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

(10) **DE 697 27 031 T2** 2004.11.25

(12) # Übersetzung der europäischen Patentschrift

(97) **EP 0 840 232 B1**
(21) Deutsches Aktenzeichen: **697 27 031.9**
(96) Europäisches Aktenzeichen: **97 119 052.5**
(96) Europäischer Anmeldetag: **31.10.1997**
(97) Erstveröffentlichung durch das EPA: **06.05.1998**
(97) Veröffentlichungstag der Patenterteilung beim EPA: **02.01.2004**
(47) Veröffentlichungstag im Patentblatt: **25.11.2004**

(51) Int Cl.[7]: **G06F 12/08**

(30) Unionspriorität:
29232 **31.10.1996** **US**

(84) Benannte Vertragsstaaten:
DE, FR, GB, IT, NL

(73) Patentinhaber:
Texas Instruments Inc., Dallas, Tex., US

(74) Vertreter:
Prinz und Partner GbR, 81241 München

(72) Erfinder:
Krueger, Steven D., Dallas, US; Shiell, Jonathan H., Plano, US

(54) Bezeichnung: **Mikroprozessor mit Mitteln zur Speicherung von nicht-cachespeicherbaren Daten**

Beschreibung

HINTERGRUND DER ERFINDUNG

[0001] Die vorliegenden Ausführungsformen betreffen Mikroprozessoren und sind vor allem auf einen Mikroprozessor gerichtet, wie er im Oberbegriff des Anspruchs 1 definiert ist.

[0002] Für den Fachmann ist klar, dass moderne hochleistungsfähige Datenverarbeitungssysteme auf herkömmliche Weise unter Verwendung von Ein-Chip-Mikroprozessoren als Zentraleinheiten (CPU) sowie unter Verwendung eines Halbleiter-Direktzugriffsspeichers (RAM) als Hauptsystemspeicher ausgeführt sind. Der Hauptspeicher ist im Allgemeinen in Form von Direktzugriffsspeichervorrichtungen (RAM), die eine hohe Speicherdichte und geringe Kosten pro Bit aufweisen, etwa als dynamischer RAM (DRAM), ausgeführt, wobei jedoch die Zugriffe und Zyklen für herkömmliche DRAM-Speicher verhältnismäßig lange dauern und nicht mit der Taktrate moderner Mikroprozessoren Schritt halten können.

[0003] Herkömmliche mikroprozessorbasierende Datenverarbeitungssysteme sind die Leistungsbeschränkungen des Hauptspeicherzugriffs mit der Verwendung von Cache-Speichern angegangen, während weiterhin der Vorteil der niedrigen Kosten von DRAMs mit hoher Speicherdichte erzielt wird. Cache-Speicher sind typischerweise kleine Blöcke sehr schneller statischer RAMs (SRAM), entweder chipintegriert mit dem Mikroprozessor oder extern (oder beides), um die Inhalte von Speicherzellen zu speichern, auf die wahrscheinlich in nächster Zeit zugegriffen wird. Typischerweise speichert ein Cache-Speicher die Inhalte von Speicherzellen, die nahe Nachbarn einer Speicherzelle sind, auf welche vor kurzem zugegriffen worden ist; da Mikroprozessoren oftmals sequenziell auf Speicher zugreifen, ist es wahrscheinlich, dass aufeinander folgende Speicherzugriffe in aufeinander folgenden Zyklen auf Speicheradressen zugreifen werden, die im Speicherraum sehr nahe beieinander sind. Folglich könnte durch Speichern der Inhalte benachbarter Speicherzellen in einen Cache ein guter Teil der Speicherzugriffe vom Mikroprozessor eher auf den Cache als auf den Hauptspeicher ausgeführt werden. Die Gesamtleistungsfähigkeit des Systems wird folglich durch Implementieren eines Cache-Speichers verbessert. Einige moderne Mikroprozessoren weisen mehrere Cache-Speicherebenen auf, wobei die Kapazität des Caches mit jeder folgenden Ebene zunimmt (und seine Geschwindigkeit abnimmt), um die Leistungsfähigkeit zu optimieren. Ein in Design und Ausführung intelligenter Cache kann die Systemleistung durch ein Minimieren der Zugriffe auf den Hauptspeicher stark verbessern.

[0004] Eine weitere Methode, um die Leistungsfähigkeit des Speicherzugriffs in mikroprozessorbasierenden Systemen zu verbessern, ist die Verwendung von speziellen Speicherzugriffszyklen, die gemeinhin als Burst-Zugriffszyklen (stoßweise Zugriffszyklen) bezeichnet werden. Burst-Speicherzugriffszyklen werden beim Betrieb von Speichervorrichtungen verwendet, um einen Zugriff auf eine Serie von Speicherzellen zu ermöglichen. Typischerweise erfolgt der Burst-Zugriff über einen Speichercontrollerchip, der zwischen den Mikroprozessor und den Hauptspeicher geschaltet ist und in Reaktion auf die vom Mikroprozessor ausgegebenen Adressinformationen und Steuersignale wirksam wird. Burst-Zyklen sind sehr effektiv, um die Leistungsfähigkeit von Speicherzugriffen zu verbessern. Beispielsweise kann ein Burst-Zyklus in einem modernen System mit einem 8-Byte-Bus bei der Ausgabe einer einzigen Speicheradresse in nur fünf Buszyklen (2-1-1-1) auf zweiunddreißig Bytes Speicher zugreifen, wenn ein Best Case Cache verwendet wird. Ein Burst-Zugriff ist außerdem sehr effizient, wenn ein Direktzugriffsspeicher für Überlagerungstechnik verwendet wird, bei dem ein 32-Byte-Zugriff in einer Buszyklussequenz von 8-3-3-3 (insgesamt siebzehn Buszyklen) ausgeführt werden könnte, und wenn spezielle DRAM-Funktionen wie etwa eine frühzeitige Datenausgabe (EDO) und ein synchroner DRAM verwendet werden, bei dem 32-Byte-Burst-Zugriffe im besten Fall mit einer Buszyklussequenz von 6-1-1-1 (insgesamt neun Buszyklen) ausgeführt werden könnten. Dies stellt eine erhebliche Verbesserung gegenüber dem nicht stoßweisen Fall dar, bei dem ein Zugriff auf eine 32-Byte-Zeile 64 Zyklen erfordert, wenn in einer Gruppe von acht einzelnen 4-Byte-Leseoperationen zugegriffen wird (wobei berücksichtigt wird, dass nicht stoßweise Zugriffe im Allgemeinen nicht länger als 4 Bytes sind). An sich ist der Stoßbetrieb-Speicherzugriff (Burst-Mode-Speicherzugriff) typisch zwei- bis sechsmal so schnell wie Nichtstoßbetriebs-Zyklen.

