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

(10) **DE 44 10 060 B4** 2006.02.09

(12) **Patentschrift**

(21) Aktenzeichen: **P 44 10 060.4**
(22) Anmeldetag: **23.03.1994**
(43) Offenlegungstag: **13.10.1994**
(45) Veröffentlichungstag der Patenterteilung: **09.02.2006**

(51) Int Cl.[8]: ***G06F 12/10*** (2006.01)

Innerhalb von drei Monaten nach Veröffentlichung der Patenterteilung kann nach § 59 Patentgesetz gegen das Patent Einspruch erhoben werden. Der Einspruch ist schriftlich zu erklären und zu begründen. Innerhalb der Einspruchsfrist ist eine Einspruchsgebühr in Höhe von 200 Euro zu entrichten(§ 6 Patentkostengesetz in Verbindung mit der Anlage zu § 2 Abs. 2 Patentkostengesetz).

(30) Unionspriorität:
08/044,854 08.04.1993 US

(73) Patentinhaber:
Hewlett-Packard Development Co., L.P., Houston, Tex., US

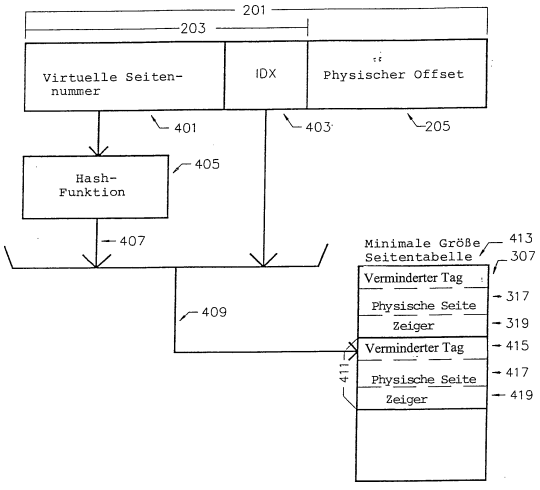
(74) Vertreter:
BOEHMERT & BOEHMERT, 80336 München

(72) Erfinder:
Morris, Dale C., Menlo Park, Calif., US; Huck, Jerome C., Palo Alto, Calif., US; Bryg, William R., Saratoga, Calif., US

(56) Für die Beurteilung der Patentfähigkeit in Betracht gezogene Druckschriften:
US 50 60 137 A
US 42 15 402 A
EP 02 12 129 A2

(54) Bezeichnung: **Übersetzungsvorrichtung zum Umsetzen einer virtuellen Speicheradresse in eine physikalische Speicheradresse**

(57) Hauptanspruch: Übersetzungsvorrichtung zum Umsetzen einer virtuellen Speicheradresse in eine physikalische Speicheradresse, mit einer Seitentabelle (413) mit einem Adressregister zum Speichern eines virtuellen Tags (203), wobei die Anzahl der Bits des virtuellen Tags die Adressenbreite des Rechners überschreitet,
wobei der virtuelle Tag (203) einen verminderten Tagabschnitt (401), dessen Bitbreite gleich oder kleiner als die Adressenbreite des Rechners ist, und einen Indexabschnitt (403) aufweist, welcher die restlichen Bits des virtuellen Tags umfaßt,
eine erste Hash-Funktion (405) eine erste Ausgangsfolge (407) aus den Bits des verminderten Tagabschnitts (401) erzeugen kann,
ein Seitentabellen-Index (409; 507) aus der ersten Ausgangsfolge (407) und der Information des Indexabschnittes (403) kombiniert wird und verschiedene Sets in der Seitentabelle (413) indiziert, und
wobei die Sets mindestens ein Paar aus einem verminderten Tag-Eintrag (307, 415) und der zugeordneten physikalischen Adresse und mindestens einen Zeiger (41) aufweisen.



Beschreibung

[0001] Die Erfindung betrifft eine Übersetzungsvorrichtung für die Übersetzung virtueller Adressen in physische Adressen.

[0002] Herkömmliche Rechnersysteme verwenden einen virtuellen Speicher, welcher mehr Speicher simuliert, als tatsächlich vorhanden ist, und ermöglicht, daß der Rechner unabhängig von deren Größe mehrere Programme gleichzeitig abarbeitet. Simultane Benutzerprogramme greifen auf Hauptspeicher-Adressen über virtuelle Adressen zu, welche von dem Betriebssystem zugewiesen worden sind. Die Abbildung der virtuellen Adresse auf den Hauptspeicher oder die physische Adresse ist ein Vorgang, der als virtuelle Adressenübersetzung oder Adreßumrechnung bekannt ist. Die virtuelle Speicherübersetzung kann mit einer großen Anzahl von Techniken erreicht werden, so daß der Prozessor auf die gewünschte Information im Hauptspeicher zugreifen kann.

[0003] Die virtuellen Adreß- und die physischen Adreßbereiche werden normalerweise in gleichgroße Speicherblöcke aufgeteilt, welche Seiten genannt werden, und eine Seitentabelle sieht die Übersetzung oder Umrechnung zwischen der virtuellen Adresse und der physischen Adresse vor. Seitentabellen können gemäß verschiedenen Strukturen organisiert werden. Auf "vorwärts-abgebildete" Tabellen wird am leichtesten zugegriffen, indem die virtuelle Seitennummer als ein Zeiger zu dem Tabelleneintrag verwendet wird, der die Übersetzung enthält. Auf "rückwärts-abgebildete" Tabelle wird am leichtesten zugegriffen, indem die physische Seitennummer als ein Zeiger zu dem Tabelleneintrag verwendet wird, welcher die Übersetzung enthält. Da es wesentlich mehr mögliche virtuelle Seitennummern als physische Seitennummern gibt, können die vorwärtsabgebildeten Tabellen sehr groß und dünn besetzt sein, sie sind jedoch recht leicht absuchbar, wenn die virtuelle Seitennummer gegeben ist. Eine rückwärts-abgebildete Tabelle enthält einen Eintrag für jede Seite des physischen Speichers. Da es eine begrenzte Anzahl von physischen Seiten, im Vergleich zu virtuellen Seiten, gibt, haben rückwärtsabgebildete Tabellen tendenziell einen effizienteren Speicheraufbau, der Zugriff auf sie ist jedoch schwieriger, wenn nur die virtuelle Seitennummer gegeben ist.

[0004] Jeder Seitentabellen-Eintrag enthält im allgemeinen die virtuelle Adresse und die physische Adresse und Schutz- und Zustandsinformationen, welche diese Seite betrifft. Der Zustand umfaßt im allgemeinen Information über die Art der Zugriffe, welche diese Seite erfahren hat. Ein "schmutziges" oder Modifikations-Bit (dirty bit) zeigt beispielsweise an, daß Daten in der Seite modifiziert worden sind. Da die Seitentabellen im allgemeinen groß sind, werden sie im Hauptspeicher gespeichert. Jeder vorschriftsmäßige Speicherzugriff kann daher tatsächlich zwei Zugriffe erfordern, einen, um die Übersetzung zu erhalten, und einen zweiten, um auf die physische Speicherstelle zuzugreifen.

[0005] Viele Rechnersysteme, welche eine virtuelle Adressenübersetzung unterstützen, benutzen Übersetzungs-Vorgriff-Puffer (TLB). Der TLB ist ein kleiner, schneller, assoziativer (inhaltsorientierter) Speicher, der normalerweise auf oder in unmittelbarer Nähe der Prozessoreinheit angeordnet ist und gerade verwendete Paare von virtuellen und physischen Adressen speichert. Der TLB enthält eine Untermenge der Übersetzungen in der Seitentabelle, und auf ihn kann wesentlich schneller zugegriffen werden. Wenn die Verarbeitungseinheit Information vom Hauptspeicher braucht, sendet sie die virtuelle Adresse zum TLB. Der TLB nimmt die Seitennummer der virtuellen Adresse entgegen und gibt eine physische Seitennummer zurück. Die physische Seitennummer wird mit der niederwertigen Adresseninformation kombiniert, um auf das gewünschte Byte oder Wort im Hauptspeicher zuzugreifen.

