] In einigen Ausführungsformen ist die Lade/Schreibeinheit ausgebildet, eine nicht korrekte Datenspekulation zu erkennen, indem ein spekulativer Wert der Adresse der Speicheroperation mit einem neuen Wert der Adresse der Speicheroperation verglichen wird und/oder indem ein spekulatives Ergebnis der Speicheroperation mit einem nicht-spekulativen Ergebnis der Speicheroperation verglichen wird.

[0015] Diverse Ausführungsformen eines Verfahrens umfassen: Ausgeben einer Operation an eine Lade/Schreibeinheit zum Ausführen; Empfangen einer Angabe durch die Lade/Schreibeinheit, wobei die Angabe angibt, dass eine an der Operation ausgeführte Datenspekulation nicht korrekt ist; in Reaktion auf diese Angabe, Erzeugen einer Wiederholangabe durch die Lade/Schreibeinheit, wobei die Wiederholangabe zumindest eine Operation kennzeichnet, die in der Lade/Schreibeinheit anhängig ist; und erneutes Ausgeben der Operation bzw. der Operationen, die von der Wiederholangabe gekennzeichnet sind, wodurch die Disponiereinheit in Reaktion darauf.

Kurze Beschreibung der Zeichnungen

[0016] Ein besseres Verständnis der vorliegenden Erfindung kann erreicht werden, wenn die folgende detaillierte Beschreibung in Verbindung mit den begleitenden Zeichnungen studiert wird, in denen:

[0017] Fig. 1 einen Mikroprozessor gemäß einer Ausführungsform zeigt;

[0018] Fig. 2 eine Blockansicht einer Lade/Schreibeinheit gemäß einer Ausführungsform zeigt;

[0019] **Fig.** 3 ein Flussdiagramm eines Verfahrens zum Wiederholen von Operationen in einer Lade/Schreibeinheit gemäß einer Ausführungsform zeigt;

[0020] Fig. 3b ein Flussdiagramm eines Verfahrens zum Wiederholen von Operationen in einer Lade/Schreibeinheit gemäß einer weiteren Ausführungsform zeigt;

[0021] Fig. 3c ein Flussdiagramm gemäß eines Verfahrens zum Wiederholen von Operationen in einer Lade/Schreibeinheit gemäß einer noch weiteren Ausführungsform zeigt;

[0022] Fig. 4 ein beispielhaftes Computersystem gemäß einer Ausführungsform zeigt; und

[0023] Fig. 5 ein weiteres beispielhaftes Computersystem gemäß einer noch weiteren Ausführungsform darstellt.

[0024] Obwohl die Erfindung diversen Modifizierungen und alternativen Formen unterliegen kann, sind dennoch spezielle Ausführungsformen davon in Form von Beispielen in den Zeichnungen gezeigt und werden nachfolgend detailliert beschrieben. Es sollte jedoch beachtet werden, dass die Zeichnungen und die detaillierte Beschreibung nicht beabsichtigen, die Erfindung auf die spezielle offenbarte Form einzuschränken, sondern die Erfindung soll vielmehr alle Modifizierungen, Äquivalente und Alternativen abdecken, die innerhalb des Grundgedankens und des Schutzbereichs der vorliegenden Erfindung liegen, wie sie durch die angefügten Patentansprüche definiert ist. Zu beachten ist auch, dass die Überschriften nur der Gliederung dienen und nicht als Einschränkung oder als Interpretation der Beschreibung oder der Ansprüche zu verwenden sind. Des weiteren ist zu beachten, dass das Wort „kann" durchwegs in dieser Anmeldung als in einem zulässigen Sinne (d. h. in dem Sinne, hat die Möglichkeit zu ist in der Lage zu) zu verstehen ist und nicht in einem zwingend vorgeschriebenen Sinne (d. h. muss). Der Begriff „enthalten" und „Ableitungen davon" bedeuten „einschließend aber nicht beschränkt darauf". Der Begriff „verbunden" bedeutet „direkt oder indirekt verbunden", und der Begriff „gekoppelt" bedeutet „direkt oder indirekt gekoppelt".

Art bzw. Arten zum Ausführen der Erfindung

[0025] Fig. 1 ist eine Blockansicht einer Ausführungsform eines Mikroprozessors **100**. Der Mikroprozessor **100** ist ausgebildet, die in einem Speichersystem **200** abgelegten Befehle auszuführen. Viele dieser Befehle operieren auf der Grundlage von Daten, die in dem Speichersystem **200** gespeichert sind. Zu beachten ist, dass das Speichersystem **200** physikalisch in einem Computersystem in verteilter Form vorgesehen sein kann und auf dieses durch einen oder mehreren Mikroprozessoren **100** zugegriffen werden kann.

[0026] Der Mikroprozessor **100** umfasst einen Befehls-Cache-Speicher **106** bzw. einen schnellen Zwischenspeicher **106** und einen Datencache-Speicher **128**. Der Mikroprozessor **100** umfasst eine Vorabholeinheit **108**, die mit dem Befehlscache-Speicher **106** verbunden ist. Eine Ausgabeeinheit **104** ist ausgebildet, Befehle aus dem Befehlscache-Speicher **106** zu erhalten und Operationen an eine oder mehrere Disponiereinheit **118** auszugeben. Eine oder mehrere Disponiereinheiten **118** sind ausgebildet, ausgegebene Operationen von der Ausgabeeinheit **104** zu empfangen und Operationen an eine oder mehrere Ausführungskerne **124** auszugeben. Der eine oder die mehreren Ausführungskerne **124** können jeweils

eine Lade/Schreibeinheit aufweisen, die ausgebildet ist, einen Zugriff auf den Daten-Cache-Speicher **128** auszuführen. Ergebnisse, die von dem einen oder den mehreren Ausführungskernen **124** erzeugt werden, können auf einem Ergebnisbus **130** ausgegeben werden. Diese Ergebnisse können als Operandenwerte für nachfolgend ausgegebene Befehle verwendet werden und/oder in einer Registerdatei **116** gespeichert werden. Eine Rücknahmewarteschlange **102** ist mit der einen oder den mehreren Disponiereinheiten **118** und der Ausgabeeinheit **104** verbunden. Die Rücknahmewarteschlange **102** ist ausgebildet, zu bestimmen, wann jede ausgegebene Operation zurückgenommen bzw. abgeschlossen werden kann. In einer Ausführungsform ist der Mikroprozessor **100** so gestaltet, dass er mit der x86-Architektur kompatibel ist. Zu beachten ist, dass der Mikroprozessor **100** auch viele andere Komponenten aufweisen kann. Beispielsweise kann der Mikroprozessor **100** eine Verzweigungsvorhersageeinheit (nicht gezeigt) aufweisen.

[0027] Der Befehls-Cache-Speicher **106** kann temporär Befehle speichern, bevor diese von der Ausgabeeinheit **104** empfangen werden. Codierung der Befehle kann den Befehls-Cache-Speicher **106** zugeleitet werden, durch Vorabholen von Codierung aus dem Speichersystem **200** mittels der Vorabholeinheit **108**. Der Befehls-Cache-Speicher **106** kann in diversen Konfigurationen eingerichtet sein (beispielsweise teilassoziativ, vollständig assoziativ oder direkt zugeordnet). In einigen Ausführungsformen kann es mehrere Ebenen an Befehls- und/oder Daten-Cache-Speicher **106** und **128** geben. Einige Ebenen können in den Mikroprozessor **100** integriert sein, wie dies gezeigt ist, während andere Ebenen des Cache-Speichers extern zu dem Mikroprozessor vorgesehen sein können.

[0028] Die Vorabholeinheit **108** kann vorläufig Befehlscodierung aus dem Systemspeicher **100** zur Speicherung in dem Befehlscache-Speicher **106** abrufen. In einer Ausführungsform ist die Vorabholeinheit **108** ausgebildet, ganze Sequenzen aus Codierung von dem Systemspeicher **100** in dem Befehls-Cache-Speicher **106** einzuladen. Die Vorabholeinheit **108** kann eine Vielzahl spezieller Vorabholverfahren und Algorithmen zum Abholen von Codierung nennen.

[0029] Die Ausgabeeinheit **104** gibt Signale aus, die Bit-codierte Operationen enthalten, die von dem einen oder den mehreren Ausführungskernen **124** ausführbar sind, sowie auch Operandenadresseninformationen, unmittelbare Daten und/oder Ersetzungsdaten enthalten. In einigen Ausführungsformen umfasst die Ausgabeeinheit **104** eine Decodierschaltung (nicht gezeigt), zum Decodieren gewisser Befehle in Operationen, die von dem einen oder den mehreren Ausführungskernen **124** ausführbar sind. Einfache Befehle können einer einzelnen Operation entsprechen. In einigen Ausführungsformen können komplexere Befehle an mehreren Operationen entsprechen. Wenn eine Operation die Aktualisierung eines Registers beinhaltet, kann eine Registerstelle in der Registerdatei **116** reserviert werden (beispielsweise beim Decodieren dieser Operation), um spekulative Registerzustände zu speichern (in einer alternativen Ausführungsform kann ein Umordnungspuffer verwendet werden, um einen oder mehrere spekulative Registerzustände für jedes Register zu speichern). Eine Registerzuordnung kann die logischen Registernamen von Ursprungs- und Zieloperanden in physikalische Registernamen übersetzen, um eine Registerumbenennung zu ermöglichen. Eine Registerzuordnung kann überwachen, welche Register innerhalb der Registerdatei **116** aktuell zugewiesen sind.

[0030] Der Mikroprozessor **100** aus Fig. 1 unterstützt die Ausführung außer der Reihenfolge. Eine Rücknahmewarteschlange **2** kann die ursprüngliche Programmreihenfolge für Register Lese- und Schreiboperationen verfolgen, kann ein spekulatives Ausführen von Befehlen und eine Wiederherstellung nach einer Verzweigungsfehlvorhersage sowie eine präzise Ausnahmebehandlung ermöglichen. Die Rücknahmewarteschlange **102** kann in Form einer „zuerst hinein zuerst heraus"-Konfiguration eingerichtet sein, in der Operationen sich zur „Unterseite" des Puffer bewegen, wenn sie validiert werden, wodurch Platz für neue Einträge an der „Oberseite" der Warteschlange geschaffen wird. Die Rücknahmewarteschlange **102** kann eine Operation zurücknehmen bzw. abschließen in Reaktion darauf, dass diese Operation in ihrer Ausführung abgeschlossen ist und das eine Datenspekulation oder eine Steuerablaufspekulation, die an Operationen bis zu und einschließlich dieser Operation in der Programmreihenfolge ausgeführt wurden, verifiziert wurden. Die Rücknahmewarteschlange **102** kann den spekulativen Zustand eines physikalischen Registers in einem tatsächlichen Zustand des Mikroprozessors **100** umwandeln, wenn die Operation, die den Wert in diesem physikalischen Register erzeugt hat, zurückgenommen bzw. abgeschlossen ist. In einigen Ausführungsformen wird die Rücknahmewarteschlange **102** als Teil eines Umordnungspuffers eingerichtet. Ein derartiger Umordnungspuffer kann ferner einen Datenwertspeicherplatz für spekulative Registerzustände bereitstellen, um eine Registerumbenennung zu unterstützen. Zu beachten ist, dass in anderen Ausführungsformen die Rücknahmewarteschlange **102** keinen Datenwertdatenplatz bietet. Stattdessen kann, wenn Operationen abgeschlossen werden, die Rücknahmewarteschlange **102** Register in der Registerdatei **116** freigeben, die nicht mehr benötigt werden, um spekulative Registerzustände zu speichern, und kann Signale an die Registerzuordnung zuleiten, die angeben, welche Register aktuell verfügbar sind. Durch Beibehalten der spekulativen Registerzustän-

de innerhalb der Registerdatei **116** (oder in alternativen Ausführungsformen innerhalb eines Umordnungspuffers) bis die Operationen, die diese Zustände erzeugt haben, als gültig erkannt sind, können die Ergebnisse von spekulativ ausgeführten Operationen entlang eines falsch vorhergesagten Programmpfades in der Registerdatei **116** als ungültig erklärt werden, wenn eine Verzweigungsvorhersage nicht korrekt ist.

[0031] Wenn ein erforderlicher Operand einer speziellen Operation eine Registerstelle ist, kann die Registeradresseninformation einer Registerzuordnung (oder einem Umordnungspuffer) zugeführt werden. Beispielsweise gibt es in der x86-Architektur acht 32-Bit-logische Register (beispielsweise EAX, EBX, ECX, EDX, EBP, ESI, EDI und ESP). Die physikalische Registerdatei **116** (oder ein Umordnungspuffer) enthält Speicherplatz für Ergebnisse, die dem Inhalt dieser logischen Register ändern, wodurch eine Bearbeitung außerhalb der Reihenfolge möglich ist. Ein physikalisches Register in der Registerdatei **116** kann reserviert werden, um das Ergebnis jeder Operation zu speichern, die erkannt wird, dass sie den Inhalt eines der logischen Register modifiziert. Daher kann zu diversen Zeitpunkten während des Ausführens eines speziellen Programms die Registerdatei **116** (oder in alternativen Ausführungsformen ein Umordnungspuffer) ein oder mehrere Register aufweisen, die den spekulativ ausgeführten Inhalt eines gegebenen logischen Registers enthalten.