[0005] In Mikroprozessoren, welche die wohl bekannte "x86"-Architektur verwenden, einschließlich der Mikroprozessoren der so genannten Pentiumklasse (womit Mikroprozessoren bezeichnet werden, die eine Funktionalitäts- und Befehlssatzkompatibilität mit den von der Firma Intel Corporation erhältlichen PENTIUM Mikroprozessoren aufweisen), sind Burst-Speicherzugriffe mit Cache-Operationen gekoppelt. Mit anderen Worten: In diesen Mikroprozessoren mit x86-Architektur werden Burst-Speicheroperationen nur in Verbindung mit Cache-Zeilen-Fülloperationen (Lesen aus dem Speicher) und Cache-Zurückschreiboperationen (Schreiben in den Speicher) ausgeführt. Unter der Voraussetzung der Cache-Architektur dieser Mikroprozessoren, wobei der überwiegende Teil des Daten- und Befehlszurückladens über den Cache-Speicher erfolgt, ist die Leistungsfä-

higkeit, die durch Ausführen von Burst-Speicherzugriffen für Cache-Operationen geboten wird, recht hoch.

[0006] Eine Cache-Verwendung funktioniert typischerweise recht gut für "reale" Speicherzellen, auf die nur der Mikroprozessor Daten schreibt oder aus denen nur der Mikroprozessor Daten liest, wobei er herkömmliche Speicherzugriffsoperationen verwendet, da der Mikroprozessor sicherstellen kann, dass seine Cache-Kopie der Speicherzelle mit der Kopie im Hauptspeicher übereinstimmt. Solange die Cache- und Hauptspeicher-Kopien derselben Speicherzellen gleich sind, wird das Lesen der Cache-Kopie statt der Hauptspeicherkopie keine Nebenwirkungen haben. Jedoch sind bestimmte Speicherzellen, wie etwa jene, die den Zustand einer E/A-Vorrichtung enthalten, oder jene, die Teile des Bildwiederholspeichers enthalten, die durch einen Graphik-Beschleuniger geändert werden könnten, in einem Ausmaß flüchtig, dass Cache-Kopien von diesen Speicherzellen häufig unaktuell wären. Das Lesen einer Cache-Kopie dieser flüchtigen Speicherzellen anstelle der Hauptspeicherzellen könnte starke Nebenwirkungen auf den Betrieb des Systems haben. Dementsprechend werden Zugriffe des Mikroprozessors auf diese flüchtigen Speicherzellen in herkömmlichen Systemen mit einer IBM PC-Architektur typisch durch den Betrieb eines Speichercontrollers gegen eine "Cachefähigkeit" (d. h. eine Speicherung in einen Cache-Speicher) verriegelt.

[0007] Beispielsweise ist das speicherabgebildete Register im Allgemeinen ein im Hinblick auf einen cachefähigen Zugriff geblockter Bereich, auf den trotzdem mittels eines herkömmlichen Speicherzugriffs zugegriffen werden kann, da das speicherabgebildete Register häufig abgefragt wird, um Veränderungen des Gerätezustands zu erfassen, in Reaktion auf welche bestimmte Steuerungsfunktionen wirksam werden. Wenn die speicherabgebildeten Register als Cache ausgeführt wären, würden sich Veränderungen des Gerätezustands zwar in der Hauptspeicher-Kopie des speicherabgebildeten Registers wiederspiegeln, jedoch nicht in der als Cache ausgeführten Kopie; eine periodische Abfrage des speicherabgebildeten Registers würde nur die Cache-Kopie lesen und daher nicht die gesuchte Änderung des Gerätezustands erfassen, wodurch die Steuerung praktisch zum erliegen gebracht wird. Um ein weiteres Beispiel zu geben: Das Cachen von Nichtspeicher-Vorrichtungen, wie etwa von speicherabgebildeten E/A-Funktionen, könnte zusätzliche Nebenwirkungen für jene Typen von E/A-Vorrichtungen herbeiführen, die den Zustand in Reaktion auf eine Leseoperation auf dem Bus ändern, da Lesevorgänge in einem chipintegrierten Cache-Speicher nicht als Buszyklen in Erscheinung treten. Ein Cachen mittels Zurückschreiben weist ebenfalls Nebenwirkungen für diese Nichtspeicher-Positionen auf, da der Cache eine neuere Kopie als der Hauptspeicher enthalten könnte; da Schreiboperationen zum Cache-Zurückschreiben nicht auf dem Bus erscheinen, würde das Cachen dieser Speicherstellen auftreten, um wiederholt Schreiboperationen anzufordern, die auf dem Bus ausgeführt werden.

