(19) Bundesrepublik Deutschland
Deutsches Patent- und Markenamt

(10) **DE 10 2004 013 676 A1** 2004.12.23

(12) # Offenlegungsschrift

(21) Aktenzeichen: **10 2004 013 676.9**
(22) Anmeldetag: **18.03.2004**
(43) Offenlegungstag: **23.12.2004**

(51) Int Cl.[7]: **G06F 9/32**

(30) Unionspriorität:
10/396063 24.03.2003 US

(71) Anmelder:
Infineon Technologies AG, 81669 München, DE

(74) Vertreter:
Jannig & Repkow Patentanwälte, 86199 Augsburg

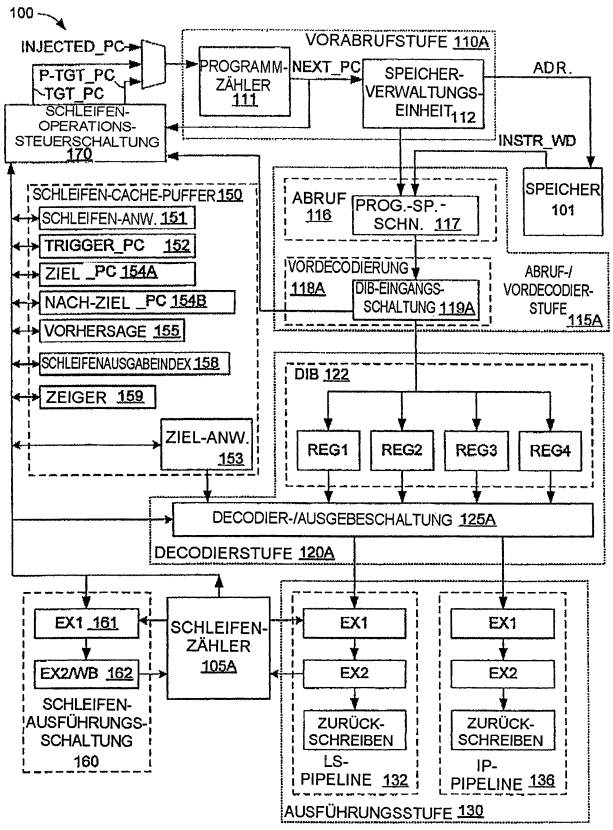
(72) Erfinder:
Ahmad, Sagheer, Sunnyvale, Calif., US; Arnold, Roger, Sunnyvale, Calif., US; Knoth, Matthias, San Jose, Calif., US

Prüfungsantrag gemäß § 44 PatG ist gestellt.

Die folgenden Angaben sind den vom Anmelder eingereichten Unterlagen entnommen

(54) Bezeichnung: **Schleifenbetrieb mit null Overhead in einem Mikroprozessor mit Anweisungspuffer**

(57) Zusammenfassung: Eine Schleifenanweisung, mindestens eine Zielanweisung und eine zugeordnete Trigger-Adresse werden während des Schleifeneintritts cache-gespeichert. Während jeder Schleifeniteration sagt der Prozessor vorher, ob die Schleife in einer nachfolgenden Interation genommen oder nicht genommen wird. Wenn das Vorabrufen der cache-gespeicherten Schleifenanweisung nachfolgend erkannt wird (d. h. durch Vergleichen der Trigger-Adresse mit dem aktuellen Programmzählwert), wird die Schleife-genommen-/-nicht-genommen-Vorhersage dazu verwendet, entweder Schleifenhauptteilanweisungen abzurufen (wenn Genommen vorhergesagt wird) oder Fall-through-Anweisungen abzurufen (wenn Nicht-genommen vorhergesagt wird). Die cache-gespeicherte Schleifenanweisung wird dann ausgeführt und die Schleife-genommen-/-nicht-genommen-Vorhersage wird unter Verwendung einer eigenen Schleifenausführungsschaltung verifiziert, während eine vorletzte Schleifenhauptteilanweisung in der Prozessorausführungsstufe (Pipeline) ausgeführt wird. Wenn eine vorherige Schleife-genommen-Vorhersage verifiziert wird, wird die cache-gespeicherte Zielanweisung ausgeführt, und dann werden die abgerufenen Schleifenhauptteilanweisungen ausgeführt. Wenn eine Schleife-nicht-genommen-Vorhersage verifiziert wird, werden die abgerufenen Fall-through-Anweisungen ausgeführt.



Beschreibung

TECHNISCHES GEBIET

[0001] Die vorliegende Erfindung betrifft allgemein Datenverarbeitungssysteme und insbesondere einen Datenverarbeitungssystemmikroprozessor, der einen Anweisungspuffer zum Speichern abgerufener Programmanweisungen vor dem Ausgeben an eine Ausführungs-Pipeline enthält.

Stand der Technik

[0002] **Fig.** 14 ist ein vereinfachtes Diagramm eines herkömmlichen Mikroprozessors (Prozessor) **1400**, der einen Anweisungspuffer (d.h. den unten besprochenen Decodierungsanweisungspuffer (DIB) **122**) zum Speichern abgerufener Programmanweisungen vor der Ausgabe an eine Ausführungs-Pipeline verwendet. Der Prozessor **1400** ist allgemein konsistent der TriCore™-Familie von Prozessoreinrichtungen, die von der Infineon Technologies AG in München hergestellt wird. Für Fachleute auf dem Gebiet der Prozessoren ist erkennbar, daß die Beschreibung des Prozessors **1400** für Erläuterungszwecke stark vereinfacht ist und daß ein Teil der unten separat beschriebenen Schaltungskomponenten mit anderen Komponenten integriert oder ganz weggelassen werden kann.

[0003] Der Prozessor **1400** ist allgemein in eine Vorabrufstufe **110**, eine Abruf-/Vordecodierstufe **115**, eine Decodierstufe **120** und eine Ausführungsstufe **130** aufgeteilt. Die Vorabrufstufe **110** enthält einen Programmzähler **111** und eine Speicherverwaltungseinheit (MMU) **112**, die kooperieren, um Adressensignale zu senden, mit denen entsprechende Programmanweisungen von einem Systemspeicher **101** (z.B. Cache-, lokaler und/oder externer Speicher) gelesen werden, der diese Programmanweisungen dann in die Abruf-/Vordecodierstufe **115** schreibt. Die Abruf-/Vordecodierstufe **115** enthält einen Abrufteil **116** mit einer Programmspeicherschnittstelle (PROG MEM INTRFC) **117** zum Empfangen der Programmanweisungen und einen Vordecodierteil **118** mit einer Decodieranweisungspuffereingangsschaltung **119**, die die Anweisungen teilweise decodiert und die Anweisungen auf unten beschriebene Weise in die Decodierstufe **120** schreibt. Die Decodierstufe **120** enthält den DIB **122** und eine Decodier-/Ausgabeschaltung **125**. Die Ausführungsstufe **130** enthält die Prozessor-„Pipeline", die die aus der Decodierstufe **120** ausgegebenen decodierten Programmanweisungen ausführt. Bei dem vorliegenden Beispiel enthält die Ausführungsstufe **130** zwei Prozessor-Pipelines: eine Lade-/Speicher-(LS-)Pipeline **132** und eine Integer-Verarbeitungs-(IP-)Pipeline **136**. Jede Pipeline enthält zwei Ausführungsstufen (d.h. EX1 und EX2) und eine Rückschreibstufe. Der Prozessor **1400** enthält außerdem ein Schleifenzählerregister **105A**, das bei dem vorliegenden Beispiel einen Schleifenzählerwert speichert. Man beachte, daß das Schleifenzählerregister **105A** eines von mehreren durch den Prozessor **1400** bereitgestellten Vielzweckregistern sein kann.

[0004] Der DIB **122** kann logisch als ein zirkulärer Puffer mit mehreren Registern (z.B. vier Registern REG1–REG4), einem Eingangs-(Schreib-)Zeiger, der durch die DIB-Eingangsschaltung **119** gesteuert wird, und einem oder mehreren Ausgangszeigern, die durch die Decodier-/Ausgabeschaltung **125** gesteuert werden, repräsentiert werden. Der Schreibzeiger zeigt auf eines der Register REG1–REG4, und eine Abruf-/Vordecodierstufe **115** schreibt in jedem Schreibzyklus eine, zwei, drei oder vier Anweisungen in das Register, auf das gezeigt wird. Zum Beispiel zeigt in einem ersten Schreibzyklus der Schreibpunkt auf REG1 und es werden vier 16-Bit-Anweisungen in REG1 geschrieben, in einem nächsten Schreibzyklus zeigt der Schreibzeiger dann auf REG2, und es werden zwei 32-Bit-Anweisungen in REG2 geschrieben ..., dann zeigt der Schreibzeiger auf REG4, und es werden eine 32-Bit-Anweisung und zwei 16-Bit-Anweisungen in REG4 geschrieben, dann kehrt der Schreibpunkt zu REG1 zurück, und es werden neue Anweisungen in REG1 geschrieben. Man beachte, daß zuvor geschriebene Anweisungen aus jedem Register ausgegeben werden, bevor neue Anweisungen in dieses Register geschrieben werden. Außerdem werden abhängig von dem Prozessor eine oder mehrere dieser Anweisungen aus den Registern REG1–REG4 während jedes Ausgabezyklus an die Ausführungsstufe **130** ausgegeben, in der die decodierten Anweisungen abhängig von dem „Typ" der ausgegebenen Anweisungen entweder an die LS-Pipeline **132** oder die IP-Pipeline **136** gehen. Zum Beispiel wird in einem ersten Ausgabezyklus eine erste 16-Bit- oder 32-Bit-Anweisung des IP-Typs an die IP-Pipeline **136** ausgegeben, und eine zweite 16-Bit- oder 32-Bit-Anweisung des LS-Typs wird aus dem DIB-Register REG1 an die LS-Pipeline **132** ausgegeben. Abhängig von dem Prozessor kann die Reihenfolge, in der die Anweisungen des LS-Typs und die Anweisungen des IP-Typs angeordnet sind, bestimmen, ob eine oder zwei Anweisungen pro Ausgabezyklus ausgegeben werden. Zum Beispiel kann in einem zweiten Ausgabezyklus eine dritte 16-Bit- oder 32-Bit-Anweisung des LS-Typs (die der zuvor ausgegebenen zweiten Anweisung des LS-Typs folgt) aus REG1 an die LS-Pipeline **132** ausgegeben werden (d.h. weil die zweite und die dritte Anweisung LS-Anweisungen sind, wird während des zweiten Ausgabezyklus keine IP-Anweisung ausgegeben). Dieser Ausgabeprozeß wird fortge-

setzt, wobei erst aus REG1 ausgegeben, dann zu REG2, REG3 bzw. REG4 übergegangen und dann zu REG1 zurückgekehrt wird. Durch Speichern und Ausgeben mehrerer Anweisungen in Registern REG1–REG4 auf diese Weise wirkt der DIB **122** als ein Anweisungspuffer, der es der Abruf-/Vorabrufstufe **115** erlaubt, mit einer anderen Geschwindigkeit als die Ausführungsstufe **130** zu arbeiten, wodurch die schnelle Verarbeitung ermöglicht wird.

[0005] Die Funktionsweise des Prozessors **1400** umfaßt in der Regel das Verarbeiten (Ausführen) eines Softwareprogramms, bei dem es sich um eine vorbestimmte Reihe von aus dem Systemspeicher **101** gelesenen Programmanweisungen handelt, die zusammen bewirken, daß der Prozessor **1400** eine gewünschte Datenverarbeitungstask durchführt. Während der Entwicklung solcher Softwareprogramme werden die Programmanweisungen im allgemeinen in der Reihenfolge angeordnet, in der sie verarbeitet (ausgeführt) werden, und die so angeordneten Programmanweisungen werden an entsprechenden sequentiellen Speicherstellen in dem Systemspeicher **101** vor der Ausführung durch den Prozessor **1400** zugewiesen (gespeichert).

[0006] Programmanweisungen können im allgemeinen folgendermaßen klassifiziert werden: Operationen, die sequentiell in der Ausführungsstufe **130** ausgeführt werden, und Verzweigungs-(oder Sprung-)Anweisungen, die bewirken, daß die Programmsteuerung von einer Anweisung zu einer Anweisung außerhalb der Reihenfolge „springt". Eine bedingte Verzweigungsanweisung, die häufig in Softwareprogrammen verwendet wird, ist eine Schleifenanweisung, die es einem Programm erlaubt, eine Anweisung (oder eine Reihe von Anweisungen) eine spezifizierte Anzahl von Malen oder bis eine bestimmte Bedingung erfüllt ist, wiederholt auszuführen. Fast alle Programmiersprachen enthalten mehrere verschiedene Schleifenanweisungen, die für verschiedene Zwecke ausgelegt sind.

[0007] **Fig.** 15 ist ein vereinfachtes Diagramm eines Teils **1500** eines Softwareprogramms, der einen oft verwendeten Schleifenanweisungstyp verwendet. Jeder Anweisung INST0 bis INST12 des Programmteils **1500** wird jeweils eine sequentiell angeordnete Adresse X0000 bis X1100 zugewiesen, die eine entsprechende Speicherstelle in dem Speicher **101** (**Fig.** 14) repräsentiert. Der Kürze halber sind die durch die Anweisungen INST0 bis INST12 durchgeführten Operationen nur für solche Anweisungen angegeben, die für die folgende Besprechung relevant sind. Zum Beispiel setzt die Anweisung INST1 einen Schleifenzähler R1 auf den ganzzahligen Wert drei (angezeigt durch „[R1 == 3]"), und die Schleifenanweisung INST9 ist eine Schleifenanweisung, die wie nachfolgend beschrieben funktioniert. Die Funktionen der anderen Anweisungen (d.h. INST0, INST2–INST8 und INST10–INST12) führen Operationen durch, deren Beschaffenheit sequentiell ist (d.h. diese Anweisungen erzeugen keine nicht sequentielle Änderung der Programmsteuerung).

[0008] In dem vorliegenden Beispiel ist die Schleifenanweisung INST9 von einem Typ, der wirkt, um einen ausgewiesenen Schleifenzähler (d.h. den Schleifenzähler R1 in diesem Beispiel) bei jeder Ausführung der Schleifenanweisung INST9 um eins zu erniedrigen, um die Programmsteuerung an eine Zielanweisung (d.h. Adresse X0010 abzugeben, wodurch in diesem Beispiel die Anweisung INST2 die Zielanweisung der Schleifenanweisung INST9 wird), solange der Schleifenzähler R1 größer als Null ist, und um die Programmsteuerung an die nächste sequentielle (Fall-through-)Anweisung nach der Schleifenanweisung (d.h. die Anweisung INST10 in diesem Beispiel) abzugeben, wenn der Schleifenzähler R1 gleich Null ist. Im vorliegenden Gebrauch bedeutet der Ausdruck „genommen" den Fall, bei dem, wenn die Schleifenanweisung ausgeführt wird, die Programmsteuerung zu der Zielanweisung springt, und der Ausdruck „nicht genommen" bedeutet den Fall, bei dem die Programmsteuerung zu den Fall-through-Anweisungen der Schleife weitergeleitet wird. Während der Schleifenzähler R1 größer als Null bleibt, ist die Schleifenanweisung INST9 folglich „genommen"-Operation, und die Programmsteuerung springt zu der Zielanweisung INST2. Der „Schleifenhauptteil" (d.h. die Anweisungen INST2–INST8) wird dadurch wiederholt ausgeführt, bis der Schleifenzähler R1 auf Null erniedrigt ist, und dann ist die Schleife „nicht genommen", und die Programmsteuerung wird zu der Fall-through-Anweisung INST10 weitergeleitet.

[0009] Wieder mit Bezug auf den Anfang von **Fig.** 14 erzeugt während der Ausführung des Softwareprogramms der Programmzähler **111** in der Regel sequentielle Programmzählerwerte NEXT_PC, die durch die MMU **112** in Speicheradressen umgesetzt werden, mit denen sequentiell auf die Speicherstellen in dem Speicher **101** zugegriffen wird, wodurch die Programmanweisungen in der vorgeordneten Reihenfolge gelesen und verarbeitet werden. Wenn Verzweigungs- oder Sprunganweisungen (z.B. Schleifenanweisungen) ausgeführt werden, wird ein nicht sequentieller Wert (INJECTED_PC) zu dem Programmzähler **111** gesendet, und eine entsprechende nicht sequentielle Adresse wird zu dem Speicher **101** gesendet. Der somit zurückgesetzte Programmzähler bzw. die somit zurückgesetzte MMU fährt dann fort, sequentielle Adressen nach der injizierten Adresse zu erzeugen, bis eine weitere Unterbrechung auftritt.

[0010] Wieder mit Bezug auf **Fig.** 15 werden während des „Schleifeneintritts" (d.h. des ersten Durchgangs durch die Anweisungen vor der Schleifenanweisung INST9) die Vorschleifenanweisungen INST0 und INST1 ausgeführt (wobei der Schleifenzähler R1 auf drei gesetzt wird), dann wird der Schleifenhauptteil das erste Mal ausgeführt, dann wird die Schleifenanweisung INST9 zum ersten Mal ausgeführt (angezeigt durch den äußersten linken Pfeil A in **Fig.** 15). Wie angegeben, erniedrigt die Schleifenanweisung INST9 den Schleifenzähler R1 auf zwei (R1 = 2), bestimmt, daß der in dem Schleifenzähler R1 gespeicherte Wert nicht gleich Null ist und bewirkt deshalb eine Operation des Typs „Schleife genommen", bei der die Programmsteuerung an die Anweisung INST2 (Adresse X0010) zurückgegeben wird. Die Verarbeitung der „inneren Schleife" des Schleifenhauptteils wird dann durchgeführt, während der Schleifenzähler R1 während einer zweiten Iteration auf eins (R1 = 1) erniedrigt wird, und während einer dritten Iteration auf Null (R1 = 0), wobei jedes Mal die Schleifenanweisung INST9 eine weitere Operation des Typs „Schleife genommen" bewirkt. „Schleifenaustritt" erfolgt, wenn die Schleifenanweisung INST9 zum vierten Mal angetroffen wird und der Schleifenzähler R1 gleich Null ist, was zu einer Operation des Typs „Schleife nicht genommen" führt, die die Programmsteuerung an die Fall-through-Anweisung INST10 abgibt. Die Programmausführung fährt dann mit der sequentiellen Ausführung von Anweisungen (z.B. der Anweisung INST11 und dann INST12) fort, bis eine weitere Verzweigung bzw. ein weiterer Sprung angetroffen wird.

[0011] Ein Problem bei Prozessoren, die Anweisungspuffer verwenden (d.h. dem oben besprochenen Prozessor **1400** ähnliche Prozessoren) besteht darin, daß die bedingte Verzweigungsoperation einer Schleifenanweisung (d.h. ob die Schleifenanweisung genommen oder nicht genommen wird) entschieden wird, wenn die Schleifenanweisung ausgeführt wird (z.B. wenn die Schleifenanweisung an die LS-Pipeline **132** ausgegeben wird; siehe **Fig.** 14). Wie bereits erwähnt, wird, wenn die Schleifenanweisung INST9 genommen wird, die Programmsteuerung zu der Zielanweisung INST2 weitergeleitet (springt dort hin). Das Problem besteht darin, daß nach dem Abrufen der Schleifenanweisung INST9 der Programmzähler **111** und die MMU **112** weiter sequentiell adressierte Anweisungen aus dem Speicher **101** abrufen, bis die Ausführungsstufe den injizierten Zählerwert erzeugt, der der Zielanweisung INST2 zugeordnet ist. Das heißt, zum Zeitpunkt der Ausführung der Schleifenanweisung wurden mehrere Fall-through-Anweisungen (z.B. INST10–INST12) abgerufen und in den verschiedenen Stufen vor der Ausführungsstufe **130** gespeichert, und die Zielanweisung INST2 wurde noch nicht abgerufen. Folglich muß der Prozessor **1400** nach jeder Schleifeniteration (jedes Mal, wenn die Schleifenanweisung INST9 ausgeführt wird) warten, während die Zielanweisung INST2 und nachfolgende Schleifenhauptteilanweisungen abgerufen, durch die verschiedenen Prozessorstufen geleitet und an die Ausführungsstufe **130** ausgegeben werden. Folglich erzeugt jede Schleifeniteration einen „Schleife-genommen-Kostenfaktor", der in der Regel durch die Anzahl von Prozessortaktzyklen zwischen der Ausführung der Schleifenanweisung und der Ausführung der Zielanweisung dieser Schleife gemessen wird. Der Schleife-genommen-Kostenfaktor ist besonders groß, wenn wie im Fall des Prozessors **1400** ein Prozessor mehrere Stufen und einen Anweisungspuffer (d.h. DIB **122**) vor der Ausführungsstufe enthält, aufgrund der Anzahl von Prozessortaktzyklen, die erforderlich ist, damit die Zielanweisung diese Stufen durchläuft.

Aufgabenstellung

[0012] Es wird ein Prozessor benötigt, der den Schleife-genommen-Kostenfaktor minimieren kann. Idealerweise wird ein Prozessor mit „Null-Overhead" benötigt, der den Schleife-genommen-Kostenfaktor eliminiert und Schleifenanweisungen ausführt, ohne jegliche Ausführungszyklen des Prozessors zu verbrauchen.

KURZE DARSTELLUNG DER ERFINDUNG

[0013] Die vorliegende Erfindung betrifft einen Prozessor, der einen Schleifenbetrieb mit Null-Overhead ermöglicht, indem vorhergesagt wird, ob eine cache-gespeicherte Schleifenanweisung während einer nächsten sequentiellen Schleifeniteration genommen oder nicht genommen wird, erkannt wird, wenn die cache-gespeicherte Schleifenanweisung in der nächsten sequentiellen Iteration ausgeführt wird und auf der Basis der Genommen-/nicht-genommen-Vorhersage entweder Anweisungen von innerhalb des Schleifenhauptteils oder Fall-through-Anweisungen abgerufen werden. Insbesondere werden während Schleifeniterationen, bei denen die Schleife genommen wird, Schleifenhauptteilanweisungen abgerufen, bevor die cache-gespeicherte Schleifenanweisung ausgeführt wird. Nach der Verifizierung der Schleife-genommen-Vorhersage werden die bereits abgerufenen Schleifenhauptteilanweisungen sofort an die Ausführungs-Pipeline des Prozessors ausgegeben. Folglich minimiert oder beseitigt die vorliegende Erfindung den in der Regel herkömmlichen Prozessoren zugeordneten Schleife-genommen-Kostenfaktor durch Beseitigung der „verschwendeten" Zyklen zwischen der Ausführung der Schleifenanweisung und der Ausführung des Schleifenhauptteils. Weiterhin werden während Schleifeniterationen, bei denen die Schleife nicht genommen wird, Fall-through-Anweisungen abgerufen, bevor die cache-gespeicherte Schleifenanweisung ausgeführt wird. Nach dem Verifizieren der Schleife-nicht-ge-

nommen-Vorhersage werden die bereits abgerufenen Fall-through-Anweisungen sofort an die Ausführungs-Pipeline des Prozessors ausgegeben, wodurch Verzögerungen beim Schleifenaustritt vermieden werden.

[0014] Gemäß einem Aspekt der vorliegenden Erfindung werden eine gewählte Schleifenanweisung und eine oder mehrere der Zielanweisung(en) der gewählten Schleife (d.h. die tatsächliche Zielanweisung und null oder mehr nachfolgende Anweisungen vom Inneren des Schleifenhauptteils) in einem speziellen Schleifen-Cache-Puffer (LCB) gespeichert, der von dem Anweisungspuffer des Prozessors (der als Decodieranweisungspuffer (DIB) bezeichnet ist) verschieden ist. Bei einer Ausführungsform wird die Schleifenanweisung beim Schleifeneintritt in der Vordecodierstufe des Prozessors erkannt und die Zielanweisung(en) wird sofort nach der Erkennung abgerufen. Die Zielanweisung(en) werden dann nach der nachfolgenden Ankunft in der Vordecodierstufe in dem LCB cachegespeichert. Jedes Mal, wenn die cache-gespeicherte Schleifenanweisung nachfolgend ausgeführt und eine vorherige Schleife-genommen-Vorhersage verifiziert wird, werden die Zielanweisung(en) sofort aus dem LCB ausgegeben (statt aus dem DIB ausgegeben zu werden). Durch Cache-Speicherung einer oder mehrerer Zielanweisungen in dem LCB auf diese Weise werden potentielle Prozessorverzögerungen, die dem Schreiben der Zielanweisung(en) aus dem Systemspeicher in den DIB zugeordnet sind, vermieden. Außerdem werden durch Cache-Speicherung einer oder mehrerer Zielanweisungen in dem LCB auf diese Weise Verzögerungen, die einer falschen Schleife-nicht-genommen-Vorhersage zugeordnet sind, minimiert (d.h. weil die Zielanweisung(en) ausgeführt werden können, während der übrige Schleifenhauptteil abgerufen wird).