[0032] Eine Registerzuordnung kann ein physikalisches Register einem speziellen logischen Register zuordnen, das als ein Zieloperand für eine Operation spezifiziert ist. Die Ausgabeeinheit **105** kann bestimmen, dass die Registerdatei **116** ein oder mehrere zuvor zugeordnete physikalische Register aufweist, die einem logischen Register zugeordnet sind, das als ein ursprünglicher bzw. Quellenoperand in einer gegebenen Operation spezifiziert ist. Die Registerzuordnung kann eine Markierung für das physikalische Register bereitstellen, das jüngst diesen logischen Register zugeordnet wurde. Diese Markierung kann verwendet werden, um auf den Datenwert des Operanden in der Registerdatei **116** zuzugreifen oder um den Datenwert über die Ergebnisweiterleitung auf den Ergebnisbus **130** zu empfangen. Wenn der Operand einer Speicherstelle entspricht, kann der Operandenwert auf dem Ergebnisbus (für die Ergebnisweiterleitung und/oder Speicherung in der Registerdatei **116**) über die Lade/Schreibeinheit **222** bereitgestellt werden. Operandendatenwerte können den einen oder den mehreren Ausführungskernen **124** zur Verfügung gestellt werden, wenn die Operation von einer der Disponiereinheiten **118** ausgegeben wird. Zu beachten ist, dass in alternativen Ausführungsformen Operandenwerte auch einer entsprechenden Disponiereinheit **118** zur Verfügung gestellt werden können, wenn eine Operation ausgegeben wird (anstatt dass die Werte einem entsprechenden Ausführungskern **124** zur Verfügung gestellt werden, wenn die Operation ausgegeben wird).

[0033] Die Bit-codierten Operationen und die unmittelbaren Daten, die an den Ausgängen der Ausgabeeinheit **104** bereitgestellt werden, können zu einer oder mehreren Disponiereinheiten **118** weitergeleitet werden. Zu beachten ist, dass im hierin verwendeten Sinne eine Disponiereinheit bzw. eine Ablauforganisationseinheit eine Einrichtung ist, die erkennt, wann Operationen für das Ausführen bereit sind und die bereitstehende Operationen zu einer oder mehreren Funktionseinheiten ausgibt. Z. B. ist eine Reservierstation eine Disponiereinheit. Operationen in einer Disponiereinheit oder einer Gruppe aus Disponiereinheiten können auch als Operationen in einem Befehls- oder Operationsfenster oder Zeitfenster bezeichnet werden. Jede Disponiereinheit **118** ist in der Lage, eine Operationsinformation zu bewahren (beispielsweise Bit-codierte Ausführungsbits sowie Operandenwerte, Operandenmarken, und/oder unmittelbare Daten) für mehrere anhängige Operationen, die auf die Ausgabe zu einem Ausführungskern **124** warten. In einigen Ausführungsformen bieten die Disponiereinheiten **118** keinen Speicherplatz für Operandenwerte. Stattdessen überwachen die Disponiereinheiten ausgegebene Operationen und Ergebnisse, die in der Registerdatei **116** verfügbar sind, um zu bestimmen, wann Operandenwerte für das Abrufen durch die Funktionseinheiten **126** aus der Registerdatei **116** oder dem Ergebnisbus **130** verfügbar sind. In einigen Ausführungsformen ist jede Disponiereinheit **118** mit einer zugeordneten Funktionseinheit **126** verknüpft. In anderen Ausführungsformen kann eine einzelne Disponiereinheit **118** Operationen an mehr als eine Funktionseinheit **126** ausgeben.

[0034] Die Disponiereinheiten **118** können vorgesehen sein, um zeitweilig Operationsinformation zu speichern, die von dem einen oder den mehreren Ausführungskernen **124** auszuführen ist. Wie zuvor beschrieben ist, kann jede Disponiereinheit **118** Operationsinformation für anhängige Operationen speichern. Des weiteren kann jede Disponiereinheit Operationsinformationen für Operationen speichern, die bereits ausgeführt sind, die aber nochmals auszugeben sind. Operationen werden an den einen oder die mehreren Ausführungskerne **124** zum Ausführen ausgegeben in Reaktion darauf, dass Werte von einem oder mehreren erforderlichen Operanden rechtzeitig für die Ausführung verfügbar sind. Folglich kann die Reihenfolge, in der die Operationen ausgeführt werden, anders sein als die Reihenfolge der ursprünglichen Programmbefehlssequenz. Operationen, die eine Datenspekulation beinhalten, können in der einen oder den mehreren Disponiereinheiten **118** bleiben, bis sie nicht-spekulativ werden, so dass sie erneut ausgegeben werden können, wenn die Datenspekulation nicht korrekt war. Wie in **Fig. 1** gezeigt

ist, liefert eine Lade/Schreib-Einheit **126c** eine Wiederholangabe, die eine oder mehrere Operationen angibt, die erneut an die Disponiereinheit **118** ausgegeben sind (beispielsweise kann eine derartige Wiederholangabe die Marke jeder Operation, die zu wiederholen ist, enthalten). Die Disponiereinheit **118** kann in Reaktion darauf Operationen erneut ausgeben, die durch eine derartige Wiederholangabe gekennzeichnet sind.

[0035] In einer Ausführungsform kann jeder der Ausführungskerne **124** mehrere Funktionseinheiten **126** (beispielsweise die Funktionseinheiten **126a** bis **126c**, wie sie in Fig. 1 gezeigt sind) enthalten. Einige Funktionseinheiten, beispielsweise die Einheit **126a**, sind ausgebildet, um Ganzzahlarithmetikoperationen der Addition und Subtraktion sowie Verschiebungen, Rotationen, logische Operationen und Verzweigungsoperationen auszuführen. Andere Funktionseinheiten, beispielsweise die Einheit **126b**, ist ausgebildet, um Gleitkommaoperationen auszuführen. Eine oder mehrere der Funktionseinheiten können ausgebildet sein, eine Adressenerzeugung für Lade- und Schreibspeicheroperationen, die von einer Funktionseinheit, beispielsweise der Einheit **126c**, auszuführen sind, zu gewährleisten, die Lade- und Schreiboperationen ausführt, um auf Daten zuzugreifen, die in dem Daten-Cache-Speicher **128** und/oder in dem Systemspeicher abgelegt sind. In einer Ausführungsform ist eine derartige Funktionseinheit **126c** mit einem Lade/Schreib-Puffer mit mehreren Speicherplätzen für Daten und Adresseninformationen für anhängige Ladeoperationen und/oder Schreiboperationen versehen.

[0036] Eine oder mehrere der Funktionseinheiten **126** können ferner Information bereitstellen im Hinblick auf das Ausführen von bedingten Verzweigungsbefehlen für eine Verzweigungsvorhersageeinheit, so dass, wenn eine Verzweigung falsch vorhergesagt wurde, die Verzweigungsvorhersageeinheit Befehle verwerfen kann, die in der Reihe nach der falsch vorhergesagten Verzweigung auftreten und in der Befehlsverarbeitungspipeline sind, und um die Vorabholeinheit **106** neu anzuweisen. Die neu angewiesene Vorabholeinheit **106** beginnt dann, den richtigen Satz an Befehlen von dem Befehlscache-Speicher **106** oder dem Systemspeicher **200** abzurufen. In derartigen Situationen können die Ergebnisse von Befehlen in der ursprünglichen Programmsequenz, die nach dem falsch vorhergesagten Verzweigungsbefehl auftraten, verworfen werden, wozu jene gehören, die spekulativ ausgeführt und temporär in der Registerdatei **116** gespeichert wurden.

[0037] Ergebnisse, die von den Funktionseinheiten **126** in den Ausführungskernen **124** erzeugt werden, können auf dem Ergebnisbus **130** an die Registerdatei **116** ausgegeben werden, wenn ein Registerwert aktualisiert wird. Wenn der Inhalt einer Speicherstelle geändert wird, können die in den Ausführungskernen **124** erzeugten Ergebnisse der Lade/Schreibeinheit **126c** zugeführt werden.

[0038] Der Daten-Cache-Speicher **128** ist ein Cache-Speicher, der zum temporären Speichern von Daten vorgesehen ist, die zwischen einem oder mehreren Ausführungskernen **124** und dem Systemspeicher **200** ausgetauscht werden. Ähnlich wie der zuvor beschriebene Befehls-Cache-Speicher **106** kann auch der Daten-Cache-Speicher **128** auf viele verschiedene Möglichkeiten von speziellen Speicherkonfigurationen eingerichtet werden, wozu eine teilassoziative Konfiguration gehört. Ferner kann der Daten-Cache-Speicher **106** und der Befehls-Cache-Speicher **128** als einheitlicher Cache-Speicher in einigen Ausführungsformen eingerichtet sein.

[0039] In einigen Ausführungsformen umfasst der Mikroprozessor **100** eine integrierte Speichersteuerung **160**, die es dem Mikroprozessor ermöglicht, direkt mit dem Systemspeicher **200** zu kommunizieren. In anderen Ausführungsformen ist die Speichersteuerung **160** in einer Busbrücke enthalten, die indirekt den Mikroprozessor **100** mit dem Systemspeicher **200** verbindet.

Datenspekulation

[0040] Wie hierin beschrieben ist, ist ein Datenwerk spekulativ, wenn die Möglichkeit besteht, dass der Datenwert als nicht korrekt ermittelt wird und folglich neu berechnet werden muss. Ein spekulativer Datenwert ist ein Wert, der nicht mit Gewissheit als korrekt oder inkorrekt bezeichnet werden kann. Ein Datenwert kann erneut berechnet werden, wenn dieser Datenwert das Ergebnis einer Operation ist, für die eine gewisse Datenspekulation ausgeführt wurde oder wenn der Datenwert von einem weiteren spekulativen Datenwert abhängt (beispielsweise wenn der Datenwert als das Ergebnis einer Operation erzeugt wird, die eine oder mehrere spekulative Operanden besitzt). Ein nicht-spekulativer Wert ist ein Wert, der nicht von einer Datenspekulation abhängt (ein derartiger Wert kann dennoch einer Steuerablaufspekulation unterliegen).

[0041] Es können diverse Mechanismen in dem Mikroprozessor **100** eine Datenspekulation ausführen. Beispielsweise können die Ausgabeeinheit **104**, die Speichersteuerung **160** und/oder eine oder mehrere Funktionseinheiten **126** jeweils eine Datenspekulation für eine spezielle Operation ausführen. Die Ausgabeeinheit **104** kann erkennen, dass ein Ergebnis einer einzelnen Operation als ein spekulativer Operand für eine weitere Operation verwendet wird. Z. B. kann die Ausgabeeinheit vorhersagen, dass eine Ladeoperation auf Daten zugreifen wird, die in dem Daten-Cache-Speicher **128** von einer früheren Schreiboperation gespeichert sind. Die Ausgabeeinheit **104**

kann darauf reagierend den Datenwert, der darin als die Quelle der Schreiboperation mit dem Register gespeichert ist, als das spekulative Ergebnis der Ladeoperation erkennen. Diese Art der Datenspekulation wird im Weiteren als Abhängigkeitsvorhersage bezeichnet. Die Abhängigkeitsvorhersage kann in der Ausgabeeinheit **104** erweitert werden, indem die Quelle der Schreiboperation als eine spekulative Operandenquelle für Operationen, die das Ergebnis der Ladeoperation spezifizieren, als ein Operand verknüpft wird. Eine weitere Art der Abhängigkeitsvorhersage kann in der Lade/Schreib-Einheit **126c** ausgeführt werden, indem Ladeoperationen Schreiboperationen mit nicht berechneten Adressen umgehen können, d. h. durch Vorhersagen, dass jüngere Ladeoperationen nicht von früheren Schreiboperationen abhängig sind.