[0008] Ein weiteres Beispiel für einen Speicherbereich, der flüchtig ist und deshalb typisch gegen einen cachefähigen Zugriff verriegelt ist, ist der Bildspeicher, der logisch innerhalb der Speicherabbildung des Mikroprozessors und physisch entweder innerhalb des Hauptspeichers oder von diesem getrennt (etwa in einem Graphikadapter) lokalisiert ist. Der Bildspeicher wird oftmals von einer vom Mikroprozessor verschiedenen Vorrichtung gesteuert, etwa von einem Graphikprozessor oder von einem Graphikadapter, und ist deshalb für einen cachefähigen Zugriff durch den Haupt-Mikroprozessor nicht geeignet, weil seine Inhalte häufig außerhalb der Steuerung durch den Mikroprozessor verändert werden. Wenn ein Teil des Bildspeichers im Mikroprozessor-Cache gespeichert wäre, würden die Cache-Inhalte für nachfolgende Zugriffe auf Grund der Änderungen, die von dem Graphikprozessor vorgenommen werden, wahrscheinlich ungültig sein.

[0009] Deshalb sind bei herkömmlichen Mikroprozessoren gemäß einer x86-Architektur burstfähige Speicherzugriffe an die Cachefähigkeit der Speicherstelle, auf die zugegriffen wird, gekoppelt. Beispielsweise fordert der PENTIUM-Mikroprozessor einen burstfähigen Speicherzugriff an, indem er während eines Zugriffs auf den Speicher (durch den hohen Logikpegel des Mikroprozessors am Anschluss M/IO# signalisiert) ein Steuersignal am Anschluss CACHE# geltend macht (das # gibt an, dass das Signal bei einem niedrigen Logikpegel aktiv ist). In Reaktion auf diese Anforderung ermittelt der Speichercontroller, ob sich die von dem Mikroprozessor ausgegebene Adresse in einem cachefähigen Bereich des Speicherraums befindet, und wenn ja, macht er die KEN#-Eingabe in den Mikroprozessor geltend und führt den Burst-Zugriff aus. Gemäß dieser herkömmlichen Implementierung wird dann, wenn der Mikroprozessor einen burstfähigen Zugriff auf einen Speicherbereich anfordert, der gegen einen cachefähigen Zugriff verriegelt ist, der Speichercontroller keinen Burst-Zugriff bewirken und dies durch Aufheben der Gültigkeit des Signals KEN# anzeigen. Es erfolgt dann ein Einzelübertragungszugriff auf die gewünschte Speicherstelle.

[0010] In diesem Zusammenhang wird auf das Dokument US-A-5 561 780 verwiesen, das ein Verfahren und eine Vorrichtung zur Kombination von nicht cachefähigen Daten-Schreiboperationen in Schreibpuffer von der Größe einer Cache-Zeile beschreibt. Ein Schreiboperationen kombinierender Puffer kombiniert für nicht cachefähige Datentypen, wie etwa Bildwiederholspeicherdaten, Daten von verschiedenen Daten-Schreibopera-

tionen in Puffereinheiten von der Größe einer Cache-Zeile. Der Schreiboperationen kombinierende Puffer ist innerhalb eines Mikroprozessors ausgeführt, der eine Daten-Cache-Einheit besitzt, die cachefähige Daten in Cache-Zeilen speichert. Die Daten-Cache-Einheit weist Bauelemente und Schaltungsanordnungen auf, die für eine effiziente Ein- und Ausgabe von eine Zeile umfassenden Cache-Daten-Einheiten vorgesehen sind. Durch Kombinieren vieler nicht cachefähiger Daten-Schreiboperationen in einem einzigen Puffer vom Umfang einer Cache-Zeile werden die Schaltungsanordnung und die Verfahren, die für die Verarbeitung von Cache-Zeilen verwendet werden, auch für die Verarbeitung von nicht cachefähigen Daten benutzt.

[0011] Des Weiteren wird auf das Dokument EP-A-0 382 396 hingewiesen, das einen Programmpufferspeicher zur Pufferung von Daten von einem externen Speicher an einen Prozessor offenbart. Der Pufferspeicher enthält einen Cache-Speicher, einen FIFO-Speicher und einen direkten Datenpfad, die entsprechend dem besonderen Abruf, der durch den Prozessor erfolgt, wählbar sind. Des Weiteren enthält der Pufferspeicher eine Schreib-Auswahleinrichtung, um Daten vom externen Speicher entweder in den Cache-Speicher, in den FIFO-Speicher oder auf den direkten Datenpfad zu schreiben, eine Lese-Auswahleinrichtung, um Daten entweder aus dem Cache-Speicher, aus dem FIFO-Speicher oder von dem direkten Datenpfad zu lesen, und einen Controller, um die Schreib- und Lese-Auswahleinrichtungen in Reaktion auf das Auftreten bestimmter Bedingungen zu steuern.

ZUSAMMENFASSUNG DER ERFINDUNG

[0012] Bei der vorliegenden Erfindung gibt es einen Mikroprozessor, der mit einem externen Schreib-Lese-Speicher gekoppelt werden kann, der einen adressierbaren Speicherraum besitzt. Dieser Speicherraum speichert cachefähige digitale Daten und nicht cachefähige digitale Daten. Der Mikroprozessor weist die im Anspruch 1 angegebenen Merkmale auf.

KURZBESCHREIBUNG DER VERSCHIEDENEN FIGUREN DER ZEICHNUNG

[0013] Die vorliegende Erfindung wird nun anhand von Beispielen mit Bezug auf die beigefügte Zeichnung beschrieben, worin

[0014] **Fig.** 1 ein Blockschaltplan eines Datenverarbeitungssystems gemäß einer erfinderischen Ausführungsform ist;

[0015] **Fig.** 2 ein Blockschaltplan einer ersten Ausführungsform des Speichers und des Ein/Ausgabe-Controllers des Systems von **Fig.** 1 ist;

[0016] **Fig.** 3 eine Prinzipskizze des Speicheradressraums und des Ein/Ausgabe-Adressraums des Datenverarbeitungssystems von **Fig.** 1 ist, die in schematischer Weise die Bedingungen veranschaulicht, unter denen auf jeden Adressraum zugegriffen wird;