[0006] In den meisten Fällen kann der TLB nicht die gesamte Seitentabelle enthalten, so daß Prozeduren implementiert werden müssen, um den TLB zu aktualisieren. Wenn auf eine virtuelle Seite zugegriffen wird, für welche die Übersetzung nicht im TLB vorhanden ist, wird auf die Seitentabelle zugegriffen, um die Übersetzung dieser virtuellen Seitennummer in eine physische Seitennummer zu ermitteln, und diese Information wird in den TLB eingegeben. Der Zugriff auf die Seitentabelle kann zwanzig mal länger dauern als der Zugriff auf den TLB, und die Programmausführungsgeschwindigkeit wird daher optimiert, wenn die gerade verwendeten Übersetzungen im TLB aufbewahrt werden.

[0007] Die meisten Rechnersysteme verwenden heute irgendeine Art von Massenspeicher, im allgemeinen eine Platte, um den physischen Direktzugriffsspeicher (RAM) im Rechner zu vergrößern. Diese Vergrößerung des Hauptspeichers ermöglicht, daß größere Programme implementiert werden, als wenn nur der Hauptspeicher zur Verfügung stünde. Zusätzlich ist ein Plattenspeicher wesentlich weniger teuer als ein RAM, er ist jedoch auch hundert mal langsamer. Abhängig von der Länge eines Programmes und dem Wettbewerb mit anderen Programmen um den Hauptspeicher kann sich zu einem bestimmten Zeitpunkt ein Teil eines Programmes im Hauptspeicher befinden, und ein Teil kann im Plattenspeicher vorhanden sein. Die Teile eines Programmes, auf welche sofort zugegriffen werden muß, werden in den Hauptspeicher gebracht, während die Teile, welche momentan nicht benötigt werden, im Plattenspeicher gelassen werden.

[0008] Wenn beispielsweise ein einzelnes Programm zwei Megabyte lang ist und auf einem Rechner eingesetzt wird, der einen Hauptspeicher von einem Megabyte hat, wird das Programm zwei Megabyte virtueller Adressen verwenden. Da der Hauptspeicher nur Information von einem Megabyte enthalten kann, würde die verbleibende Programminformation von einem Megabyte in dem Plattenspeichersystem gespeichert werden. Zu jedem Zeitpunkt kann dann eine Hälfte der virtuellen Adressen auf Programminformation im Hauptspeicher abgebildet werden, und eine Hälfte kann nicht abgebildet werden, da eine Hälfte des Programms den vorhandenen physischen Speicher füllt. Der Zugriff auf die Information im Hauptspeicher läuft normal ab. D.h., zunächst wird in dem TLB gesucht, um zu sehen, ob er die Übersetzung enthält, und wenn er sie nicht enthält, wird der TLB mit der Information in der Seitentabelle aktualisiert, und dann wird wieder auf den TLB Bezug genommen, um die Übersetzungs-Information zu bekommen.

[0009] Wenn ein Zugriff auf Information erfolgt, die nicht im Hauptspeicher ist, wird zuerst im TLB nach der Übersetzung gesucht, die darin nicht enthalten sein wird. Dann wird auf die Seitentabelle Bezug genommen, um die Übersetzungs-Information zu bekommen, um den TLB zu aktualisieren. Die Seitentabelle enthält jedoch nur die Übersetzungen für die Information im Hauptspeicher, und daher wird sie nicht die erforderliche Übersetzungs-Information enthalten. Dieser Umstand wird Seitenfehler genannt. Abhängig von einem Seitenfehler wird der virtuellen Seite, auf welche Bezug genommen wurde, eine physische Seite zugeordnet, und diese Information wird in das Seitenverzeichnis eingefügt. Wenn alle physischen Seiten bereits mit anderen virtuellen Seiten verbunden worden sind, dann muß die Seitenfehler-Steuerung (handler) wählen, welche der physischen Seiten der Seite mit der virtuellen Adresse, auf welche momentan Bezug genommen wird, neu zugeordnet werden soll. Es gibt viele Algorithmen für eine derartige Auswahl, beispielsweise FIFO (first-in-first-out)-Algorithmen und am wenigsten-zuletzt-verwendet (least-recently-used)-Algorithmen. Die Seitenfehler-Steuerung ist im allgemeinen in Software implementiert, während das TLB-Aktualisierungsverfahren entweder mittels Hardware oder mittels Software gehandhabt werden kann, wie in der Industrie allgemein bekannt ist.

[0010] Fig. 1 zeigt das oben beschriebene Verfahren. In Schritt **112** wird dem TLB eine virtuelle Adresse vorgegeben. Wenn die Übersetzung für diese virtuelle Adresse im TLB vorhanden ist (TLB-Treffer), wird die zugeordnete physische Adresse vom TLB abgeleitet und für den Zugriff auf den physischen Speicher (Schritt **114**) verwendet. Wenn die Übersetzung für diese virtuelle Adresse nicht im TLB ist (TLB-Fehltreffer), wird die Übersetzung für diese Adresse in dem Seitenverzeichnis/Tabelle gesucht bzw. nachgeschlagen (Schritt **116**). Wenn die Übersetzung im Seitenverzeichnis ist, wird diese Information in den TLB eingefügt (Schritt **118**), und die virtuelle Adresse wird erneut vorgegeben (Schritt **112**). Diesmal gibt es einen TLB-Treffer, so daß die sich ergebende physische Adresse für den Zugriff auf den physischen Speicher verwendet wird.

[0011] Wenn die virtuelle Adresse in einer Seite von virtuellen Adressen ist, der keine Seite von physischen Adressen zugeordnet ist, dann gibt es keinen Eintrag für diese Seite im Seitenverzeichnis, und es wird ein Seitenfehler auftreten. In dieser Situation wird eine Software-Seitenfehler-Steuerung (Schritt **120**) der virtuellen Seite eine physische Seite zuordnen und die Seitentabelle aktualisieren. Diese virtuelle Adresse wird dann wiederum dem TLB vorgegeben, wobei der TLB die Übersetzung noch nicht enthält, so daß ein TLB-Fehltreffer auftritt, und der TLB wird aus dem Seitenverzeichnis aktualisiert. Dann wird die virtuelle Adresse dem TLB wieder vorgegeben, und diesmal ist ein TLB-Treffer sichergestellt, so daß die sich ergebende physische Adresse für den Zugriff auf den physischen Speicher verwendet wird.

[0012] Fig. 2 zeigt ein vereinfachtes Verfahren zum Zugreifen auf einen Eintrag in einem Übersetzungs-Vorgriff-Puffer (TLB) abhängig von der Vorgabe einer virtuellen Adresse. Zu Zwecken der Erläuterung hat der gezeigte TLB nur einen Eintrag, wohingegen ein TLB normalerweise sehr viel mehr Einträge aufweisen würde. Die virtuelle Adresse wird in ein Register **201** geladen. Diese virtuelle Adresse ist aus zwei Teilen zusammengesetzt, einer virtuellen Seitennummer **203** und einem physischen Offset **205**. Der physische Offset **205** wird normalerweise durch die unteren 12 Bit (Bits 11 – 0) der Adresse gebildet, welche ein bestimmtes Byte innerhalb einer Seite spezifizieren. Die übrigen Bits in dem Register geben die virtuelle Seitennummer an.

[0013] Der Begriff "Seitenoffset" ist ein Begriff, der in der Industrie häufig verwendet wird, und er verhält sich zu dem Begriff "physischer Offset" synonym, welcher bei der Beschreibung der vorliegenden Erfindung verwendet wird. Andere Bits werden häufig verwendet, um eine Übersetzung oder Umsetzung in eine physische Seitennummer eindeutig zu spezifizieren. Solche Bits umfassen beispielsweise "Adreßbereich-Identifikations"-Bits. Für die Zwecke der vorliegenden Erfindung können alle diese Bits jedoch als ein Teil der virtuellen Seitennummer betrachtet werden.

[0014] Bei dem gezeigten Beispiel wird die virtuelle Seitennummer zum virtuellen (Identifizierungs-)Tag, welche ein Eingangssignal für den TLB-Vergleicher **207** liefert. Ein TLB **209** enthält zwei miteinander verknüpfte Teile, einen TLB-Tag **211** und eine zugeord-

nete physische Seitennummer **213**. Der TLB-Tag **211** liefert das zweite Eingangssignal für den TLB-Vergleicher **207**, und der Vergleicher vergleicht den TLB-Tag mit der virtuellen Tag. Wenn die Tags übereinstimmen, zeigt der Vergleicher einen TLB-Treffer an, und die physische Seitennummer **213** wird mit dem physischen Offset **205** kombiniert, um die physische (echte) Speicheradresse vorzusehen. Wenn die Tags nicht übereinstimmen, ist ein TLB-Fehltreffer aufgetreten, und das TLB-Fehltreffer-Verfahren, welches in Verbindung mit Fig. 1 beschrieben wurde, wird zum Aktualisieren des TLB eingesetzt.