[0015] Gemäß einem weiteren Aspekt der vorliegenden Erfindung werden außerdem Trigger-Adresse und eine Nachzieladresse beim Schleifeneintritt in dem LCB gespeichert. Die Trigger-Adresse dient in nachfolgenden Schleifeniterationen zum Erkennen des Endes des Schleifenhauptteils (d.h. zum Auslösen des nächsten spekulativen Vorabrufens entweder von Schleifenhauptteilanweisungen oder von Fall-through-Anweisungen). Bei einer Ausführungsform entspricht die Trigger-Adresse der vorletzten Anweisung in dem Schleifenhauptteil (d.h. der Anweisung, die der cache-gespeicherten Schleifenanweisung unmittelbar vorausgeht), und die Nachzieladresse entspricht der Anweisung in dem Schleifenhauptteil, der der letzten in dem LCB gespeicherten Zielanweisung unmittelbar nachfolgt. Während des Prozessorbetriebs wird die gespeicherte Trigger-Adresse mit durch die Vorabrufstufe des Prozessors erzeugten Programmzählerwerten verglichen. Wenn eine Übereinstimmung auftritt und die Schleife als genommen vorhergesagt wird, wird die Nachzieladresse sofort an die Vorabrufstufe ausgegeben. Die so abgerufenen Nachzielanweisungen (d.h. die nicht in dem LCB cache-gespeicherten Schleifenhauptteilanweisungen) werden anschließend aus dem DIB an die Prozessorausführungsstufe ausgegeben, nachdem die cache-gespeicherten Zielanweisungen aus dem LCB ausgegeben sind, wodurch die Schleifenausführung mit Null-Overhead ermöglicht wird.

[0016] Gemäß einem weiteren Aspekt der vorliegenden Erfindung enthält ein Prozessor sowohl eine „normale" Ausführungsstufe (Pipeline) als auch eine spezialisierte Schleifenausführungsschaltung, die Schleifenanweisungen ausführen kann, wenn die Ausführungsstufe gleichzeitig eine oder mehrere Anweisungen von innerhalb des Schleifenhauptteils ausführt. Bei einer Ausführungsform empfängt die Schleifenausführungsschaltung die cache-gespeicherte Schleifenanweisung aus dem LCB zusammen mit einem zugeordneten Schleifenzählerwert. Die Schleifenausführungsschaltung verifiziert dann die vorherige Genommen-/nicht-genommen-Vorhersage (die in dem LCB gespeichert ist), erniedrigt den Zählerwert und aktualisiert dann die Genommen-/nicht-genommen-Vorhersage für die nächste Schleifeniteration. Die Vorhersageaktualisierung wird durchgeführt, indem man bestimmt, ob der erniedrige Zählerwert größer oder gleich „1" (eins) oder gleich „0" (Null) ist. Wenn der erniedrigte Zählerwert größer oder gleich eins ist, dann wird die nächste Schleifeniteration als genommen vorhergesagt. Wenn der erniedrigte Zählerwert gleich Null ist, dann wird die nächste Schleifeniteration als nicht genommen vorhergesagt. Wenn die vorherige Vorhersage „Schleife genommen" ist und der Zählerwert Null ist, dann werden Korrekturmaßnahmen eingeleitet (z.B. werden Fall-through-Anweisungen abgerufen). Wenn die vorherige Vorhersage „Schleife nicht genommen" und der Zählerwert größer oder gleich eins ist, dann werden die cache-gespeicherten Zielanweisungen ausgegeben, und der übrige Schleifenhauptteil wird abgerufen. Die Schleifenausführungsschaltung schreibt außerdem den erniedrigten Schleifenzählerwert zurück in sein zugeordnetes Register.

[0017] Gemäß einem weiteren Aspekt der vorliegenden Erfindung werden einer oder mehrere Zeigerwerte in dem LCB gespeichert, um das Ausgeben von Anweisungen an die Schleifenausführungsschaltung und/oder die Ausführungsstufe zu koordinieren. Bei einer Ausführungsform dient ein (erster) Wert eines Schleifenausgabeindex zum Koordinieren des Ausgebens der cache-gespeicherten Schleifenanweisung aus dem LCB in die Schleifenausführungsschaltung, wenn der DIB-Ausgabezeiger die vorletzte Anweisung aus dem DIB in die Ausführungsstufe ausgibt. Folglich werden die cache-gespeicherte Schleifenanweisung und die vorletzte An-

weisung gleichzeitig in der Schleifenausführungsschaltung bzw. der „normalen" Prozessorausführungsstufe ausgeführt. Wenn die Ausführung der Schleifenanweisung eine vorherige Schleife-genommen-Vorhersage verifiziert, dann wird, wie bereits erwähnt, die DIB-Ausgabesteuerung auf die in dem LCB cache-gespeicherten Zielanweisungen verlegt. Zusätzlich wird mit einem DIB-Zeigerwert die nächste Anweisung angezeigt, die nach der Ausführung der cache-gespeicherten Schleifenanweisung aus dem DIB ausgegeben werden soll. Das heißt, der DIB-Zeigerwert identifiziert das DIB-Register, das die erste Nachzielanweisung (falls Schleife genommen) oder die erste Fall-through-Anweisung (falls Schleife nicht genommen) enthält. Der Schleifenausgabeindexwert und der DIB-Zeigerwert werden in jeder Schleifeniteration aktualisiert. Folglich liefern diese Zeiger ein flexibles und effizientes Verfahren zum koordinierten Ausgeben von Anweisungen aus dem LCB und dem DIB, wodurch weiterhin Schleifenoperationen mit Null-Overhead ermöglicht werden.

[0018] Gemäß einem weiteren Aspekt der vorliegenden Erfindung wird eine Vorabrufsperrfunktion für bestimmte kleine Schleifen verwendet (d.h. Schleifen mit Hauptteilen, die vollständig in den LCB und den DIB geschrieben werden können), bei der der Anweisungsabrufprozeß angehalten wird, während solche kleinen Schleifen ausgeführt werden. Ein in dem LCB gespeichertes Vorabrufsperrbit wird gesetzt, wenn eine kleine Schleife abgerufen wird, und der Schleifenausgabeindex und der DIB-Zeiger dienen zum wiederholten Ausführen des Schleifenhauptteils und der cache-gespeicherten Schleifenanweisung auf die oben beschriebene Weise. Schließlich wird das Vorabrufsperrbit ausgeschaltet, wenn die Schleifenausführungsschaltung eine Schleife-nicht-genommen-Vorhersage erzeugt, und an diesem Punkt werden Fall-through-Anweisungen auf die oben beschriebene Weise abgerufen und in den DIB geschrieben. Folglich wird eine hocheffiziente Schaltung zur Ausführung kleiner Schleifen bereitgestellt, die andernfalls bei herkömmlichen Prozessoren einen signifikanten Schleifenkostenfaktor erzeugen würden.

[0019] Gemäß einem weiteren Aspekt der vorliegenden Erfindung wird ein Prozessor, der zwei oder mehr LCBs enthält, so gesteuert, daß jeder LCB „verriegelt" (reserviert) wird, wenn dieser LCB eine spekulative Vorabrufoperation erzeugt, und wird „entriegelt", wenn die aktuelle Genommen-/nicht-genommen-Vorhersage auf die oben beschriebene Weise validiert wird. Wenn mehr Schleifen als die Anzahl von LCBs angetroffen werden, wird die Zuweisung eines LCB zu der zuletzt angetroffenen Schleife gemäß einem verriegelbaren, modifizierten LRU-Schema (LRU = least recently used) bestimmt. Wenn zum Beispiel zwei LCBs vorgesehen sind und jeweils zwei Schleifen darin cache-gespeichert werden und dann eine dritte Schleife angetroffen wird, wird die dritte Schleife unter den folgenden Umständen in einem der beiden LCBs cache-gespeichert. Wenn sich die dritte Schleife „unterhalb" (außerhalb) der ersten und der zweiten Schleife befindet, dann wird der erste LCB, der „verfügbar" wird, zum Cache-Speichern der dritten Schleife verwendet (d.h. der LCB, der zuletzt benutzt wurde, behält seine Schleifeninformationen). Wenn die dritte Schleife in den ersten beiden Schleifen vernestet ist, dann wird die Schleife, die weniger oft iteriert wurde, durch die dritte Schleife ersetzt.

[0020] Gemäß einem weiteren Aspekt der vorliegenden Erfindung basiert in Prozessoren, die Anweisungen, die Mehrfachanweisungsdatenwörter (z.B. Anweisungsdoppelwörter (IDWs)) verwenden, abrufen, die Entscheidung, eine erkannte Schleifenanweisung im Cache zu speichern, zum Beispiel auf der Position der Schleifenanweisung in ihrem Datenwort und der Anwesenheit anderer cache-gespeicherter Schleifenanweisungen in dem Datenwort. Ähnlich werden Trigger-Adressen und Nachzieladressen durch die Positionen der jeweiligen vorletzten und Nachzielanweisungen innerhalb der IDWs bestimmt.

KURZE BESCHREIBUNG DER ZEICHNUNGEN

[0021] Diese und andere Merkmale, Aspekte und Vorteile der vorliegenden Erfindung werden im Hinblick auf die folgende Beschreibung, die angefügten Ansprüche und die beigefügten Zeichnungen besser verständlich. Es zeigen:

[0022] **Fig.** 1 ein Blockschaltbild eines Prozessors, der Schleifenoperationen mit Null-Overhead ermöglicht, gemäß einer ersten Ausführungsform der vorliegenden Erfindung;

[0023] **Fig.** 2 ein vereinfachtes Flußdiagramm einer durch eine Schleifenausführungsschaltung des in **Fig.** 1 gezeigten Prozessors durchgeführten Funktion;

[0024] **Fig.** 3 ein vereinfachtes Flußdiagramm von Funktionen, die durch eine Schleifenoperationssteuerschaltung des in **Fig.** 1 gezeigten Prozessors durchgeführt werden;

[0025] **Fig.** 4(A), 4(B) und 4(C) vereinfachte Darstellungen des in **Fig.** 1 gezeigten Prozessors, wobei während des Schleifeneintritts durchgeführte spezifische Operationen angegeben sind;

[0026] **Fig.** 5(A) und 5(B) vereinfachte Darstellungen des in **Fig.** 1 gezeigten Prozessors, wobei während innerer Schleifenoperationen durchgeführte spezifische Operationen angegeben sind;

[0027] **Fig.** 6(A) und 6(B) vereinfachte Darstellungen des in **Fig.** 1 gezeigten Prozessors, wobei während des Schleifenaustritts durchgeführte spezifische Operationen angegeben sind;

[0028] **Fig.** 7(A) und 7(B) vereinfachte Darstellungen des in **Fig.** 1 gezeigten Prozessors, wobei als Reaktion auf falsche Schleife-genommen- und Schleife-nicht-genommen-Vorhersagen durchgeführte spezifische Operationen angegeben sind;

[0029] **Fig.** 8 ein vereinfachtes Diagramm eines Teils eines Softwareprogramms, der eine beispielhafte kleine Schleife enthält;

[0030] **Fig.** 9 eine vereinfachte Darstellung eines Prozessors gemäß einer weiteren Ausführungsform, wobei der Ausführung der in **Fig.** 8 abgebildeten kleinen Schleife zugeordnete spezifische Operationen angegeben sind;

[0031] **Fig.** 10 ein Blockschaltbild eines Prozessors mit mehreren Schleifen-Cache-Puffern gemäß einer weiteren Ausführungsform der vorliegenden Erfindung;

[0032] **Fig.** 11 ein vereinfachtes Diagramm eines Teils eines Softwareprogramms, der zwei vernestete Schleifen enthält;

[0033] **Fig.** 12(A) und 12(B) vereinfachte Diagramme von Softwareprogrammteilen, die drei Schleifen enthalten;

[0034] **Fig.** 13 ein Zustandsdiagramm eines verriegelbaren modifizierten LRU-Schemas (least-recently-used), das von dem Prozessor von **Fig.** 10 verwendet wird;

[0035] **Fig.** 14 ein Blockschaltbild eines herkömmlichen Mikroprozessors; und

[0036] **Fig.** 15 ein vereinfachtes Diagramm eines Teils eines Softwareprogramms, der eine Schleife enthält.

Ausführungsbeispiel

AUSFÜHRLICHE BESCHREIBUNG DER ZEICHNUNGEN

[0037] **Fig.** 1 ist ein Blockschaltbild eines Mikroprozessors (im folgenden einfach als „Prozessor" bezeichnet) **100**, der Schleifenoperationen mit Null-Overhead ermöglicht, gemäß einer vereinfachten ersten Ausführungsform der vorliegenden Erfindung. Der Prozessor **100** ist im allgemeinen in eine Vorabrufstufe **110A**, eine Abruf-/Vordecodierstufe **115A**, eine Decodierstufe **120A**, die einen DIB (Anweisungspuffer) **122** enthält, und eine Ausführungsstufe **130** aufgeteilt. Jede dieser Stufen und deren Komponenten werden durch Bezugszahlen identifiziert, die entsprechenden Stufen/Komponenten des herkömmlichen Prozessors **1400** (der oben mit Bezug auf **Fig.** 14 beschrieben wird) ähneln, und funktioniert im wesentlichen wie oben beschrieben (mit den nachfolgend dargelegten Ausnahmen). Folglich wird die ausführliche Beschreibung der im wesentlichen herkömmlichen Funktionen, die durch diese Stufen/Komponenten durchgeführt werden, der Kürze halber im folgenden weggelassen.

[0038] Zusätzlich zu den obenerwähnten, im wesentlichen herkömmlichen Stufen/Komponenten enthält der Prozessor **100** einen Schleifen-Cache-Puffer (LCB) **150**, eine Schleifenausführungsschaltung **160** und eine Schleifenoperationssteuerschaltung **170**, die kooperativ wirken, um eine Schleifenausführung mit Null-Overhead auf die nachfolgend beschriebene Weise zu ermöglichen.

Schleifen-Cache-Puffer

[0039] Mit Bezug auf die obere linke Seite von **Fig.** 1 enthält der LCB **150** zahlreiche Datenfelder, mit denen verschiedene Anweisungsdatenwörter und Steuerdatenwörter bzw. -Bit cache-gespeichert werden, die von der Schleifenausführungsschaltung **160** und der Schleifenoperationssteuerschaltung **170** auf nachfolgend beschriebene Weise verwendet werden. Diese Datenfelder enthalten ein Feld **151** für die Schleifenanweisung (LOOP INST), ein Feld **152** für die Trigger-Adresse (TRIGGER_PC), ein Feld **153** für die Zielanweisung (TAR-

GET INST), ein Feld **154A** für die Zieladresse (TARGET_PC), ein Feld **154B** für die Nachzieladresse (POST-TARGET_PC), ein Feld **155** für die Schleife-genommen-/nicht-genommen-Vorhersage (PREDICTION), ein Schleifenausgabeindexfeld **158** und ein DIB-Zeigerfeld **159**. Die in jedem dieser Felder gespeicherten Daten werden nachfolgend eingeführt, und die Verwendung dieser Daten während des Betriebs des Prozessors **100** wird in nachfolgenden Beispielen angegeben.

[0040] Das Schleifenanweisungsfeld **151** dient zum Speichern von Informationen, die einer ausgewählten (cache-gespeicherten) Schleifenanweisung zugeordnet sind. Insbesondere werden, wenn eine Schleifenanweisung auf die nachfolgend beschriebene Weise cache-gespeichert wird, die folgenden, der Schleifenanweisung zugeordneten Informationen in dem Schleifenanweisungsfeld **151** gespeichert: Der Operationscode (opcode; d.h. Informationen, die den Schleifenanweisungstyp identifizieren), Daten, die identifizieren, ob die Schleifenanweisung bedingt oder unbedingt ist, die Schleifengröße (z.B. 16 Bit oder 32 Bit) und das zugeordnete Schleifenzählerregister. Wie nachfolgend beschrieben, dienen diese Informationen dann zur Ausführung der cache-gespeicherten Schleifenanweisung unter Verwendung der Schleifenausführungsschaltung **160**.

[0041] Das Trigger-Adressenfeld **152** dient zum Erkennen (oder Vorhersagen), wenn die Schleifenanweisung, die in dem Feld **151** cache-gespeichert ist, in der Vorabrufstufe **110A** abgerufen wird (oder davor steht). Bei einer Ausführungsform ist das Trigger-Adressenfeld **152** die tatsächliche oder virtuelle Speicheradresse (bzw. der Programmzählerwert) der Speicherstelle, die entweder die cache-gespeicherte Schleifenanweisung enthält (wenn mehrere Anweisungen zusammen aus dem Speicher **101** gelesen werden; siehe unten) oder die Adresse der vorletzten Anweisung. Unter nochmaliger Bezugnahme auf die beispielhafte Schleife von **Fig.** 15 und unter der Annahme, daß pro Abrufzyklus eine Anweisung geschrieben wird, kann zum Beispiel, wenn die Schleifenanweisung INST9 in dem Schleifenanweisungsfeld **151** cachegespeichert wird, das Trigger-Adressenfeld **152** dann einen Wert enthalten, der der Speicheradresse X1000 (d.h. der Adresse, die der vorletzten Anweisung INST8 zugeordnet ist) entspricht. Die in dem Feld **152** gespeicherte Adresse wird nur dann aktualisiert, wenn die Schleife cache-gespeichert wird (d.h. während des Schleifeneintritts).

[0042] Das Zielanweisungsfeld **153** dient zum Speichern der tatsächlichen Zielanweisung (und von null oder mehr nachfolgenden Anweisungen) der in dem Schleifenanweisungsfeld **151** cache-gespeicherten Schleifenanweisung. Wenn zum Beispiel unter Verwendung des Beispiels von **Fig.** 15 die Schleifenanweisung INST9 in dem Feld **151** cachegespeichert wird, dann wird die Zielanweisung INST2 in dem Zielanweisungsfeld **153** cache-gespeichert. Eine oder mehrere nachfolgende Anweisungen (d.h. INST3, INST4 usw.) können abhängig von der Größe des Zielanweisungsfelds **153** ebenfalls cache-gespeichert werden. Bei einer tatsächlichen Ausführungsform wird das Zielanweisungsfeld **153** so bemessen, daß es Anweisungen aus zwei 64-Bit-Mehrfachanweisungswörtern (Anweisungsdoppelwörtern (IDWs), die unten weiter behandelt werden) speichert.

[0043] Das Zieladressenfeld **154A** ist die tatsächliche oder virtuelle Speicheradresse (bzw. der Programmzählerwert) der tatsächlichen Zielanweisung der cache-gespeicherten Schleifenanweisung. Wenn zum Beispiel das Zielanweisungsfeld **153** nur die Zielanweisung INST2 (**Fig.** 15) cache-speichert, dann speichert das Zieladressenfeld **154A** eine Adresse, die der Zielanweisung INST2 (z.B. Adresse X0010) zugeordnet ist. Das Zieladressenfeld **154A** wird nur während des Schleifeneintritts verwendet.

[0044] Das Nachzieladressenfeld **154B** ist die tatsächliche oder virtuelle Speicheradresse (oder der Programmzählerwert) der Anweisung, die der letzten in dem Zielanweisungsfeld **153** gespeicherten Anweisung unmittelbar nachfolgt. Wenn zum Beispiel das Zielanweisungsfeld **153** nur die Zielanweisung INST2 (**Fig.** 15) cache-speichert, dann speichert das Nachzieladressenfeld **154B** eine Adresse, die der „Nachziel"-Anweisung INST3 (z.B. Adresse X0011) zugeordnet ist. In einem anderen Beispiel speichert, wenn das Zielanweisungsfeld **153** sowohl die Zielanweisung INST2 als auch die Anweisung INST3 cache-speichert, das Nachzieladressenfeld **154B** dann Adresseninformationen, die die „Nachziel"-Anweisung INST4 (z.B. Adresse X0100) identifizieren. Man beachte, daß das Zieladressenfeld **154A** und das Nachzieladressenfeld **154B** kombiniert werden können, indem nach dem Schleifeneintritt ein vorbestimmtes Offset zu der tatsächlichen Zieladresse addiert wird.

[0045] Das Vorhersagefeld **155** speichert Daten, die eine aktuelle Vorhersage anzeigen, ob die cache-gespeicherte Schleifenanweisung während einer nächsten sequentiellen Iteration genommen oder nicht genommen wird. Bei einer Ausführungsform basiert diese Genommen-/nicht-genommen-Vorhersage auf einem Schleifenzählerwert, der der cache-gespeicherten Schleifenanweisung zugeordnet ist, der jedes Mal, wenn die cache-gespeicherte Schleifenanweisung ausgeführt wird, um eins erniedrigt wird und Schleife genommen vorhersagt, während der Schleifenzählerwert größer oder gleich eins ist, und Schleife nicht genommen vorhersagt, wenn der Schleifenzählerwert gleich null ist. Zum Beispiel speichert mit Bezug auf das in **Fig.** 15 ange-

gebene Beispiel das Vorhersagefeld **155** während der ersten drei Iterationen (z.B. während der Schleifenzähler R1 gleich drei, zwei bzw. eins ist) einen „True"-Wert (d.h. sagt Schleife genommen vorher), und wechselt dann auf einen „False"-Wert (d.h. sagt Schleife nicht genommen voraus), wenn der Schleifenzähler R1 auf null erniedrigt ist. Bei einer Ausführungsform wird das Vorhersagefeld **155** beim Schleifeneintritt immer auf den „True"-Wert gesetzt.

[0046] Das Schleifenausgabeindexfeld **158** und das DIB-Zeigerfeld **159** werden während des Ausgebens von Schleifenanweisungen aus dem Schleifenanweisungsfeld **151** (oder dem DIB **122**) und von Zielanweisungen aus dem Zielanweisungsfeld **153** verwendet. Das Schleifenausgabeindexfeld **158** identifiziert die Speicherstelle der cache-gespeicherten Schleifenanweisung (während des Schleifeneintritts) und der vorletzten Schleifenanweisung im DIB **122** (während aller nachfolgenden Schleifeniterationen). Das heißt, während des Schleifeneintritts zeigt das Schleifenausgabeindexfeld **158** auf das DIB-Register, das die Schleifenanweisung speichert. Nach dem Schleifeneintritt wird jedes Mal, wenn die vorletzte Anweisung abgerufen wird, das Schleifenausgabeindexfeld **158** berechnet (d.h. weil sich diese Anweisung in jeder Iteration in einem verschiedenen DIB-Register befinden kann). Wenn man zum Beispiel annimmt, daß jedes Register des DIB **122** nur eine Anweisung speichert, würde, wenn die vorletzte Anweisung INST8 (**Fig.** 15) während einer inneren Schleifeniteration in das Register REG2 geschrieben wird, der in dem Schleifenausgabeindexfeld **158** gespeicherte Wert das Register REG2 anzeigen. Mit dem DIB-Zeiger **159** wird die Speicherstelle der Nachzielanweisung in dem DIB **122**, nachdem sie unter Verwendung des (oben besprochenen) Nachzieladressenfelds **154B** abgerufen wurde, angezeigt. Ähnlich wie bei dem Schleifenausgabeindexfeld **158** wird der DIB-Zeiger **159** in jeder Schleifeniteration berechnet. Das Schleifenausgabeindexfeld **158** und der DIB-Zeiger **159** werden unten in zusätzlichem Detail besprochen.

[0047] Zusätzliche Datenfelder, die für Vielzweck- und Mehrfachschleifenausführungsformen der vorliegenden Erfindung verwendet werden, werden nachfolgend besprochen.