[0042] In einem Mehrprozessorsystem kann die Speichersteuerung **160** Kohärenzprüfungen ausführen, um die Kohärenz des Cache-Speichers zu erhalten. Die Speichersteuerung **160** kann spekulativ eine Kopie einer Cache-Zeile aus dem Speichersystem **200** zurückgeben, bevor die Kohärenzprüfungen mit den Cache-Speichern anderer Mikroprozessoren abgeschlossen sind. Wenn die Kohärenzprüfungen nachfolgend erkennen, dass die korrekte Kopie der abzurufenden Cache-Zeile aktuell in den Cache-Speichers eines anderen Prozessors enthalten ist, kann die Kopie der Cache-Zeile, die spekulativ aus dem Systemspeicher **200** abgerufen wurde, als ungültig erklärt werden. Folglich sind entsprechende Ergebnisse der Ladeoperation, die durch das Zugreifen auf diese Cache-Zeile erzeugt wurden, spekulativ bis die Kohärenzprüfungen abgeschlossen sind. Die Art der Spekulation wird im Weiteren als Speichervorhersage bezeichnet.

[0043] Die Ausgabeeinheit **104** kann eine Datenspekulation ausführen, indem das Ergebnis einer Operation vorhergesagt wird. Beispielsweise neigen gewisse Operationen dazu, das gleiche Ergebnis zu erzeugen, und somit kann jedes mal, wenn eine dieser Operationen gehandhabt wird, das Ergebnis spekulativ durch die Ausgabeeinheit **104** vor dem eigentlichen Ausführen der Operation durch eine der Funktionseinheiten **126** erzeugt werden. Diese Art der Datenspekulation wird hierin als Datenvorhersage bezeichnet. Zu beachten ist, dass die Datenvorhersage auch in anderen Bereichen des Mikroprozessors ausgeführt werden kann (beispielsweise in einer Lade/Schreibeinheit **126c**).

[0044] Eine Lade/Schreib-Einheit **126c** kann spekulativ die Adresse und basierend auf der spekulativen Adresse das Ergebnis eines Ladebefehls erzeugen, dessen Adresse noch nicht auf der Grundlage eines Musters von zuvor gehandhabten Ladeoperationen berechnet ist. Wenn beispielsweise die vorhergehenden N-Ladeoperationen die Zieladressen A1 bis AN aufweisen, die um einen konstanten Abstand C voneinander getrennt sind (beispielsweise A1, A2 = A1 + C, ..., AN = A(N – 1) + C), kann die Lade/Schreibeinheit **126c** spekulativ Daten aus der angesprochenen Adresse AN + dem konstanten Abstand C als Ergebnis der Ladeoperation zurückgeben. Diese Art der Datenspekulation wird im Weiteren als Adressenvorhersage bezeichnet. Zu beachten ist, dass andere Formen der Adressenvorhersage in vielen Ausführungsformen ebenfalls angewendet werden können.

[0045] Es können mehrere unterschiedliche Arten an Datenspekulationen ausgeführt werden, um einige spekulative Ergebnisse zu erzeugen. z. B. kann das spekulative Ergebnis einer Ganzzahloperation unter Anwendung einer Datenvorhersage erzeugt werden. Dieses spekulative Ergebnis kann dann in einer Schreiboperation gespeichert werden. Eine Ladeoperation kann als abhängig von dieser Schreiboperation durch eine Abhängigkeitsvorhersage vorhergesagt werden, und somit ist das spekulative Ergebnis der Ladeoperation das spekulative Ergebnis der Ganzzahloperation.

[0046] Operationen, die von dem Ergebnis von Operationen abhängen, an denen eine Datenspekulation ausgeführt wurde, können ebenfalls spekulative Ergebnisse erzeugen. Wenn beispielsweise eine Adressenvorhersage verwendet wird, um das spekulative Resultat einer Ladeoperation zu erzeugen, können abhängige Operationen, die unter Anwendung des spekulativen Ergebnisses dieser Ladeoperation als ein Operand ausgeführt werden, spekulative Ergebnisse hervorrufen, die wiederum als Operanden von anderen abhängigen Operationen verwendet werden. Wenn folglich die zugrundeliegende Spekulation in der Ladeoperation als nicht korrekt bestimmt wird, können die Ergebnisse der abhängigen Operationen ebenfalls nicht korrekt sein, und somit kann es notwendig sein, die gesamte Abhängigkeitskette aus Operationen, die von dieser Ladeoperation abhängig sind, erneut auszuführen, um korrekte Ergebnisse zu erzeugen. Wenn andererseits die zu Grunde liegende Spekulation als korrekt ermittelt wird, können die Ergebnisse der abhängigen Operationen ebenfalls korrekt sein (unter der Annahme, dass jene Ergebnisse nicht auf weiteren spekulativen Werten beruhen).

[0047] Viele Operationen, für die eine Datenspekulation ausgeführt wird, können verifiziert werden, wenn diese Operationen durch eine Funktionseinheit ausgeführt werden. Beispielsweise kann die Datenvorhersage, die zum spekulativen Erzeugen des Ergebnisses einer Operation angewendet wird, von der Funktionseinheit **126** verifiziert werden, die diese Operation ausführt, indem das tatsächliche Ergebnis der Operation mit dem spekulativen Ergebnis verglichen wird. Derartige Operationen müssen ggf. nicht erneut ausgeführt werden, wenn die Datenspekulati-

on nicht korrekt ist, da das richtige Ergebnis bereits verfügbar ist. Andere Operationen können verifiziert werden, ohne dass diese vollständig ausgeführt werden. Wenn beispielsweise eine Ladeoperation mit einer nicht berechneten Adresse ihr Ergebnis von einer früheren Schreiboperation weitergibt (beispielsweise auf Grund der Abhängigkeits- oder Adressenvorhersage), kann das spekulative Ergebnis der Ladeoperation verifiziert werden, wenn die Ladeadresse berechnet ist. Wenn die Datenspekulation nicht korrekt ist, muss eine derartige Operation ggf. erneut ausgeführt werden (zumindest teilweise), um das korrekte Ergebnis zu erzeugen.

[0048] Da Operationen, für die Datenspekulationen ausgeführt worden sind, und ihre abhängigen Operationen erneut ausgeführt werden müssen, kann die Rücknahmewarteschlange **102** ausgebildet sein, lediglich Operationen abzuschließen, für die eine zu Grunde liegende Datenspekulation abgeschlossen ist.

Lade/Schreib-Einheiten mit Wiederholmechanismus

[0049] In einem Mikroprozessor, der eine datenspekulative Ausführung unterstützt, müssen eine oder mehrere Operationen auf Grund einer nicht korrekten Datenspekulation ggf. erneut ausgeführt werden. Der Mikroprozessor **100** kann diverse Wiederholmechanismen aufweisen, um ein erneutes Ausführen von Operationen zu ermöglichen. Beispielsweise ist die Disponiereinheit **118** ausgebildet, eine Operation an eine Funktionseinheit **126c** in Reaktion auf eine Angabe erneut auszugeben, dass einer der datenspekulativen Operanden dieser Operation nicht korrekt ist. In ähnlicher Weise kann eine Lade/Schreib-Einheit **126c** auch einen Wiederholmechanismus aufweisen, der erkennt, welche anhängigen Operationen innerhalb der Lade/Schreibeinheit wiederholt werden müssen in Reaktion darauf, dass nach abgeschlossener Beurteilung der Datenspekulation eine Operation die Spekulation als nicht korrekt erkannt wurde. Die Lade/Schreibeinheit **126c** kann der Disponiereinheit anzeigen, welche anhängigen Operationen wiederholt werden sollen, wodurch die Disponiereinheit veranlasst wird, diese Operationen erneut an die Lade/Schreibeinheit **126c** zu einer späteren Zeit auszugeben.

[0050] Fig. 2 zeigt eine Ausführungsform einer Lade/Schreibeinheit **126c**, die ausgebildet ist, Operationen auf Grund einer nicht korrekten Datenspekulation, die von der Lade/Schreibeinheit **126c** erkannt wird, zu wiederholen. Eine Datenspekulationsverifizierlogik **305** ist ausgebildet, jede Art an Datenspekulation zu verifizieren, die von der Lade/Schreibeinheit **126c** ausgeführt wird. Wenn beispielsweise die Lade/Schreibeinheit sowohl eine Adressenvorhersage (beispielsweise durch Vorhersagen der Adresse einer Ladeoperation) und einer Abhängigkeitsvorhersage (beispielsweise durch ermöglichen, dass Ladeoperationen frühere Schreiboperationen mit nicht berechneten Adressen umgehen) ausführt, ist die Lade/Schreibeinheit ausgebildet, sowohl die Adressen- als auch die Abhängigkeitsvorhersage zu verifizieren. Die Datenspekulationsverifizierlogik **305** kann zusätzlich (oder alternativ) ausgebildet sein, die von anderen Bereichen des Mikroprozessors **100** ausgeführte Datenspekulation zu verifizieren. Wenn beispielsweise die Ausgabeeinheit **104** ausgebildet ist, eine Abhängigkeitsprüfung (beispielsweise durch spekulatives Verknüpfen eines Ladeergebnisses mit der Quelle eines früheren Schreibbefehls), kann die Datenspekulationsverifizierlogik **305** ausgebildet sein, diese Abhängigkeitsvorhersage zu verifizieren.

[0051] Die Lade/Schreibeinheit **126c** umfasst ferner einen Operationsspeicherbereich **307** für Operationen, die an die Lade/Schreibeinheit (beispielsweise durch die Disponiereinheit **118**) ausgegeben wurden, die aber noch nicht abgeschlossen sind. Eine an die Lade/Schreibeinheit ausgegebene Operation, an der eine Datenspekulation ausgeführt wurde, wird ggf. nicht abgeschlossen, bis diese Operation von der Datenspekulationsverifizierlogik **305** verifiziert ist. Der Operationsspeicherbereich **307** kann alle anhängigen Operationen innerhalb der Lade/Schreibeinheit **126c** verfolgen. Der Operationsspeicherbereich **307** kann einen Eintrag **310** für jede anhängige Lade- und Schreiboperation aufweisen.

[0052] Ein Eintrag **310** kann auch Information **313** enthalten, die angibt, ob der Eintrag einer Ladeoperation oder einer Schreiboperation zugeordnet ist (oder in einigen Ausführungsformen kann ein Eintrag angeben, dass dieser sowohl eine Ladeoperation als auch eine Schreiboperation enthält, wenn diese einer Operation entspricht, die auf einen Wert operiert, der von einer Speicheradresse eingeladen wird und das Ergebnis in einer Speicheradresse abgelegt wird). Des weiteren kann ein Eintrag **310** eine Markierung **315** (die beispielsweise die Operation und ihr Ergebnis in dem Mikroprozessor **100** kennzeichnet), eine Adresse **317** und/oder Daten **319** enthalten. Das Datenfeld **319** jedes Eintrags kann in einigen Ausführungsformen einen Speicherbereich für sowohl spekulative als auch nicht-spekulative Daten enthalten. In ähnlicher Weise kann das Adressenfeld **317** Speicherplatz für mehr als einen Wert einer Adresse einer Operation in einigen Ausführungsformen aufweisen (beispielsweise für eine spekulative Adresse, die durch Adressenvorhersage erzeugt wird und einen neuen Adressenwert, der durch Ausführen einer Operation erzeugt wird). In einigen Ausführungsformen können Einträge zusätzliche Felder enthalten, um Operationen und/oder Operanden als datenspekulativ zu kennzeichnen. Ein Eintrag **310** kann in Reaktion darauf zugewiesen werden, dass die Disponiereinheit **118** eine Operation an die Lade/Schreibeinheit **126c** ausgibt, und dieser Eintrag kann freige-

geben werden in Reaktion darauf, dass die Lade/Schreibeinheit **126c** das Ausführen der Operation abschließt.

[0053] Die Datenspekulationsverifizierlogik **305** kann ferner gewisse Arten an Datenspekulation verifizieren (beispielsweise Datenvorhersage und/oder einige Arten an Abhängigkeitsvorhersage), indem das spekulative Ergebnis einer Operation mit dem tatsächlichen Ergebnis der Operation verglichen wird. Beispielsweise wird das spekulative Ergebnis einer Ladeoperation in dem Eintrag **310** dieser Ladeoperation in dem Operationsspeicherbereich **307** gespeichert. Wenn das tatsächliche Ergebnis dieser Ladeoperation auf dem Daten-Cache-Speicher **128** erhalten wird, kann die Datenspekulationsverifizierlogik das tatsächliche Ergebnis mit dem spekulativ erhaltenen Ergebnis, das in den Operationsspeicherbereich **307** abgelegt ist, vergleichen. Die Datenspekulationsverifizierlogik **305** kann andere Arten an Datenspekulation verifizieren (beispielsweise einige Arten an Abhängigkeitsvorhersage), indem die Adresse einer Operation mit Adressen einer oder mehreren früheren Operationen verglichen wird. Die Datenspekulationsverifizierlogik **305** kann andere Arten an Datenspekulation (beispielsweise Adressenvorhersage) verifizieren, indem die spekulative Adresse einer Operation mit einem neuen Wert dieser Adresse in Reaktion darauf, dass der neue Wert allgemein adressiert auf den Ergebnisbus **130** gelegt wird, verglichen wird. Wenn die spekulative Adresse nicht mit dem neuen Wert der Adresse übereinstimmt, dann bestimmt die Datenspekulationsverifizierlogik **305**, dass die Datenspekulation für die Adresse nicht korrekt ist.