[0017] **Fig.** 4 ein Steuerungsdiagramm ist, das die Anforderung und die Ausführung eines Burst-Lesens aus einem nicht cachefähigen Speicher gemäß der ersten Ausführungsform veranschaulicht;

[0018] **Fig.** 5 ein Steuerungsdiagramm ist, das die Anforderung und die Ausführung eines Burst-Schreibens aus einem nicht cachefähigen Speicher gemäß der ersten Ausführungsform veranschaulicht;

[0019] **Fig.** 6 ein Blockschaltplan einer zweiten Ausführungsform des Speichers und des Ein/Ausgabe-Controllers des Systems von **Fig.** 1 ist;

[0020] **Fig.** 7 eine schematische Darstellung einer Ausführungsform zum Speichern nicht cachefähiger Daten und einer zugeordneten Adresse ist, die von der CPU gelesen und geschrieben sowie während einer Aktivitätsdauer, die durch einen zugeordneten Zähler bestimmt ist, modifiziert werden könnten;

[0021] **Fig.** 8 ein Ablaufplan einer Ausführungsform des Verfahrens für die Operation der schematischen Darstellung von **Fig.** 7 ist;

[0022] **Fig.** 9 eine schematische Darstellung einer Ausführungsform zum Speichern mehrerer Zeilen nicht cachefähiger Daten und zugeordneter Adressen ist, wobei jede Zeile von der CPU gelesen und geschrieben sowie während einer Aktivitätsdauer, die durch einen zugeordneten Zähler bestimmt ist, modifiziert werden könnte; und

[0023] **Fig.** 10 eine schematische Darstellung einer Ausführungsform zum Speichern mehrerer Zeilen von Daten und zugeordneten Adressen ist, wobei einige der Zeilen mittels bekannter Cache-Grundsätze verwaltet werden, während andere von der CPU gelesen und geschrieben sowie während einer Aktivitätsdauer, die durch einen Zähler bestimmt ist, der durch das Tag für die Zeile bestimmt ist, modifiziert werden könnten.

AUSFÜHRLICHE BESCHREIBUNG DER ERFINDUNG

[0024] Mit Bezug auf **Fig.** 1 wird nun ein beispielhaftes mikroprozessorbasierendes System **2** ausführlich beschrieben, in dem eine erste bevorzugte Ausführungsform umgesetzt ist. Wie in **Fig.** 1 gezeigt ist, weist das System **2** eine Zentraleinheit (CPU) **5** auf, die in dieser Ausführungsform der Erfindung ein Mikroprozessor der wohl bekannten "x86-Architektur" und vorzugsweise ein Mikroprozessor der Pentium-Klasse ist. Die CPU **5** weist eine Busschnittstelleneinheit (BIU) **8** auf. Die Busschnittstelleneinheit **8** ist eine Schaltungsanordnung innerhalb der CPU **5**, die zur Steuerung und Ausführung der Kommunikation zwischen der CPU **5** und dem Rest des Systems **2** dient. In dieser Ausführungsform ist die Busschnittstelleneinheit **8** der CPU **5** an einen Bus gekoppelt, der den Adressbus ABUS, den Datenbus DBUS und den Steuerbus CBUS umfasst. Wie im Fachgebiet üblich ist der Adressbus ABUS ein Bus, auf dem die CPU **5** eine im Binärsystem ausgedrückte Adresse ausgibt, um auf weitere Elemente des Systems **2** zugreifen zu können, der Datenbus DBUS ist ein Bus für die Übertragung von digitalen Daten zwischen der CPU **5** und den übrigen Systemelementen und der Steuerbus CBUS ist ein Bus, über den Steuersignale zu den Elementen des Systems **2** übertragen werden.

[0025] In dem System **2** von **Fig.** 1 sind verschiedene periphere Elemente über zugeordnete Controller an die Busse ABUS, DBUS, CBUS angeschlossen, um die üblichen Systemfunktionen auszuführen. Der Hauptspeicher **20** des Systems **2** ist über den Speichercontroller **10** mit den Bussen ABUS, DBUS, CBUS gekoppelt; an sich empfängt der Speichercontroller **10** Adresswerte und Steuersignale von der CPU **5** und präsentiert dem Hauptspeicher **20** entsprechende Steuersignale, um die angestrebte Operation auszuführen, die im Allgemeinen die Übertragung von Daten zur CPU **5** oder von dieser auf dem Datenbus DBUS mit sich bringt. Außerdem umfasst das System **2** einen Level-3-Cache-Speicher **22**, der auf herkömmliche Weise an den Cache-Controller **12** angeschlossen ist; der Cache-Controller **12** ist an die Busse ABUS, DBUS, CBUS angeschlossen, um die Datenübertragung zwischen der CPU **5** und dem Level-3-Cache-Speicher **22** zu steuern. In diesem Beispiel ist der Level-3-Cache-Speicher **22** ein Cache-Speicher der dritten Ebene für das System **2**, wobei sich Cache-Speicher der Ebenen **1** und **2** innerhalb der CPU **5** befinden (wobei in **Fig.** 1 der Level-2-Cache **6** gezeigt ist). Das System **2** enthält außerdem einen E/A-Controller **14**, der über die Busse ABUS, DBUS, CBUS an die CPU **5** angeschlossen ist und außerdem an verschiedene Ein/Ausgabe-Vorrichtungen **24** angeschlossen ist. Die Ein/Ausgabe-Vorrichtungen **24** könnten typische Eingabe- und Ausgabe-Peripheriegeräte des Systems **2** wie etwa ein Bildschirmgerät, eine Tastatur, ein Zeigegerät, Plattenlaufwerks-Untersysteme und dergleichen umfassen. Die Controller **10**, **12**, **14** sind typisch mittels eines sogenannten "Baustein-Satzes" implementiert, der so beschaffen ist, dass er mit der CPU **5** zusammenwirkt. Des Weiteren umfasst das System **2** eine Taktgeberschaltung **16**, die auf der Leitung CLK ein periodisches Taktsignal erzeugt, das an jedes der Elemente des Systems **2**, einschließlich der CPU **5** über die Busschnittstelleneinheit **8**, weitergeleitet wird und von dem in jedem der verschiedenen Systemelemente interne Taktsignale erzeugt werden könnten. Dementsprechend wird das System **2** als einem typischen modernen Computer entsprechend, etwa vom Desktop-, Workstation- oder tragbaren Typ, angesehen, bei dem Computerprogramme in Plattenspeichern (durch eines der Ein/Ausgabe-Vorrichtungen **24** repräsentiert) gespeichert sind und für den Betrieb in den Hauptspeicher **20** geladen werden.