Schleifenausführungsschaltung

[0048] **Fig.** 2 ist ein Flußdiagramm einer vereinfachten Darstellung der durch die Schleifenausführungsschaltung **160** durchgeführten Funktion. Die Schleifenausführungsschaltung **160** ist eine Schaltung, die spezifisch so konstruiert ist, daß sie Schleifenanweisungen außerhalb der allgemeinen Prozessor-Pipeline (d.h. außerhalb der LS-Pipeline **132** und/oder der IP-Pipeline **136**) ausführt. Wie unten in zusätzlichem Detail beschrieben wird, wird während des Schleifeneintritts eine frisch cache-gespeicherte Schleifenanweisung in der Ausführungsstufe **130** ausgeführt, aber während aller anderen Iterationen (einschließlich Schleifenaustritt) durch die Schleifenausführungsschaltung **160**. Mit Bezug auf den Anfang von **Fig.** 2 wird eine cache-gespeicherte Schleifenanweisung an die Schleifenausführungsschaltung **160** aus dem Schleifenanweisungsfeld **151** des LCB **150** (oben besprochen) ausgegeben, und ein zugeordneter Schleifenzählerwert wird aus dem Schleifenzählerregister **105A** ausgegeben (gelesen) (beides ist in **Fig.** 1 gezeigt). Wie in **Fig.** 1 gezeigt, enthält die Schleifenausführungsschaltung **160** separate Ausführungsstufen EX1 und EX2/WB („WB" bedeutet „write-back"). Wieder mit Bezug auf **Fig.** 2 wird mit der Stufe EX1 bestimmt, ob der Zählerwert eine Vorhersageänderung erfordert (z.B. wenn der Zählerwert eins ist; Block **210**). Mit dieser Bestimmung wird die vorherige Genommen-/nicht-genommen-Vorhersage (Block **220**) verifiziert und außerdem werden damit Korrekturmaßnahmen eingeleitet (z.B. das Abrufen einer unvorhergesagten Anweisung und/oder das Invalidieren des aktuellen Inhalts des LCB **150**, wie unten besprochen), wenn die vorherige Genommen-/nicht-genommen-Vorhersage als falsch bestimmt wird (Block **230**). Der Zählerwert wird dann systematisch verändert (z.B. um eins erniedrigt; Block **240**), ein aktualisierter Genommen-/nicht-genommen-Vorhersagewert wird erzeugt und in dem Feld **155** gespeichert (Block **250**), und der erniedrigte Adressenzähler wird dann aus der Stufe EX2/WB wieder in das Schleifenzählerregister **105A** zurückgeschrieben (Block **260**). Wie unten ausführlicher dargelegt wird, ermöglicht die Schleifenausführungsschaltung **160** einen Betrieb mit Null-Overhead durch Ermöglichung der gleichzeitigen Ausführung der cache-gespeicherten Schleifenanweisung (in der Schleifenausführungsschaltung **160**) und der vorletzten Anweisung (in der Ausführungsstufe **130**).

Schleifenoperationssteuerschaltung

[0049] Die Schleifenoperationssteuerschaltung **170** verwendet einen oder mehrere Automaten (FSMs) und zugeordnete Schaltkreise zur Koordination des Betriebs des LCB **150** und der Schleifenausführungsschaltung **160** mit den Operationen der Vorabrufstufe **110A**, der Abruf-/Vordecodierstufe **115A**, der Decodierstufe **120** und der Ausführungsstufe **130** auf die nachfolgend beschriebene Weise. Bei einer Ausführungsform kann die Schleifenoperationssteuerschaltung **170** in drei Funktionsblöcke aufgeteilt werden: einen Vorabrufblock, der mit der Vorabrufstufe **110A** in Wechselwirkung tritt, einen Vordecodierblock, der mit der Abruf-/Vordecodierstu-

fe **115A** in Wechselwirkung tritt, und einen Ausgabeblock, der mit der Decodierstufe **120A** in Wechselwirkung tritt. Wie in dem folgenden Beispiel dargelegt wird, sind die durch diese Funktionsblöcke durchgeführten Operationen abhängig von der durchgeführten Schleifeniteration (d.h. Schleifeneintritt, innere Schleifeniterationen und Schleifenaustritt) unterschiedlich.

[0050] **Fig.** 3 enthält eine Reihe von Flußdiagrammen, die Operationen abbilden, die durch die Schleifenoperationssteuerschaltung **170** während des Schleifeneintritts (äußerste linke Spalte), der inneren Schleife (mittlere Spalte) bzw. des Schleifenaustritts (äußerste rechte Spalte) durchgeführt werden, gemäß einer vereinfachten Ausführungsform der vorliegenden Erfindung. Weiterhin sind Operationen, die dem Vorabruffunktionsblock der Schleifenoperationssteuerschaltung **170** zugeordnet sind, im oberen Drittel von **Fig.** 3, Operationen, die dem Vordecodierfunktionsblock der Schleifenoperationssteuerschaltung **170** zugeordnet sind, in dem mittleren Drittel von **Fig.** 3 und Operationen, die dem Ausgabefunktionsblock der Schleifenoperationssteuerschaltung **170** zugeordnet sind, in dem unteren Drittel von **Fig.** 3 positioniert.

[0051] Die in **Fig.** 3 abgebildeten Flußdiagramme werden unten mit Bezug auf **Fig.** 4(A) bis 6(B) beschrieben, die sich ihrerseits auf den in **Fig.** 15 abgebildeten beispielhaften Softwareprogrammteil **1500** beziehen. Man beachte, daß die nachfolgende Beschreibung annimmt, daß der LCB **150** zu Anfang „leer" ist (d.h. keine Informationen enthält, die mit zuvor cache-gespeicherten Schleifenanweisungen zusammenhängen), und daß eine einzige relativ lange Schleife durch den Prozessor **100** ohne Unterbrechung ausgeführt wird (d.h. keine injizierten Anweisungen aus asynchronen Verzweigungen, Traps oder anderen Interrupt-Operationen).

Schleifeneintrittsphase

[0052] Mit Bezug auf die äußerste linke Spalte von **Fig.** 3 beginnt die Schleifeneintrittsphase der Schleifenausführung, wenn eine neue Schleifenanweisung identifiziert wird (Block **310**), die bei der vorliegenden Ausführungsform durch den Vordecodierfunktionsblock der Schleifenoperationssteuerschaltung **170** durchgeführt wird. Nachdem die Schleifenanweisung identifiziert und eine Entscheidung, die Schleifenanweisung im Cache zu speichern, getroffen wurde (z.B. durch Bestimmung, daß der LCB verfügbar ist), wird das Ziel der Schleifenanweisung abgerufen und die Schleifenanweisung wird zum Aktualisieren mehrerer Felder des LCB **150** verwendet (Block **314**).

[0053] **Fig.** 4(A) ist ein Blockschaltbild eines Teils des Prozessors **100** während der Ausführung der Operationen, die den Blöcken **310** und **314** zugeordnet sind, gemäß einer vereinfachten Ausführungsform. Wie in **Fig.** 4(A) angegeben, wird die Schleifenanweisung INST9 identifiziert, wenn sie in die Vordecodierstufe **118A** eintritt, indem zum Beispiel jede die Vordecodierstufe **118A** durchlaufende Anweisung mit Opcodes verglichen wird, die bekannten Schleifenanweisungen zugeordnet sind. Wenn entschieden wird, die Schleifenanweisung INST9 im Cache zu speichern, wird die Adresse für die Zielanweisung INST2 (z.B. X0010) durch die Schleifenoperationssteuerschaltung **170** an die Vorabrufstufe **110A** weitergeleitet. Zusätzlich werden verschiedene Informationen bezüglich der Schleifenanweisung INST9 in dem Schleifenanweisungsfeld **151** gespeichert, die Adresse der vorletzten Anweisung INST8 (z.B. X1000) wird in dem Trigger-Adressenfeld **152** gespeichert, die Nachzielanweisung (z.B. die Adresse der Anweisung INST3, d.h. X0011) wird berechnet und in dem Nachzieladressenfeld **154B** gespeichert, die Genommen-/nicht-genommen-Vorhersage wird auf Schleife genommen („T") gesetzt, und der Schleifenausgabeindex wird auf das DIB-Register REG2 gesetzt. Man beachte, daß die Fall-through-Anweisung INST10 in der Abrufstufe **116** empfangen wird, wenn die Schleifenanweisung INST9 identifiziert und cache-gespeichert wird.

[0054] Wieder mit Bezug auf **Fig.** 3 wird die im Block **314** abgerufene Zielanweisung, wenn sie in der Vordecodierstufe des Prozessors ankommt, in dem LCB cache-gespeichert (Block **317**).

[0055] **Fig.** 4(B) ist ein Blockschaltbild des Prozessors **100** nach der dem Block **314** zugeordneten Operation gemäß dem etablierten vereinfachten Beispiel. Wie in **Fig.** 4(B) angegeben, wird die Zielanweisung INST2 aus der Vordecodierstufe **118A** in das Zielanweisungsfeld (TRGT INST) **153** geschrieben, und das DIB-Zeigerfeld wird aktualisiert, um aus Gründen, die unten deutlich werden, das DIB-Register REG1 zu identifizieren. Man beachte, daß an diesem Punkt in dem Betrieb ein der Decodierstufe **120A** zugeordneter Ausgabezeiger **410** auf das DIB-Register REG1 zeigt, das gerade die vorletzte Anweisung INST8 speichert. Man beachte außerdem, daß ein der Vordecodierstufe **118A** zugeordneter DIB-Schreibzeiger auf das DIB-Register REG4 zeigt. Bei einer Ausführungsform wird die Zielanweisung INST2 in das DIB-Register REG4 geschrieben, wird aber aus dem LCB **150** ausgegeben, wie unten besprochen.

[0056] Mit Bezug auf den unteren linken Teil von **Fig.** 3 umfassen Operationen, die dem Ausgabefunktions-

block der Schleifenoperationssteuerschaltung **170** zugeordnet sind, das Ausgeben der Schleifenanweisung an die Ausführungs-Pipeline des Prozessors (Block **320**) und das anschließende Weiterleiten der Ausgabesteuerung an den LCB per Schleifenausgabeindex, um die Zielanweisung(en) auszugeben (Block **324**), und das nachfolgende Rückleiten der Ausgabesteuerung an den DIB per Schleifenzeiger, um die Nachzielanweisungen auszugeben (Block **327**).

[0057] **Fig.** 4(C) ist ein Blockschaltbild des Prozessors **100** nach der den Blöcken **320**, **324** und **327** zugeordneten Operation gemäß dem etablierten vereinfachten Beispiel. Wie in **Fig.** 4(C) angegeben, gibt nach dem Erreichen des DIB-Registers REG2 (d.h. des DIB-Registers, das durch das Schleifenausgabeindexfeld **158** identifiziert wird) der DIB-Ausgabezeiger die Schleifenanweisung INST9 (die durch den Ausgabezeiger **410**(t1) angezeigt wird), aus, verschiebt dann die Ausgabesteuerung an das Zielanweisungsregister **153** und gibt die cache-gespeicherte Zielanweisung INST2 (die durch den Ausgabezeiger **410** (t2) angezeigt wird) aus. Man beachte, daß bei bestimmten Prozessoren das Ausgeben von Zielanweisung(en) aus dem LCB anstelle des DIB die Schleifenausführung verschnellern kann, wodurch gegenüber Prozessoren, die Zielanweisungen nicht in einem eigenen Puffer cache-speichern, ein Vorteil erreicht wird. Nach dem Ausgeben der Zielanweisung(en) wird die Ausgabesteuerung durch den in dem DIB-Zeigerfeld **159** gespeicherten Wert an das DIB-Register REG4 zurückgeschoben, wodurch die Nachzielanweisung INST3 (die durch den Ausgabezeiger **410**(t3) angezeigt wird) ausgegeben wird. Man beachte, daß das Verwenden des DIB-Zeigerfelds **159** dem Prozessor **100** gestattet, die Fall-through-Anweisung INST10 zu „überspringen" (d.h. sie nicht auszugeben/auszuführen), die in das DIB-Register REG3 geschrieben wurde.

Innere Schleifenphase

[0058] Mit Bezug auf den Anfang der mittleren Spalte von **Fig.** 3 werden nun Operationen beschrieben, die dem Vorabruffunktionsblock der Schleifenoperationssteuerschaltung **170** während innerer Schleifeniterationen zugeordnet sind. Die nachfolgend gegebene Besprechung der inneren Schleifeniteration hängt mit der in **Fig.** 15 gezeigten zusammen (d.h, wo der Schleifenzähler R1 = 2 oder R1 = 1 ist). Während die Anweisungen aus dem Schleifenhauptteil sequentiell abgerufen und ausgeführt werden, erzeugt der Programmzähler schließlich einen Wert, der mit der gespeicherten Trigger-Adresse übereinstimmt (Block **330**). Nach der Erkennung der Trigger-Adresse injiziert, weil die aktuelle Vorhersage darin besteht, daß die Schleife „genommen" werden wird, der Vorabruffunktionsblock der Schleifenoperationssteuerschaltung die cache-gespeicherte Nachzieladresse (Block **337**), wodurch der Programmzähler auf einen Wert zurückgesetzt wird, der der Zielanweisung der Schleife zugeordnet ist.

[0059] **Fig.** 5(A) ist ein Blockschaltbild des Prozessors **100** nach der den Blöcken **330** und **337** zugeordneten Operation gemäß dem etablierten vereinfachten Beispiel. Wie in **Fig.** 5(A) gezeigt, werden die sequentiell erzeugten Programmzählerwerte, die durch die Vorabrufstufe **110A** erzeugt werden, durch die Schleifenoperationssteuerschaltung **170** überwacht und mit dem in dem Trigger-Adressenfeld **152** gespeicherten Wert verglichen. Wenn der Programmzählerwert, der der vorletzten Anweisung INST8 (d.h. X1000_PC(t4)) zugeordnet ist, durch die Schleifenoperationssteuerschaltung **170** erkannt wird, wird die in dem Nachzieladressenfeld **154B** gespeicherte Nachzieladresse (POST-TARGET_PC(t5)) zu der Vorabrufstufe **110A** gesendet, wodurch bewirkt wird, daß die Vorabrufstufe die Nachzielanweisung INST3 abruft.

[0060] Mit Bezug auf die Mitte der mittleren Spalte von **Fig.** 3 umfassen Operationen, die dem Vordecodierfunktionsblock der Schleifenoperationssteuerschaltung **170** während einer inneren Schleifeniteration zugeordnet sind, das Aktualisieren des Schleifenausgabeindex und des DIB-Zeigers (Block **340**). Diese Operation ist in **Fig.** 5(A) durch das gestrichelte Oval t6 angegeben. Insbesondere wird das Schleifenausgabeindexfeld **158** aktualisiert, so daß es das DIB-Register REG1 identifiziert, und das DIB-Zeigerfeld **159** wird so aktualisiert, daß es das DIB-Register REG2 identifiziert.

[0061] Am Ende in der mittleren Spalte von **Fig.** 3 ist zu sehen, daß Operationen, die dem Ausgabefunktionsblock der Schleifenoperationssteuerschaltung **170** während einer inneren Schleifeniteration zugeordnet sind, mit dem Ausgeben der vorletzten Anweisung aus dem DIB an die Prozessorausführungs-Pipeline beginnen, und zum selben Zeitpunkt wird die cache-gespeicherte Schleifenanweisung aus dem LCB an die Schleifenausführungsschaltung ausgegeben (Block **350**). Die Schleifenausführungsschaltung verarbeitet dann die Schleifenanweisung auf die oben beschriebene Weise, um die zuvor hergestellte Genommen-/nicht-genommen-Vorhersage zu verifizieren, und erzeugt dann eine aktualisierte Genommen-/nicht-genommen-Vorhersage, die für die nächste sequentielle Schleifeniteration bereitgestellt wird (Block **353**). Unter der Annahme, daß die vorherige „Genommen"-Vorhersage gültig war, wird die Ausgabesteuerung dann verschoben, um die in dem LCB cache-gespeicherte Zielanweisung auszugeben (Block **355**), und als letztes wird die Ausgabesteu-

erung verschoben, um die in dem DIB cache-gespeicherte Nachzielanweisung auszugeben (Block **357**).

[0062] **Fig.** 5(B) ist ein Blockschaltbild des Prozessors **100** während den Operationen, die den Blöcken **350**, **353**, **355** und **337** zugeordnet sind, gemäß dem etablierten vereinfachten Beispiel. Wie in **Fig.** 5(B) gezeigt wird, wenn der Ausgabezeiger **410**(t7) das DIB-Register REG1 (das in dem Schleifenausgabeindex **158** identifiziert wird) erreicht und die vorletzte Anweisung INST8 ausgibt, die Schleifenanweisung INST9 aus dem Feld **151** an die Schleifenausführungsschaltung **160** ausgegeben (wie durch den Schleifenausführungszeiger **510**(t7) angegeben). Auf der Basis des in **Fig.** 15 gezeigten Beispiels umfaßt die Ausführung cache-gespeicherter Schleifenanweisung während der inneren Schleifenphase das Verifizieren der vorherigen Vorhersage (d.h. daß der Schleifenzählerwert größer oder gleich eins ist, wie bei dem vorliegenden Beispiel), das Erniedrigen des Schleifenzählers, und dann wird das Vorhersagefeld **155** aktualisiert. Man beachte, daß für das Beispiel von **Fig.** 15, wenn der Schleifenzählerwert von eins auf null erniedrigt wird, das Vorhersagefeld **155** auf „Nicht genommen" (N) gewechselt wird, wie durch das gestrichelte Oval t8 angegeben. Da die vorherige Vorhersage als gültig befunden wurde, verschiebt sich anschließend die Ausgabesteuerung zu dem Zielanweisungsfeld **153**, wie durch den Ausgabezeiger **410**(t9) angegeben, und die cache-gespeicherte Zielanweisung INST2 wird an die Ausführungsstufe **130** ausgegeben. Als letztes verschiebt sich die Ausgabesteuerung zurück zu dem DIB-Register REG2 mittels des in dem DIB-Zeigerfeld **159** gespeicherten Werts und die Nachzielanweisung INST3 wird an die Ausführungsstufe **130** ausgegeben, wie durch den Ausgabezeiger **410**(t10) angegeben.

Schleifenaustrittsphase

[0063] Mit Bezug auf den Anfang der äußersten rechten Spalte von **Fig.** 3 werden nun Operationen beschrieben, die dem Ausgabefunktionsblock der Schleifenoperationssteuerschaltung **170** während des Schleifenaustritts zugeordnet sind. Bei dem in **Fig.** 15 eingeführten Beispiel erfolgt der Schleifenaustritt, wenn der Schleifenzähler R1 gleich null ist. Ähnlich wie in der inneren Schleifenphase beginnt der Schleifenaustritt mit dem Erkennen des Übertragens der Trigger-Adresse durch die Prozessorvorabrufstufe (Block **360**). Nach der Erkennung der Trigger-Adresse unternimmt der Vorabruffunktionsblock nichts, weil die aktuelle Vorhersage darin besteht, daß die Schleife „nicht genommen" wird, und die Vorabrufstufe erzeugt die Adresse, die den Fall-through-Anweisungen der Schleife zugeordnet ist (Block **365**).

[0064] **Fig.** 6(A) ist ein Blockschaltbild des Prozessors **100** nach den Operationen, die den Blöcken **360** und **367** zugeordnet sind, gemäß dem etablierten vereinfachten Beispiel. Wie in **Fig.** 6(A) gezeigt, werden die durch die Vorabrufstufe **110A** erzeugten sequentiell erzeugten Programmzählerwerte durch die Schleifenoperationssteuerschaltung **170** überwacht und mit dem in dem Trigger-Adressenfeld **152** gespeicherten Wert verglichen. Wenn der Programmzählerwert, der der vorletzten Anweisung INST8 (d.h. X1000-PC(t11)) zugeordnet ist, durch die Schleifenoperationssteuerschaltung **170** erkannt wird, unternimmt die Schleifenoperationssteuerschaltung **170** wegen der Schleife-nicht-genommen-Vorhersage (angegeben durch das „N" in dem Vorhersagefeld **155**) nichts. Wie auf der rechten Seite der Vorabrufstufe **110A** angegeben, entspricht folglich die nächste zu dem Speicher **101** gesendete Adresse (d.h. X10011(t12)) der Fall-through-Anweisung INST10.

[0065] Mit Bezug auf die Mitte der äußersten rechten Spalte von **Fig.** 3 umfassen Operationen, die dem Vordecodierfunktionsblock der Schleifenoperationssteuerschaltung **170** während des Schleifenaustritts zugeordnet sind, das Aktualisieren des Schleifenausgabeindex (Block **370**) auf ähnliche Weise wie oben mit Bezug auf innere Schleifeniterationen beschrieben. In diesem Beispiel wird, wie durch das gestrichelte Oval t13 in **Fig.** 6(A) gezeigt, das Schleifenausgabeindexfeld **158** so aktualisiert, daß es das DIB-Register REG3 identifiziert.

[0066] Wie am Ende der äußersten rechten Spalte von **Fig.** 3 gezeigt, beginnt ähnlich wie bei inneren Schleifenoperationen der Ausgabefunktionsblock der Schleifenoperationssteuerschaltung **170** während des Schleifenaustritts mit dem Ausgeben der vorletzten Anweisung aus dem DIB an die Prozessorausführungs-Pipeline, und gleichzeitig wird die cache-gespeicherte Schleifenanweisung aus dem LCB an die Schleifenausführungsschaltung ausgegeben (Block **380**). Die Schleifenausführungsschaltung verarbeitet dann die Schleifenanweisung auf die oben beschriebene Weise, um die zuvor hergestellte Nicht-genommen-Vorhersage zu verifizieren (Block **384**). Unter der Annahme, daß die vorherige „Nicht-genommen"-Vorhersage gültig war, wird die Ausgabesteuerung dann verschoben, um die erste in dem DIB cache-gespeicherte Fall-through-Anweisung auszugeben (Block **387**), wodurch der Schleifenbetrieb beendet wird. Man beachte, daß das Ausgeben der ersten Fall-through-Anweisung ein Überspringen einer DIB-Registerspeicherstelle, die eine nicht ausgeführte Kopie der Schleifenanweisung speichert (siehe die Besprechung unten mit dem Thema Prozessoren, die IDWs verwenden) erfordern kann.

[0067] **Fig.** 6(B) ist ein Blockschaltbild des Prozessors **100** während den Operationen, die den Blöcken **380**, **384** und **387** zugeordnet sind, gemäß dem etablierten vereinfachten Beispiel. Wie in **Fig.** 6(B) gezeigt, wird, wenn der Ausgabezeiger **410**(t14) das DIB-Register REG3 (das in dem Schleifenausgabeindex **158** identifiziert wird) erreicht und die vorletzte Anweisung INST8 ausgibt, die Schleifenanweisung INST9 aus dem Feld **151** an die Schleifenausführungsschaltung **160** ausgegeben (angegeben durch den Schleifenausführungszeiger **510**(t14)). Auf der Basis des in **Fig.** 15 gezeigten Beispiels umfaßt die Ausführung cache-gespeicherter Schleifenanweisung das Verifizieren der vorherigen Nicht-genommen-Vorhersage (d.h. daß der Schleifenzählerwert gleich null ist). Da die vorherige Vorhersage als gültig befunden wurde, verschiebt sich die Ausgabesteuerung anschließend zu dem DIB-Register REG4 mittels des in dem DIB-Zeigerfeld **159** gespeicherten Werts, und die Fall-through-Anweisung INST10 wird an die Ausführungsstufe **130** ausgegeben, wie durch den Ausgabezeiger **410**(t15) angegeben.

Spezialfälle

[0068] Während der Prozessor **100** (wie oben beschrieben) Operationen mit Null-Overhead ermöglicht, an denen eine einzige relativ lange ununterbrochene Schleifenanweisung beteiligt ist, werden außerdem mehrere zusätzliche neuartige Aspekte der vorliegenden Erfindung bereitgestellt, die das Behandeln von Mehrfachanweisungswörtern und das Auftreten falsch vorhergesagter Schleifen, injizierte Anweisungen aus asynchronen Verzweigungen oder anderen Interrupt-Mechanismen, kleine Schleifen, Schleifen mit Nullhauptteil und mehrfache (vernestete und nicht vernestete) Schleifenoperationen betreffen. Jeder dieser Spezialfälle wird nachfolgend beschrieben.

[0069] Falsche Genommen-/nicht-genommen-Vorhersagen Gemäß einem weiteren Aspekt der vorliegenden Erfindung wird die einer cache-gespeicherten Schleife zugeordnete Genommen-/nicht-genommen-Vorhersage aus dem aktuellen Wert des Schleifenzählers, der systematischen Veränderung des der cache-gespeicherten Schleifenanweisung zugeordneten Schleifenzählerwerts und einer Annahme, daß der Schleifenzählerwert zwischen Ausführungen der cache-gespeicherten Schleifenanweisung nicht verändert wird, bestimmt. Unter Verwendung des oben dargelegten vereinfachten Beispiels wird, weil die cache-gespeicherte Schleifenanweisung INST9 von dem Typ ist, der den Schleifenzähler R1 in jeder Iteration um eins erniedrigt und austritt, wenn der Schleifenzähler R1 Null ist, die Schleife bei der nächsten Iteration als nicht genommen vorhergesagt, wenn der Schleifenzähler R1 gleich Null ist. In allen anderen Fällen (d.h. wenn der Schleifenzähler R1 größer oder gleich eins ist) wird die nachfolgende Schleifeniteration als genommen vorhergesagt.