Stand der Technik

[0015] **Fig.** 3 zeigt das Verfahren zum Wiedergewinnen der physischen Seiteninformation, wenn die virtuelle Seitennummer gegeben ist, so wie es zum Aktualisieren des TLB nach einem TLB-Fehltreffer erforderlich wäre. Wie oben beschrieben, werden die Abbildungen von virtuellen auf physische Adressen in einer Seitentabelle aufbewahrt. Ein Lösungsweg, um eine gegebene virtuelle Adresse in eine physische Adresse zu übersetzen, ist, eine Hash-Funktion mit der virtuellen Adresse durchzuführen, um einen Index für die Seitentabelle zu bilden. Dies erzeugt einen Zeiger zu einer verknüpften Liste von Einträgen. Diese Einträge werden dann nach einem passenden Eintrag abgesucht. Um ein Gegenstück zu ermitteln, wird die virtuelle Seitennummer mit einem Eintrag in der Seitentabelle (virtueller Tag) verglichen. Wenn die beiden gleich sind, sieht dieser Seitentabelleneintrag die physische Adressenübersetzung vor.

[0016] Bei dem gezeigten Beispiel wird eine Hash-Funktion **301** mit der virtuellen Seitennummer **203** durchgeführt, um einen Index zu bilden. Dieser Index ist ein Offset in die Seitentabelle **303** hinein. Bei dem gezeigten Beispiel ist der Index 0, d.h., der Index zeigt zum ersten Eintrag **305** in der Seitentabelle. Jeder Eintrag in der Seitentabelle besteht aus mehreren Teilen, er enthält jedoch typischerweise mindestens einen virtuellen Tag **307**, eine physische Seite **309** und einen Zeiger **311**. Wenn die virtuelle Seitennummer **203** gleich dem virtuellen Tag **307** ist, dann gibt die physische Seite **309** die gewünschte physische (echte) Speicherseiten-Adresse an. Wenn der virtuelle Tag nicht paßt, zeigt der Zeiger **311** zu einer Kette von Einträgen im Speicher, welche Information zur Übersetzung virtueller in physische Adressen enthält. Diese zusätzliche in der Kette enthaltene Information wird benötigt, weil mehr als eine virtuelle Seitennummer zum selben Seitentabellen-Eintrag zeigen kann (Hash-Code-Anwendung).

[0017] Wie gezeigt, zeigt der Zeiger **311** zu einem Kettensegment **313**. Dieses Kettensegment enthält dieselbe Art von Information wie die Seitentabelle. Wie zuvor wird die virtuelle Seitennummer **203** mit dem nächsten virtuellen Tag **315** verglichen, um festzustellen, ob es ein Gegenstück gibt. Wenn ein übereinstimmendes Gegenstück da ist, gibt die zugeordnete physische Seite **317** die Adresse der gewünschten physischen Speicherseite an. Wenn kein passendes Gegenstück da ist, wird der Zeiger **319** untersucht, um das nächste Kettensegment zu ermitteln, wenn eines vorhanden ist. Wenn der Zeiger **319**, wie gezeigt, nicht zu einem anderen Kettensegment zeigt, dann ist ein Seitenfehler aufgetreten. In diesem Fall wird ein Seitenfehler-Softwareprogramm verwendet, wie in Verbindung mit **Fig.** 1 beschrieben, um die Seitentabelle zu aktualisieren.

[0018] Ein Verfahren dieser Art ist aus US 5,060,137 bekannt.

[0019] Das oben beschriebene Verfahren arbeitet gut für Systeme, bei dem der virtuelle Tag kleiner ist als die Adressbreite des Computers, oder für Systeme mit nur einer Zentraleinheit (CPU). Wenn der virtuelle Tag jedoch größer ist als die Adressbreite, werden zwei Vergleiche benötigt, um zu überprüfen, ob der virtuelle Tag und die virtuelle Seitennummer dieselben sind. Dieser zusätzliche Vergleichsschritt vermindert die Geschwindigkeit des Rechnersystems. Ferner muß bei Systemen mit mehreren CPUs große Sorgfalt darauf verwendet werden, zu gewährleisten, daß eine CPU nicht denselben Seitentabellen-Eintrag aktualisiert, welchen eine andere CPU überprüft. Sonst wäre es möglich, daß eine erste CPU einen Seitentabellen-Eintrag modifiziert, während eine zweite CPU den zweiten Vergleichsschritt mit demselben Seitentabellen-Eintrag durchführt. Dies wäre ein unannehmbares Ergebnis, und daher wäre zusätzliche Hardware notwendig, um sicherzustellen, daß keine Konflikte auftreten.

Aufgabenstellung

[0020] Aufgabe der Erfindung ist es, eine Übersetzungsvorrichtung anzugeben, bei der die Bitbreite der virtuellen Tags größer ist als die Adressbreite des Rechnersystems, ohne bei Seitentabellen-Operationen mehrere Vergleichsschritte durchführen zu müssen.

[0021] Diese Aufgabe wird durch die Übersetzungsvorrichtung nach Anspruch 1 erfüllt.

[0022] Die Unteransprüche 2 – 6 sind vorteilhafte Ausführungen der erfindungsgemäßen Übersetzungsvorrichtung.

[0023] Durch die erfindungemäße Übersetzungsvorrichtung können virtuelle Tags verwendet werden, die größer sind als die Adressbreite eines Rechners, ohne daß mehrere Vergleichsoperationen nötig sind. Anstatt den gesamten Tag in den Seitentabellen-Einträgen zu speichern, werden mehrere Vergleiche vermieden, indem ein verminderter Tag gespeichert

wird. Der verminderte Tag ist nicht größer als die Adressbreite, so daß nur eine einzige Vergleichsoperation benötigt wird. Der verminderte Tag wird aus der virtuellen Adresse des Systems gebildet, indem einige Bits entfernt werden, wodurch ein kleinerer Tag gebildet wird. Um die Eindeutigkeit der Seitentabelle-Einträge zu erhalten, werden die von der virtuellen Adresse entfernten Bits verwendet, um einen Index in die Seitentabelle einzuführen.

[0024] Ferner können die Bits, die nicht zum Bilden des verminderten Tags verwendet werden, mit anderen Bits aus der virtuellen Adresse kombiniert werden, um einen Hash-Index zu bilden. Dieser Hash-Index gewährleistet eine gleichmäßige Verteilung der Einträge in der Seitentabelle.

[0025] Zudem kann ein Hash-Tag aus den Bits der virtuellen Seitennummer und den Indexbits gebildet werden. Dieser Hash-Tag ist nicht größer als die Adressbreite des Rechners, und er wird in der Seitentabelle an der Stelle des verminderten Tags gespeichert.

[0026] Die Erfindung hat den Vorteil, daß große virtuelle Adressen verwendet werden können, ohne daß mehrere Vergleichsschritte notwendig wären. Rechner, welche die vorliegende Erfindung benutzen, genießen daher die Vorteile eines großen Adressbereiches ohne die Nachteile und Probleme für die Leistungsfähigkeit bei Konstruktionen nach dem Stand der Technik. Dies ist ein besonders wichtiger Fortschritt für Hochleistungs-Rechnersysteme, welche mehrere CPUs verwenden, die jeweils auf einen gemeinsamen physischen Speicher Zugriff haben.

Ausführungsbeispiel

[0027] Die Erfindung ist im folgenden anhand bevorzugter Ausführungsbeispiele mit Bezug auf die Zeichnung näher erläutert. Es zeigen:

[0028] Fig. 1 ein Verfahren nach dem Stand der Technik, das zeigt, wie auf virtuelle Adressen reagiert wird, welche während der Ausführung eines Programmes übergeben werden,

[0029] Fig. 2 ein Verfahren nach dem Stand der Technik zum Zugreifen auf einen Eintrag in einem Übersetzungs-Vorgriff-Puffer,

[0030] Fig. 3 ein Verfahren nach dem Stand der Technik zum Wiedergewinnen physischer Seiteninformation, um einen TLB nach einem TLB-Fehltreffer zu aktualisieren,

[0031] Fig. 4 eine erste bevorzugte Ausführungsform der vorliegenden Erfindung, bei der Indexbits mit einem Hash-Tag kombiniert werden, um auf die Seitentabelle zuzugreifen,

[0032] Fig. 5 eine zweite bevorzugte Ausführungsform der vorliegenden Erfindung, bei der Hash-Indexbits mit einem Hash-Tag kombiniert werden, um einen Zugriff auf die Seitentabelle vorzusehen,

[0033] Fig. 6 eine dritte bevorzugte Ausführungsform der vorliegenden Erfindung, bei der Hash-Tags in der Seitentabelle gespeichert werden, und

[0034] Fig. 7 ein zweites Seitentabellenformat.