[0026] Speicherzugriffe werden durch das Übergeben eines Adresswertes auf dem Bus ABUS durch die CPU **5** in Kombination mit den entsprechenden Steuersignalen auf dem Steuerbus CBUS (einschließlich eines Lese/Schreib-Auswahlsignals) ausgeführt, wobei der Speicherkontroller **10** wiederum die entsprechenden Steuersignale an den Hauptspeicher **20** übergibt, um auf die angestrebte Position zuzugreifen. In diesem Beispiel ist ein Abschnitt **20v** des Hauptspeichers **20** Bildspeicher, auf den von der einen der E/A-Funktionen **24**, die dem Graphikadapter (oder Controller) entspricht, entweder über die Busse ABUS, DBUS, CBUS oder über einen (nicht gezeigten) zweiten Bus unabhängig zugegriffen werden kann; oder als eine andere Möglichkeit könnte der Bildspeicher **20v** wie im Fall einer PCI-Videokarte als eigenständiges Gerät auf dem Bus implementiert sein. Unter der Steuerung durch den Speichercontroller **10**, die im Allgemeinen mittels eines Quittungsaustausch-Protokolls zwischen diesem und der CPU **5** über den Steuerbus CBUS erfolgt, übergibt die CPU **5** entweder Daten an den Datenbus DBUS (in einer Speicher-Schreiboperation) oder empfängt Daten vom Datenbus DBUS (in einer Speicher-Leseoperation).

[0027] Wie in **Fig.** 1 gezeigt ist, tritt sowohl der Speicher- als auch der Ein/Ausgabe-Datenverkehr über denselben Bussen, nämlich dem Adressbus ABUS, dem Datenbus DBUS und dem Steuerbus CBUS auf. Dem-

entsprechend werden in dieser Ausführungsform, wie dies für mikroprozessorbasierende Systeme mit einer x86-Architektur typisch ist, Ein/Ausgabe-Zugriffe auf ähnliche Weise wie die obenbeschriebenen Speicherzugriffe ausgeführt, wobei die CPU **5** in Kombination mit den entsprechenden Steuersignalen auf der Leitung CBUS dem Adressbus ABUS eine Adresse übergibt. Bei einer E/A-Operation entspricht die Adresse auf dem Adressbus ABUS einer bestimmten der Ein/Ausgabe-Funktionen **24**. Unter der Steuerung durch den E/A-Controller **14** werden dann von der CPU **5** über den Datenbus DBUS an die ausgewählte E/A-Funktion **24** (für eine Ausgabeoperation) oder von der ausgewählten E/A-Funktion **24** über den Datenbus DBUS an die CPU **5** (für eine Eingabeoperation) Daten übertragen.

[0028] Mit Bezug auf **Fig.** 2 werden nun der Aufbau und die Funktionsweise des Speichercontrollers **10** gemäß der ersten Ausführungsform und in Verbindung mit der CPU **5** und dem Hauptspeicher **20** ausführlich beschrieben. Wie in **Fig.** 2 gezeigt ist, kommuniziert die Busschnittstelleneinheit **8** der CPU **5** über den Adressbus ABUS, den Datenbus DBUS und den Steuerbus CBUS, die mit bestimmen Anschlüssen der CPU **5** verbunden sind, mit dem Speichercontroller **10**. Die Abschlüsse der CPU **5** sind ihre externen Verbinder, die in Form von Anschlussstiften (wie in PGA-Gehäusen), Lötperlen, Gehäusezuleitungen, Bondinseln (wenn in Chipform) oder von irgendeinem anderen herkömmlichen Typ von Außenverbindung für Gehäuse integrierter Schaltungen sein könnten. Wie Fachleuten bekannt ist, werden die Anschlüsse der CPU **5** herkömmlich mit Anschlussstift- oder Signalnamen bezeichnet; für diese Beschreibung haben die Busleitungen, die mit den Anschlüssen der CPU **5** verbunden sind, den gleichen Namen wie ihr entsprechender Anschluss. Wie in **Fig.** 2 gezeigt ist, umfasst der Steuerbus CBUS Leitungen, die Anschlüssen der CPU **5** entsprechen und die in diesem Beispiel herkömmliche Pentium-Klassen-Mikroprozessoranschlüsse wie etwa Speicher/EA-Auswahl (memory/IO select) M/IO#, Cache-Anforderung CACHE#, Burst-Betriebsbereitschaft (burst-ready) BRDY# und Cache-Freigabe (cache enable) KEN# enthalten. Weitere herkömmliche Anschlüsse der Pentium-Klasse, die zur Ausführung von Speicherzugriffen verwendet werden (jedoch in **Fig.** 2 nicht gezeigt sind), umfassen die Byte-Freigabe-Signale (byte enable) BE7#-BE0, das Adresszustandssignal (address status) ADS#, das Daten/Steuerungs-Auswahlsignal (data/control select) D/C#, das Lesen/Schreiben-Auswahl-Signal (write/read select) W/R#, das Zurückschreiben/Durchschreiben-Signal (write-back/write-through) WB/WT# usw. Der Taktgeber **16** liefert auf der Leitung CLK ein Taktsignal (über die Busschnittstelleneinheit **8**) an die CPU **5** und an den Speichercontroller **10**.