[0070] Um einen Prozessorfehler aufgrund der falsch vorhergesagten Schleifenoperation zu verhindern, ist die Schleifenoperationssteuerschaltung **170** mit einem Fehlerkorrekturblock ausgestattet, der Vorhersageverifizierungssignal(e) aus der Schleifenoperationssteuerschaltung **170** empfängt und eine Fehlerkorrektur einleitet, wenn eine falsche Vorhersage auftritt (z.B. wenn der Schleifenzähler innerhalb des Schleifenhauptteils verändert wird). Solche Verifizierungssignale zeigen zwei Formen von Fehlern an: erstens wurde die Schleife als genommen vorhergesagt, aber der Zählerwert hat einen Nicht-genommen-Wert (z.B. Null); und zweitens wurde die Schleife als nicht genommen vorhergesagt, und der Zählerwert hat einen Genommen-Wert (z.B. eins oder mehr).

[0071] **Fig.** 7(A) ist ein Blockschaltbild des Prozessors **100** während einer inneren Schleifenoperation (oben besprochen), wobei der Fall dargestellt ist, in dem die beispielhafte Schleife als genommen vorhergesagt wurde, aber der Schleifenzähler R1 zu Null bestimmt wurde. Wenn der Ausgabezeiger **410**(t7) die vorletzte Anweisung INST8 ausgibt, wird wie oben beschrieben die Schleifenanweisung INST9 aus dem Feld **151** an die Schleifenausführungsschaltung **160** ausgegeben (wie durch den Schleifenausführungszeiger **510**(t7) angegeben). In diesem Beispiel ist der Schleifenzählerwert Null (oder eine negative Zahl), wenn die vorherige Vorhersage verifiziert wird. Da die in dem Vorhersagefeld **155** gespeicherte Vorhersage „genommen" (T) ist, ist die abgerufene Anweisung, auf die der DIB-Zeiger **159** zeigt, die Nachzielanweisung INST3, und dies ist falsch, weil der Schleifenzählerwert Null ist. Um diesen Fehler zu korrigieren, injiziert die Schleifenoperationssteuerschaltung **170** dann den Programmzählerwert 2+TRIGGER_PC(t8A), der bewirkt, daß die Vorabrufstufe die Fall-through-Anweisung INST10 adressiert (d.h. die Adresse X1010(t8A) erzeugt). Zusätzlich wird der Ausgabezeiger bei Erkennung der Fall-through-Anweisung INST10 in der Vordecodierstufe **118A** zurückgesetzt.

[0072] **Fig.** 7(B) ist ein Blockschaltbild des Prozessors **100** während einer Schleifenaustrittsoperation (oben besprochen), wobei der Fall dargestellt ist, in dem die beispielhafte Schleife als nicht genommen vorhergesagt wurde, aber der Schleifenzähler R1 als größer als Null bestimmt wird. In diesem Fall gibt der Ausgabezeiger **410**(t14) die vorletzte Anweisung INST8 aus, und der Schleifenausführungszeiger **510**(t14) gibt die Schleifenanweisung INST9 an die Schleifenausführungsschaltung **160** aus. In diesem Beispiel ist der Schleifenzähler-

wert eins (oder ein anderer positiver Wert), wenn die vorherige Vorhersage verifiziert wird. Da die in dem Vorhersagefeld **155** gespeicherte vorherige Vorhersage „Nicht genommen" (N) ist, ist die abgerufene Anweisung, auf die der DIB-Zeiger **159** zeigt, die Fall-through-Anweisung INST10, und das ist wegen des positiven Schleifenzählerwerts falsch. Um diesen Fehler zu korrigieren, injiziert die Schleifenoperationssteuerschaltung **170** dann den Programmzählerwert POST-TRIGGER_PC(t15A), der bewirkt, daß die Vorabrufstufe die Nachtriggeranweisung INST3 adressiert (d.h. die Adresse X0011(t15A) erzeugt). Weiterhin wird der Ausgabezeiger **410**(t15A) auf das Zielanweisungsfeld **153** verschoben, von dem aus die Zielanweisung INST2 ausgegeben wird, und der DIB-Zeiger wird nach der Erkennung der Zielanweisung INST3 in der Vordecodierstufe **118A** zurückgesetzt. Man beachte, daß durch Speichern mehrerer Anweisungen in dem Zielanweisungsfeld **153** ein durch eine solche falsche Vorhersage verursachter Kostenfaktor (Verzögerung) vermieden werden kann. Das heißt, wenn Anweisungen aus dem Zielanweisungsfeld **153** ausgegeben werden, bis die Nachzielanweisung den DIB **122** erreicht, dann führt das falsch vorhergesagte Schleifenende zu einer Prozessorverzögerung von Null.

Unterbrechungen

[0073] Gemäß einem weiteren Aspekt der vorliegenden Erfindung wird eine cache-gespeicherte Schleife invalidiert, wenn eine asynchrone Verzweigung, ein Trap oder ein anderer Mechanismus eine Anweisung während des Schleifeneintritts (d.h. bevor der LCB **150** vollständig aktualisiert ist) von außerhalb des Schleifenhauptteils injiziert. Bei einer Ausführungsform ist ein LCB-gültig-Bit in dem LCB **150** vorgesehen, um einen Betrieb mit einem unvollständigen LCB-Datensatz zu verhindern. Das Gültig-Bit wird auf „false" gesetzt, wenn der LCB-Aktualisierungsprozeß beginnt (z.B. in dem in **Fig.** 4(A) angegebenen Zustand, wenn die Schleifenanweisung INST9 cache-gespeichert ist, aber bevor die Zielanweisung INST2 cache-gespeichert ist). Dieses Gültig-Bit wird nach dem Abschluß des LCB-pktualisierungsprozesses (z.B. wenn die Zielanweisung INST2 cache-gespeichert ist und der Schleifenausgabeindex und DIB-Zeigerfelder aktualisiert sind) auf „true" gewechselt. Wenn eine Anweisung von auflerhalb des Schleifenhauptteils bei der Vordecodierung während des Cache-Speicherns einer Schleifenanweisung (d.h. während des Schleifeneintritts) erkannt wird, dann wird das Gültig-Bit auf „false" gesetzt, und die Schleifen-Cache-Speicherung wird beendet (d.h. die Steuerung wird auf die injizierte Anweisung bzw. die injizierten Anweisungen übergewechselt). Andere mit Mehrfachschleifenoperationen zusammenhängende Gültigkeitsprobleme werden mit Bezug auf die unten dargelegte zweite Ausführungsform besprochen.

Kleine Schleifen

[0074] Kleine Schleifen (d.h. mit einer relativ kleinen Anzahl von Anweisungen) stellen ein Synchronisierungsproblem dar, wenn die Trigger-Adresse erkannt wird, wodurch der Start einer aktuellen Iteration angezeigt wird, bevor die cache-gespeicherte Schleifenanweisung aus einer vorherigen Iteration ausgeführt ist. Um diese Situation zu behandeln, wird in dem LCB ein Vorhersage-gültig-Bit vorgesehen, mit dem Operationen der Vorabrufstufe **110A** mit denen der Schleifenausführungsschaltung **160** synchronisiert werden. Jedes Mal, wenn eine cache-gespeicherte Schleifenanweisung durch die Schleifenausführungsschaltung **160** ausgeführt wird, wird das Vorhersage-gültig-Bit auf einen „true"-Wert (z.B. 1) gesetzt, und ein nachfolgendes Abrufen der cache-gespeicherten Schleifenanweisung (tatsächlich der vorletzten Anweisung) setzt das Vorhersage-gültig-Bit auf einen „False"-Wert (z.B. 0). Wenn sowohl das Vorabrufen als auch die Ausführung gleichzeitig auftreten, dann bleibt das Vorhersage-gültig-Bit gleich (die Ausführung hebt den Effekt der Vorabrufung auf). Im Fall bestimmter kleiner Schleifen wird jedoch, wenn das Vorhersage-gültig-Bit „false" ist, wenn die Trigger-Adresse erkannt wird, das Vorabrufen angehalten, bis das Vorhersage-gültig-Feld **156** zu „true" wechselt.

[0075] Gemäß einem weiteren Aspekt der vorliegenden Erfindung stellen bestimmte kleine Schleifen (z.B. weniger als fünf Anweisungen in dem oben dargelegten vereinfachten Beispiel) insofern einen Spezialfall dar, als die Ausführung solcher kleinen Schleifen unter Verwendung des oben dargelegten Prozesses (d.h. Abrufen des Schleifenhauptteils aus dem Speicher **101** in jeder Schleifenoperation) unnötige Verzögerungen einführen könnte, während die Pipeline gefüllt wird. Um diese Verzögerungen zu vermeiden, wird, wie in dem nachfolgend angegebenen Beispiel dargelegt, der gesamte Schleifenhauptteil in dem Zielanweisungsfeld **153** (und notwendigenfalls in einem oder mehreren Registern des DIB **122**) gespeichert, und das Abrufen von Anweisungen wird bis zum Schleifenaustritt gesperrt (d.h. es wird Schleife nicht genommen vorhergesagt).

[0076] **Fig.** 8 ist ein vereinfachtes Diagramm eines Abschnitts **810** eines Softwareprogramms mit einer kleinen Schleife, die durch Anweisungen INST02 bis INST05 gebildet wird. Ähnlich wie bei dem in **Fig.** 15 angegebenen Beispiel setzt die Anweisung INST01 einen Schleifenzähler R2 auf einen ganzzahligen Wert von drei (angegeben durch „[R2 == 3]"), und die Schleifenanweisung INST05 ist eine Schleifenanweisung, die wie

nachfolgend beschrieben arbeitet.

[0077] **Fig.** 9 ist ein vereinfachtes Blockschaltbild eines Prozessors **100A** zur Ausführung des Programmabschnitts **810** (**Fig.** 8) gemäß einer weiteren Ausführungsform der vorliegenden Erfindung. Der Prozessor **100A** ist dem (oben beschriebenen) Prozessor **100** ähnlich, mit der Ausnahme, daß der LCB **150A** ein Sperrfeld **910** enthält (alle anderen Stufen und Schaltungen arbeiten im wesentlichen wie oben beschrieben). Wie bereits erwähnt, wird, wenn die kleine Schleife (z.B. die Anweisungen INST02 bis INST05) des Softwareprogrammabschnitts **810** aus dem Speicher **101** abgerufen wird, das Sperrfeld **910** auf „true" (Y) gesetzt, und das Vorabrufen wird angehalten. Wie in **Fig.** 9 gezeigt, wird an diesem Punkt jede Anweisung der kleinen Schleife entweder in dem LCB **150A** oder in dem DIB **122** gespeichert. In diesem Beispiel werden die Schleifenanweisung INST05 und die Zielanweisung INST02 in den zugeordneten Feldern von **150A** gespeichert, die Nachzielanweisung INST03 wird in dem DIB-Register REG1 gespeichert und die vorletzte Anweisung INST04 in dem DIB-Register REG2. Das Schleifenausgabeindexfeld **158** und der DIB-Zeiger **159** werden permanent (d.h. bis zum Schleifenaustritt) gespeichert. Folglich umfassen innere Schleifeniterationen das Ausgeben der Zielanweisung INST02 aus dem Zielanweisungsfeld **153** (angegeben durch den Zeiger **410**(t21)) nach dem Verifizieren einer vorherigen Schleife-genommen-Vorhersage und das anschließende Verschieben der Ausgabesteuerung gemäß dem DIB-Zeigerfeld **159** (wie durch den Zeiger **410**(t22) angegeben) zu dem DIB-Register REG1, wodurch die Nachzielanweisung INST03 ausgegeben wird. Die Ausgabesteuerung schreitet dann zu dem DIB-Register REG2 (angegeben durch den Zeiger **410**(t23)) voran, wodurch die vorletzte Anweisung INST04 ausgegeben wird. Da das DIB-Register REG2 mit dem in dem Schleifenausgabeindexfeld **158** gespeicherten Wert übereinstimmt, wird die Schleifenanweisung INST05 auch aus dem LCB **150A** (angegeben durch den Zeiger **150**(t23)) an die Schleifenausführungsschaltung **160** ausgegeben, und dort wird die vorherige Vorhersage verifiziert und eine neue Vorhersage erzeugt. Dieser Prozeß wird fortgesetzt, bis die neue Vorhersage „nicht genommen" ist, wobei die Schleifenoperationssteuerschaltung **170A** den Programmzähler für die Fall-through-Anweisung INST06 injiziert, wodurch bewirkt wird, daß die Vorabrufstufe **110A** eine entsprechende Adresse (d.h. Y0110(t24)) ausgibt.

Schleifen mit Nullhauptteil

[0078] Eine Schleife mit Nullhauptteil ist eine Schleife, die keine Anweisungen in ihrem Hauptteil aufweist (d.h. die Schleifenanweisung führt in einer Schleife zu sich selbst). Eine Schleife, deren Ziel vor der Schleifenanweisung liegt, wird hier als eine Vorwärtsschleife bezeichnet. Diese beiden Schleifentypen werden in einer Ausführungsform der vorliegenden Erfindung funktional unterstützt (d.h. die Leistungsfähigkeit dieser Schleifentypen mit Null-Overhead wird ermöglicht), indem ein LCB-Bit zum Identifizieren solcher Schleifen vorgesehen wird. Diese Schleifen werden zum Beispiel dadurch identifiziert, daß die Zieladresse mit der Trigger-Adresse verglichen wird, wenn die Schleifenanweisung cache-gespeichert wird. Wenn das LCB-Bit eine Null-Hauptteil-Schleifenanweisung anzeigt, dann wird das Vorabrufen des Schleifenhauptteils nicht initialisiert und die Schleifenanweisung wird wie eine andere Verzweigungsanweisung behandelt, wobei in diesem Fall das Schleifenziel durch die EX1-Stufe der LS-Pipeline injiziert wird, wenn der Schleifenzähler von Null verschieden ist.

Mehrfache Schleifen

[0079] Gemäß einem weiteren Aspekt der vorliegenden Erfindung werden mehrfache Schleifen (z.B. vernestete Schleifen) behandelt, indem zwei oder mehr LCBs vorgesehen werden und die Schleifenoperationssteuerschaltung so modifiziert wird, daß sie die Schleifen-Cache-Speicherung in den beiden LCBs koordiniert.

[0080] **Fig.** 10 ist ein vereinfachtes Diagramm eines Prozessors **100B**, der Schleifenoperationen mit Null-Overhead ermöglicht, gemäß einer weiteren Ausführungsform der vorliegenden Erfindung. Der Prozessor **100B** ist dem (oben beschriebenen) Prozessor **100** ähnlich, mit der Ausnahme, daß die beiden LCBs (LCB1 und LCB2) vorgesehen sind, wobei jeder dieser LCBs Felder zum Speichern der Anweisung, von Adressen- und Steuerdaten, wie oben mit Bezug auf den LCB **150** beschrieben, enthält. Andere Stufen und Komponenten des Prozessors **100B** funktionieren im wesentlichen wie oben beschrieben (mit den nachfolgend angegebenen Ausführungen).

[0081] **Fig.** 11 ist ein vereinfachtes Diagramm eines Abschnitts **1110** eines Softwareprogramms, der zwei vernestete Schleifen enthält, die durch die Anweisungen IMST21 bis INST31 gebildet werden. Wenn sie durch den Prozessor **100B** abgerufen werden, werden die Anweisungen INST21 bis INST27 auf die oben beschriebene Weise sequentiell abgerufen und ausgeführt. Nachdem die Schleifenanweisung INST28 das erste Mal (d.h. zu einem Zeitpunkt t1) erkannt wurde, prüft der Prozessor **100B** auf verfügbare LCBs (an diesem Punkt

in dem Beispiel werden beide als verfügbar angenommen). Als nächstes wird die durch die Anweisungen INST24 bis INST28 gebildete Schleife (die als die „vernestete Schleife" bezeichnet wird) cachegespeichert und gemäß dem oben beschriebenen Prozeß unter Verwendung des LCB1 des Prozessors **110B** ausgeführt. Insbesondere führt der Prozessor **100B** folgendes durch: Cache-Speicherung der Schleifenanweisung INST28 in dem Feld **151** des LCB1, Abrufen der Zielanweisung INST24, Schreiben der Adresse der vorletzten Anweisung INST27 in das Feld **152** des LCB1 und Schreiben der Nachzieladresse, die der Nachzielanweisung INST24 zugeordnet ist, in das Feld **154B** des LCB1. Zusätzlich wird ein dem LCB1 zugeordnetes „Verriegelungs"-Bit gesetzt, wenn das Vorabrufen der Zielanweisung INST24 eingeleitet wird. Wenn anschließend die Zielanweisung INST24 zu einem Zeitpunkt t2 in der Vordecodierstufe ankommt, wird sie in dem Feld **153** des LCB1 cache-gespeichert. Das „Verriegelungs"-Bit wird ausgeschaltet, wenn INST28 zu der EX1-Stufe (d.h. **161**) des Prozessors **100B** reicht, in der der Prozessor **100B** die Genommen-/nicht-genommen-Vorhersage validiert (das „Verriegelungs"-Bit wird ausgeschaltet). Das „Verriegelungs"-Bit wird anschließend jedes Mal dann eingeschaltet, wenn ein spekulativ vorabgerufenes Ziel in den DIB **122** geschrieben wird (d.h. der LCB1 wird jedes Mal auf der Basis der vorabgerufenen Zielinformationen – Ausgabeindex und DIB-Zeiger – aktualisiert), und wird jedes Mal ausgeschaltet, wenn die aktuelle Genommen-/nicht-genommen-Vorhersage auf die oben beschriebene Weise validiert wird. Die vernestete Schleife wird auf diese Weise bis zum Schleifenaustritt verarbeitet, und an diesem Punkt wird die Steuerung an die Anweisung INST29 abgegeben. Man beachte jedoch, daß die in dem LCB1 gespeicherten Werte gehalten werden.

[0082] Die Ausführung des Programmabschnitts **1110** wird dann fortgesetzt, bis die Schleifenanweisung INST31 erkannt wird. Obwohl der LCB1 verfügbar sein kann, wird die Schleifenanweisung INST31 zu einem Zeitpunkt t3 in dem Feld **151** des LCB2 cache-gespeichert (wie auf der rechten Seite von **Fig.** 11 angegeben). Zu diesem Zeitpunkt wird das Vorabrufen der Zielanweisung INST21, wie oben beschrieben, initialisiert, und es wird ein dem LCB2 zugeordnetes „Verriegelungs"-Bit gesetzt. Zusätzlich wird die Adresse der vorletzten Anweisung INST30 in dem Feld **152** des LCB2 gespeichert, und die der Nachzielanweisung INST22 zugeordnete Nachzieladresse wird auf der Basis von durch die Schleifenanweisung IMST31 bereitgestellten Informationen in dem Feld **154B** des LCB2 gespeichert.

[0083] Wenn nachfolgend die Zielanweisung INST21 in der Vordecodierstufe zu einem Zeitpunkt t4 ankommt, wird sie ebenfalls in dem Feld **153** des LCB2 cache-gespeichert. Wie bei LCB1 wird das LCB2 zugeordnete „Verriegelungs"-Bit anschließend bei jeder LCB2-Aktualisierung ein- und jedes Mal, wenn die vorherige Vorhersage verifiziert wird, ausgeschaltet.

[0084] Die sequentielle Anweisungsverarbeitung wird dann fortgesetzt, bis die vorletzte Anweisung INST30 ein zweites Mal erkannt wird (d.h. durch Vergleichen des ausgegebenen Programmzählers mit der in dem Feld **152** des LCB2 gespeicherten Trigger-Adresse). Die cache-gespeicherte Schleifenanweisung INST31 wird dann auf die oben beschriebene Weise aus dem LCB1 ausgegeben und ausgeführt. Wenn die äußere Schleife genommen wird, wird die Ausgabesteuerung auf die oben beschriebene Weise an die Zielanweisung INST21 zurückgegeben. Andernfalls wird die Steuerung an die Fall-through-Anweisung INST32 abgegeben.

[0085] Eine weitere Mehrfachschleifensituation entsteht, wenn mehr Schleifen in einem Programm angetroffen werden, als LCBs verfügbar sind. Wenn zum Beispiel sowohl der LCB1 als auch der LCB2 des Prozessors **110B** belegt sind und eine dritte Schleife angetroffen wird, muß der Prozessor **100B** bestimmen, ob er LCB1 ersetzen (überschreiben), LCB2 ersetzen oder die dritte Schleife einfach nicht cache-speichern soll. Wie durch das in **Fig.** 12(A) gezeigte vereinfachte Diagramm angegeben, wird, wenn sich eine dritte nicht vernestete Schleife (d.h. einschließlich der Schleifenanweisung LP-3) außerhalb der ersten beiden Schleifen befindet und beide LCBs nicht verriegelt sind (d.h. die sowohl LCB1 als auch LCB2 zugeordneten „Verriegelungs"-Bit sind ausgeschaltet), der erste verfügbare LCB (d.h. der LCB, der zuvor die der Schleifenanweisung LP-1 zugeordnete Schleife speichert) mit der Schleifenanweisung LP-3 ersetzt. Im Gegensatz dazu wird, wie durch das in **Fig.** 12(B) gezeigte vereinfachte Diagramm gezeigt, wenn die dritte Schleife in den ersten beiden Schleifen vernestet ist, die Schleife (LP-1 oder LP-2), die weniger oft iteriert wird, ersetzt. Das heißt, in dem in **Fig.** 12(B) gezeigten Fall wird der die Schleifenanweisung LP-2 speichernde LCB mit der Schleifenanweisung LP-3 ersetzt.

[0086] **Fig.** 13 ist ein Zustandsdiagramm eines verriegelbaren modifizierten Schemas des Typs LRU (least-recently-used), das von der Schleifenoperationsteuerschaltung **170B** des Prozessors **100B** (**Fig.** 10) zum Zuweisen des LCB1 und des LCB2 gemäß einem weiteren Aspekt der vorliegenden Erfindung verwendet wird. Das LRU-Schema enthält vier Zustände: einen Zustand „0" (Null), in dem kein LCB cache-gespeichert wird, einen Zustand „1" (eins), in dem nur LCB1 cache-gespeichert wird, und Zustände „2" (zwei) und „3" (drei), in denen sowohl LCB1 als auch LCB2 cache-gespeichert werden. Wenn beide LCBs cache-gespeichert wer-

den, verschiebt sich die Steuerung zwischen den Zuständen „2" und „3" abhängig davon, welche der cache-gespeicherten Schleifen am häufigsten iteriert wird (d.h. abhängig davon, welche LCBs „getroffen" werden).

Prozessoren, die IDWs verwenden

[0087] Die oben angegebene Beschreibung verwendet stark vereinfachte Beispiele zur Erläuterung. Zum Beispiel betreffen die oben angegebenen Beispiele einen vereinfachten Prozessor, bei dem in jedem Abrufzyklus eine Anweisung aus dem Systemspeicher gelesen wird. Wie bereits erwähnt, transferieren bestimmte Prozessoren jedoch Anweisungen unter Verwendung von Anweisungsdoppelwörtern (IDWs) aus dem Systemspeicher in die Prozessor-Abruf-/Vordecodierstufe. Diese Prozessoren leiten die abgerufenen IDWs (oder umgeordnete Teile davon) häufig bis zum Ausgeben aus dem DIB in die Ausführungsstufe durch die Prozessorstufen. Zum Beispiel ist der von der Infineon Technologies AG hergestellte TriCore-Mikroprozessor ein eingebetteter Doppel-Pipeline-RISC+DSP-Prozessor, der bei jeder Abrufanforderung 64 Bit adressenausgerichteten Programmcode abruft. Diese 64-Bit-Programmcode-IDWs werden auf ähnliche Weise wie oben beschrieben in die Abruf-/Vordecodierstufe des Prozessors abgerufen. Jede TriCore-Anweisung kann entweder eine 16-Bit-Anweisung oder eine 32-Bit-Anweisung sein und ist bezüglich Architektur entweder als eine IP-Anweisung (d.h. gekennzeichnet für Ausführung in der IP-Pipeline) oder als eine LS-Anweisung (z.B. Schleifenanweisungen sind LS-Anweisungen) definiert. Wenn eine 32-Bit-Anweisung in zwei 16-Bit-Teile aufgetrennt wird, die in zwei sequentiellen IDWs übertragen werden, werden die beiden Teile in der Vordecodierstufe vor der Ausgabe an den DIB wieder zusammengestellt. Während des nachfolgenden Ausgebens werden pro Prozessorzyklus maximal zwei Anweisungen aus dem DIB an die Ausführungsstufe ausgegeben (d.h. eine 16-Bit- oder 32-Bit-IP-Anweisung an die IP-Pipeline und/oder eine 16-Bit- oder 32-Bit-LS-Anweisung an die LS-Pipeline).

[0088] Da ein IDW bis zu vier Anweisungen (I1, I2, I3 und I4) aufweisen kann, könnten beliebige oder alle dieser vier Anweisungen eine cache-gespeicherte oder eine nicht cache-gespeicherte Schleifenanweisung sein. Da nur ein Schleifenzielvorabruf auf einmal eingeleitet werden kann, muß entschieden werden, welches Schleifenanweisungsziel für jedes IDW abgerufen werden soll. Bei einer Ausführungsform basiert die Entscheidung, eine nicht cache-gespeicherte Schleife (L.U) im Cache zu speichern, auf der Anordnung der Anweisungen I1–I4 in dem IDW (d.h. die in der Vordecodierstufe umgeordnet wird), die in Tabelle 1 (unten) gezeigt ist.