[0035] Ein Grundmerkmal der vorliegenden Erfindung ist das Aufspalten der virtuellen Seitennummer **203** in zwei Teile. Einen ersten Teil, welcher der verminderte Tag wird, und einen zweiten Teil, welcher als ein Index verwendet wird. Der verminderte Tag wird dadurch gebildet, daß die für den Tag verwendeten Bits auf nicht mehr als die Adressbreite des Rechners begrenzt werden. Wenn das Rechnersystem beispielsweise eine Datenbreite von 32 Bit hat, würde das verminderte Tag nicht mehr als 32 Bit enthalten. Alle Bits, die nicht zum Bilden des verminderten Tags verwendet werden, werden dann zum Bilden eines Rest-Indexes verwendet.

[0036] Fig. 4 zeigt eine bevorzugte Realisierung der vorliegenden Erfindung zum Wiedergewinnen der physischen Seiteninformation nach einem TLB-Fehltreffer. Wie gezeigt, wird die virtuelle Seitennummer **203** in zwei Teile aufgespaltet; ein vermindertes Tag **401** und ein Rest-Index **403**. Der verminderte Tag **401** besteht aus nicht mehr Bits als die Adressbreite des Rechners. Die zum Bilden des verminderten Tags nicht verwendeten Bits bilden den Rest-Index **403**. Wenn also die virtuelle Seitennummer **203** eine Länge von 36 Bits hat, wäre der verminderte Tag **401** 32 Bit lang, und der Rest-Index **403** hätte eine Länge von 4 Bit, wenn man annimmt, daß die Datenbreite des Rechners 32 Bit beträgt.

[0037] Eine Hash-Funktion **405** wird mit dem verminderten Tag **401** durchgeführt, um einen Tagindex **407** zu erzeugen. Der Tagindex **407** wird mit dem Rest-Index **403** kombiniert, beispielsweise durch Verkettung, um einen Seitentabellen-Index **409** zu bilden. Wie gezeigt, ist der Seitentabellen-Index Eins und zeigt zum zweiten Eintrag **411** der Seitentabelle **413**. Jeder Eintrag in der Tabelle besteht aus mehreren Teilen, wie bereits in Verbindung mit Fig. 3 beschrieben. Wenn der verminderte Tag **401** gleich dem verminderten Tag **415** ist, gibt die physische Seite **417** die physische (echte) gewünschte Speicherseiten-Adresse an. Wenn die Tags **401** und **415** nicht übereinstimmen, wird der Zeiger **419** untersucht, um das nächste Glied in der Kette zu finden, soweit vorhanden, wie oben beschrieben.

[0038] Dieses Verfahren und die Vorrichtung erfor-

dern, daß die Seitentabelle eine minimale Anzahl von Indizes aufweist. Da der Rest-Index **403** eine Größe von 4 Bit hat, muß die Seitentabelle bei diesem Beispiel mindestens 16 Einträge aufweisen. Da die Größe von Seitentabellen im allgemeinen wesentlich größer ist, macht diese Beschränkung keine Schwierigkeiten.

[0039] Fig. 5 zeigt eine zweite bevorzugte Ausführungsform der vorliegenden Erfindung. Wie zuvor, ist die virtuelle Seitennummer **203** in einen verminderten Tag **401** und einen Rest-Index **403** aufgespaltet. Der verminderten Tag **401** ist so breit wie die Adressbreite des Rechners. Bits, welche nicht zum Bilden des verminderten Tags verwendet werden, bilden den Rest-Index **403**. Bei dem gezeigten Beispiel hat der Rest-Index eine Länge von 4 Bit, und der verminderte Tag **401** hat eine Länge von 32 Bit.

[0040] Die 32 Bits des verminderten Tags **401** werden über eine Hash-Funktion **405** verarbeitet, um einen Markenindex **407** zu erzeugen. Eine zweite Hash-Funktion **501** wird mit dem Rest-Index **403** durchgeführt, und "n" Bits **503** bilden den virtuelle Tag **401**, wobei "n" einen Wert von 0 bis zur maximalen Anzahl der Bits in dem verminderten Tag haben kann. Die Ausgabe **505** der zweiten Hash-Funktion **501** wird mit dem Markenindex **407** kombiniert, beispielsweise durch eine Verkettung, um einen Seitentabellen-Index **507** zu bilden.

[0041] Durch Verwenden einer zweiten Hash-Funktion **501** ist eine gleichmäßigere Verteilung der Seitentabellen-Indizes möglich, als wenn nur die Bits des Rest-Indexes **403** verwendet würden. Im Idealfall wäre die Verteilung so, daß der erste Index zur Seitentabelle einen Treffer ergäbe. Aufgrund von Größenbeschränkungen für die Seitentabelle ist jedoch eine Suchlänge in die Seitentabelle hinein von 1,1 typischer, und sie gewährt eine vernünftige Seitentabellengröße.

[0042] Die Organisation und Funktion der Seitentabelle **509** ist im wesentlichen wie in Verbindung mit Fig. 3 und Fig. 4 beschrieben. Die Größe der Seitentabelle wird dagegen durch die Anzahl von Bits des Ausgangs **505** der zweiten Hash-Funktion bestimmt. In einem Minimalfall besteht die Ausgabe **505** aus der Anzahl von Bits im Rest-Index **403**. In dem Minimalfall, bei dem der Rest-Index eine Länge von vier Bit hat und "n" = 0, hat die Seitentabelle **16** Einträge und arbeitet wie in Verbindung mit Fig. 4 beschrieben. Wenn "n" > 0, dann ist die Ausgabe **505** größer als 4 Bit, und die Seitentabelle hat 2^m Eintragsbereiche, wobei m die Anzahl der Bits in der Ausgabe **505** ist. Es muß nicht notwendigerweise eine direkte Beziehung zwischen "n" und "m" geben, da die zweite Hash-Funktion eine beliebige Anzahl von Bits von dem verminderten Tag entnehmen kann und eines oder mehrere zusätzliche Bits zur Kombination mit den Bits des Rest-Indexes **403** erzeugen kann. Je mehr Bits die Ausgabe **505** hat, desto gleichmäßiger sind die Seiten in der Seitentabelle verteilt.

[0043] Die "n" Bits von dem verminderten Tag können über Software konfigurierbar sein, so daß die Anzahl der Bits, und die einzelnen verwendeten Bits, mittels Software ausgewählt werden können. Dies ermöglicht die intelligente Konfiguration des Rechnersystems, was wünschenswert ist, wenn die virtuelle Adresse **203** wesentlich größer als die Datenwegbreite ist und wenn daher die Zwangsbedingungen für die Seitentabellengröße zu einem Problem werden.

[0044] Fig. 6 zeigt eine weitere Ausführungsform der vorliegenden Erfindung. Der Betrieb des Seitentabellen-Nachschlage-verfahrens läuft mit den folgenden Ausnahmen im wesentlichen genauso ab, wie in Verbindung mit Fig. 4 beschrieben.

[0045] Die Seitentabelle **413** enthält "Hash-Tags" **601** und **603** anstelle der verminderten Tags von Fig. 4. Diese Hash-Tags werden gebildet, indem die Indexbits **403** und die Bits **401** der virtuellen Seitennummer genommen werden und eine Hash-Funktion mit diesen Bits durchgeführt wird, wobei das Ergebnis nicht größer als die Adressbreite des Rechners ist. Dieses Hash-Ergebnis wird dann in der Seitentabelle (**601** und **603**) anstelle der zuvor beschriebenen verminderten Tags gespeichert.