[0029] Der Speichercontroller **10** enthält mehrere Funktionsblöcke, die für eine Kommunikation zwischen der CPU **5** und dem Hauptspeicher **20** sorgen. Diese Blöcke sind in **Fig.** 2 funktionell dargestellt, da angenommen wird, dass ein Durchschnittsfachmann den Aufbau und die Funktionsweise des Speichercontrollers **10** aus einer Funktionsbeschreibung vollständig erfassen wird. Wie im Fach üblich empfängt der Adresspuffer **21** die Speicheradresse von der CPU **5** auf dem Adressbus ABUS und befördert diese Speicheradresse zum Adressmultiplexer **26** für eine Weitergabe an den Hauptspeicher **20** auf den Adressleitungen AN bis A0. Genauso empfängt der Datenpuffer **23** im Speichercontroller **10** Daten von der CPU **5** auf dem Datenbus DBUS und befördert diese Daten in Schreiboperationen zum Schreibpuffer **27**, der über Datenleitungen D_{in} mit dem Hauptspeicher **20** in Verbindung steht; dieser Pfad ist bidirektional, jedoch könnte an sich der Schreibpuffer **27** Daten vom Hauptspeicher **20** auf Leitungen D_{out} empfangen und diese Daten für eine Weiterleitung an die CPU **5** in Speicher-Leseoperationen zum Datenpuffer **23** befördern.

[0030] Die Steuerung der Kommunikation zwischen der CPU **5** und dem Hauptspeicher **20** erfolgt mittels der Bussteuerschaltung **25**, die mit dem Steuerbus CBUS und der Speichersteuereinheit **29** verbunden ist; außerdem enthält der Speichercontroller **10** eine Steuerlogik **28**, welche die internen Operationen des Speichercontrollers **10** steuert. Insbesondere empfängt die Bussteuerschaltung **25** Steuersignale, einschließlich der Signale M/IO# und CACHE#, von der CPU **5** auf dem Steuerbus CBUS und weist in Reaktion auf eine Speicherzugriffsanforderung die Speichersteuerschaltung **29** an, die entsprechenden herkömmlichen DRAM-Steuer- und Synchronisiersignale an den Hauptspeicher **20** auszugeben, um den Speicherzugriff auszuführen, wobei solche Signale Row Address Strobe (RAS#), Column Address Strobe (CAS#), Schreib-Freigabe (write enable: WE#) einschließen. Außerdem gibt die Bussteuerschaltung **25** Steuersignale an die CPU **5** aus, die den Zustand des Speicherzugriffs anzeigen, wobei solche Steuersignale sowohl ein Cache-Freigabesignal (cache enable) KEN# als auch ein Burst-Betriebsbereitschaftssignal (burst ready) BRDY# enthalten, die von besonderer Bedeutung für Burst-Zugriffe sind, wie weiter unten beschrieben ist.

[0031] Die Steuerlogik **28** des Speichercontrollers **10** umfasst außerdem vorzugsweise programmierbare Register für die Steuerung des Betriebs des Speichercontrollers **10**, wobei Register eingeschlossen sind, die nicht cachefähige Speicherstellen des Hauptspeichers **20** definieren. In Reaktion auf jeden von der CPU **5** angeforderten Speicherzugriff prüft die Steuerlogik **28** diese Register, um zu ermitteln, ob sich die angestrebte Speicheradresse, die auf dem Adressbus ABUS weitergeleitet wird, in einem nicht cachefähigen Bereich des Spei-

cheradressraums befindet. Wie Fachleuten bekannt ist, macht der Speichercontroller **10** die Leitung KEN# geltend, um der CPU **5** anzuzeigen, dass der angestrebte Speicherzugriff cachefähig ist, und hebt das Geltendmachen der Leitung KEN# auf, wenn sich die angestrebte Speicheradressen in einem nicht cachefähigen Bereich befindet.

[0032] Wie weiter oben beschrieben worden ist, standen gemäß dem Stand der Technik burstfähige Speicherzugriffe nur für Speicheradressen zur Verfügung, die sich nicht in einem nicht cachefähigen Bereich befanden. In Mikroprozessorimplementierungen des Standes der Technik würde deshalb das Geltendmachen der Leitung KEN# in Reaktion auf eine Anforderung eines Speicherzugriffs, der auf einen nicht cachefähigen Bereich erfolgt, aufgehoben werden; außerdem würde der Anschluss BRDY# des Mikroprozessors statt für mehrere Buszyklen, wie dies für einen Burst-Zugriff der Fall wäre, nur für die Dauer einer Einzelübertragungsoperation (Lesen oder Schreiben), die in einem nicht cachefähigen Bereich des Speichers ausgeführt wird, geltend gemacht.

[0033] **Fig.** 3 veranschaulicht in schematischer Form die entsprechenden Adressräume im System **2** gemäß dieser Ausführungsform. In dem beispielhaften System **2** sind wie für Mikroprozessoren der Pentiumklasse üblich ein 4-GByte-Speicheradressraum **30** (Speicheradressen von 0000 0000hex bis FFFF FFFFhex) und ein 64 kByte-E/A-Adressraum **40** (Speicheradressen von 0000 0000hex bis 0000 FFFFhex) vorgesehen. Wie es für Mikroprozessoren der Pentiumklasse einschließlich der CPU **5** typisch ist, verwenden der Speicheradressraum **30** und der E/A-Adressraum **40** einige der Adresswerte gemeinsam (d. h. Adresswerte zwischen 0000 0000hex und 0000 FFFFhex entsprechen sowohl Speicherstellen im Speicheradressraum **30** als auch im E/A-Adressraum **40**). Wie weiter oben beschrieben worden ist, legen herkömmliche Mikroprozessoren mit x86-Architektur ein Steuersignal an einem mit M/IO# bezeichneten Anschluss vor, das bei einem hohen Logikpegel anzeigt, dass ein Speicherzugriff stattfinden sollte, und mit einem niedrigen Logikpegel anzeigt, dass eine E/A-Operation stattfinden sollte.