Tabelle 1

Nr.	Anw. I1	Anw. I2	Anw. I3	Anw. I4
1	L.U	-	-	-
2	nL	L.U	-	-
3	nL	nL	L.U	-
4	nL	nL	nL	L.U
5	L.C.nT	L.U	-	-
6	L.C.nT	nL	L.U	-
7	L.C.nT	nL	nL	L.U
8	nL	L.C.nT	L.U	-
9	nL	L.C.nT	nL	L.U
10	nL	nL	L.C.nT	L.U

Legende: L = Schleifenanweisung, U = nicht cache-gespeichert, nL = keine Schleife, C = cache-gespeichert, nT = nicht genommen (Vorhersage) und „-" = gleichgültig.

[0089] Nunmehr mit Bezug auf die (obige) Tabelle 1, wird jede nicht cache-gespeicherte Schleife (L.U) nur

dann cachegespeichert, wenn das (in der Vordecodierstufe umgeordnete) IDW zu einem der obigen 10 Typen gehört (jede Zeile der Tabelle stellt ein IDW in der Vordecodierung dar). Wenn zum Beispiel im Fall Nr. 1 in Tabelle 1 die nicht cache-gespeicherte Schleifenanweisung die erste Anweisung (I1) in einem IDW ist, dann wird die Schleifenanweisung cache-gespeichert (d.h. LCB-Aktualisierung ist qualifiziert). Mit Bezug auf Fall Nr. 5, wenn die erste Anweisung (I1) eine cache-gespeicherte Schleife ist, die als nicht genommen vorhergesagt wird (d.h. L.C.nT) und die zweite Anweisung (I2) eine nicht cache-gespeicherte Schleife ist, dann wird die nicht cache-gespeicherte Schleifenanweisung cache-gespeichert (d.h. LCB-Aktualisierung ist qualifiziert). Die übrigen Fälle geben andere Anordnungen an, in denen nicht cache-gespeicherte Schleifen cache-gespeichert werden. Man beachte, daß zusätzlich zu der Positionierung in einem IDW, wie in Tabelle 1 angegeben, das Schleifen-Cache-Speichern nur dann beginnen würde, wenn ein LCB verfügbar ist (die LCB-Verfügbarkeit wird zum Beispiel durch die in **Fig.** 13 gezeigte und oben besprochene LRU entschieden).

[0090] Zusätzlich zu den Entscheidungen bezüglich der Schleifen-Cache-Speicherung (oben besprochen) erfordert die Verwendung von IDWs außerdem eine Modifikation der Art der Definition der Felder jedes LCB. Falls zum Beispiel jedes DIB-Register zwei IDWs speichert und jeder LCB Register zum Speichern von zwei IDWs enthält, müssen der Schleifenausgabeindex und der DIB-Zeiger so modifiziert werden, daß sie jede der potentiellen Anweisungsadressen in diesen Schaltungen adressieren. Zum Beispiel kann eine cache-gespeicherte Zielanweisung in einem IDW enthalten sein, das Vorschleifenanweisungen enthält, die alle in einem zugeordneten LCB-Register cache-gespeichert werden. In diesem Fall wird ein Zeiger verwendet, um das Ausgeben aus der tatsächlichen Speicherstelle der Zielanweisung zu beginnen. Ähnlich würde die Trigger-Adresse jeder cache-gespeicherten Schleifenanweisung die tatsächliche Schleifenanweisung abrufen, wenn sich sowohl die Schleifenanweisung als auch die zugeordnete vorletzte Anweisung in demselben IDW befinden. Auch wenn die abgerufene Version der cache-gespeicherten Schleifenanweisung (und etwaiger Fall-through-Anweisungen) den DIB erreicht, wird wieder die tatsächliche cache-gespeicherte Schleifenanweisung aus dem LCB ausgegeben. Obwohl die vorliegende Erfindung mit Bezug auf bestimmte spezifische Ausführungsformen beschrieben wurde, ist für Fachleute ersichtlich, daß die erfindungsgemäßen Merkmale der vorliegenden Erfindung auch auf andere Ausführungsformen anwendbar sind, die alle in den Schutzumfang der vorliegenden Erfindung fallen sollen. Zum Beispiel kann das Zielfeld **153** (**Fig.** 1) weggelassen werden, eine oder mehrere Zielanweisungen nicht cache-zu-speichern, kann jedoch die Schleifenausführung während des Schleifeneintritts verzögern und außerdem die Ausführung bestimmter kleinerer Schleifen verzögern. Obwohl die oben angegebenen Beispiele hauptsächlich eine Art von Prozessor und eine Art von Schleifenanweisung betreffen, ist außerdem für Fachleute erkennbar, daß neuartige Aspekte der vorliegenden Erfindung verwendet werden können, um den Betrieb mit Null-Overhead anderer Prozessortypen und für andere Schleifenanweisungstypen bereitzustellen.

Patentansprüche

1. Schaltung in einem Prozessor, der eine Vorabrufstufe zum Abrufen von Programmanweisungen aus einer Systemspeichereinrichtung und eine Ausführungsstufe zur Ausführung der abgerufenen Programmanweisungen enthält, zur Steuerung einer iterativen Ausführung einer Gruppe der Programmanweisungen, die eine Schleifenanweisung und eine Zielanweisung umfassen, wobei jede Iteration mit der Ausführung der Zielanweisung beginnt und mit der Ausführung der Schleifenanweisung endet und wobei bei der Ausführung der Schleifenanweisung am Ende jeder Iteration ein Schleifenzählerwert bestimmt, ob eine neue Iteration eingeleitet und die Zielanweisung ausgeführt wird oder ob die iterative Ausführung beendet und eine Fall-through-Anweisung ausgeführt wird, wobei die Schaltung folgendes umfaßt:
ein Mittel, das am Anfang einer ersten Iteration bestimmt, ob am Ende der ersten Iteration eine zweite Iteration eingeleitet oder ob die iterative Ausführung am Ende der ersten Iteration beendet wird; und
ein an die Vorabrufstufe angekoppeltes Mittel, das erkennt, daß eine erste vorbestimmte Anweisung der Gruppe der Programmanweisungen durch die Vorabrufstufe abgerufen wurde; und
ein Mittel, das die Vorabrufstufe so steuert, daß sie eine zweite vorbestimmte Anweisung der Gruppe der Programmanweisungen abruft, wenn die zweite Iteration vorhergesagt wird, und das die Vorabrufstufe so steuert, daß sie das Fall-through abruft, wenn die Beendigung der iterativen Ausführung vorhergesagt wird.

2. Schaltung nach Anspruch 1, wobei das Vorhersagemittel eine Schleifenausführungsschaltung zum Ausführen der Schleifenanweisung, während die erste vorbestimmte Anweisung in der Ausführungsstufe ausgeführt wird, umfaßt.

3. Schaltung nach Anspruch 1, weiterhin mit einem Mittel, das einen durch das Vorhersagemittel erzeugten aktuellen Vorhersagewert speichert, und einem Mittel, das die Vorhersage am Ende der ersten Iteration durch Vergleichen eines der Schleifenanweisung zugeordneten Zählerwerts mit dem aktuellen Vorhersagewert veri-

fiziert.

4. Schaltung nach Anspruch 1, wobei das Vorhersagemittel ein Mittel zum Lesen eines der Schleifenanweisung zugeordneten Schleifenzählerwerts aus einem vordefinierten Register, ein Mittel, das den Schleifenzählerwert erniedrigt, und ein Mittel zum Schreiben des erniedrigten Schleifenzählerwerts in das vordefinierte Register umfaßt.

5. Schaltung nach Anspruch 1, wobei das Erkennungsmittel einen Schleifen-Cache-Puffer mit einem Speicherfeld zum Speichern eines Trigger-Adressenwerts und ein Mittel zum Vergleichen des Trigger-Adressenwerts mit durch die Vorabrufstufe erzeugten Adressenwerten umfaßt.

6. Schaltung nach Anspruch 5, wobei die Vorabrufstufe einen Programmzähler zum Erzeugen einer Reihe von Programmzählerwerten und eine Speicherverwaltungseinheit zum Erzeugen von Speicheradressenwerten als Reaktion auf die Reihe von Programmzählerwerten umfaßt und wobei das Vergleichsmittel ein Mittel zum Vergleichen der Reihe von Programmzählerwerten mit den Trigger-Adressenwerten umfaßt.

7. Schaltung nach Anspruch 6, wobei das Mittel zum Steuern der Vorabrufstufe ein Mittel zum Rücksetzen des Programmzählers, so daß die Speicherverwaltungseinheit eine der zweiten vorbestimmten Anweisung entsprechende Adresse erzeugt, umfaßt.

8. Verfahren zum Betrieb eines Prozessors, der eine Vorabrufstufe zum Abrufen von Programmanweisungen aus einer Systemspeichereinrichtung und eine Ausführungsstufe zur Ausführung der abgerufenen Programmanweisungen enthält, wobei eine Gruppe der Programmanweisungen eine Schleifenanweisung und eine Zielanweisung umfaßt, wobei jede Iteration einer iterativen Ausführung der Gruppe von Anweisungen mit der Ausführung der Zielanweisung beginnt und mit der Ausführung der Schleifenanweisung endet, und wobei bei Ausführung der Schleifenanweisung am Ende jeder Iteration ein Schleifenzählerwert bestimmt, ob eine neue Iteration eingeleitet und die Zielanweisung ausgeführt wird oder ob die iterative Ausführung beendet und eine Fall-through-Anweisung ausgeführt wird, mit den folgenden Schritten:
Vorhersagen am Anfang einer ersten Iteration, ob eine zweite Iteration am Ende der ersten Iteration eingeleitet wird oder ob die iterative Ausführung am Ende der ersten Iteration beendet wird; und
Erkennen, daß eine erste vorbestimmte Anweisung der Gruppe der Programmanweisungen durch die Vorabrufstufe abgerufen wurde; und
spekulatives Abrufen entweder einer zweiten vorbestimmten Anweisung der Gruppe der Programmanweisungen, wenn die zweite Iteration vorhergesagt wird, oder des Fall-through, wenn eine Beendigung der iterativen Ausführung vorhergesagt wird.

9. Verfahren nach Anspruch 8, wobei das Vorhersagen weiterhin das Ausführen der Schleifenanweisung unter Verwendung einer Schleifenausführungsschaltung, während die erste vorbestimmte Anweisung unter Verwendung der Ausführungsstufe ausgeführt wird, umfaßt.

10. Verfahren nach Anspruch 8, weiterhin mit dem Speichern eines aktuellen Vorhersagewerts vor der ersten Iteration und dem Verifizieren der Vorhersage am Ende der ersten Iteration durch Vergleichen eines der Schleifenanweisung zugeordneten Zählerwerts mit dem aktuellen Vorhersagewert.

11. Verfahren nach Anspruch 8, wobei das Vorhersagen weiterhin das Lesen eines der Schleifenanweisung zugeordneten Schleifenzählerwerts aus einem vordefinierten Register, das Erniedrigen des Schleifenzählerwerts und das Schreiben des erniedrigten Schleifenzählerwerts in das vordefinierte Register umfaßt.

12. Verfahren nach Anspruch 8, wobei das Erkennen weiterhin das Speichern eines Trigger-Adressenwerts und das Vergleichen des Trigger-Adressenwerts mit durch die Vorabrufstufe erzeugten Adressenwerten umfaßt.

13. Verfahren nach Anspruch 12, wobei die Vorabrufstufe einen Programmzähler zum Erzeugen einer Reihe von Programmzählerwerten und eine Speicherverwaltungseinheit zum Erzeugen von Speicheradressenwerten als Reaktion auf die Reihe von Programmzählerwerten umfaßt und wobei das Erkennen weiterhin das Vergleichen der Reihe von Programmzählerwerten mit den Trigger-Adressenwerten umfaßt.

14. Verfahren nach Anspruch 13, wobei das spekulative Abrufen weiterhin das Rücksetzen des Programmzählers dergestalt, daß die Speicherverwaltungseinheit eine der zweiten vorbestimmten Anweisung entsprechende Adresse erzeugt, umfaßt.

15. Schaltung in einem Prozessor, der einen Anweisungspuffer zum Speichern mehrerer Programmanweisungen, eine Ausführungsstufe zur Ausführung der sequentiell aus dem Anweisungspuffer ausgegebenen Programmanweisungen und einen Schleifenzählerspeicher zum Speichern eines Schleifenzählerwerts enthält, umfassend:
einen Schleifen-Cache-Puffer mit einem ersten Speicherfeld zum Speichern einer gewählten Schleifenanweisung und einem zweiten Speicherfeld zum Speichern einer der gewählten Schleifenanweisung zugeordneten Zielanweisung;
eine Schleifenausführungsschaltung zum Ausführen der gewählten Schleifenanweisung, zum Aktualisieren des in dem Schleifenzählerspeicher gespeicherten Schleifenzählerwerts und zum Erzeugen eines Schleife-genommen-Steuersignals, wenn der Schleifenzähler gleich einem vorbestimmten Wert ist; und
ein Mittel zum Ausgeben der Zielanweisung aus dem zweiten Speicherfeld an die Ausführungsstufe, wenn die Schleifenausführungsschaltung das Schleife-genommen-Steuersignal erzeugt.

16. Schaltung nach Anspruch 15,
wobei der Schleifen-Cache-Puffer weiterhin ein Trigger-Adressenfeld zum Speichern eines ersten Adressenwerts, der einer vorletzten Anweisung eines der ausgewählten Schleifenanweisung zugeordneten Schleifenhauptteils entspricht, und ein Nachzieladressenfeld zum Speichern eines zweiten Adressenwerts, der einer Nachzielanweisung des Schleifenhauptteils entspricht, enthält; und
wobei der Prozessor weiterhin folgendes umfaßt:
eine Vorabrufstufe zum Abrufen von Programmanweisungen durch Senden von Adressenwerten zu einer Systemspeichereinrichtung; und
ein Mittel zum Vergleichen des ersten Adressenwerts mit den durch die Vorabrufstufe gesendeten Adressenwerten und zum Bewirken, daß die Vorabrufstufe den zweiten Adressenwert sendet, wenn der erste Adressenwert mit einem durch die Vorabrufstufe gesendeten aktuellen Adressenwert übereinstimmt.

17. Schaltung nach Anspruch 16, weiterhin mit einem Mittel zum gleichzeitigen Ausgeben der vorletzten Anweisung aus dem Anweisungspuffer an die Ausführungsstufe und zum Ausgeben der gewählten Schleifenanweisung an die Schleifenausführungsschaltung.

18. Verfahren zum Betreiben eines Prozessors, der einen Anweisungspuffer zum Speichern mehrerer Programmanweisungen, eine Ausführungsstufe zum Ausführen der sequentiell aus dem Anweisungspuffer ausgegebenen Programmanweisungen und einen Schleifenzählerspeicher zum Speichern eines Schleifenzählerwerts enthält, mit den folgenden Schritten:
Speichern einer gewählten Schleifenanweisung in einem ersten Speicherfeld,
Speichern einer Zielanweisung in einem zweiten Speicherfeld,
Ausführen der gewählten Schleifenanweisung, einschließlich des Aktualisierens des in dem Schleifenzählerspeicher gespeicherten Schleifenzählerwerts, und Erzeugen eines Schleife-genommen-Steuersignals, wenn der Schleifenzähler gleich einem vorbestimmten Wert ist; und
Ausgeben der Zielanweisung aus dem zweiten Speicherfeld an die Ausführungsstufe, wenn die Schleifenausführungsschaltung das Schleife-genommen-Steuersignal erzeugt.

19. Verfahren nach Anspruch 18, wobei der Prozessor weiterhin eine Vorabrufstufe zum Abrufen von Programmanweisungen durch Senden von Adressenwerten an eine Systemspeichereinrichtung enthält, und wobei das Verfahren weiterhin die folgenden Schritte umfaßt
Speichern eines ersten Adressenwerts und eines zweiten Adressenwerts, wobei der erste Adressenwert einer vorletzten Anweisung eines der gewählten Schleifenanweisung zugeordneten Schleifenhauptteils entspricht und der zweite Adressenwert einer Nachzielanweisung des Schleifenhauptteils entspricht; und
Vergleichen des ersten Adressenwerts mit den durch die Vorabrufstufe gesendeten Adressenwerten; und Bewirken, daß die Vorabrufstufe den zweiten Adressenwert sendet, wenn der erste Adressenwert mit einem durch die Vorabrufstufe gesendeten aktuellen Adressenwert übereinstimmt.

20. Verfahren nach Anspruch 19, bei dem weiterhin gleichzeitig die vorletzte Anweisung aus dem Anweisungspuffer an die Ausführungsstufe ausgegeben und die ausgewählte Schleifenanweisung an die Schleifenausführungsschaltung ausgegeben wird.

21. Schaltung in einem Prozessor, der eine Vorabrufstufe zum Erzeugen von Adressenwerten, mit denen Programmanweisungen aus einer Systemspeichereinrichtung abgerufen werden, und eine Ausführungsstufe zum Ausführen der abgerufenen Programmanweisungen enthält, zum iterativen Ausführen eines Schleifenhauptteils, der mehrere Anweisungen enthält, wobei die Schaltung folgendes umfaßt:
ein Mittel zum Speichern eines Trigger-Adressenwerts, der einer ersten Anweisung des Schleifenhauptteils zu-

geordnet ist, während einer ersten Iteration des Schleifenhauptteils; und
ein Mittel zum Erkennen der ersten Anweisung während einer nachfolgenden Iteration des Schleifenhauptteils durch Vergleichen der durch die Vorabrufstufe erzeugten Adressenwerte mit dem gespeicherten Trigger-Adressenwert.

22. Schaltung nach Anspruch 21,
wobei das Speichermittel ein Speicherfeld zum Speichern einer Nachzieladresse enthält, die einer zweiten Anweisung des Schleifenhauptteils zugeordnet ist, und
wobei das Erkennungsmittel weiterhin ein Mittel zum Senden der Nachzieladresse zu der Vorabrufstufe bei Erkennung der ersten Adresse umfaßt.

23. Verfahren zum Betreiben eines Prozessors, der eine Vorabrufstufe zum Erzeugen von Adressenwerten, mit denen Programmanweisungen aus einer Systemspeichereinrichtung abgerufen werden, und eine Ausführungsstufe zum Ausführen der abgerufenen Programmanweisungen enthält, mit einer Schaltung zum iterativen Ausführen eines Schleifenhauptteils, der mehrere Anweisungen enthält, mit den folgenden Schritten:
Speichern eines Trigger-Adressenwerts, der einer ersten Anweisung des Schleifenhauptteils zugeordnet ist, während einer ersten Iteration des Schleifenhauptteils; und
Erkennen der ersten Anweisung während einer nachfolgenden Iteration des Schleifenhauptteils durch Vergleichen der durch die Vorabrufstufe erzeugten Adressenwerte mit dem gespeicherten Trigger-Adressenwert.

24. Verfahren nach Anspruch 23, weiterhin mit den folgenden Schritten:
Speichern, während der ersten Iteration, einer Nachzieladresse, die einer zweiten Anweisung des Schleifenhauptteils zugeordnet ist, und
bei Erkennung der ersten Adresse, Einleiten des spekulativen Vorabrufens der zweiten Anweisung durch Senden der Nachzieladresse zu der Vorabrufstufe.

25. Prozessor, umfassend:
eine Vorabrufstufe zum Abrufen mehrerer Programmanweisungen aus einer Systemspeichereinrichtung;
einen Anweisungspuffer mit mehreren Registern zum vorübergehenden Speichern der abgerufenen Programmanweisungen;
eine Ausführungsstufe zum Ausführen von Programmanweisungen, die systematisch aus dem Anweisungspuffer ausgegeben werden;
einen Schleifen-Cache-Puffer zum Speichern gewählter Schleifenanweisungen der mehreren Programmanweisungen; und
eine Schleifenausführungsschaltung zur Ausführung der in dem Schleifen-Cache-Puffer gespeicherten gewählten Schleifenanweisungen.

26. Prozessor nach Anspruch 25,
wobei der Schleifen-Cache-Puffer ein erstes Speicherfeld zum Speichern einer Zielanweisung enthält; und
wobei der Prozessor weiterhin ein Mittel zum Ausgeben der Schleifenanweisung aus dem Schleifen-Cache-Puffer an die Schleifenausführungsschaltung, wenn eine vorbestimmte Anweisung aus dem Anweisungsregister an die Ausführungsstufe ausgegeben wird, und zum nachfolgenden Ausgeben der Zielanweisung aus dem ersten Speicherfeld des Schleifen-Cache-Puffers an die Ausführungsstufe umfaßt.

27. Prozessor nach Anspruch 25,
wobei der Schleifen-Cache-Puffer weiterhin ein zweites Speicherfeld zum Speichern eines ersten Zeigerwerts, der eine erste Registerspeicherstelle des Anweisungspuffers angibt, und ein drittes Speicherfeld zum Speichern eines zweiten Zeigerwerts, der eine zweite Registerspeicherstelle des Anweisungspuffers angibt, umfaßt,
wobei das Ausgabemittel weiterhin folgendes umfaßt
ein Mittel zum Erkennen, wenn eine Ausgabezeigerposition des Anweisungspuffers gleich der durch den ersten Zeigerwert angegebenen ersten Registerspeicherstelle ist; und
ein Mittel zum Wechseln der Ausgabezeigerposition auf die durch den zweiten Ausgabezeiger angegebene zweite Registerposition, nachdem die Zielanweisung aus dem Schleifen-Cache-Puffer ausgegeben wurde.

28. Prozessor nach Anspruch 25, wobei die mehreren Programmanweisungen eine Schleifenanweisung, eine Zielanweisung und eine zweite Vielzahl von Nachzielanweisungen, die sequentiell zwischen der Schleifenanweisung und der Zielanweisung angeordnet sind, umfassen, und wobei der Prozessor ein Mittel zum Sperren der Vorabrufstufe, wenn die zweite Vielzahl von Nachzielanweisungen kleiner oder gleich der Vielzahl von Registern des Anweisungspuffers ist, umfaßt.

29. Prozessor, umfassend:
eine Vorabrufstufe zum Abrufen mehrerer Programmanweisungen aus einer Systemspeichereinrichtung;
eine Ausführungsstufe zum Ausführen mindestens eines Teils der abgerufenen Programmanweisungen;
eine Schleifenausführungsschaltung;
einen ersten Schleifen-Cache-Puffer mit einem ersten Schleifenanweisungsfeld;
einen zweiten Schleifen-Cache-Puffer mit einem zweiten Schleifenanweisungsfeld;
ein Mittel zum Zuweisen einer ersten Schleifenanweisung zu dem ersten Schleifenanweisungsfeld des ersten Schleifen-Cache-Puffers und zum Zuweisen einer zweiten Schleifenanweisung zu dem zweiten Schleifenanweisungsfeld des zweiten Schleifen-Cache-Puffers; und
ein Mittel zum selektiven Ausgeben der ersten und der zweiten Schleifenanweisung aus dem ersten bzw. zweiten Schleifen-Cache-Puffer an die Schleifenausführungsschaltung.

30. Prozessor nach Anspruch 29,
wobei der erste Schleifen-Cache-Puffer weiterhin ein Nachzieladressenfeld umfaßt und
wobei der Prozessor weiterhin folgendes umfaßt:
ein Mittel zum Einleiten eines spekulativen Vorabrufprozesses, einschließlich des Sendens der Nachzieladresse zu der Vorabrufstufe, und
ein Mittel zum Verhindern der Zuweisung des ersten Schleifen-Cache-Puffers zu einer dritten Schleifenanweisung zwischen der Einleitung des spekulativen Vorabrufprozesses, bis zur Ausführung der ersten Schleifenanweisung durch die Schleifenausführungsschaltung.