[0054] In Reaktion auf das Erkennen, dass die Datenspekulation für eine spezielle Operation nicht korrekt ist, veranlasst die Datenspekulationsverifizierlogik **305**, dass eine oder mehrere Operationen wiederholt werden. Eine Operation kann wiederholt werden, indem ein Wiederholsignal, das diese Operation kennzeichnet, der Disponiereinheit **118** zugeleitet wird. In Reaktion auf ein derartiges Signal markiert die Disponiereinheit **118** die Operation für eine Wiederholung (beispielsweise in dem Zustandsinformation modifiziert wird, die mit dieser Operation verknüpft ist und die angibt, dass die Operation wiederholt werden soll). In einer Ausführungsform bewirkt die Datenspekulationsverifizierlogik **305**, dass eine Operation wiederholt wird, indem die Markierung dieser Operation zusammen mit einer Marke bzw. Flagge, die angibt, dass diese Operation wiederholt werden soll, an die Disponiereinheit **118** gesendet wird.

[0055] Wenn die nicht korrekt spekulierte Operation erneut ausgeführt werden muss (beispielsweise auf Grund einer nicht korrekten Abhängigkeitsvorhersage oder einer nicht korrekten Adressenvorhersage), kann die Datenspekulationsverifizierlogik **305** veranlassen, dass diese Operation wiederholt wird. Wenn das korrekte Ergebnis des nicht korrekt spekulierten Ergebnisses bereits verfügbar ist (wenn beispielsweise das nicht korrekte Ergebnis sich auf Grund einer nicht korrekten Daten- oder Abhängigkeitsvorhersage ergibt), kann die Datenspekulationsverifiziereinheit die Lade/Schreibeinheit **126c** veranlassen, das korrekte Ergebnis der nicht korrekten spekulierten Operation den anderen Komponenten des Mikroprozessors in allgemein adressierter Weise zur Verfügung zu stellen, so dass abhängige Operationen innerhalb anderer Bereiche des Mikroprozessors unter Anwendung des korrekten Wertes erneut ausgeführt werden können. Die Datenspekulationsverifizierlogik **305** kann in einigen Ausführungsformen das Wiederholen einer Operation nicht veranlassen. Zu beachten ist, dass die Lade/Schreibeinheit **126c** das Ausführen einer anhängigen Operation abschließen kann, selbst wenn die Lade/Schreibeinheit **126c** auch veranlasst, dass diese Operation wiederholt wird.

[0056] Zusätzlich zu dem Wiederholen der nicht korrekt spekulierten Operation, falls dies erforderlich ist, kann die Datenspekulationsverifizierlogik **305** auch das Wiederholen anderer Operationen, die aktuell in der Lade/Schreibeinheit anhängig sind, veranlassen. Z. B. können gewisse Ladeoperationen Daten aus einer Schreiboperation auf der Grundlage der spekulativen Adresse der Schreiboperation weitergeben. Wenn die spekulative Adresse der Schreiboperation als nicht korrekt erkannt wird (beispielsweise wenn das Ausführen einer Operation, die die Adresse der Schreiboperation erzeugt, einen anderen Adressenwert erzeugt, der nicht gleich der spekulativen Adresse ist), können Ladeoperationen, die Daten auf der Grundlage der spekulativen Adresse des Schreibbefehls weiterleiteten, wiederholt werden. In ähnlicher Weise können Ladeoperationen, deren Adresse mit der neuen Adresse für die Schreiboperationen übereinstimmen, ebenfalls wiederholt werden, so dass diese Operationen nun Daten von dem Schreibbefehl weiter verwenden. In ähnlicher Weise kann die Datenspekulationsverifizierlogik **305** ausgebildet sein zu erkennen, wenn der Operand für eine Schreiboperation erneut allgemein adressiert auf dem Ergebnisbus gelegt wird, und ist ferner ausgebildet, in Reaktion darauf das Wiederholen einer abhängigen Ladeoperation, die den vorhergehenden Wert des Operanden der Schreiboperation verwendet hat, zu veranlassen.

[0057] In einer Ausführungsform kann die Datenspekulationsverifizierlogik **305** abhängige Operationen wiederholen, indem veranlasst wird, dass alle anhängigen Operationen in der Lade/Schreibeinheit, die jünger sind als die nicht korrekt spekulierte Operation, wiederholt werden. In einigen Ausführungsformen bezeichnet eine Komparatorlogik **303** die jüngeren Operationen in dem Operationsspeicherbereich

307 im Hinblick auf die Wiederholung. Die Komparatorlogik **303** kann diverse Mittel aufweisen, um Werte zu vergleichen und/oder übereinstimmende Werte zu erkennen (beispielsweise Komparatoren, inhaltsadressierbare Speicher, etc.).

[0058] In anderen Ausführungsformen verwendet die Datenspekulationsverifizierlogik **305** eine Komparatorlogik **303**, um selektiv gewisse jüngere Operationen in Reaktion auf das Erkennen einer nicht korrekten Datenspekulation für eine spezielle Operation zu wiederholen. Beispielsweise vergleicht in einer Ausführungsform in Reaktion darauf, dass eine nicht-korrekte Datenspekulation für die Adresse einer Schreiboperation erkannt wird, die Komparatorlogik **303** den spekulativen Wert und den neuen Wert der Adresse der Schreiboperation mit den Adressen jeder jüngeren Operation innerhalb des Operationsspeicherbereichs **307**. Wenn die Adresse einer der jüngeren Operationen mit den Adressen der Schreiboperation übereinstimmt, dann wird veranlasst, dass die übereinstimmenden jüngeren Operationen erneut ausgeführt werden, indem die Disponiereinheit mit einer Angabe über die Notwendigkeit zur Wiederholung dieser Operation beliefert wird.

[0059] In einigen Ausführungsformen kann die Lade/Schreibeinheit **126c** eine Schreib-zu-Lade-Weiterleitung einrichten, indem die Adressen von anhängigen Ladeoperationen mit den Adressen älterer Schreiboperationen verglichen werden. Wenn eine Adresse einer Ladeoperation mit der Adresse einer älteren Schreiboperation übereinstimmt, kann die Lade/Schreibeinheit **126c** die Daten, die von der älteren Schreiboperation gespeichert werden, als das Ergebnis der Ladeoperation ausgeben. Jedes mal, wenn eine Ladeoperation Daten von einer Schreiboperation weiter verwendet, kann die Lade/Schreibeinheit **126c** eine Markierung, die die Schreiboperation bezeichnet, in einem Eintrag eines Weiterleitungsverfolgungspuffers bzw. Weiterverwendungsverfolgungspuffers **309** speichern, der mit der Ladeoperation, die die Daten weiter verwendet, verknüpft ist. In einigen dieser Ausführungsformen verwendet die Komparatorlogik **303** die Information, die von dem Weiterleitungsverfolgungspuffer **309** bereitgestellt wird, um Operationen für die erneute Ausführung in Reaktion darauf zu kennzeichnen, dass die Datenspekulationsverifizierlogik **305** eine nicht korrekte Datenspekulation für eine Operation erkennt. Wenn beispielsweise die Adressenvorhersage für eine Schreiboperation als nicht korrekt erkannt wird, ist die Komparatorlogik **303** ausgebildet, den neuen Wert der Adresse für die Schreiboperation mit der Adresse jeder jüngeren Ladeoperation in dem Operationsspeicherbereich **307** zu vergleichen. Jegliche übereinstimmende Operationen sollten Daten von der nicht korrekt spekulierten Schreiboperation weitergeleitet haben, haben dies aber nicht getan auf Grund der nicht korrekt spekulierten Schreibadresse. Folglich kann die Lade/Schreibeinheit **126c** veranlassen, dass diese übereinstimmenden Operationen erneut ausgeführt werden. Des weiteren kann die Markierung der nicht korrekt spekulierten Schreiboperation mit den Markierungen verglichen werden, die in dem Weiterleitungsverfolgungspuffer **309** für jüngere Ladeoperationen enthalten sind. Wenn die Markierung nicht korrekt spekulierten Schreiboperation mit einer Markierung übereinstimmt, die für eine jüngere Ladeoperation gespeichert ist, wodurch angegeben wird, dass die jüngere Ladeoperation in nicht korrekter Weise Daten auf der Grundlage der nicht korrekt spekulierten Adresse des Schreibbefehls weiter verwendet hat, kann diese Lade/Schreibeinheit **126c** veranlassen, dass die jüngere Ladeoperation erneut ausgeführt wird.

[0060] Fig. 3a zeigt eine Ausführungsform eines Verfahrens zum Wiederholen von anhängigen Operationen innerhalb einer Lade/Schreibeinheit in Reaktion auf das Erkennen einer nicht korrekten Datenspekulation. Bei **501** wird eine Angabe, dass die Datenspekulation für eine Operation nicht korrekt ist, erkannt. Beispielsweise empfängt eine Lade/Schreibeinheit einen neuen Wert der Adresse für die Operation aus einer Adressenerzeugungseinheit (ein Teil der Lade/Schreibeinheit in einigen Ausführungsformen ist). Der neue Adressenwert kann sich von einem spekulativen Adressenwert, der von der Lade/Schreibeinheit zum Erzeugen eines spekulativen Ergebnisses dieser Operation oder einer weiteren Operation verwendet wurde, unterscheiden, wodurch angezeigt wird, dass die an der Adresse ausgeführte Datenspekulation nicht korrekt ist. In einem weiteren Beispiel erkennt eine Lade/Schreibeinheit eine nicht korrekte Abhängigkeitsvorhersage, indem das spekulative Ergebnis einer Ladeoperation (die beispielsweise durch Vorhersagen erzeugt wurde, dass das Ladeergebnis gleich dem Quellenwert einer früheren Schreiboperation) mit dem tatsächlichen Ergebnis der Ladeoperation, dass durch Zugreifen auf den Daten-Cache-Speicher gewonnen wird, verglichen wird.

[0061] In der Ausführungsform aus Fig. 3a veranlasst die Lade/Schreibeinheit, dass alle jüngeren anhängigen Operationen innerhalb der Lade/Schreibeinheit erneut ausgeführt werden, wenn erkannt wird, dass die Datenspekulation für eine Operation nicht korrekt war, wie dies bei **503** gezeigt ist. Die Lade/Schreibeinheit kann ferner veranlassen, dass die jüngeren Operationen erneut ausgeführt werden, indem die Markierung jeder zu wiederholender Operation einer Disponiereinheit zugeleitet wird. Wenn die Markierungen von der Lade/Schreibeinheit bereitgestellt werden, kann die Disponiereinheit den Status der Operation in der Disponiereinheit aktualisieren (beispielsweise von „ausgegeben" zu „nicht ausgegeben" oder „muss erneut ausgegeben werden"). Ein derartiger Wiederholmechanismus ermöglicht es, dass die jüngeren Operationen erneut ausgeführt

werden, ohne dass diese aus der Bearbeitungspipeline des Mikroprozessors verworfen werden und erneut aus dem Befehls-Cache-Speicher **106** abgerufen werden müssen.

[0062] Fig. 3b zeigt eine weitere Ausführungsform eines Verfahrens zum erneuten Ausführen anhängiger Operationen in einer Lade/Schreibeinheit. Bei **505** wird eine Angabe erkannt, dass die Datenspekulation für eine Adresse einer Schreiboperation nicht korrekt ist. Wenn jüngere anhängige Ladeoperationen oder Schreiboperationen Adressen besitzen, die mit den spekulativen Wert oder dem neuen Wert der Adresse der nicht korrekt spekulierten Schreiboperation übereinstimmen, wie die bei **507** bestimmt wird, dann werden die übereinstimmenden Operationen erneut ausgeführt. Wenn keine Adresse der jüngeren Operationen mit den spekulativen Wert oder dem neuen Wert der Adresse der Schreiboperation übereinstimmt, werden die jüngeren Operationen nicht wiederholt, wie dies bei **509** angegeben ist.

[0063] Fig. 3c zeigt schematisch eine noch weitere Ausführungsform eines Verfahrens zum erneuten Ausführen anhängiger Operationen innerhalb einer Lade/Schreibeinheit. Bei **515** wird eine Angabe erkannt, dass die Datenspekulation für die Adresse einer Schreiboperation nicht korrekt ist. Wenn eine Adresse einer jüngeren Ladeoperation mit dem neuen Wert der Adresse der Schreiboperation übereinstimmt, werden die übereinstimmenden jüngeren Ladeoperationen erneut ausgeführt. Wenn eine jüngere Ladeoperation Daten aus der nicht korrekt spekulierten Schreiboperation weiter verwendet hat (beispielsweise wie dies erkannt werden kann, indem die Markierungen von Schreiboperationen, von denen die jüngeren Ladeoperationen Daten weiter benutzt haben, mit der Markierung der nicht korrekt spekulierten Schreiboperation verglichen werden), wie dies bei **521** gezeigt ist, werden die jüngeren Ladeoperationen erneut ausgeführt.