[0034] Wie in **Fig.** 3 gezeigt ist, schließt der Speicheradressraum **30** einen nicht cachefähigen Bereich **32** ein. Der nicht cachefähige Bereich **32** entspricht einem Bereich des Speicheradressraums **30**, der gegen einen Zugriff mittels eines Cache-Schreibens oder eines Cache-Lesens verriegelt ist, wie weiter oben beschrieben ist. Ein cachefähiger Zugriff auf nicht cachefähige Bereiche wird entsprechend dem Inhalt bestimmter Register, welche die Adressen enthalten, für die ein cachefähiger Zugriff zu verhindern ist, durch den Speichercontroller **10** und insbesondere seine Steuerlogik **28** verhindert. Der nicht cachefähige Bereich **32** könnte den Bildspeicher **20v** (siehe **Fig.** 1) oder speicherabgebildete Steuerregister enthalten, die typisch alle als für Cache-Speicherungen nicht geeignet angesehen werden. An sich wird verhindert, dass während des Betriebs des Speichercontrollers **10** und seiner Ausgabe eines nicht die Geltung beanspruchenden Zustands auf der Leitung KEN# die Inhalte des nicht cachefähigen Bereichs **32** in internen Caches innerhalb der CPU **5** (wie etwa dem in **Fig.** 1 gezeigten Level-2-Cache **6**) gespeichert werden.

[0035] Es ist jedoch von den Erfindern festgestellt worden, dass es zweckmäßig sein könnte, nicht cachefähigen Bereichen des Hauptspeichers **20** in Mikroprozessoren mit einer x86-Architektur die Fähigkeit zu einem Burst-Zugriff zu verschaffen. Beispielsweise könnte die CPU **5** schnell (d. h. in einer Burst-Betriebsart) auf die Inhalte eines Teils des Bildspeichers **20v** im nicht cachefähigen Bereich **32** zugreifen wollen. Beispielsweise könnte die CPU das Rasterbild in den Speicher kopieren wollen oder einen anderen Teil des Speichers in den Bildspeicher kopieren wollen. Da der Bildspeicher **20** nicht cachefähig ist, würden Systeme mit einer x86-Architektur einen burstfähigen Zugriff auf den nicht cachefähigen Bereich **32** verhindern.

[0036] Gemäß einer ersten bevorzugten Ausführungsform ist jedoch die CPU **5** durch Nutzen von vorhandenen Steuersignalen auf dem Steuerbus CBUS in der Lage, einen burstfähigen, jedoch nicht cachefähigen Zugriff auf den Hauptspeicher **20** anzufordern. Wie weiter oben angegeben worden ist, wird das Steuerausgangssignal M/IO# von Mikroprozessoren mit herkömmlicher x86-Architektur verwendet, um anzuzeigen, ob ein Speicherzugriff (M/IO# auf hohem Pegel) oder eine E/A-Operation ((M/IO# auf niedrigem Pegel) auszuführen ist. Hingegen wird gemäß dieser Ausführungsform die Kombination aus einem niedrigen Logikpegel auf der Leitung M/IO# und einem Geltendmachen des Signals CACHE# (mit einem niedrigen Logikpegel) von der CPU **5** verwendet, um einen burstfähigen Speicherzugriff auf einen nicht cachefähigen Bereich des Speicheradressraums **30** anzufordern. Vom Speichercontroller **10** und insbesondere von seine Bussteuerschaltung **25** wird diese Kombination aus M/IO# auf niedrigem Pegel und CACHE# auf niedrigem Pegel als eine Anforderung eines Burst-Speicherzugriffs interpretiert (statt als eine E/A-Anforderung, wie nach der Interpretation gemäß dem Stand der Technik), und es wird auch dann ein nicht cachefähiger Burst-Speicherzugriff auf den Hauptspeicher **20** ausgeführt, wenn die Steuerlogik **28** angibt, dass sich die Speicheradresse innerhalb des nicht cachefähigen Bereichs **32** des Speicheradressraums **30** befindet. Wenn die CPU **5** in Kombination mit M/IO# und

CACHE# auf niedrigem Pegel eine Adresse ausgibt, die im cachefähigen Bereich des Speicheradressraums **30** ist, wird ein Burst-Zugriff angefordert und ausgeführt, wobei die CPU **5** jedoch nicht versucht, den Zugriff zu cachen, selbst wenn in diesem Fall für die adressierte Speicherstelle ein Cache-Zugriff verfügbar ist.

[0037] Die Speicher- und E/A-Operationen gemäß dieser Ausführungsform könnten folglich von der CPU **5** mittels der Steuersignale M/IO# und CACHE# entsprechend der folgenden Tabelle 1 und der entsprechenden Logik von **Fig.** 3 angefordert werden.

Operationstyp	M/IO#	CACHE#
burstfähiger, cachefähiger Speicherzugriff	1	0
E/A-Operation (Eingabe oder Ausgabe)	0	1
nicht burstfähiger, nicht cachefähiger Speicherzugriff	1	1
burstfähiger, nicht cachefähiger Speicherzugriff	0	0

Tabelle 1

[0038] Diese Operation ist in **Fig.** 3 durch den Teil der Bussteuerschaltung **25**, der benutzt wird, um die Steuersignale MEN1 für einen einzelnen Speicherübertragungszyklus (d. h. nicht burstfähig, nicht cachefähig), MENBC für einen burstfähigen, cachefähigen Speicherzyklus, IOEN für eine E/A-Operation und MENBNC für einen burstfähigen, nicht cachefähigen Speicherzugriff zu treiben, schematisch dargestellt. Das UND-Gatter **31** treibt die Leitung MEN1 nur dann auf den hohen Pegel, wenn die beiden Leitungen M/IO# und CACHE# auf hohem Pegel sind, das UND-Gatter **33** treibt die Leitung IOEN nur dann auf den hohen Pegel, wenn die Leitung M/IO# auf niedrigem Pegel ist, während die Leitung CACHE# auf hohem Pegel ist, das UND-Gatter **35** treibt die Leitung MENBC nur dann auf den hohen Pegel, wenn die Leitung M/IO# auf hohem Pegel und die Leitung CACHE# auf niedrigem Pegel ist, und das UND-Gatter **37** treibt die Leitung MENBNC nur dann auf den hohen Pegel, wenn die beiden Leitungen M/IO# und CACHE# auf niedrigem Pegel sind.