Es folgen 13 Blatt Zeichnungen

Anhängende Zeichnungen

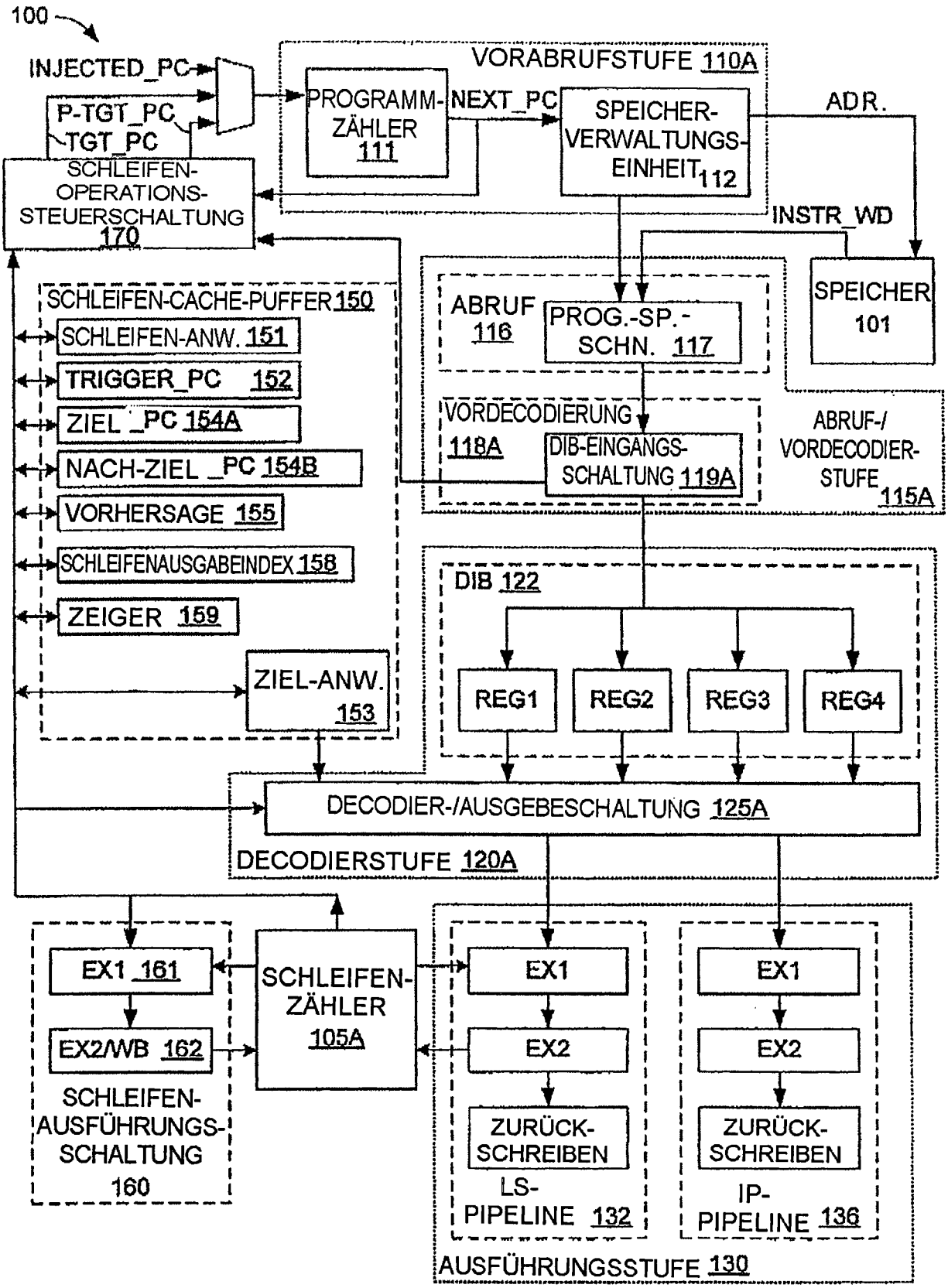


FIG. 1

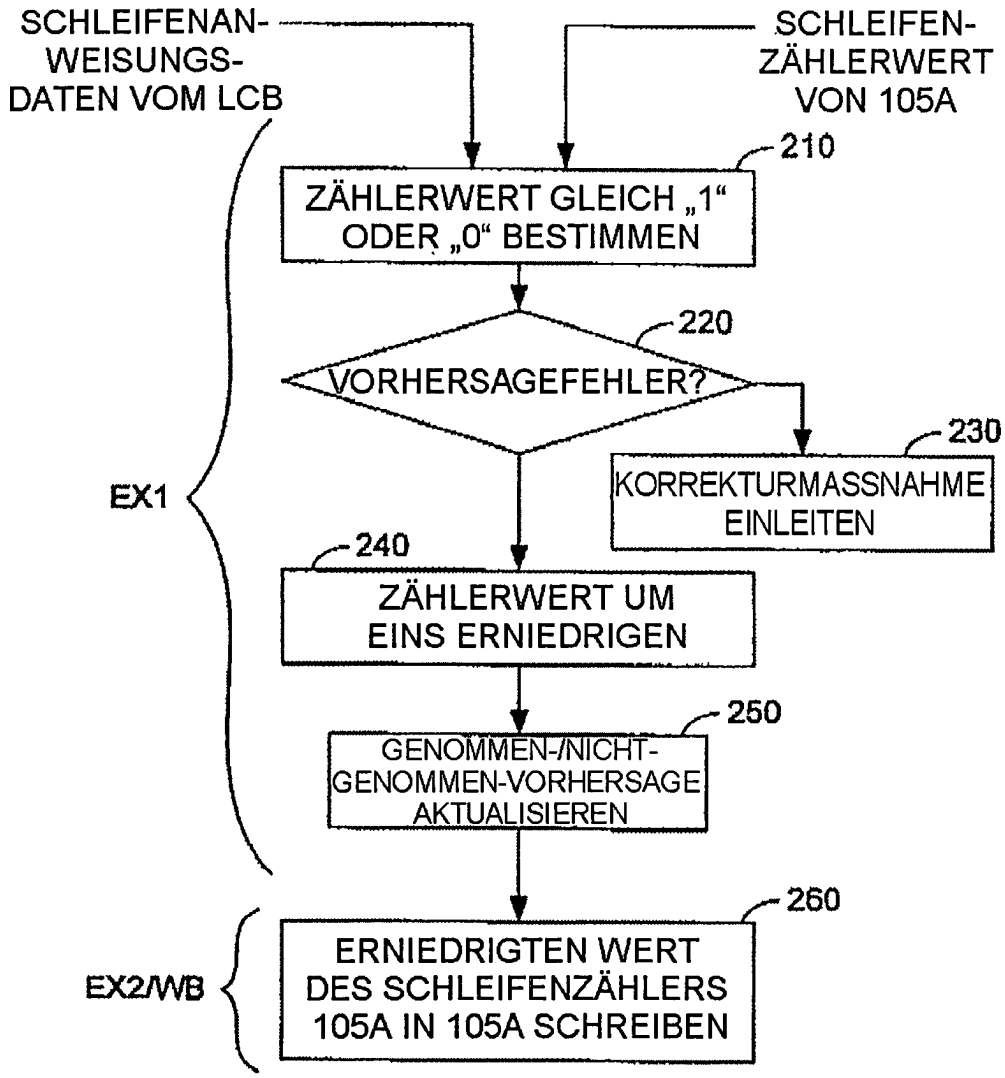

SCHLEIFENAN-
WEISUNGS-
DATEN VOM LCB
SCHLEIFEN-
ZÄHLERWERT
VON 105A
210
ZÄHLERWERT GLEICH „1“
ODER „0“ BESTIMMEN
220
VORHERSAGEFEHLER?
230
KORREKTURMASSNAHME
EINLEITEN
EX1
240
ZÄHLERWERT UM
EINS ERNIEDRIGEN
250
GENOMMEN-/NICHT-
GENOMMEN-VORHERSAGE
AKTUALISIEREN
260
EX2/WB
ERNIEDRIGTEN WERT
DES SCHLEIFENZÄHLERS
105A IN 105A SCHREIBEN

FIG. 2

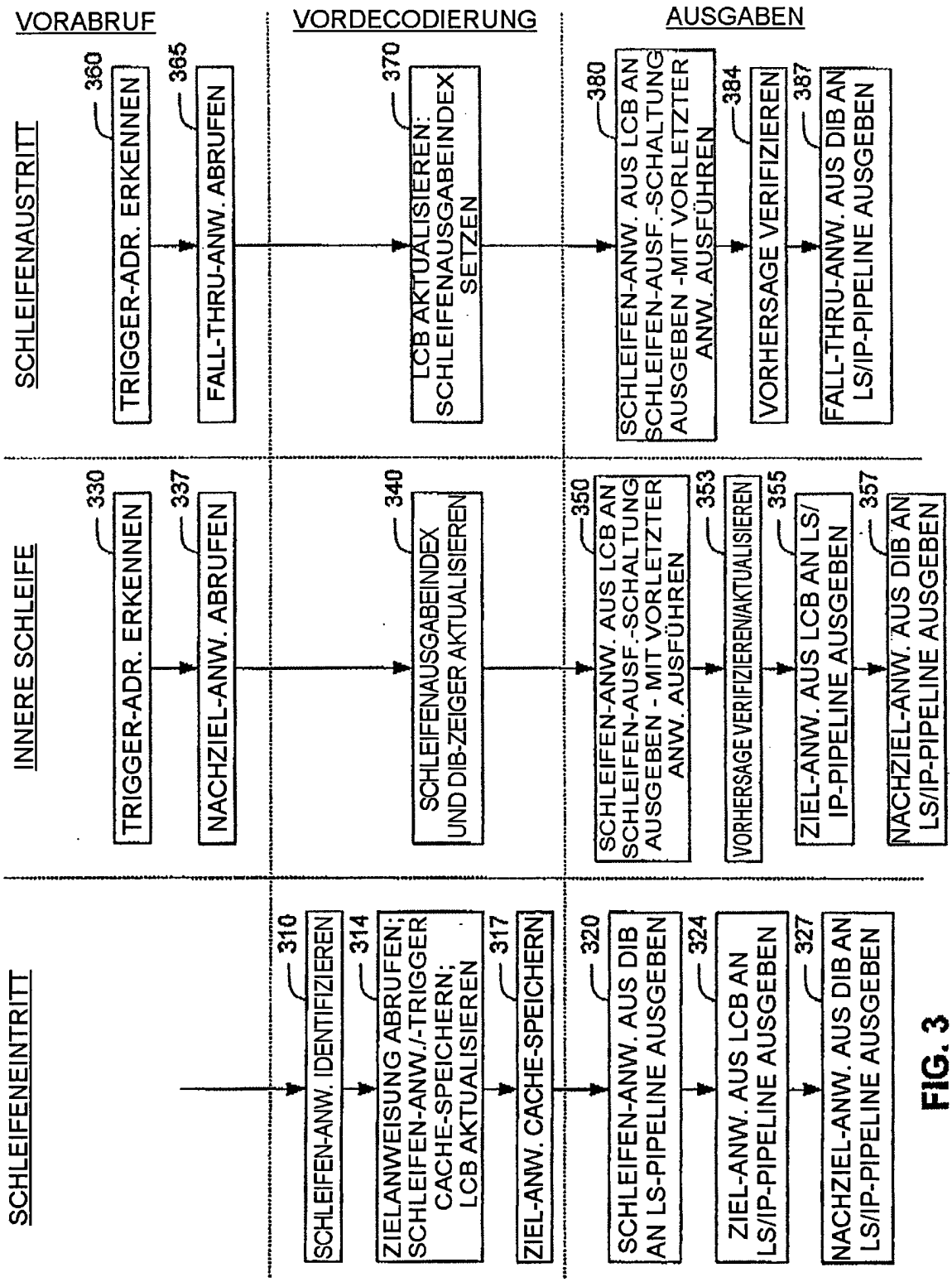

VORABRUF
VORDECODIERUNG
AUSGABEN
SCHLEIFENAUSTRITT
INNERE SCHLEIFE
SCHLEIFENEINTRITT
360
TRIGGER-ADR. ERKENNEN
365
FALL-THRU-ANW. ABRUFEN
370
LCB AKTUALISIEREN: SCHLEIFENAUSGABEINDEX SETZEN
380
SCHLEIFEN-ANW. AUS LCB AN SCHLEIFEN-AUSF.-SCHALTUNG AUSGEBEN -MIT VORLETZTER ANW. AUSFÜHREN
384
VORHERSAGE VERIFIZIEREN
387
FALL-THRU-ANW. AUS DIB AN LS/IP-PIPELINE AUSGEBEN
330
TRIGGER-ADR. ERKENNEN
337
NACHZIEL-ANW. ABRUFEN
340
SCHLEIFENAUSGABEINDEX UND DIB-ZEIGER AKTUALISIEREN
350
SCHLEIFEN-ANW. AUS LCB AN SCHLEIFEN-AUSF.-SCHALTUNG AUSGEBEN - MIT VORLETZTER ANW. AUSFÜHREN
353
VORHERSAGE VERIFIZIEREN/AKTUALISIEREN
355
ZIEL-ANW. AUS LCB AN LS/IP-PIPELINE AUSGEBEN
357
NACHZIEL-ANW. AUS DIB AN LS/IP-PIPELINE AUSGEBEN
310
SCHLEIFEN-ANW. IDENTIFIZIEREN
314
ZIELANWEISUNG ABRUFEN; SCHLEIFEN-ANW./-TRIGGER CACHE-SPEICHERN; LCB AKTUALISIEREN
317
ZIEL-ANW. CACHE-SPEICHERN
320
SCHLEIFEN-ANW. AUS DIB AN LS-PIPELINE AUSGEBEN
324
ZIEL-ANW. AUS LCB AN LS/IP-PIPELINE AUSGEBEN
327
NACHZIEL-ANW. AUS DIB AN LS/IP-PIPELINE AUSGEBEN

FIG. 3

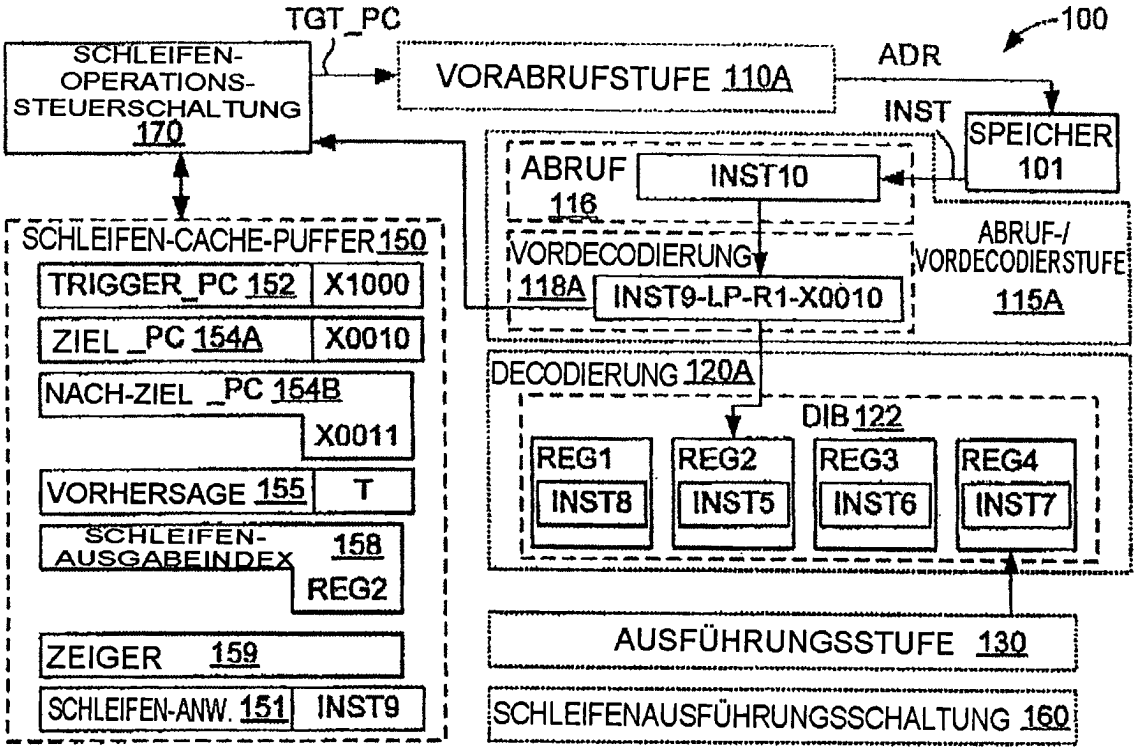

TGT_PC
100
SCHLEIFEN-OPERATIONS-STEUERSCHALTUNG 170
VORABRUFSTUFE 110A
ADR
INST
SPEICHER 101
ABRUF 116
INST10
SCHLEIFEN-CACHE-PUFFER 150
TRIGGER_PC 152
X1000
ZIEL _PC 154A
X0010
NACH-ZIEL _PC 154B
X0011
VORHERSAGE 155
T
SCHLEIFEN-AUSGABEINDEX 158
REG2
ZEIGER 159
SCHLEIFEN-ANW. 151
INST9
VORDECODIERUNG 118A
INST9-LP-R1-X0010
ABRUF-/VORDECODIERSTUFE 115A
DECODIERUNG 120A
DIB 122
REG1
INST8
REG2
INST5
REG3
INST6
REG4
INST7
AUSFÜHRUNGSSTUFE 130
SCHLEIFENAUSFÜHRUNGSSCHALTUNG 160

FIG. 4(A)

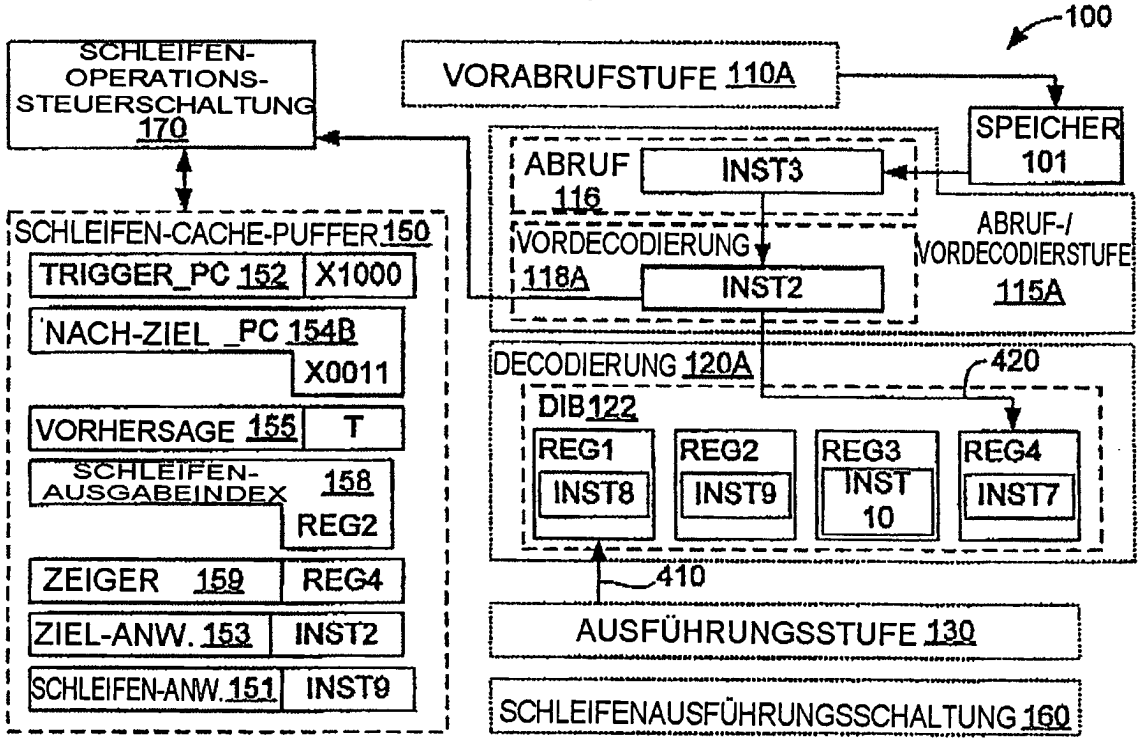

100
SCHLEIFEN-OPERATIONS-STEUERSCHALTUNG 170
VORABRUFSTUFE 110A
SPEICHER 101
ABRUF 116
INST3
SCHLEIFEN-CACHE-PUFFER 150
TRIGGER_PC 152
X1000
NACH-ZIEL _PC 154B
X0011
VORHERSAGE 155
T
SCHLEIFEN-AUSGABEINDEX 158
REG2
ZEIGER 159
REG4
ZIEL-ANW. 153
INST2
SCHLEIFEN-ANW. 151
INST9
VORDECODIERUNG 118A
INST2
ABRUF-/VORDECODIERSTUFE 115A
DECODIERUNG 120A
420
DIB 122
REG1
INST8
REG2
INST9
REG3
INST 10
REG4
INST7
410
AUSFÜHRUNGSSTUFE 130
SCHLEIFENAUSFÜHRUNGSSCHALTUNG 160

FIG. 4(B)

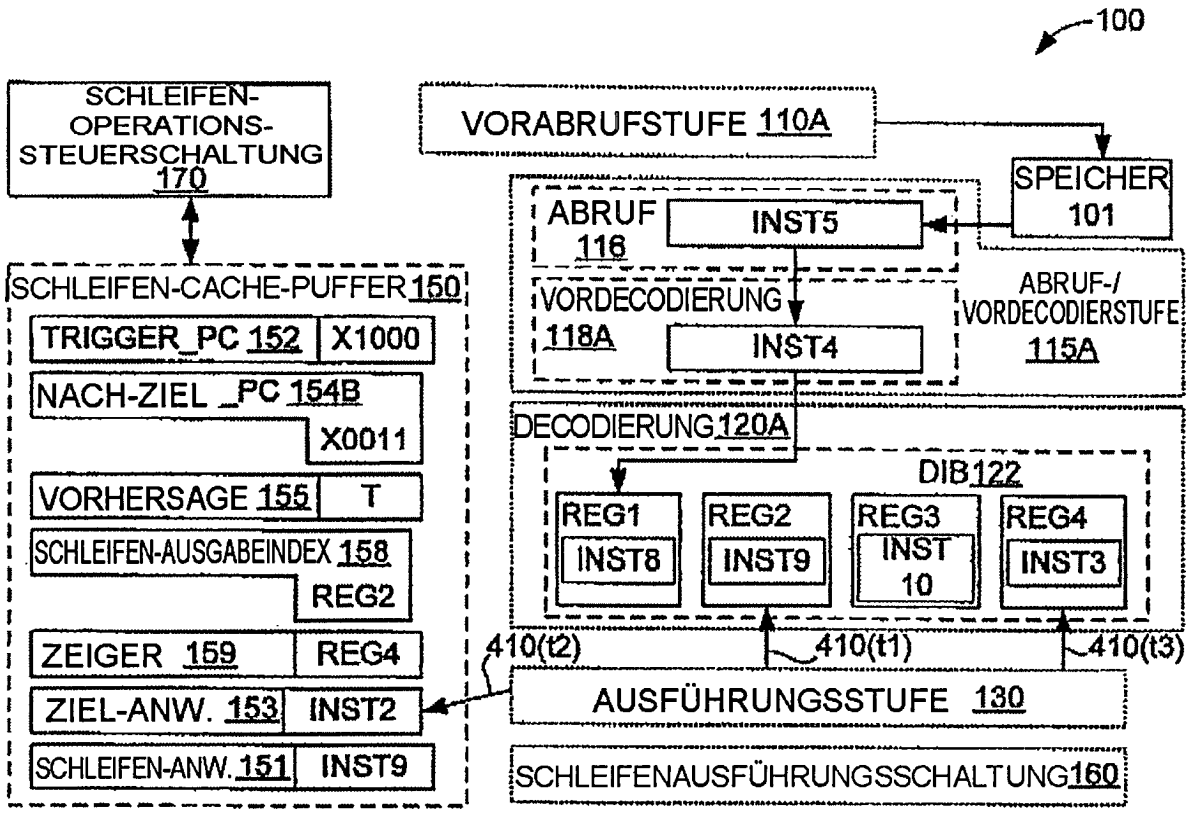

100
SCHLEIFEN-OPERATIONS-STEUERSCHALTUNG 170
VORABRUFSTUFE 110A
SPEICHER 101
ABRUF 116
INST5
VORDECODIERUNG 118A
INST4
ABRUF-/VORDECODIERSTUFE 115A
DECODIERUNG 120A
DIB 122
REG1
INST8
REG2
INST9
REG3
INST 10
REG4
INST3
SCHLEIFEN-CACHE-PUFFER 150
TRIGGER_PC 152
X1000
NACH-ZIEL _PC 154B
X0011
VORHERSAGE 155
T
SCHLEIFEN-AUSGABEINDEX 158
REG2
ZEIGER 159
REG4
ZIEL-ANW. 153
INST2
SCHLEIFEN-ANW. 151
INST9
410(t2)
410(t1)
410(t3)
AUSFÜHRUNGSSTUFE 130
SCHLEIFENAUSFÜHRUNGSSCHALTUNG 160

FIG. 4(C)