[0064] Zu beachten ist, dass andere Ausführungsformen in unterschiedlicher Weise ablaufen können, wie dies in den Fig. 3a bis Fig. 3c gezeigt ist. Beispielsweise werden in einer Ausführungsform ähnlich zu jener aus Fig. 3b lediglich jüngere Ladeoperationen, deren Adresse mit den spekulativen Wert oder den neuen Wert der Adresse der nicht korrekt spekulierten Schreiboperation übereinstimmen, erneut ausgeführt. In ähnlicher Weise werden in einer Ausführungsform ähnlich zu jener aus Fig. 3c jüngere Ladeoperationen und Schreiboperationen, deren Adressen mit dem neuen Wert der Adresse der nicht korrekt spekulierten Schreiboperation übereinstimmen, erneut ausgeführt. Es sind auch viele andere Variationen möglich.

Beispielhafte Computersysteme

[0065] Fig. 4 zeigt eine Blockansicht einer Ausführungsform eines Computersystems **900**, das einen Prozessor **100** aufweist, der mit einer Vielzahl von Systemkomponenten über eine Busbrücke **902** verbunden ist. Der Prozess **100** umfasst eine Ausführungsform einer Lade/Schreibeinheit, wie dies zuvor beschrieben ist. Andere Ausführungsformen eines Computersystems sind ebenfalls möglich und hierin mit eingeschlossen. In dem dargestellten System ist ein Hauptspeicher **200** mit der Busbrücke **902** über einen Speicherbus **906** verbunden, und eine Graphiksteuerung **908** ist mit der Busbrücke **902** über einen AGP-Bus **910** verbunden. Es sind diverse PCI-Geräte **912a** bis **912b** mit der Busbrücke **902** über einen PCI-Bus **914** verbunden. Eine zweite bzw. sekundäre Busbrücke **916** kann ebenfalls vorgesehen sein, um eine elektrische Schnittstelle zu einem oder mehreren EISA- oder ISA-Geräten **918** über einen EISA/ISA-Bus **920** bereitzustellen. In diesem Beispiel ist der Prozessor **10** mit der Busbrücke **902** über einen CPU-Bus **924** und mit einem optionalen L2-Cache-Speicher **928** verbunden. In einigen Ausführungsformen umfasst der Prozessor **100** einen integrierten L1-Cache-Speicher (nicht gezeigt).

[0066] Die Busbrücke **902** bildet eine Schnittstelle zwischen dem Prozessor **100**, dem Hauptspeicher **200**, der Graphiksteuerung **908** und den Geraten, die mit dem PCI-Bus **914** verbunden sind. Wenn eine Operation von einer der Einrichtungen empfangen wird, die mit der Busbrücke **902** verbunden ist, erkennt die Busbrücke **902** das Ziel der Operation (beispielsweise ein spezielles Gerät oder im Falle des PCI-Busses **914**, dass das Ziel auf dem PCI-Bus **914** liegt). Die Busbrücke **902** leitet die Operation an das Zielgerät weiter. Die Busbrücke **902** übersetzt im Allgemeinen eine Operation auf dem Protokoll, das von der ursprünglichen Einrichtung oder dem Bus verwendet wird, in das Protokoll, das von dem Zielgerät oder dem Zielbus verwendet wird.

[0067] Zusätzlich zur Bereitstellung einer Schnittstelle für den PCI-Bus **914** in Bezug auf einen ISA/EISA-Bus kann die sekundäre Busbrücke **916** weitere Funktionen enthalten. Eine Eingabe/Ausgabe-Steuerung (nicht gezeigt), die extern vorliegen kann oder in der sekundären Busbrücke **916** integriert sein kann, ist ebenfalls in dem Computersystem **900** vorgesehen, um eine zusätzliche Unterstützung für eine Tastatur und eine Maus **922** und diverse serielle und parallele Anschlüsse zu ermöglichen. Eine externe Cache-Speichereinheit (nicht gezeigt) kann mit dem CPU-Bus **924** zwischen dem Prozessor **100** und der Busbrücke **902** in weiteren Ausführungsformen angeschlossen sein. Alternativ ist der externe Cache-Speicher mit der Busbrücke **902** verbunden und eine Cache-Steuerlogik für den externen Cache-Speicher ist in der Busbrücke **902** integriert. Der

L2-Cache-Speicher **928** ist in Art einer Rückseitenkonfiguration im Hinblick auf den Prozessor **100** dargestellt. Zu beachten ist, dass der L2-Cache-Speicher **928** auch separat zu dem Prozessor **100** vorgesehen sein kann, oder er kann in einem Gehäuse (beispielsweise Einschub **1** oder Einschub A) im Zusammenwirken mit dem Prozessor **100** vorgesehen sein, oder er kann sogar gemeinsam auf einem Halbleitersubstrat mit dem Prozessor **100** vorgesehen sein.

[0068] Der Hauptspeicher **200** ist ein Speicher, in welchem Anwenderprogramme gespeichert sind und aus dem heraus der Prozessor **100** hauptsächlich arbeitet. Ein geeigneter Hauptspeicher **200** umfasst einen DRAM (dynamischen Speicher mit wahlfreiem Zugriff). Beispielsweise können mehrere Bänke aus SDRAM (synchrone DRAM) oder RAMBUS-DRAM (RDRAM) angewendet werden.

[0069] Die PCI-Geräte **912a** bis **912b** sind lediglich anschaulicher Natur für eine Vielzahl peripherer Geräte, etwa Netzwerkschnittstellenkarten, Videobeschleuniger, Audiokarten, Festplatten oder Diskettenlaufwerke oder Laufwerkssteuerungen, SCSI (Kleincomputersystemschnittstellen-) Adapter und Telefonkarten. In ähnlicher Weise ist das ISA-Gerät **918** stellvertretend für diverse Arten von peripheren Geräten zu sehen, etwa ein Modem, eine Klangkarte, eine Vielzahl von Datennahmekarten, etwa GPIB oder Feldbusschnittstellenkarten.

[0070] Die Graphiksteuerung **908** ist vorgesehen, um das Erzeugen von Text und Bildern auf einer Anzeige **926** zu steuern. Die Graphiksteuerung **908** kann einen typischen Graphikbeschleuniger repräsentieren, der im Allgemeinen bekannt ist, um dreidimensionale Datenstrukturen zu erzeugen, die effizient in den Hauptspeicher **200** geschrieben und daraus ausgelesen werden können. Die Graphiksteuerung **908** kann daher ein übergeordnetes Gerät bzw. ein Master des AGP-Busses **910** sein, dahingehend, dass die Steuerung Zugriff auf eine Zielschnittstelle innerhalb der Busbrücke **902** anfordern kann und diesen Zugriff auch erhält, um damit auch Zugriff auf den Hauptspeicher **200** zu erhalten. Ein spezieller Graphikbus kann das rasche Abrufen von Daten aus dem Hauptspeicher **200** ermöglichen. Für gewisse Operationen ist die Graphiksteuerung **908** ferner ausgebildet, PCI-Protokolltransaktionen auf dem AGP-Bus **910** zu erzeugen. Die AGP-Schnittstelle der Busbrücke **902** kann damit Funktionen enthalten, um sowohl AGP-Protokolltransaktionen sowie PCI-Protokollziel- und Initiatortransaktionen zu unterstützen. Die Anzeige **926** ist eine beliebige elektronische Anzeige, auf der Bild oder Text dargestellt werden können. Zu geeigneten Anzeigen **926** gehören eine Kathodenstrahlröhre („CRT"), eine Flüssigkristallanzeige („LCD"), etc.

[0071] Zu beachten ist, dass obwohl die AGP-, PCI- und ISA- oder EISA-Busse als Beispiele in der obigen Beschreibung verwendet sind, beliebige Busarchitekturen nach Bedarf eingesetzt werden können. Des weiteren ist zu beachten, dass das Computersystem **900** ein Multiverarbeitungscomputersystem mit zusätzlichen Prozessoren (beispielsweise einen Prozessor **100a**, der als eine optionale Komponente des Computersystems **900** gezeigt ist) aufweisen kann. Der Prozessor **100a** kann ähnlich zu dem Prozessor **100** sein. Insbesondere kann der Prozessor **100a** eine identische Kopie des Prozessors **100** sein. Der Prozessor **100a** kann mit der Busbrücke **902** über einen unabhängigen Bus (wie in Fig. 4 gezeigt) und kann den CPU-Bus **924** gemeinsam mit dem Prozessor **100** benutzen. Des weiteren kann der Prozessor **100a** mit einem optionalen L2-Cache-Speicher **928a** ähnlich dem L2-Cache-Speicher **928** verbunden sein.

[0072] Fig. 5 zeigt eine weitere Ausführungsform eines Computersystem **900**, das einen Prozessor **100** aufweist, der eine Ausführungsform einer Lade/Schreibeinheit besitzt, wie sie zuvor beschrieben ist. Es sind andere Ausführungsformen möglich und hierin mit eingeschlossen. In der Ausführungsform aus Fig. 5 umfasst das Computersystem **900** mehrere Verarbeitungsknoten **1012A**, **12012B**, **1212C** und **1012D**. Jeder Verarbeitungsknoten ist mit einem entsprechenden Speicher **200a** bis **200b** über eine Speichersteuerung **1016A** bis **1016D** verbunden, die jeweils in dem entsprechenden Verarbeitungsknoten **1012A** bis **1012D** vorgesehen ist. Des weitem enthalten die Verarbeitungsknoten **1012A** bis **1012D** Schnittstellenlogiken, die zur Kommunikation zwischen den Verarbeitungsknoten **1012A** bis **1012D** verwendet werden. Beispielsweise enthält der Verarbeitungsknoten **1012A** eine Schnittstellenlogik **1018A** zur Kommunikation mit dem Verarbeitungsknoten **1012B**, eine Schnittstellenlogik **1018B** zur Kommunikation mit dem Verarbeitungsknoten **1012C** und eine dritte Schnittstellenlogik **1018C** zur Kommunikation mit noch einem weiteren Verarbeitungsknoten (nicht gezeigt). In ähnlicher Weise umfasst der Verarbeitungsknoten **1012B** Schnittstellenlogiken **1018D**, **1018E** und **1018F**; der Verarbeitungsknoten **1012C** umfasst Schnittstellenlogiken **1018G**, **1018H** und **1018I**; und der Verarbeitungsknoten **1012D** umfasst Schnittstellenlogiken **1018J**, **1018K** und **1018L**. Der Verarbeitungsknoten **1012D** ist ferner ausgebildet, mit mehreren Eingabe/Ausgabe-Geräten (beispielsweise den Geräten **1020A** bis **1020B** in einer Prioritätskettenkonfiguration) über die Schnittstellenlogik **1018L** zu kommunizieren. Andere Verarbeitungsknoten können mit anderen I/O-Geräten in ähnlicher Weise in Verbindung treten.

[0073] Die Verarbeitungsknoten **1012A** bis **1012D** stellen eine paketbasierte Verbindung für die Kommunikation zwischen den Verarbeitungsknoten dar.

In der vorliegenden Ausführungsform wird die Verbindung als Gruppen aus unidirektionalen Leitungen eingerichtet (beispielsweise werden Leitungen **1024A** verwendete, um Pakete von dem Verarbeitungsknoten **1012A** zu dem Verarbeitungsknoten **1012B** zu übertragen, und Leitungen **1024B** werden verwendet, um Pakete von dem Verarbeitungsknoten **1012B** zu dem Verarbeitungsknoten **1012A** zu übertragen). Andere Gruppen aus Leitungen **1024C** bis **1024H** werden verwendet, um Pakete zwischen anderen Verarbeitungsknoten auszutauschen, wie dies in Fig. 5 gezeigt ist. Im Allgemeinen enthält jede Gruppe aus Leitungen **1024** eine oder mehrer Datenleitungen, eine oder mehrere Taktleitungen entsprechend den Datenleitungen und eine oder mehrere Steuerleitungen, die die Art der übertragenen Pakete kennzeichnen. Die Verbindung kann in einer mit dem Cache-Speicher kohärenten Weise zum Datenaustausch zwischen Verarbeitungsknoten oder in einer nicht kohärenten Weise zum Datenaustausch zwischen einem Verarbeitungsknoten und einem I/O-Gerät betrieben werden (oder einer Busbrücke zu einem I/O-Bus mit konventionellem Aufbau, etwa dem PCI-Bus oder dem ISA-Bus). Des weiteren kann die Verbindung in einer nicht-kohärenten Weise unter Anwendung einer Prioritätskettenstruktur zwischen I/O-Geräten betrieben werden, wie dies gezeigt ist. Zu beachten ist, dass ein Paket, das von einem Verarbeitungsknoten zu einem weiteren zu übermitteln ist, durch einen oder mehrere Zwischenknoten laufen kann. Beispielsweise kann ein von dem Verarbeitungsknoten **1012A** an den Verarbeitungsknoten **1012D** übertragenes Paket durch den Verarbeitungsknoten **1012B** oder dem Verarbeitungsknoten **1012C** laufen, wie dies in Fig. 5 gezeigt ist. Es kann ein beliebiger geeigneter Signalführungsalgorithmus angewendet werden. In anderen Ausführungsformen des Computersystems **900** können mehr oder weniger Verarbeitungsknoten als in der in Fig. 5 gezeigten Ausführungsform vorgesehen sein.