[0046] Im Betrieb wird ein Seitentabellen-Index **409**, wie zuvor erläutert, gebildet. Gleichzeitig mit dem Bilden des Seitentabellen-Indexes wird ein Hash-Ergebnis **605** gebildet, indem eine Hash-Funktion **607** mit den Indexbits **403** und den Bits **401** der virtuellen Seitennummer durchgeführt wird. Diese Hash-Funktion ist dieselbe wie die, die zum Erzeugen aller Hash-Tags verwendet wurde, welche in der Seitentabelle gespeichert wurden. Der Seitentabellen-Index zeigt zu einem Eintrag **411** in der Seitentabelle **413**, und der Hash-Tag **603** wird mit dem Hash-Ergebnis **605** verglichen (**609**). Wenn der Tag **603** und das Ergebnis **605** übereinstimmen, gibt eine physische Seite **417** die gewünschte Adresse der physischen Speicherseite an. Wenn es kein passendes Gegenstück gibt, wird der Zeiger **419** untersucht, um das nächste Glied in der Kette, soweit vorhanden, herauszufinden.

[0047] Diese Vorrichtung zum Erzeugen eines Seitentabellen-Tags ist besonders vorteilhaft, wenn das Betriebssystem des Rechners gewährleisten kann, daß hochwertige Bits in der virtuellen Seitennummer Null sind. Wenn beispielsweise der gesamte virtuelle Adressbereich nicht benötigt wird, können einige Bits immer Null sein. Dieses Verfahren und die Vorrichtung erlauben also eine Verminderung der Größe des IDX-Indexes, wodurch die Effizienz der Seitentabelle

verbessert wird, da die minimale Größe der Seitentabelle vermindert wird.

[0048] Fig. 7 zeigt eine von vielen möglichen anderen Seitentabellen-Organisationsformen, welche verwendet werden können, ohne den Bereich der vorliegenden Erfindung zu verlassen. Bei diesem Beispiel ist ein Eintrag **701** in einer Seitentabelle gezeigt. Dieser Eintrag hat drei Sätze (**703**, **705** und **707**) von verminderten Tags und physischen Seiteneinträgen. Der Betrieb dieser Seitentabelle ist im wesentlichen derselbe, wie in Verbindung mit Fig. 4 beschrieben, außer daß drei verminderte Tags (**709**, **711** und **713**) mit der virtuellen Seitennummer **401** verglichen werden, bevor der Zeiger **715** untersucht wird, um das nächste Glied, soweit vorhanden, aufzufinden.

Patentansprüche

1. Übersetzungsvorrichtung zum Umsetzen einer virtuellen Speicheradresse in eine physikalische Speicheradresse, mit einer Seitentabelle (**413**) mit einem Adressregister zum Speichern eines virtuellen Tags (**203**), wobei die Anzahl der Bits des virtuellen Tags die Adressenbreite des Rechners überschreitet,
wobei der virtuelle Tag (**203**) einen verminderten Tagabschnitt (**401**), dessen Bitbreite gleich oder kleiner als die Adressenbreite des Rechners ist, und einen Indexabschnitt (**403**) aufweist, welcher die restlichen Bits des virtuellen Tags umfaßt,
eine erste Hash-Funktion (**405**) eine erste Ausgangsfolge (**407**) aus den Bits des verminderten Tagabschnitts (**401**) erzeugen kann,
ein Seitentabellen-Index (**409**; **507**) aus der ersten Ausgangsfolge (**407**) und der Information des Indexabschnittes (**403**) kombiniert wird und verschiedene Sets in der Seitentabelle (**413**) indiziert, und
wobei die Sets mindestens ein Paar aus einem verminderten Tag-Eintrag (**307**, **415**) und der zugeordneten physikalischen Adresse und mindestens einen Zeiger (**41**) aufweisen.

2. Übersetzungsvorrichtung nach Anspruch 1, bei der die Seitentabelle (**413**) 2^{I} verminderte Tag-Einträge aufweist, wobei "I" gleich der Anzahl von Bits im Indexabschnitt (**403**) ist.

3. Übersetzungsvorrichtung nach Anspruch 1, wobei die aus der Seitentabelle (**413**) gewonnene physikalische Seitenadresse und die virtuelle Seitennummer in einem Translation-Lookaside-Puffer (TLB) abgespeichert werden.

4. Übersetzungsvorrichtung nach einem der vorangehenden Ansprüche, bei der der Seitentabellen-Index (**409**; **507**) aus der ersten Ausgangsfolge und der Ausgabe (**505**) einer zweiten Hash-Funktion (**501**) kombiniert wird, wobei die zweite Hash-Funktion die gesamte Information des Indexabschnitts (**403**) und Bits des verminderten Tagabschnitts (**401**) als Eingabe empfängt.

5. Übersetzungsvorrichtung nach einem der vorangehenden Ansprüche, mit einer dritten Hash-Funktion (**607**) zur Bildung eines Hash-Ergebnisses (**605**) aus dem verminderten Tagabschnitt (**401**) und dem Indexabschnitt (**403**), und mit einem Vergleicher (**609**) zum Vergleichen des Hash-Ergebnisses (**605**) mit einem in der Seitentabelle (**413**) gespeicherten Hash-Tag (**603**).

6. Übersetzungsvorrichtung nach Anspruch 5, wobei dann, wenn die Hash-Tag und das Hash-Ergebnis übereinstimmen, der dem Hash-Tag-Eintrag zugeordnete physikalische Seiten-Eintrag in dem TLB gespeichert wird.

Es folgen 7 Blatt Zeichnungen

Anhängende Zeichnungen

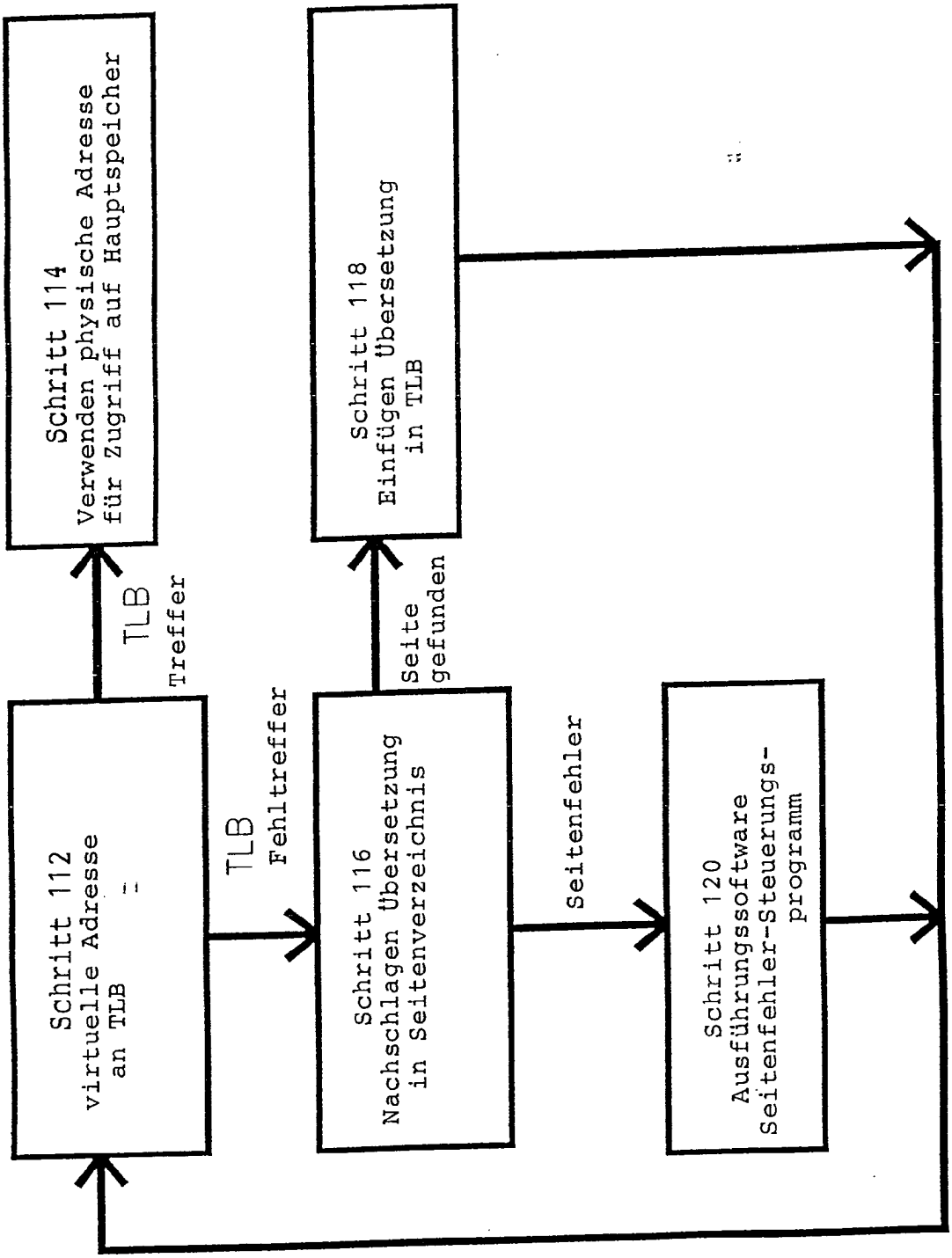


FIG 1 (Stand der Technik)