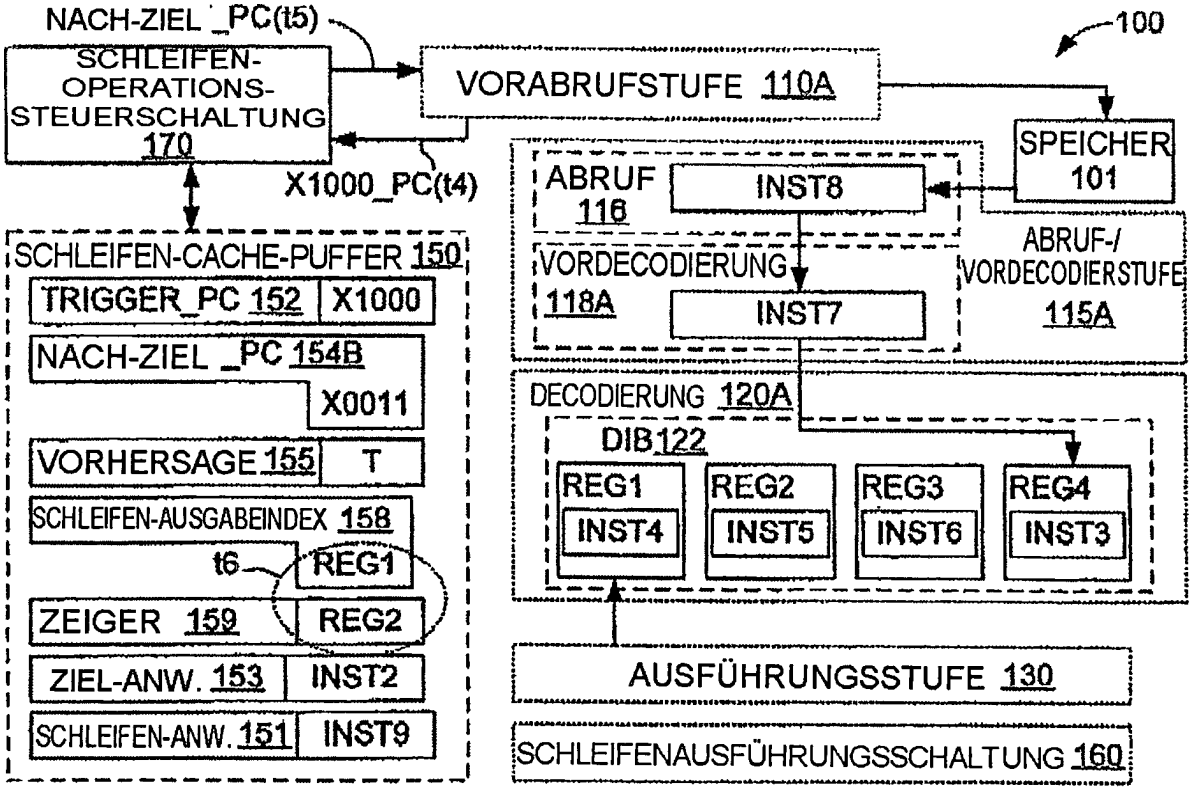

NACH-ZIEL _PC(t5)
SCHLEIFEN-OPERATIONS-STEUERSCHALTUNG 170
X1000_PC(t4)
100
VORABRUFSTUFE 110A
SPEICHER 101
ABRUF 116
INST8
VORDECODIERUNG 118A
INST7
ABRUF-/VORDECODIERSTUFE 115A
DECODIERUNG 120A
DIB 122
REG1
INST4
REG2
INST5
REG3
INST6
REG4
INST3
SCHLEIFEN-CACHE-PUFFER 150
TRIGGER_PC 152
X1000
NACH-ZIEL _PC 154B
X0011
VORHERSAGE 155
T
SCHLEIFEN-AUSGABEINDEX 158
t6
REG1
ZEIGER 159
REG2
ZIEL-ANW. 153
INST2
SCHLEIFEN-ANW. 151
INST9
AUSFÜHRUNGSSTUFE 130
SCHLEIFENAUSFÜHRUNGSSCHALTUNG 160

FIG. 5(A)

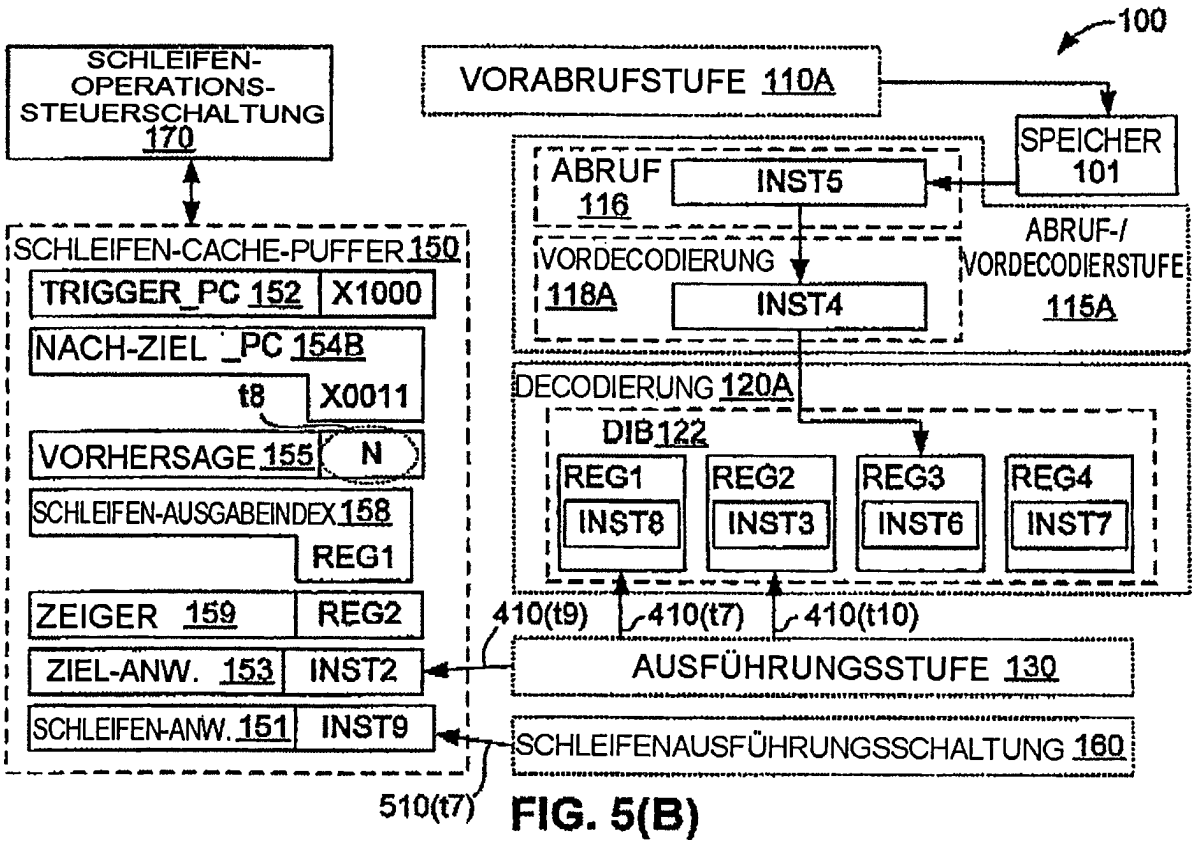

SCHLEIFEN-OPERATIONS-STEUERSCHALTUNG 170
100
VORABRUFSTUFE 110A
SPEICHER 101
ABRUF 116
INST5
VORDECODIERUNG 118A
INST4
ABRUF-/VORDECODIERSTUFE 115A
DECODIERUNG 120A
DIB 122
REG1
INST8
REG2
INST3
REG3
INST6
REG4
INST7
SCHLEIFEN-CACHE-PUFFER 150
TRIGGER_PC 152
X1000
NACH-ZIEL _PC 154B
t8
X0011
VORHERSAGE 155
N
SCHLEIFEN-AUSGABEINDEX 158
REG1
ZEIGER 159
REG2
ZIEL-ANW. 153
INST2
SCHLEIFEN-ANW. 151
INST9
410(t9)
410(t7)
410(t10)
AUSFÜHRUNGSSTUFE 130
SCHLEIFENAUSFÜHRUNGSSCHALTUNG 160
510(t7)

FIG. 5(B)

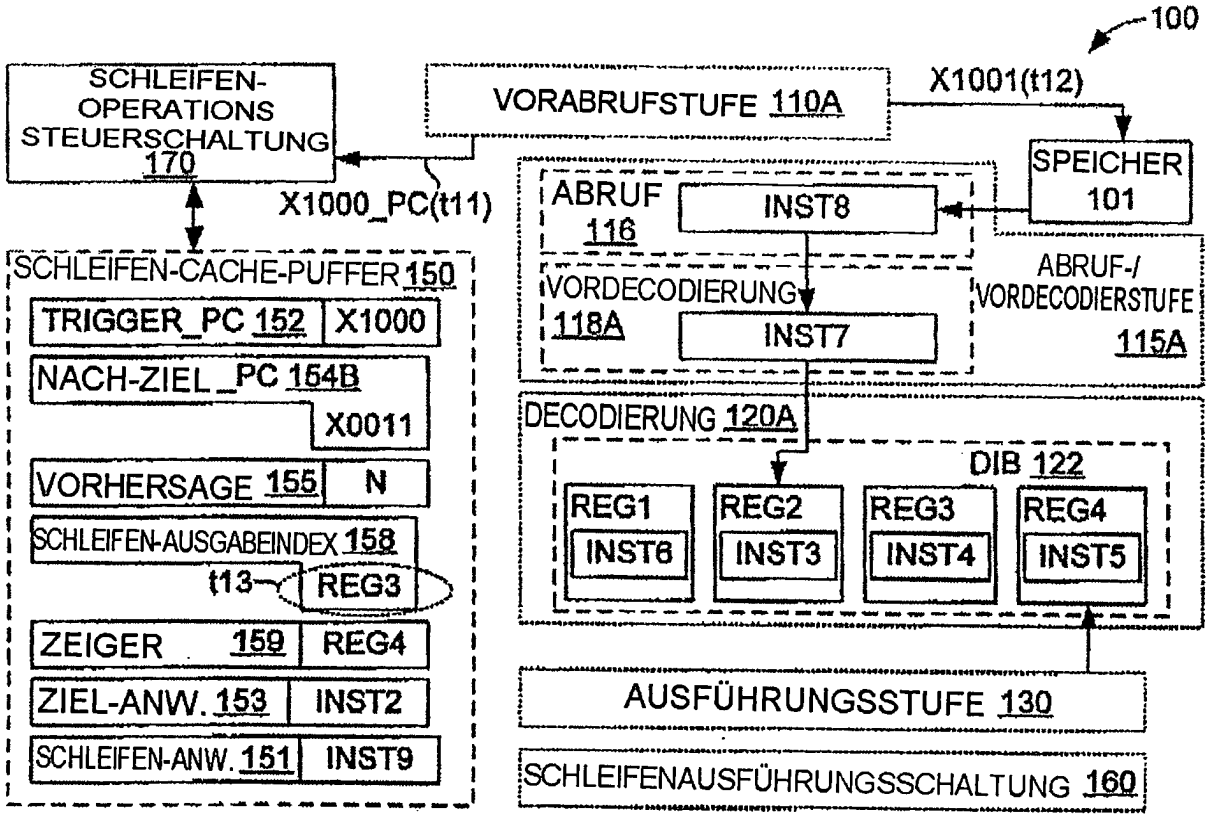

100
SCHLEIFEN-OPERATIONS STEUERSCHALTUNG 170
VORABRUFSTUFE 110A
X1001(t12)
SPEICHER 101
X1000_PC(t11)
ABRUF 116
INST8
VORDECODIERUNG 118A
INST7
ABRUF-/VORDECODIERSTUFE 115A
SCHLEIFEN-CACHE-PUFFER 150
TRIGGER_PC 152
X1000
NACH-ZIEL_PC 154B
X0011
VORHERSAGE 155
N
SCHLEIFEN-AUSGABEINDEX 158
t13
REG3
ZEIGER 159
REG4
ZIEL-ANW. 153
INST2
SCHLEIFEN-ANW. 151
INST9
DECODIERUNG 120A
DIB 122
REG1
INST6
REG2
INST3
REG3
INST4
REG4
INST5
AUSFÜHRUNGSSTUFE 130
SCHLEIFENAUSFÜHRUNGSSCHALTUNG 160

FIG. 6(A)

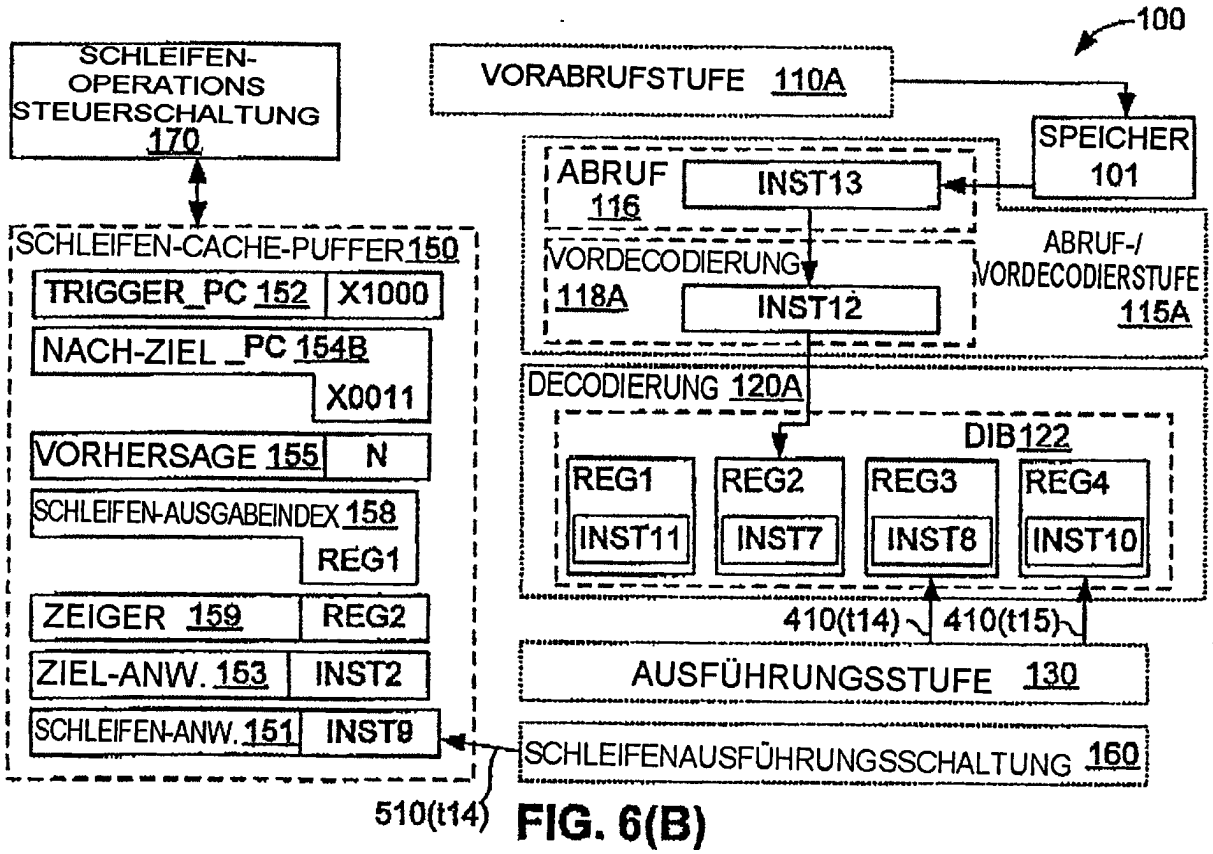

100
SCHLEIFEN-OPERATIONS STEUERSCHALTUNG 170
VORABRUFSTUFE 110A
SPEICHER 101
ABRUF 116
INST13
VORDECODIERUNG 118A
INST12
ABRUF-/VORDECODIERSTUFE 115A
SCHLEIFEN-CACHE-PUFFER 150
TRIGGER_PC 152
X1000
NACH-ZIEL_PC 154B
X0011
VORHERSAGE 155
N
SCHLEIFEN-AUSGABEINDEX 158
REG1
ZEIGER 159
REG2
ZIEL-ANW. 153
INST2
SCHLEIFEN-ANW. 151
INST9
DECODIERUNG 120A
DIB 122
REG1
INST11
REG2
INST7
REG3
INST8
REG4
INST10
410(t14)
410(t15)
AUSFÜHRUNGSSTUFE 130
SCHLEIFENAUSFÜHRUNGSSCHALTUNG 160
510(t14)

FIG. 6(B)

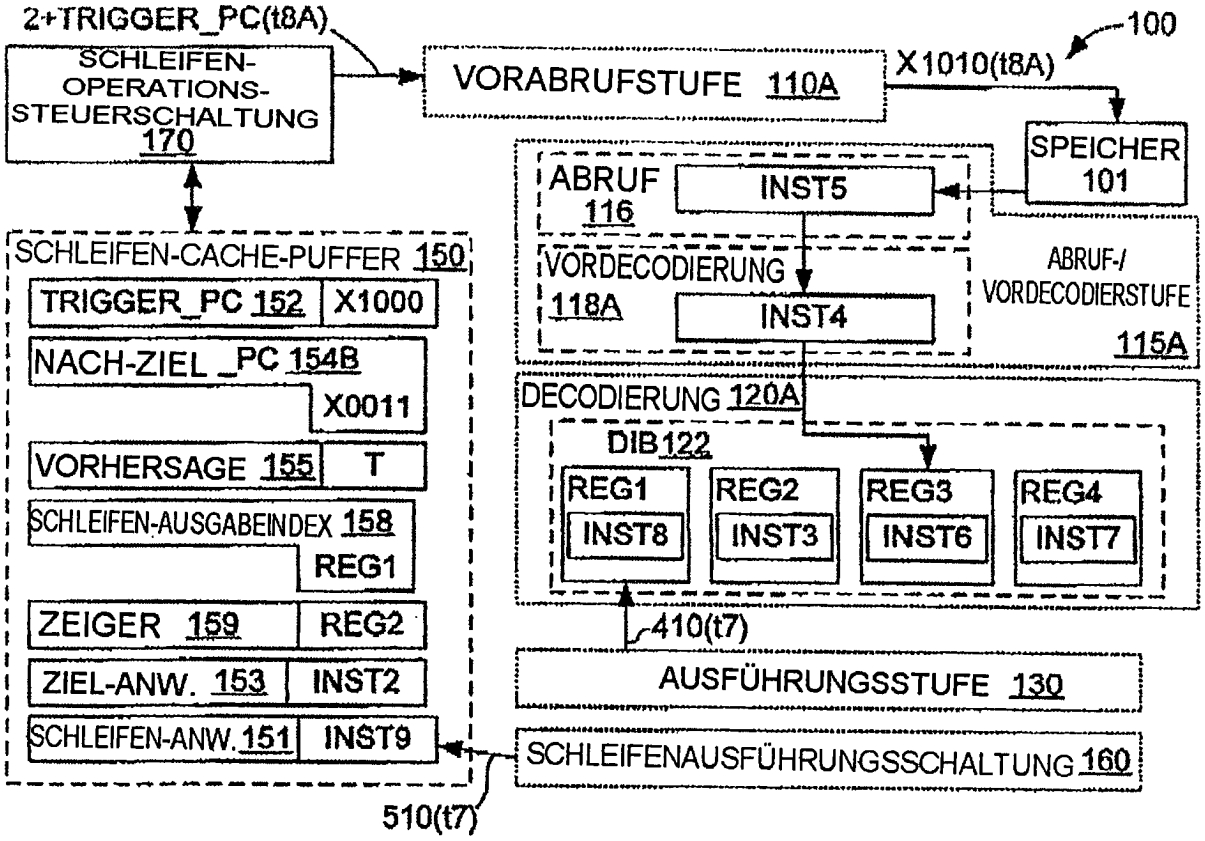

2+TRIGGER_PC(t8A)
SCHLEIFEN-OPERATIONS-STEUERSCHALTUNG 170
VORABRUFSTUFE 110A
X1010(t8A)
100
SPEICHER 101
ABRUF 116
INST5
VORDECODIERUNG 118A
INST4
ABRUF-/VORDECODIERSTUFE 115A
DECODIERUNG 120A
DIB 122
REG1 INST8
REG2 INST3
REG3 INST6
REG4 INST7
410(t7)
AUSFÜHRUNGSSTUFE 130
SCHLEIFENAUSFÜHRUNGSSCHALTUNG 160
SCHLEIFEN-CACHE-PUFFER 150
TRIGGER_PC 152 X1000
NACH-ZIEL _PC 154B X0011
VORHERSAGE 155 T
SCHLEIFEN-AUSGABEINDEX 158 REG1
ZEIGER 159 REG2
ZIEL-ANW. 153 INST2
SCHLEIFEN-ANW. 151 INST9
510(t7)

FIG. 7(A)

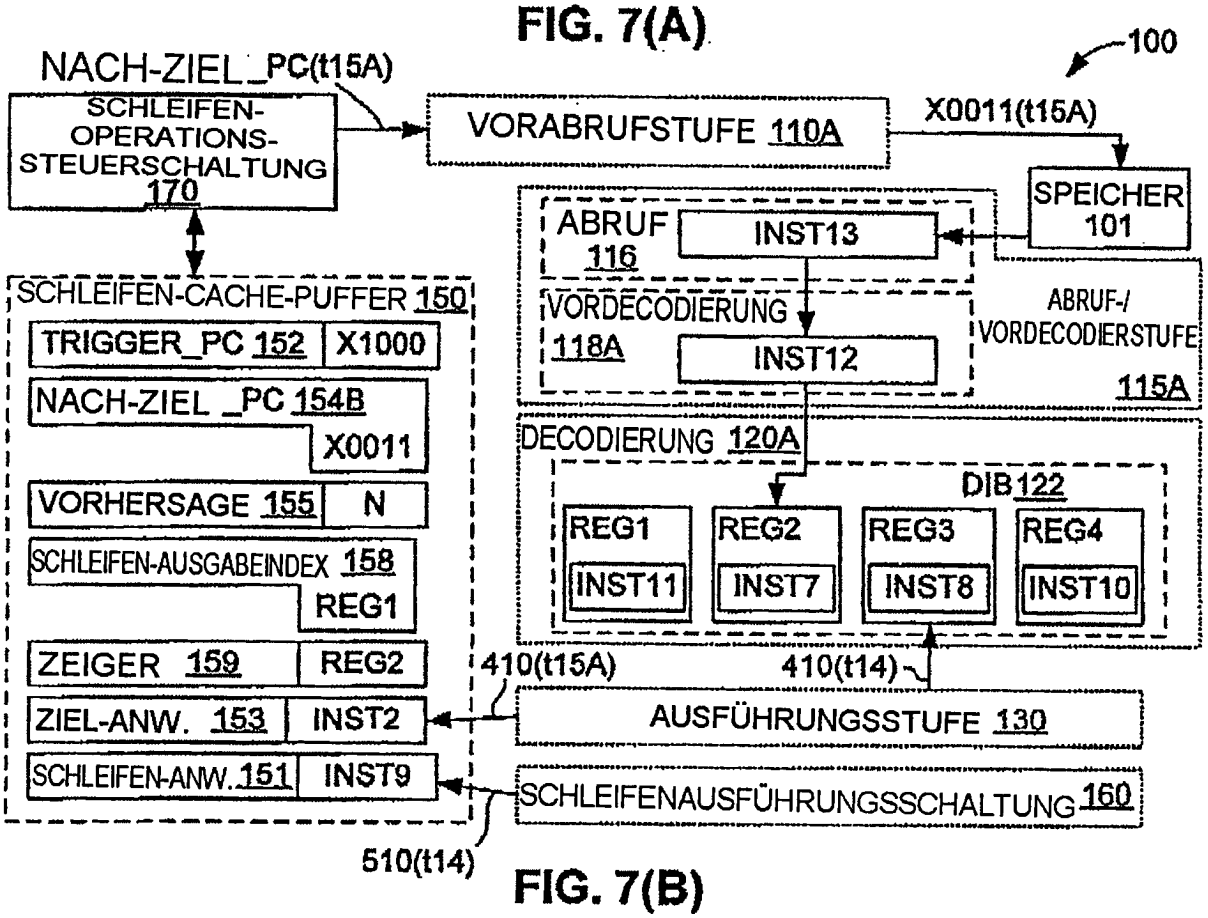

NACH-ZIEL_PC(t15A)
SCHLEIFEN-OPERATIONS-STEUERSCHALTUNG 170
VORABRUFSTUFE 110A
X0011(t15A)
100
SPEICHER 101
ABRUF 116
INST13
VORDECODIERUNG 118A
INST12
ABRUF-/VORDECODIERSTUFE 115A
DECODIERUNG 120A
DIB 122
REG1 INST11
REG2 INST7
REG3 INST8
REG4 INST10
410(t15A)
410(t14)
AUSFÜHRUNGSSTUFE 130
SCHLEIFENAUSFÜHRUNGSSCHALTUNG 160
SCHLEIFEN-CACHE-PUFFER 150
TRIGGER_PC 152 X1000
NACH-ZIEL _PC 154B X0011
VORHERSAGE 155 N
SCHLEIFEN-AUSGABEINDEX 158 REG1
ZEIGER 159 REG2
ZIEL-ANW. 153 INST2
SCHLEIFEN-ANW. 151 INST9
510(t14)