[0074] Im Allgemeinen werden die Pakete als eine oder mehrere Bit-Zeiten auf den Leitungen **1024** zwischen den Knoten ausgetauscht. Eine Bit-Zeit kann die ansteigende oder die abfallende Flanke des Taktsignals in den entsprechenden Taktleitungen sein. Die Pakete können Befehlspakete zum Initiieren von Transaktionen, Abfrage bzw. Sondierungspakete zur Beibehaltung der Cache-Kohärenz und Antwortpakete zum Antworten auf Fragen und Befehle enthalten.

[0075] Die Verarbeitungsknoten **1012A** bis **1012D** können zusätzlich zu einer Speichersteuerung und einer Schnittstellenlogik einen oder mehrere Prozessoren aufweisen. Allgemein gesagt, umfasst ein Verarbeitungsknoten mindestens einen Prozessor und umfasst optional eine Speichersteuerung zur Kommunikation mit einem Speicher und anderen Logikkomponenten nach Bedarf. Insbesondere kann jeder Verarbeitungsknoten **1012A** bis **1012D** eine oder mehrere Kopien des Prozessors aufweisen. Eine externe Schnittstelleneinheit kann die Schnittstellenlogik **1018** innerhalb der Knoten sowie die Speichersteuerung **1016** beinhalten.

[0076] Speicher **200A** bis **2000** können beliebige geeignete Speichereinrichtungen aufweisen. Beispielsweise können die Speicher **200A** bis **200D** ein oder mehrere RAMBUS-DRAM's (RDRAM's), synchrone DRAM's (SDRAM's), statische RAM's, etc. aufweisen. Der Adressenraum des Computersystems **900** wird auf die Speicher **200A** bis **200D** aufgeteilt. Jeder Verarbeitungsknoten **1012A** bis **1012D** kann eine Speicherzuordnung aufweisen, die verwendet wird zu bestimmen, welche Adressen welchen Speichern **200A** bis **200D** zugeordnet sind, und somit zu welchen Verarbeitungsknoten **1012A** bis **1012D** eine Speicheranforderung für eine spezielle Adresse zuzuleiten ist. In einer Ausführungsform ist der Kohärenzpunkt für eine Adresse innerhalb des Computersystems **900** die Speichersteuerung **1016A** bis **1016D**, die mit dem Speicher verbunden ist, der Bytes entsprechend der Adresse enthält. Anders ausgedrückt, die Speichersteuerung **1016A** bis **1016D** ist dafür verantwortlich sicherzustellen, dass jeder Speicherzugriff auf den entsprechenden Speicher **200A** bis **200D** einer mit dem Cache-Speicher kohärenten Weise erfolgt. Die Speichersteuerungen **1016A** bis **1016D** umfassen eine Steuerschaltung zur Kommunikation mit den Speichern **200A** bis **200D**. Ferner können die Speichersteuerungen **1016A** bis **1016D** Anforderungswarteschlangen zum Speichern von Speicheranforderungen aufweisen.

[0077] Die Schnittstellenlogiken **1018A** bis **1018L** können eine Vielzahl von Puffer zum Empfangen von Paketen aus der Verbindung und zum Zwischenspeichern der Pakete, die über die Verbindung zu senden sind, aufweisen. Das Computersystem **900** umfasst ferner einen geeigneten Ablaufsteuerungsmechanismus zum Senden von Paketen. Beispielsweise speichert in einer Ausführungsform jede Schnittstellenlogik **1018** die Nummer jeder Art von Puffer innerhalb des Empfängers am anderen Ende der Verbindung, mit dem die Schnittstellenlogik verbunden ist. Die Schnittstellenlogik sendet kein Paket, sofern nicht die empfangende Schnittstellenlogik einen freien Pufferplatz zur Speicherung des Pakets aufweist. Wenn ein empfangender Puffer freigegeben wird, indem ein Paket weitergeleitet wird, sendet die empfangende Schnittstellenlogik eine Nachricht zu der sendenden Schnittstellenlogik, um anzuzeigen, dass der Puffer verfügbar ist. Ein derartiger Mechanismus wird als ein „Coupon-basiertes" System bezeichnet.

[0078] Die I/O-Geräte **1020A** bis **1020B** können beliebige geeignete I/O-Geräte sein. Beispielsweise beinhalten die I/O-Geräte **1020A** bis **1020B** Einrichtungen zur Kommunikation mit einem anderen Computersystem, mit dem die Geräte verbunden sind (bei-

spielsweise Netzwerkschnittstellenkarten oder Modems). Ferner können zu den I/O-Geräten **1020A** bis **1020B** gehören: Wiederbeschleuniger, Audiokarten, Festplatten oder Diskettenlaufwerke oder Laufwerkssteuerungen, SCSI-(Kleincomputersystemschnittstellenadapter) und Telefonkarten, Klangkarten und eine Vielzahl von Datenahmekarten, etwa GPIB- oder Feldbusschnittstellenkarten. Zu beachten ist, dass der Begriff „I/O-Gerät" und der Begriff "peripheres Gerät" hierin Synonyme verwendet sind.

[0079] Im hierin verwendeten Sinne bezeichnet der Begriff „Taktzyklus" ein Zeitintervall, in welchem diverse Stufen der Befehlsverarbeitungspipelines bzw. Bildverarbeitungsvektorstrukturen ihre Aufgaben abschließen. Befehle und berechnete Werte werden von Speicherelementen, etwa Register oder Arrays, entsprechend einem Taktsignal, das den Taktzyklus definiert, aufgenommen. Beispielsweise kann ein Speicherelement gemäß der ansteigenden oder abfallenden Flanke des Taktsignals aufnehmen.

[0080] In der vorhergehenden E Läuterung werden Signale als „gesetzt" beschrieben. Ein Signal kann als gesetzt definiert werden, wenn es einen Wert übermittelt, der eine spezielle Information kennzeichnet. Ein spezielles Signal kann als gesetzt definiert werden, wenn es einen binären Wert 1 oder alternativ wenn es einen binären Wert 0 überträgt.

[0081] Der Fachmann erkennt, dass diverse Variationen und Modifizierungen bei Berücksichtigung der vorliegenden Offenbarung möglich sind. Es ist beabsichtigt, dass die folgenden Patenansprüche alle derartigen Variationen und Modifizierungen mit einschließen.

Industrielle Anwendbarkeit

[0082] Diese Erfindung kann im Allgemeinen auf das Gebiet der Mikroprozessoren angewendet werden.

Patentansprüche

1. Mikroprozessor (**100**) mit:
einer Disponiereinheit (**118**), die zum Ausgeben von Operationen ausgebildet ist;
einer Lade/Schreibeinheit (**126c**), die angeschlossen und ausgebildet ist, von der Disponiereinheit ausgegebene Speicheroperationen zu empfangen und die Speicheroperationen auszuführen;
wobei die Lade/Schreib-Einheit ferner ausgebildet ist, Informationen zu speichern, die mehrere an die Lade/Schreib-Einheit ausgegebene Speicheroperationen kennzeichnen; **dadurch gekennzeichnet**, dass beim Erkennen einer nicht korrekten Datenspekulation für eine der mehreren Speicheroperationen die Lade/Schreib-Einheit ausgebildet ist, eine Wiederholangabe für die Disponiereinheit bereitzustellen, die angibt, dass mindestens eine der mehreren Speicheroperationen innerhalb der Lade/Schreib-Einheit erneut auszugeben ist; und
wobei die Disponiereinheit ferner ausgebildet ist, die mindestens eine der mehreren Speicheroperationen in Reaktion auf die von der Lade/Schreib-Einheit erhaltenen Angabe erneut auszugeben.

2. Mikroprozessor nach Anspruch 1, wobei die Lade/Schreibeinheit ausgebildet ist, die Wiederholangabe zu erzeugen, indem jede der mehreren Speicheroperationen mit einer Adresse, die mit einer Adresse der einen der mehreren Speicheroperationen übereinstimmt, ermittelt wird.

3. Mikroprozessor nach Anspruch 1, wobei die Lade/Schreib-Einheit ausgebildet ist, die Wederholangabe zu erzeugen, indem jede der mehreren Speicheroperationen mit einer Adresse, die mit einem spekulativen Wert einer Adresse der einen der mehreren Speicheroperationen oder einem neuen Wert der Adresse der einen der mehreren Speicheroperationen übereinstimmt, ermittelt wird.

4. Mikroprozessor nach Anspruch 1, wobei die Lade/Schreib-Einheit ausgebildet ist, die Wiederholangabe zu erzeugen, indem jede der mehreren Speicheroperationen, die Ladeoperationen sind und die eine Adresse aufweisen, die mit einer Adresse der einen der mehreren Speicheroperationen übereinstimmt, ermittelt wird.

5. Mikroprozessor nach Anspruch 1, wobei die Lade/Schreib-Einheit ausgebildet ist, zu überwachen, welche Ladeoperationen, die in den mehreren Speicheroperationen enthalten sind, Daten von Schreiboperationen weitergeleitet haben; und wobei, wenn eine Adresse einer Schreiboperation als nicht korrekt spekuliert erkannt wird, die Lade/Schreib-Einheit ausgebildet ist, die Wiederholangabe zu erzeugen, indem eine jüngere der Ladeoperationen, die von der Schreiboperation Daten weiterleitete, ermittelt wird.

6. Computersystem (**900**) mit:
einem Speicher (**200**); und
einem mit dem Speicher gekoppelten Prozessor (**100**),
dadurch gekennzeichnet, dass der Prozessor umfasst:
eine Disponiereinheit (**118**), die zum Ausgeben von Speicheroperationen ausgebildet ist;
eine Lade/Schreib-Einheit (**126c**), die angeschlossen und ausgebildet ist, von der Disponiereinheit ausgegebene Speicheroperationen zu empfangen und die Speicheroperationen auszuführen;
wobei die Lade/Schreib-Einheit ausgebildet ist, Information zu speichern, die mehrere an die Lade/Schreib-Einheit ausgegebene Speicheroperationen kennzeichnet;

dadurch gekennzeichnet, dass in Reaktion auf das Erkennen einer nicht korrekten Datenspekulation für eine der mehreren Speicheroperationen die Lade/Schreib-Einheit ausgebildet ist, eine Wiederholangabe an die Disponiereinheit auszugeben, die angibt, dass mindestens eine der mehreren Speicheroperationen in der Lade/Schreib-Einheit erneut auszugeben ist; und
wobei die Disponiereinheit ausgebildet ist, die mindestens eine der mehreren Speicheroperationen in Reaktion auf die Angabe aus der Lade/Schreib-Einheit erneut auszugeben.

7. Verfahren mit:
Ausgeben einer Operation durch eine Disponiereinheit (**118**) an eine Lade/Schreib-Einheit (**126c**) zur Ausführung der Operation;
Speichern von Information durch die Lade/Schreib-Einheit, wobei die Information mehrere Speicheroperationen, die an die Lade/Schreib-Einheit ausgegeben werden, kennzeichnet;
dadurch gekennzeichnet, dass
die Lade/Schreib-Einheit erkennt, dass eine an der Operation ausgeführte Datenspekulation nicht korrekt ist;
in Reaktion auf dieses Erkennen die Lade/Schreibeinheit eine Wiederholangabe erzeugt, die mindestens eine Operation kennzeichnet, die aktuell an die Lade/Schreib-Einheit ausgebeben ist, und die Wiederholangabe an die Disponiereinheit ausgibt; und
in Reaktion auf das Empfangen der Wiederholangabe die Disponiereinheit die mindestens eine Operation erneut an die Lade/Schreib-Einheit ausgibt.

8. Verfahren nach Anspruch 7, wobei das Erzeugen umfasst, dass die Lade/Schreib-Einheit eine oder mehrere Wiederholangaben erzeugt, die alle anhängigen Operationen innerhalb der Lade/Schreib-Einheit kennzeichnen, die jünger sind als die Operation, für die die Datenspekulation nicht korrekt ist.