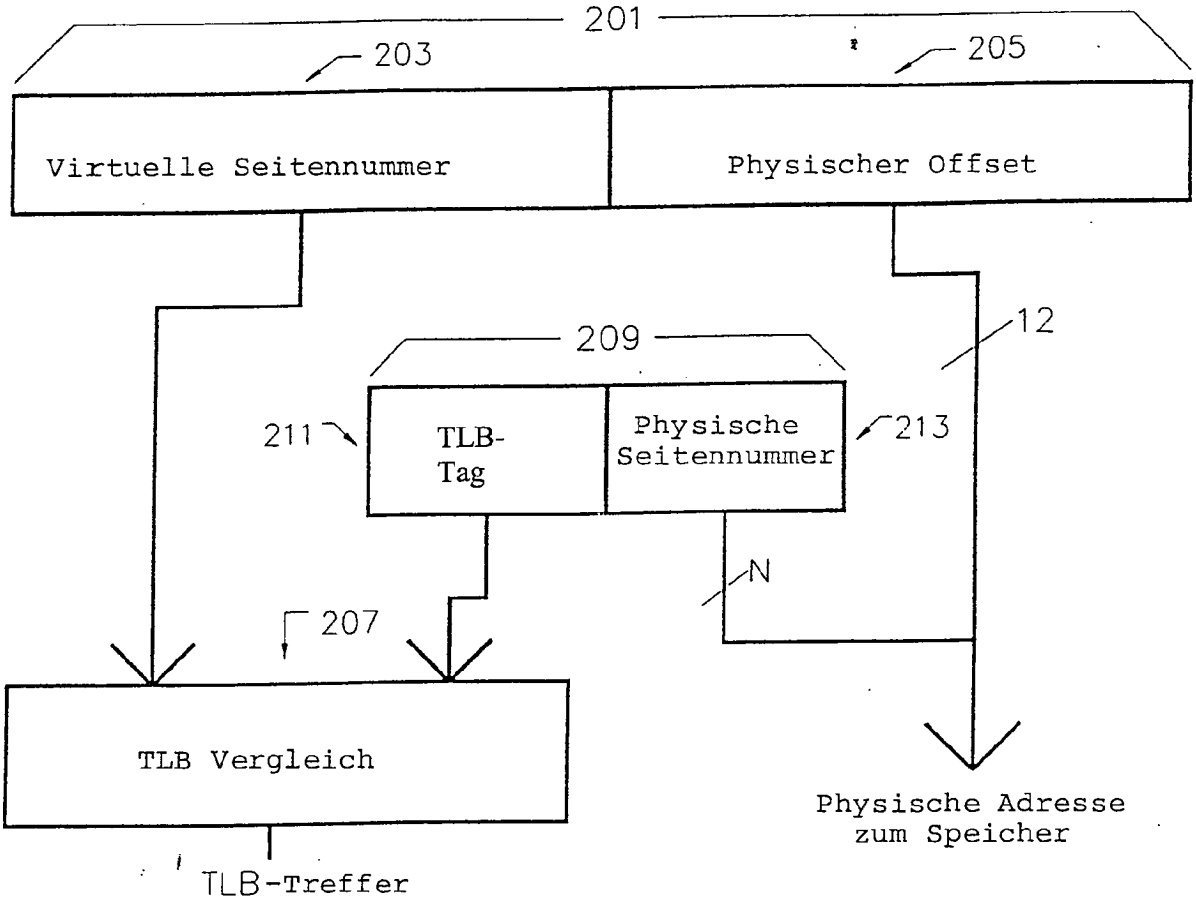


FIG 2 (Stand der Technik)

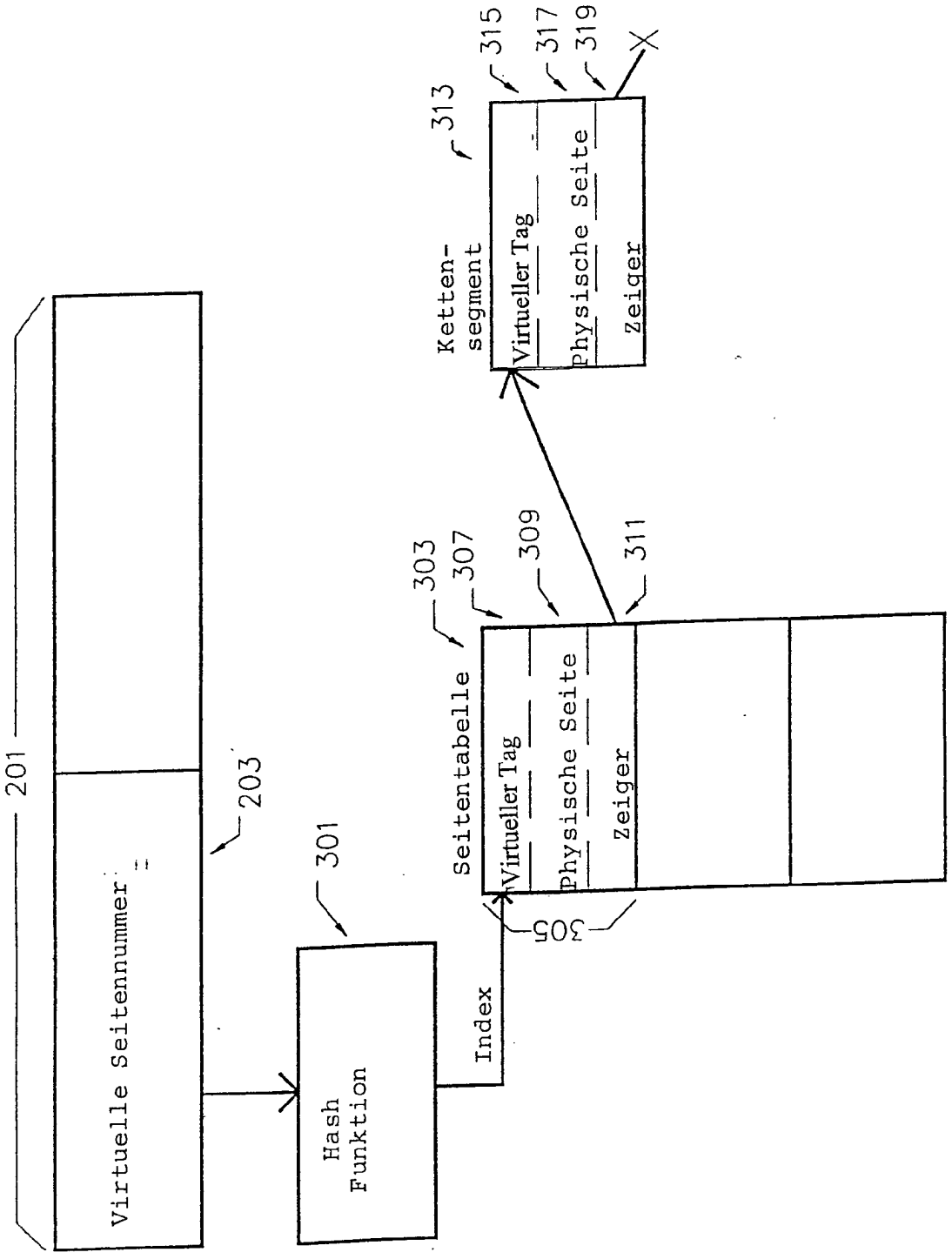


FIG 3 (Stand der Technik)