FIG. 7(B)

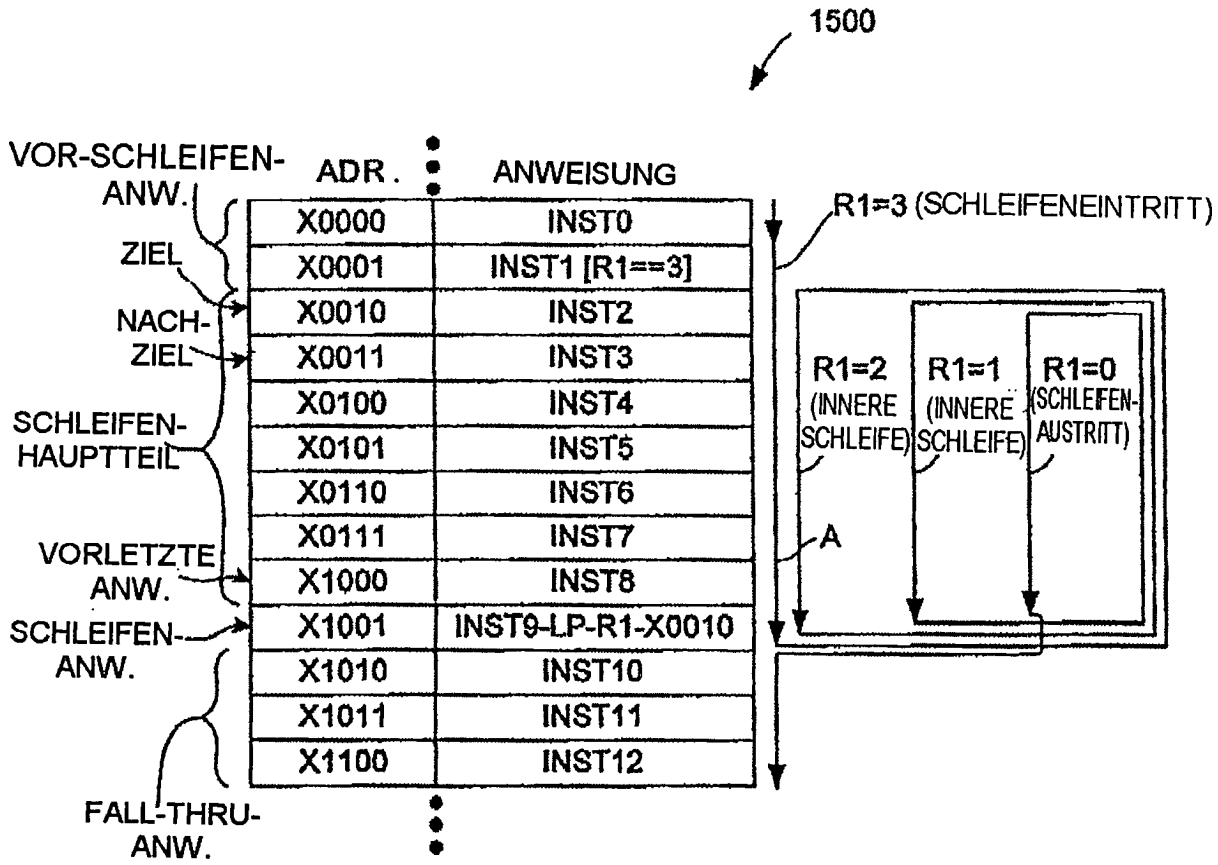

1500
VOR-SCHLEIFEN-ANW.
ZIEL
NACH-ZIEL
SCHLEIFEN-HAUPTTEIL
VORLETZTE ANW.
SCHLEIFEN-ANW.
FALL-THRU-ANW.
ADR.
ANWEISUNG
X0000 INST0
X0001 INST1 [R1==3]
X0010 INST2
X0011 INST3
X0100 INST4
X0101 INST5
X0110 INST6
X0111 INST7
X1000 INST8
X1001 INST9-LP-R1-X0010
X1010 INST10
X1011 INST11
X1100 INST12
R1=3 (SCHLEIFENEINTRITT)
R1=2 (INNERE SCHLEIFE)
R1=1 (INNERE SCHLEIFE)
R1=0 (SCHLEIFEN-AUSTRITT)
A

FIG. 15

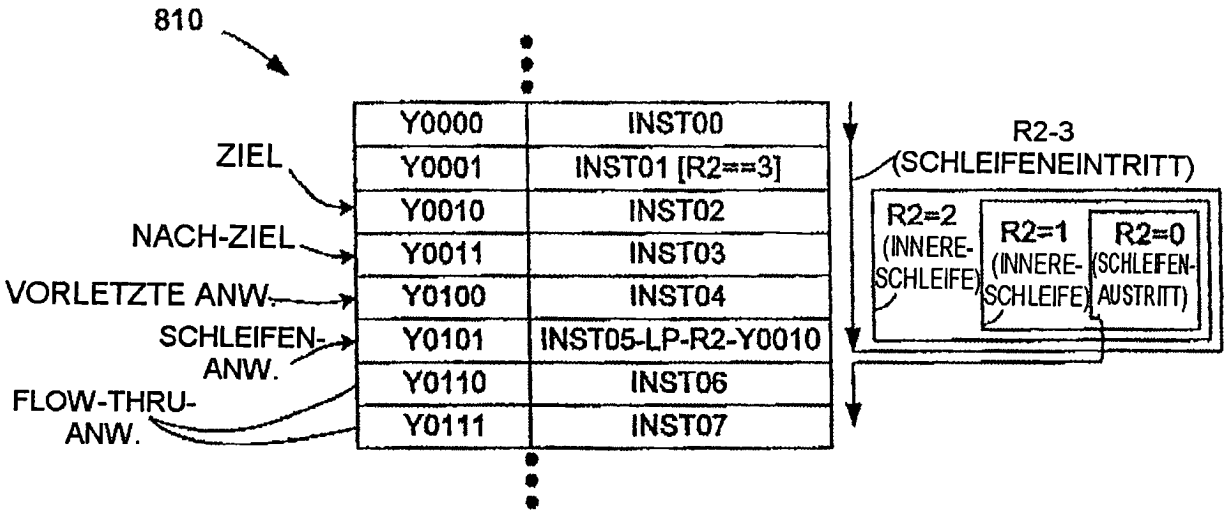

810
ZIEL
NACH-ZIEL
VORLETZTE ANW.
SCHLEIFEN-ANW.
FLOW-THRU-ANW.
Y0000 INST00
Y0001 INST01 [R2==3]
Y0010 INST02
Y0011 INST03
Y0100 INST04
Y0101 INST05-LP-R2-Y0010
Y0110 INST06
Y0111 INST07
R2-3 (SCHLEIFENEINTRITT)
R2=2 (INNERE-SCHLEIFE)
R2=1 (INNERE-SCHLEIFE)
R2=0 (SCHLEIFEN-AUSTRITT)

FIG. 8

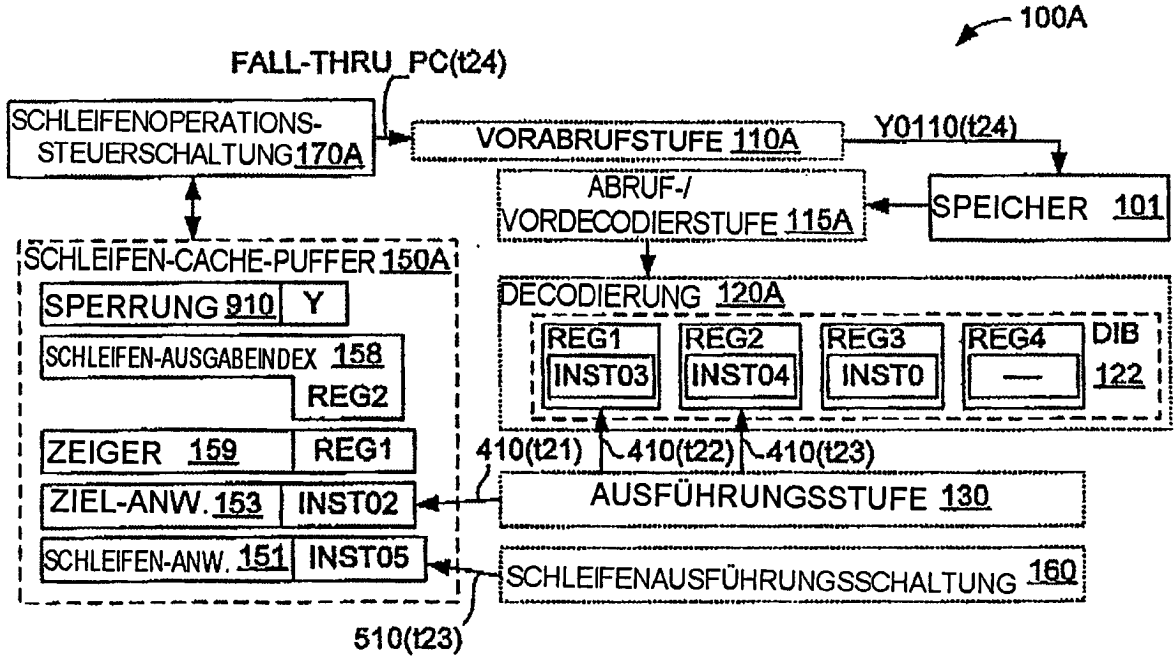

100A
FALL-THRU_PC(t24)
SCHLEIFENOPERATIONS-STEUERSCHALTUNG 170A
VORABRUFSTUFE 110A
Y0110(t24)
ABRUF-/ VORDECODIERSTUFE 115A
SPEICHER 101
SCHLEIFEN-CACHE-PUFFER 150A
SPERRUNG 910 Y
SCHLEIFEN-AUSGABEINDEX 158 REG2
ZEIGER 159 REG1
ZIEL-ANW. 153 INST02
SCHLEIFEN-ANW. 151 INST05
DECODIERUNG 120A
REG1 INST03
REG2 INST04
REG3 INST0
REG4 —
DIB 122
410(t21)
410(t22)
410(t23)
AUSFÜHRUNGSSTUFE 130
SCHLEIFENAUSFÜHRUNGSSCHALTUNG 160
510(t23)

FIG. 9

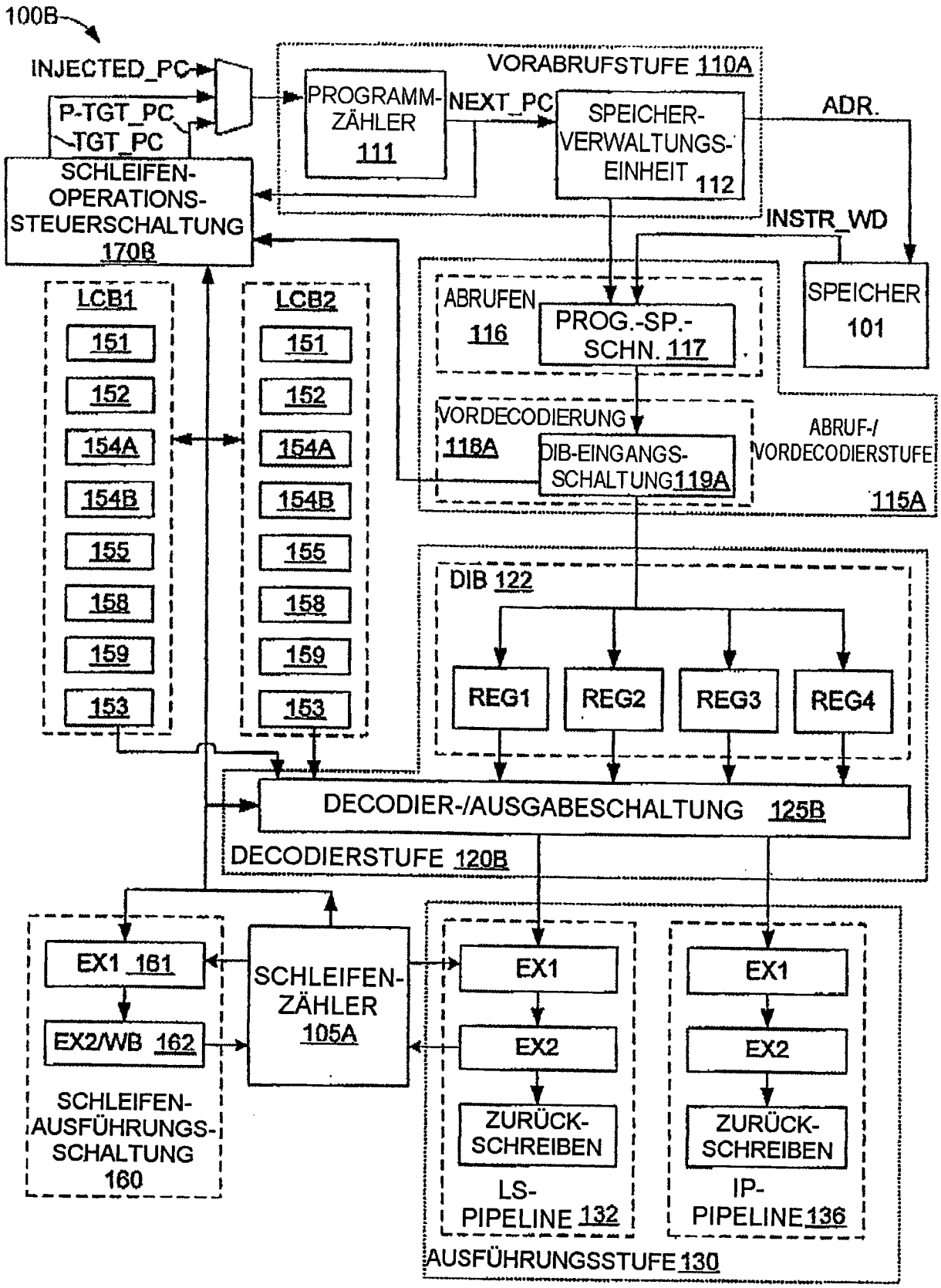

100B
INJECTED_PC
P-TGT_PC
TGT_PC
SCHLEIFEN-OPERATIONS-STEUERSCHALTUNG 170B
VORABRUFSTUFE 110A
PROGRAMM-ZÄHLER 111
NEXT_PC
SPEICHER-VERWALTUNGS-EINHEIT 112
ADR.
INSTR_WD
SPEICHER 101
LCB1
LCB2
151
152
154A
154B
155
158
159
153
ABRUFEN 116
PROG.-SP.-SCHN. 117
VORDECODIERUNG 118A
DIB-EINGANGS-SCHALTUNG 119A
ABRUF-/VORDECODIERSTUFE 115A
DIB 122
REG1
REG2
REG3
REG4
DECODIER-/AUSGABESCHALTUNG 125B
DECODIERSTUFE 120B
EX1 161
EX2/WB 162
SCHLEIFEN-AUSFÜHRUNGS-SCHALTUNG 160
SCHLEIFEN-ZÄHLER 105A
EX1
EX2
ZURÜCK-SCHREIBEN
LS-PIPELINE 132
IP-PIPELINE 136
AUSFÜHRUNGSSTUFE 130

FIG. 10

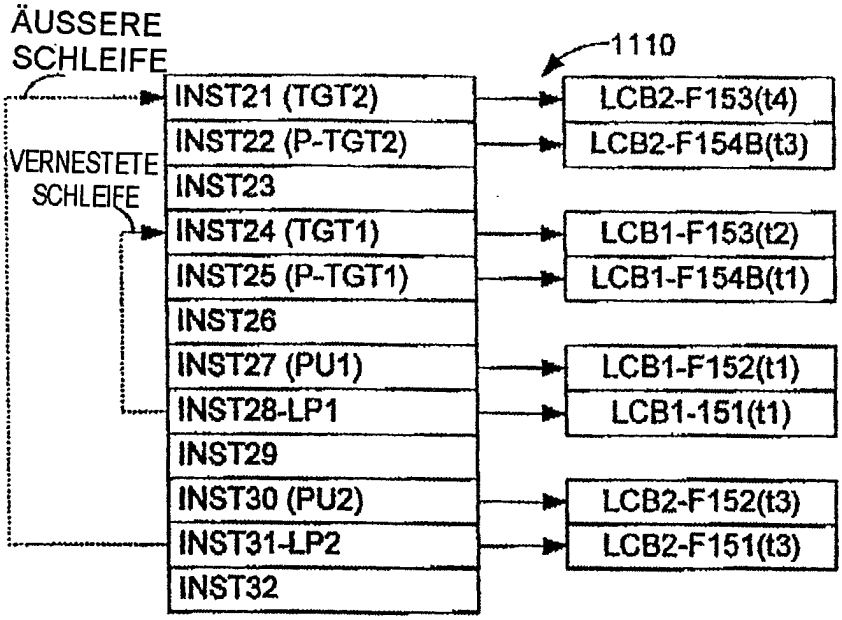


FIG. 11

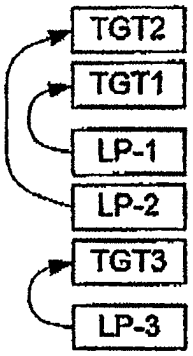


FIG. 12(A)

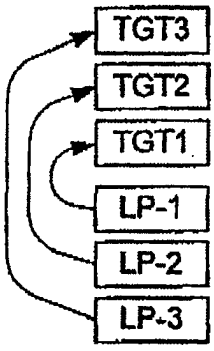


FIG. 12(B)

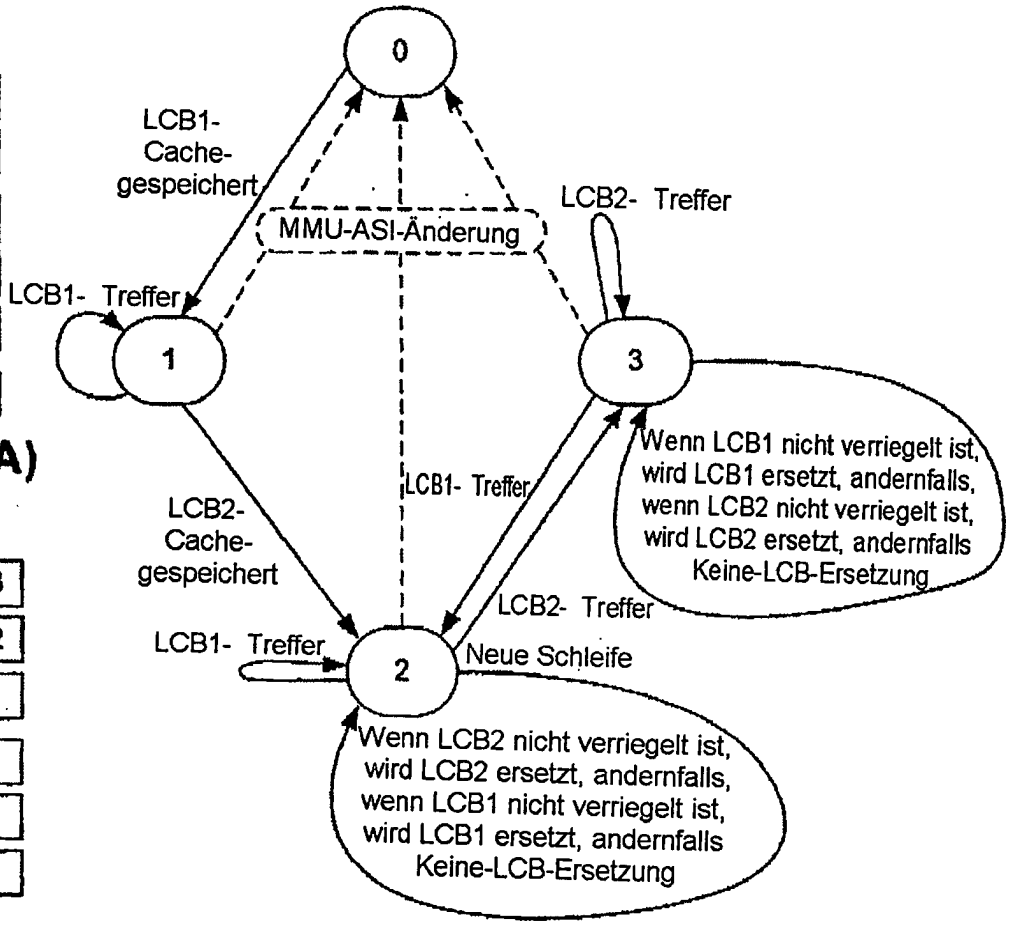


FIG. 13

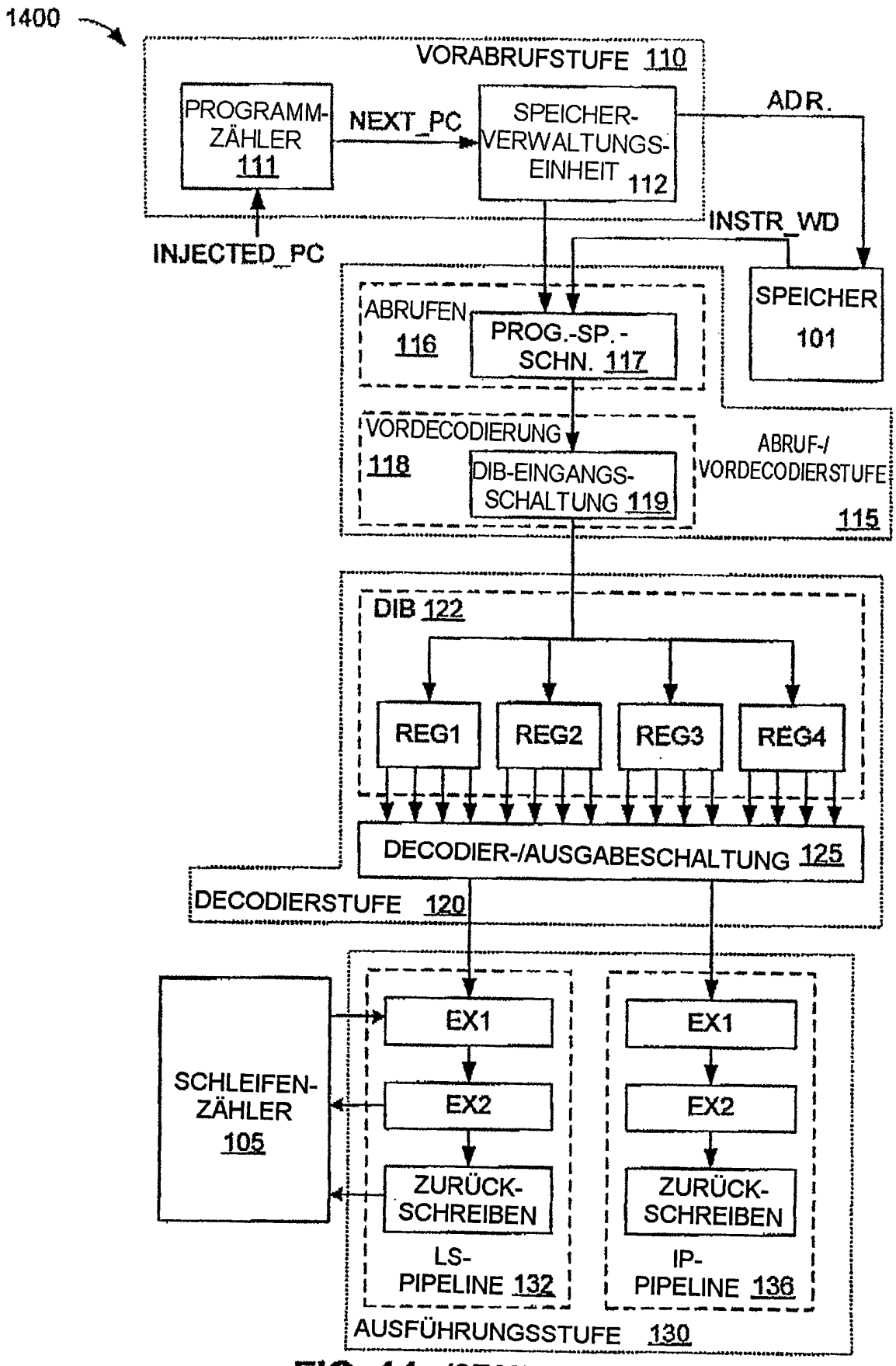


FIG. 14 (STAND DER TECHNIK)