9. Verfahren nach Anspruch 7, wobei das Erzeugen umfasst, dass die Lade/Schreib-Einheit eine oder mehrere Wiederholangaben erzeugt, die anhängige Operationen in der Lade/Schreib-Einheit kennzeichnen, die eine Adresse besitzen, die mit einer Adresse der Operation übereinstimmt, für die die Datenspekulation nicht korrekt ist.

10. Verfahren nach Anspruch 7, wobei das Erzeugen umfasst, dass die Lade/Schreib-Einheit eine oder mehrere Wiederholangaben erzeugt, die eine anhängige Operation innerhalb der Lade/Schreib-Einheit mit einer Adresse kennzeichnen, die mit einem spekulativen Wert einer Adresse der Operation übereinstimmt, für die die Datenspekulation nicht korrekt ist.

Es folgen 6 Blatt Zeichnungen

Anhängende Zeichnungen

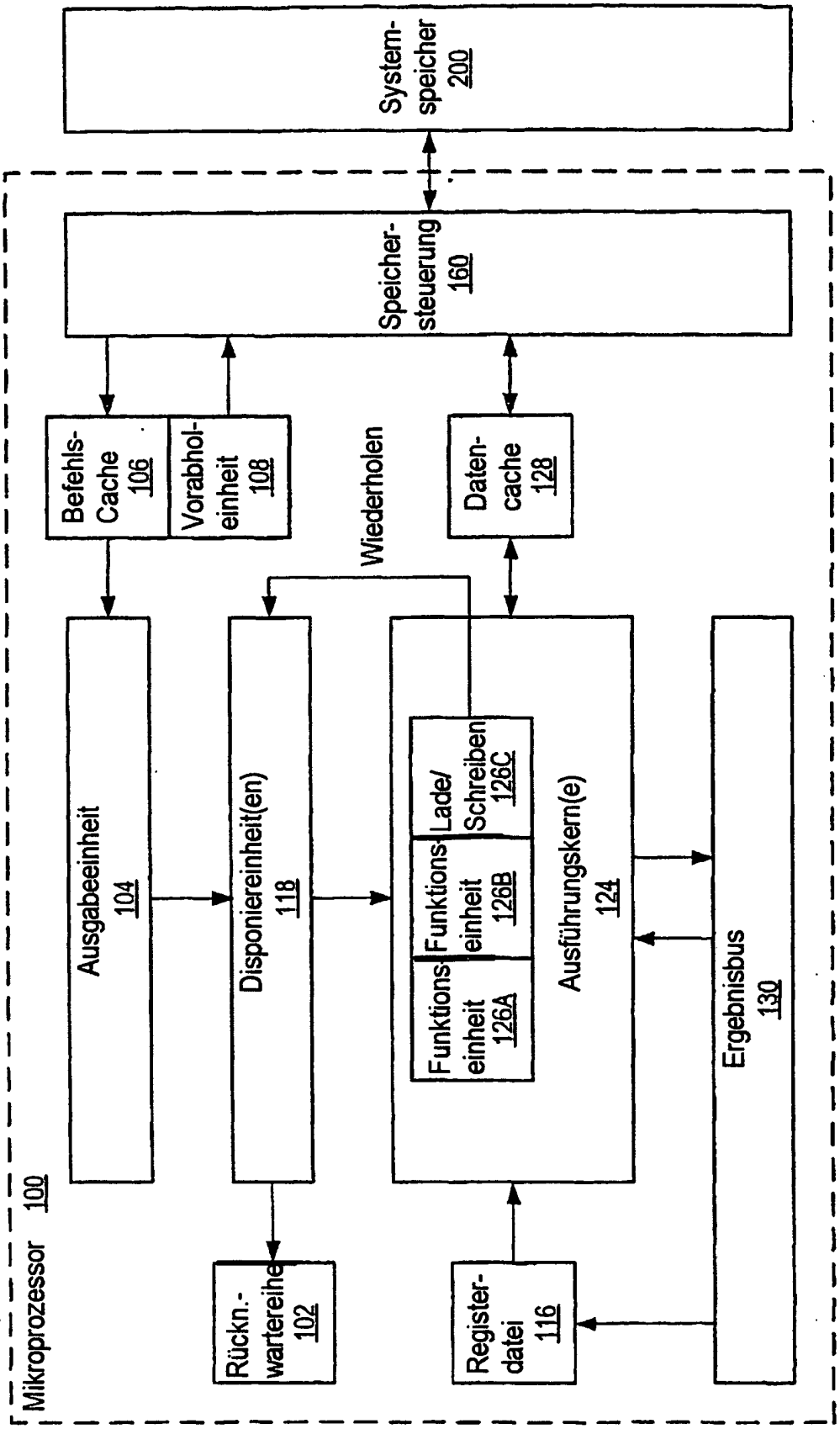

FIG. 1

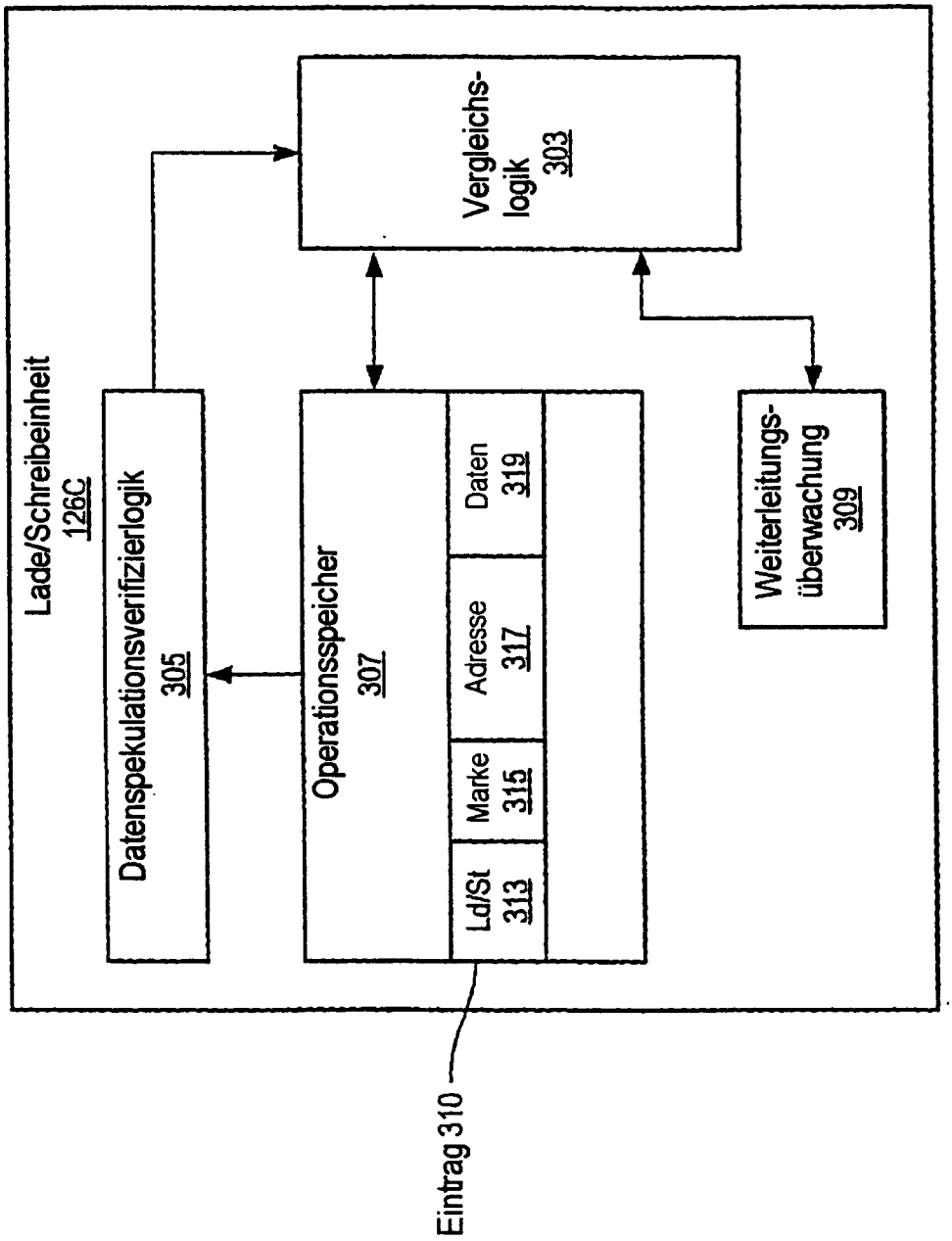
Lade/Schreibeinheit
126C
Vergleichs-
logik
303
Datenspekulationsverifizierlogik
305
Operationsspeicher
307
Ld/St
313
Marke
315
Adresse
317
Daten
319
Weiterleitungs-
überwachung
309
Eintrag 310

FIG. 2

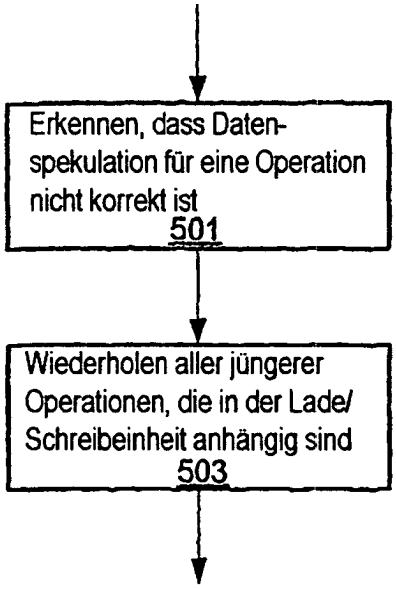
Erkennen, dass Daten-spekulation für eine Operation nicht korrekt ist
501
Wiederholen aller jüngerer Operationen, die in der Lade/ Schreibeinheit anhängig sind
503

FIG. 3A

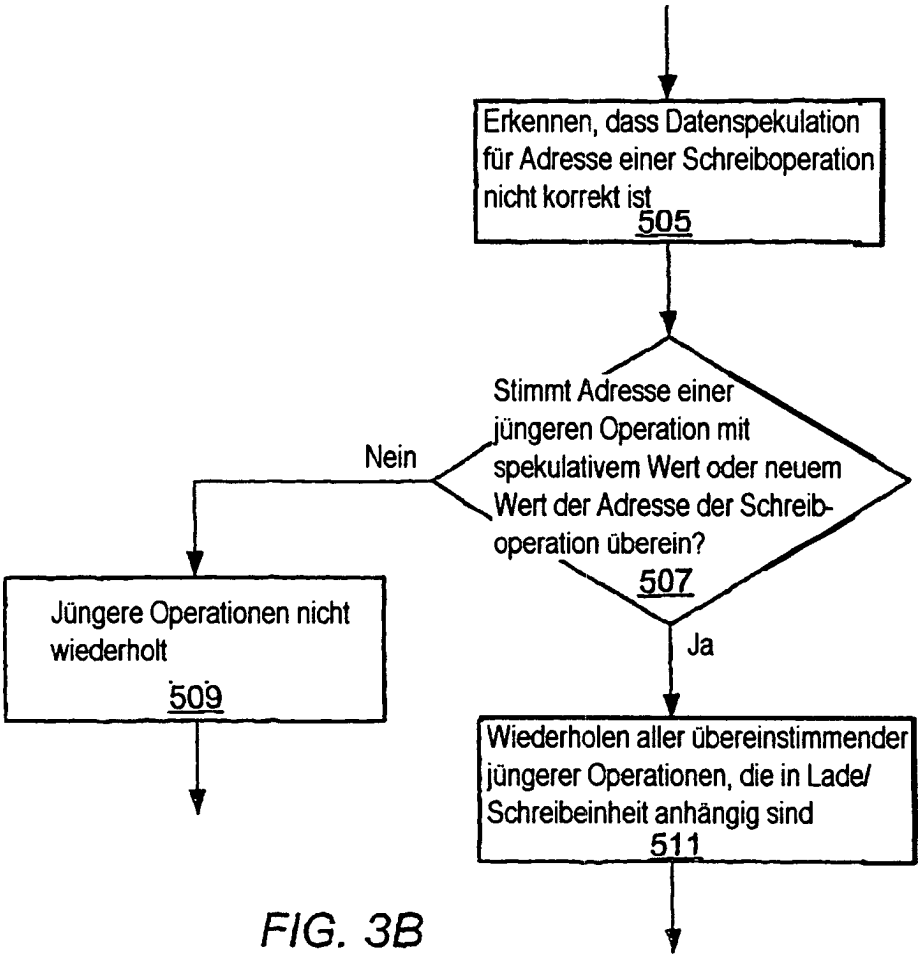
Erkennen, dass Datenspekulation für Adresse einer Schreiboperation nicht korrekt ist
505
Stimmt Adresse einer jüngeren Operation mit spekulativem Wert oder neuem Wert der Adresse der Schreib-operation überein?
507
Nein
Ja
Jüngere Operationen nicht wiederholt
509
Wiederholen aller übereinstimmender jüngerer Operationen, die in Lade/ Schreibeinheit anhängig sind
511