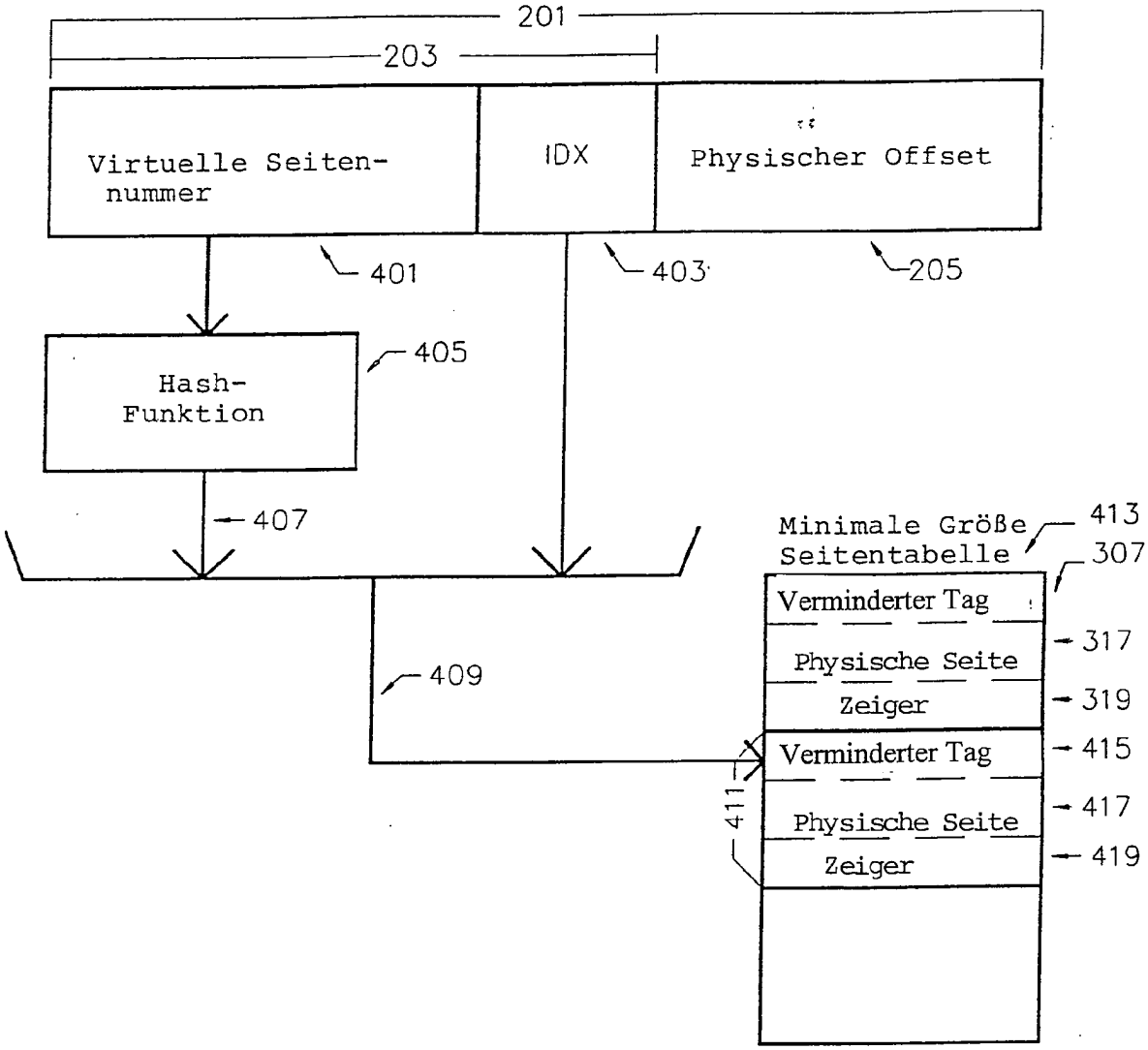

201
203
Virtuelle Seiten-nummer
IDX
Physischer Offset
401
403
205
Hash-Funktion
405
407
409
Minimale Größe Seitentabelle
413
307
Verminderter Tag
317
Physische Seite
319
Zeiger
Verminderter Tag
415
411
Physische Seite
417
Zeiger
419

FIG 4

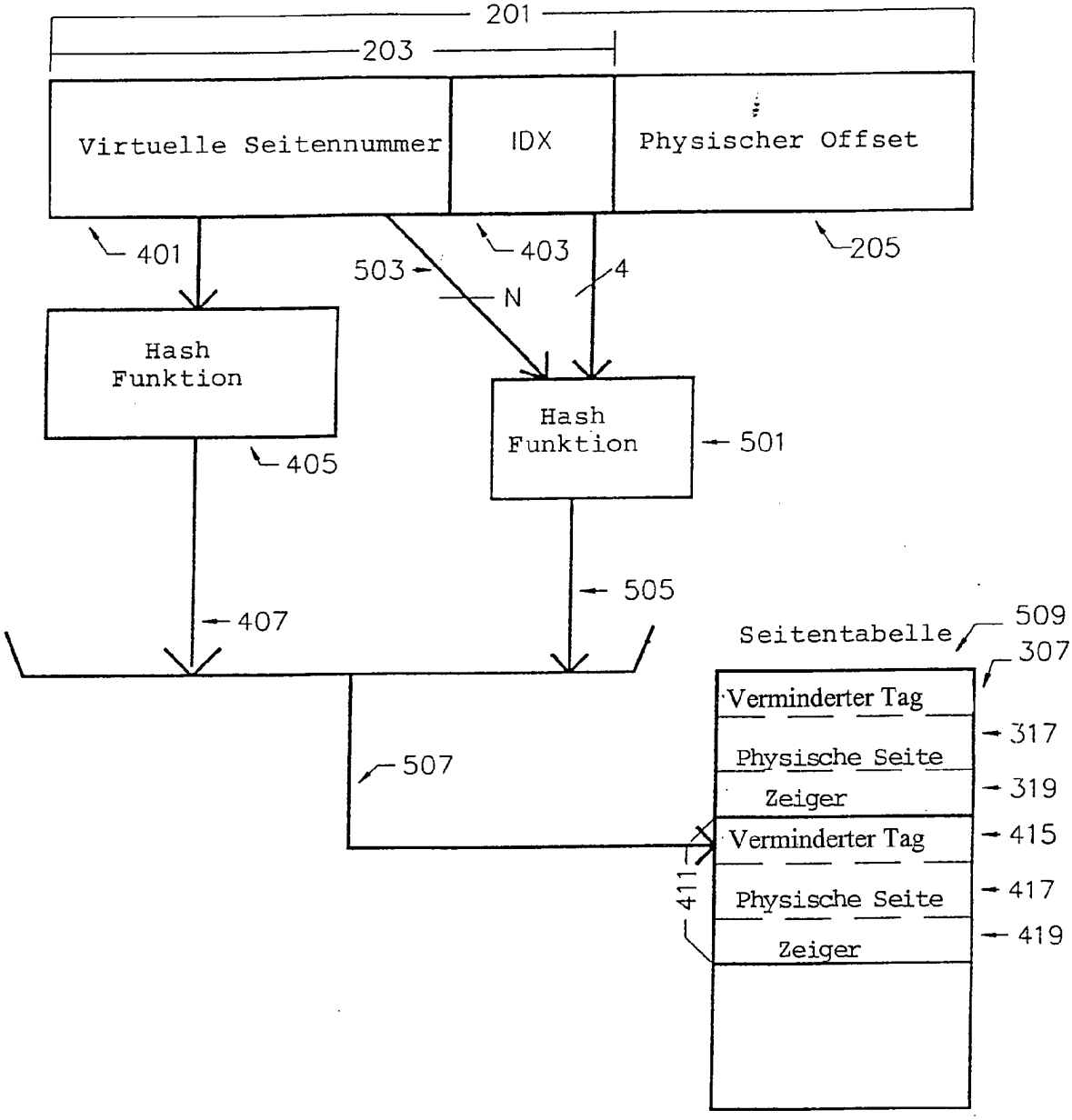

201
203
Virtuelle Seitennummer
IDX
Physischer Offset
401
403
205
503
N
4
Hash Funktion
405
Hash Funktion
501
505
407
Seitentabelle
509
307
Verminderter Tag
317
Physische Seite
319
Zeiger
507
Verminderter Tag
415
411
Physische Seite
417
Zeiger
419

FIG 5

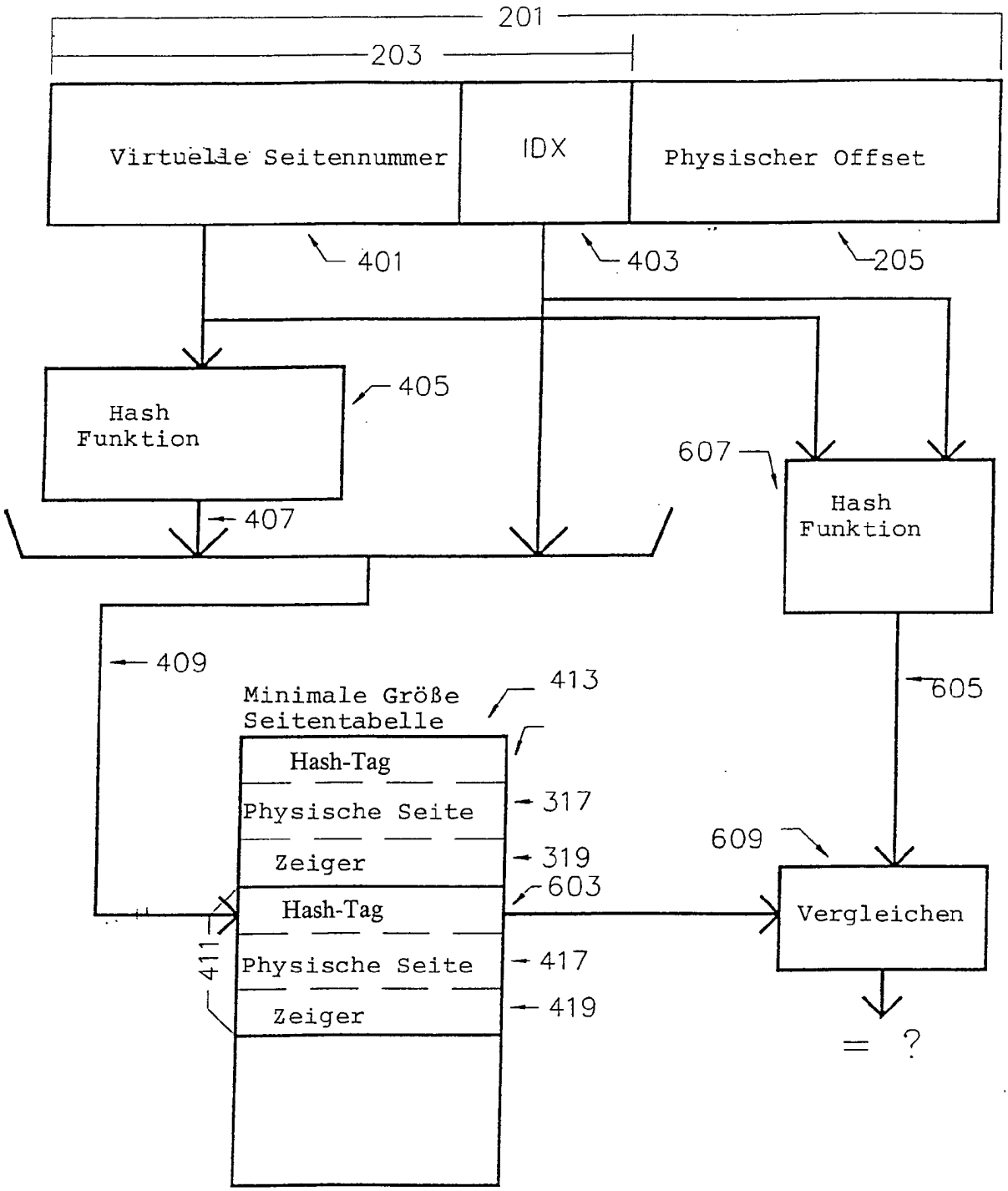

201
203
Virtuelle Seitennummer
IDX
Physischer Offset
401
403
205
Hash Funktion
405
607
Hash Funktion
407
409
Minimale Größe Seitentabelle
413
605
Hash-Tag
Physische Seite
317
Zeiger
319
609
603
Hash-Tag
Vergleichen
411
Physische Seite
417
Zeiger
419
= ?

FIG 6

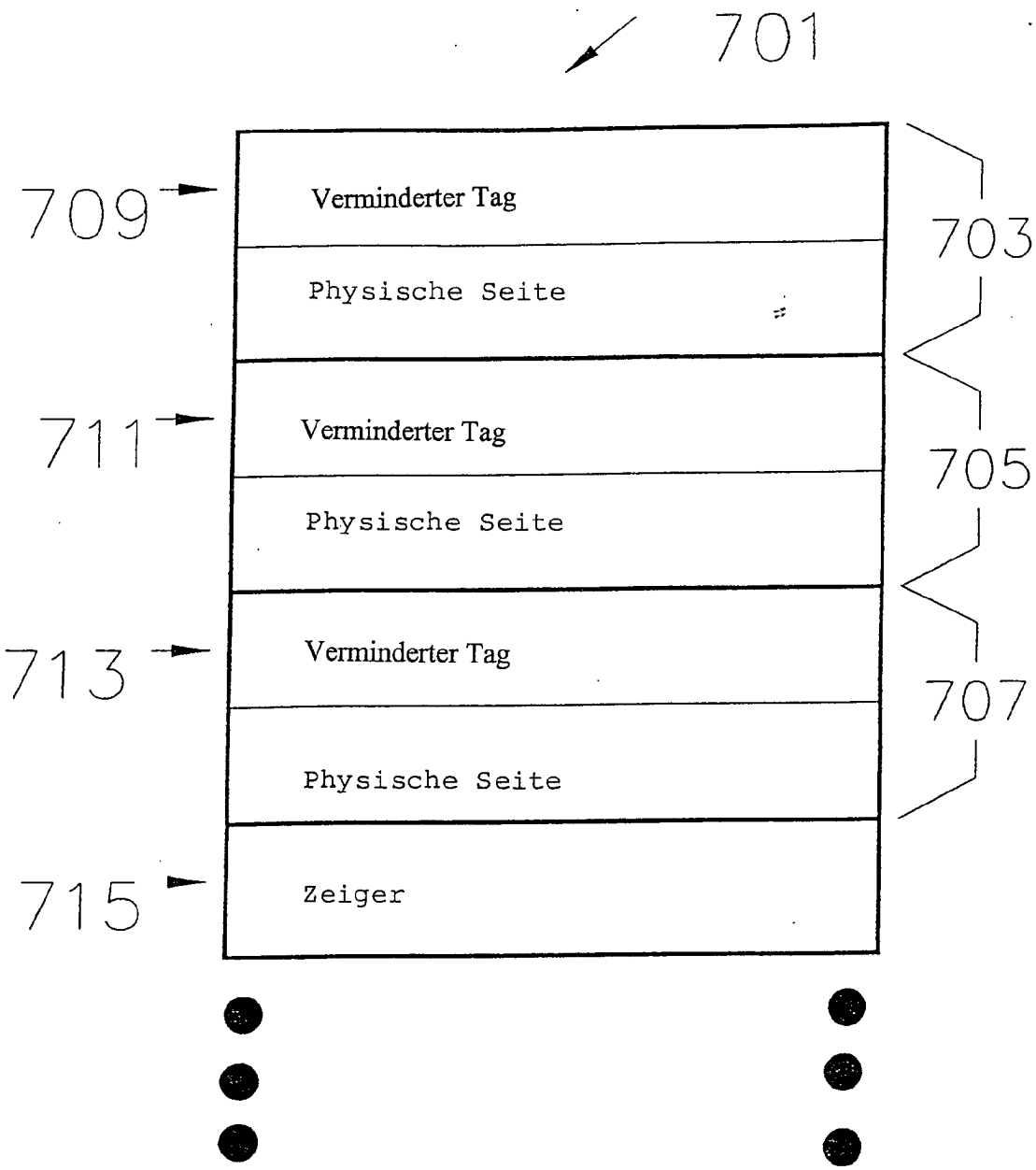

701
709
Verminderter Tag
Physische Seite
703
711
Verminderter Tag
Physische Seite
705
713
Verminderter Tag
Physische Seite
707
715
Zeiger

FIG 7