FIG. 3B

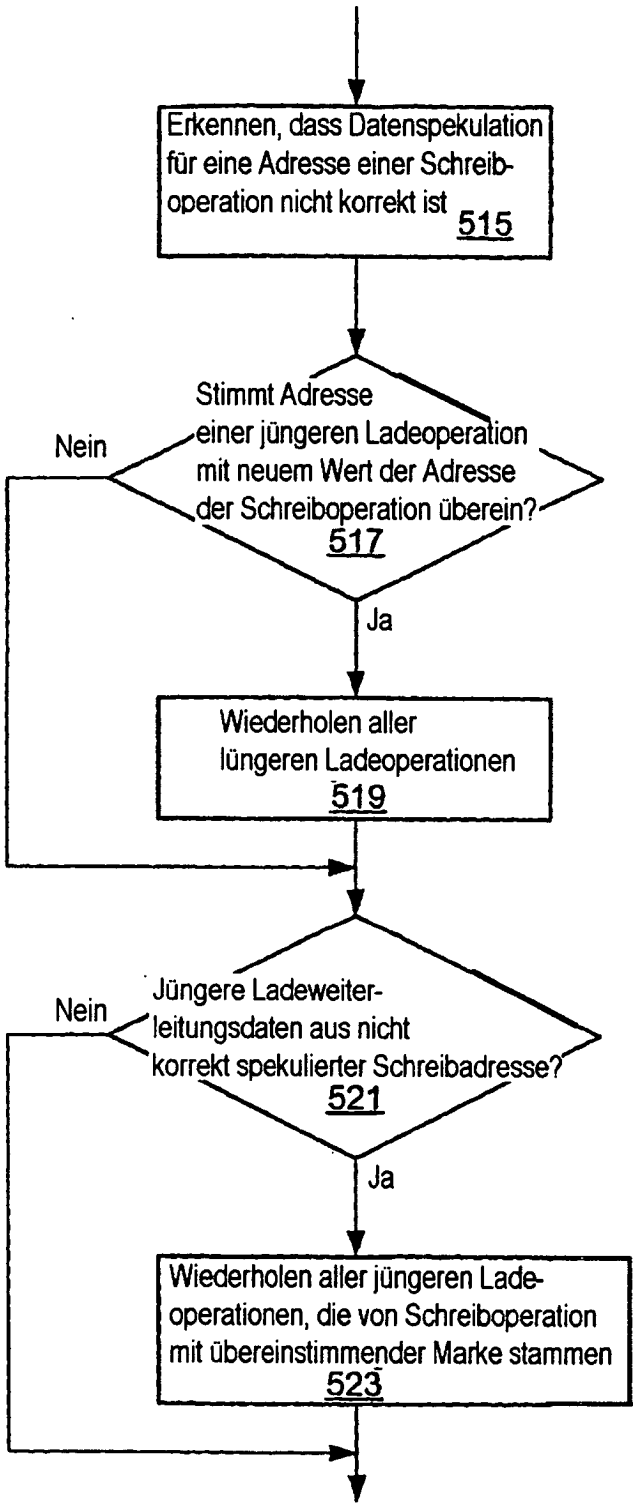
Erkennen, dass Datenspekulation für eine Adresse einer Schreib-operation nicht korrekt ist
515
Stimmt Adresse einer jüngeren Ladeoperation mit neuem Wert der Adresse der Schreiboperation überein?
517
Nein
Ja
Wiederholen aller lüngeren Ladeoperationen
519
Jüngere Ladeweiter-leitungsdaten aus nicht korrekt spekulierter Schreibadresse?
521
Nein
Ja
Wiederholen aller jüngeren Lade-operationen, die von Schreiboperation mit übereinstimmender Marke stammen
523

FIG. 3C

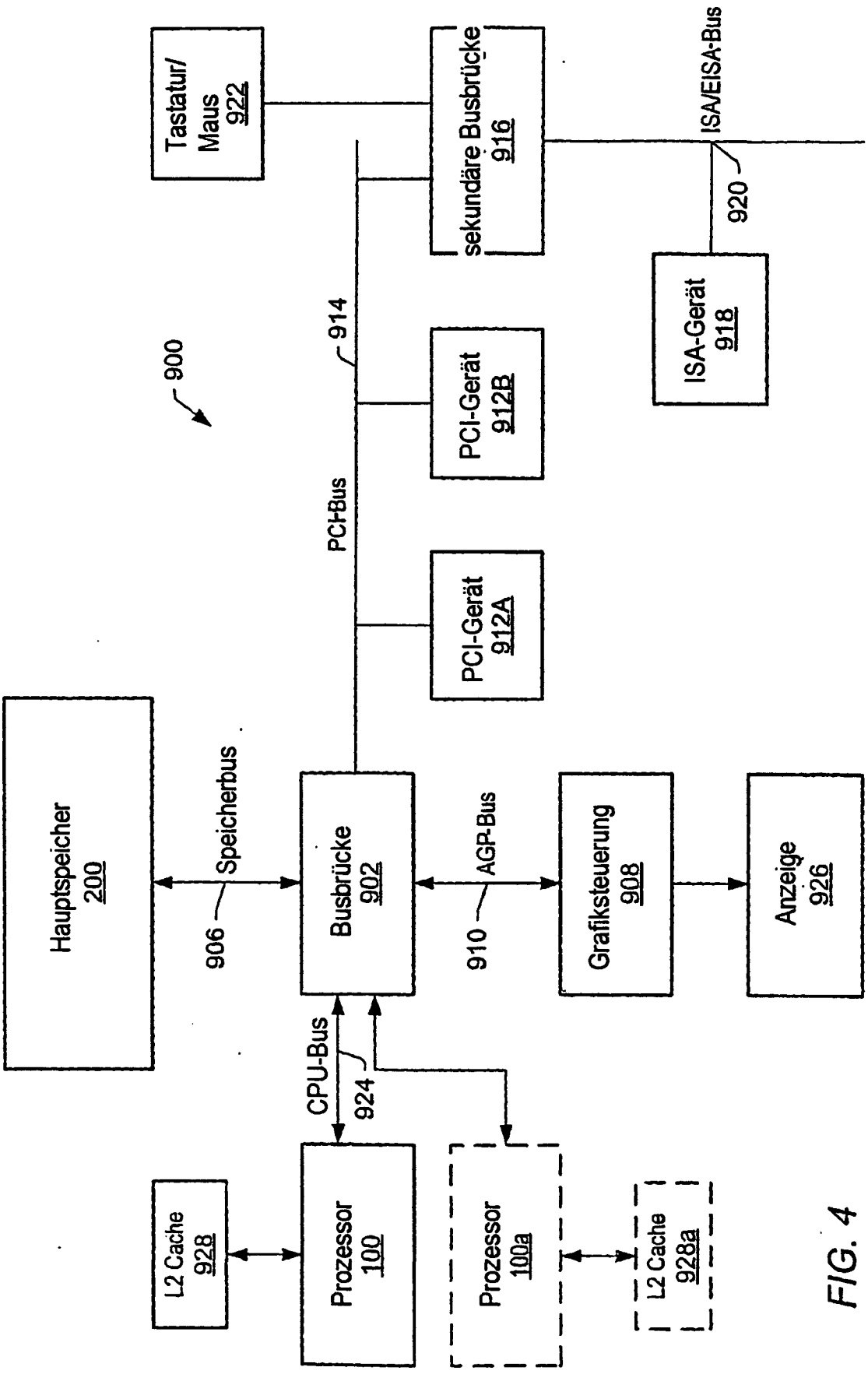
Tastatur/
Maus
922
sekundäre Busbrücke
916
ISA/EISA-Bus
920
ISA-Gerät
918
900
914
PCI-Gerät
912B
PCI-Bus
PCI-Gerät
912A
Hauptspeicher
200
Speicherbus
906
Busbrücke
902
AGP-Bus
910
Grafiksteuerung
908
Anzeige
926
CPU-Bus
924
L2 Cache
928
Prozessor
100
Prozessor
100a
L2 Cache
928a

FIG. 4

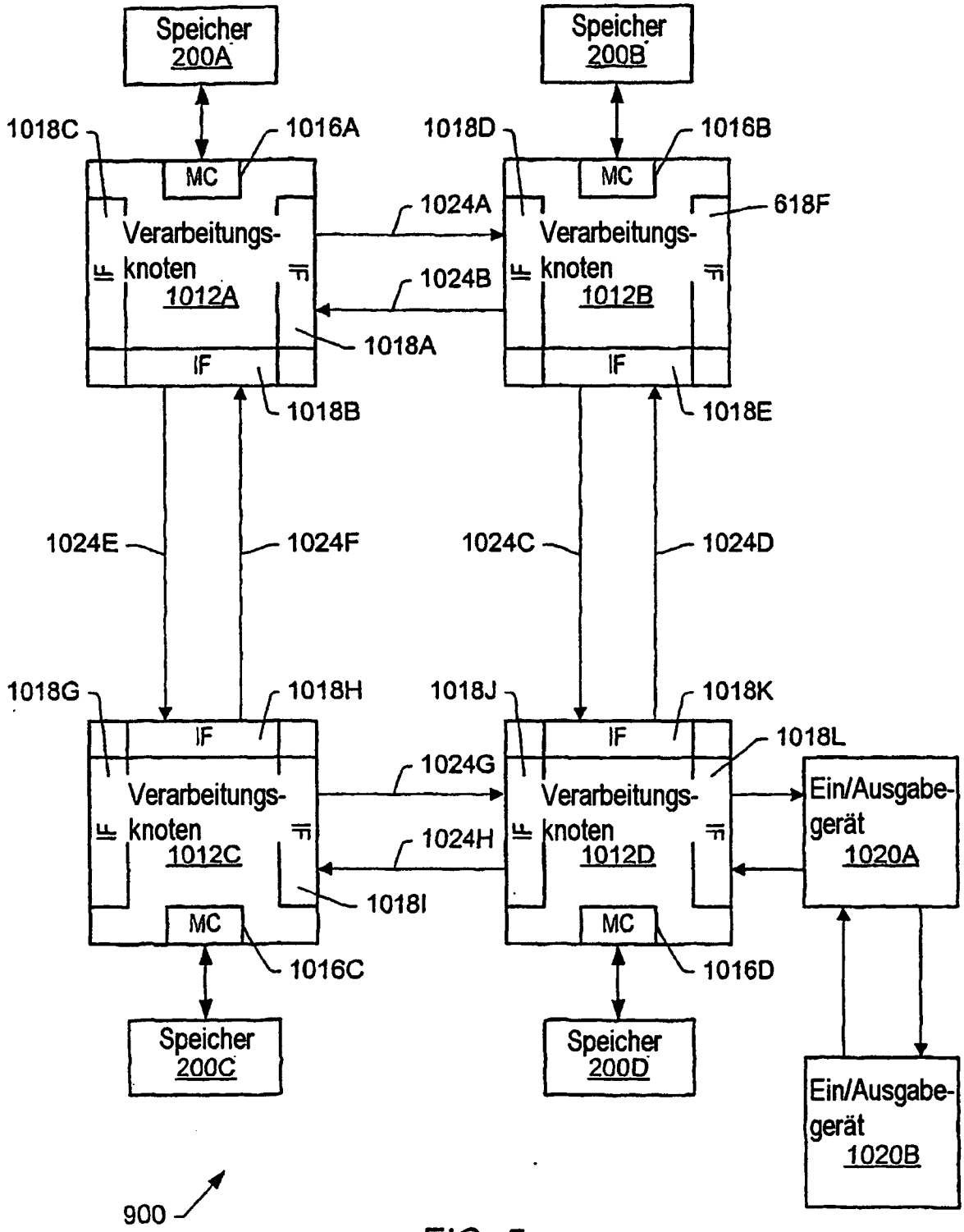
Speicher
200A
Speicher
200B
1018C
1016A
1018D
1016B
MC
MC
1024A
618F
Verarbeitungs-
knoten
1012A
1024B
Verarbeitungs-
knoten
1012B
IF
IF
IF
IF
1018A
1018B
1018E
1024E
1024F
1024C
1024D
1018G
1018H
1018J
1018K
IF
IF
1018L
1024G
Verarbeitungs-
knoten
1012C
1024H
Verarbeitungs-
knoten
1012D
Ein/Ausgabe-
gerät
1020A
1018I
MC
MC
1016C
1016D
Speicher
200C
Speicher
200D
Ein/Ausgabe-
gerät
1020B
900

FIG. 5