[0039] Selbstverständlich ist gemäß dieser Ausführungsform der E/A-Controller **14** so konfiguriert, dass er nicht auf einen niedrigen Logikpegel auf der Leitung M/IO# in Kombination mit der Leitung CACHE# auf niedrigem Pegel reagiert, so dass ein Buskonflikt auf Grund eines burstfähigen, nicht cachefähigen Speicherzugriffs, der gleichzeitig mit einer unbeabsichtigten E/A-Operation auftritt, vermieden wird.

[0040] Mit Bezug auf **Fig.** 4 wird nun, um eine nähere Erläuterung zu geben, ein Steuerungsdiagramm, das die Operation eines burstfähigen, nicht cachefähigen Speicherlesens veranschaulicht, die von der CPU **5** angefordert und durch den Hauptspeicher **20** und den Speichercontroller **10** gemäß dieser ersten bevorzugten Ausführungsform ausgeführt wird, beschrieben. In diesem Beispiel wird der Speicherlesezugriff durch die CPU **5** angefordert, die eine Speicheradresse auf die Adressbusleitungen A31–A3 in Kombination mit einem Wert 0 auf den Byte-Freigabeleitungen BE7#–BE0# ausgibt, wobei diese Adresse durch die CPU **5** bei Geltendmachen eines niedrigen Logikpegels auf der Leitung ADS# als gültig erklärt wird. Bei diesem Beispiel eines burstfähigen, nicht cachefähigen Speicherlesens befindet sich die von der CPU **5** ausgegebene Adresse innerhalb eines nicht cachefähigen Bereichs **32** des Speicheradressraums **30**. In Kombination mit der Speicheradresse gibt die CPU **5** außerdem einen hohen Logikpegel auf der Leitung D/C# aus, um anzuzeigen, dass diese Operation eine Datenoperation ist, und macht auf der Leitung W/R# einen niedrigen Pegel geltend, um ein Speicherlesen anzufordern. Diese Signale sind für Leseoperationen von Mikroprozessoren mit x86-Architektur üblich. Gemäß dieser ersten Ausführungsform der Erfindung gibt die CPU **5** außerdem in Kombination mit einem niedrigen Logikpegel auf der Leitung CACHE# einen niedrigen Logikpegel auf der Leitung M/IO# aus. Diese Signale treten alle in dem Buszyklus B0 von **Fig.** 4 auf.

[0041] Diese Signalkombination gibt dem Speichercontroller **10** an, dass ein burstfähiger Zugriff auf einen

nicht cachefähigen Bereich des Hauptspeichers **20** angefordert wird. Der Speichercontroller **10** reagiert auf diese Anforderung mit der Ausgabe entsprechender Steuersignale und Adressen an die adressierten Speichereineinrichtungen im Hauptspeicher **20**. Im Buszyklus B2 (während der Buszyklus B1 ein Wartezustand ist) zeigt der Speichercontroller **10** der CPU **5** durch Ausgeben eines niedrigen Pegels auf der Leitung KEN# an, dass ein burstfähiger Zugriff auf den Hauptspeicher **20** gewährt worden ist, und zeigt durch Ausgeben eines niedrigem Pegels auf der Leitung BRDY# an, dass am Ende des aktuellen Taktzyklus gültige Speicherdaten auf dem Datenbus DBUS platziert sein werden. Die Leitung WB/WT# ist für diese Operation gleichgültig, da das Speicherlesen nicht cachefähig ist. Es erscheinen dann am Ende des aktuellen Buszyklus B2 und der nächsten drei folgenden Buszyklen B3 bis B5 (wobei vorausgesetzt wird, dass keine Wartezustände vorliegen) gültige Daten in Form von Vierfachwörtern QW0 bis QW3 (von jeweils 64 Bit oder 8 Bytes). Das Burst-Speicherlesen gemäß dieser Ausführungsform wird dann im Buszyklus B6 als abgeschlossen angezeigt, indem der Speichercontroller die Leitung BRDY# auf einen Zustand mit einem hohen Pegel treibt. Nachfolgende Speicherzugriffe des burstfähigen, nicht cachefähigen Typs oder eines anderen Typs könnten dann auf herkömmliche Weise ausgeführt werden.

[0042] Mit Bezug auf **Fig.** 5 wird nun die Wirkungsweise einer burstfähigen, nicht cachefähigen Speicherschreiboperation ausführlich beschrieben. Ähnlich wie im Fall des Speicherlesens von **Fig.** 4 leitet die CPU **5** die Operation mit der Ausgabe der angestrebten Adresse auf den Adressleitungen A31 bis A3 und des Wertes null für die Byte-Freigabebits BE# zusammen mit einem hohen Pegel auf der Leitung D/C# zu dem Zeitpunkt, zu dem die CPU **5** das Signal ADS# mit niedrigem Pegel im Buszyklus WB0 geltend macht, ein. Außerdem zeigt die CPU **5** in diesem Zyklus WB0 durch Ausgeben des hohen Pegels auf der Leitung W/R# an, dass sie ein Speicherschreiben ausführen möchte, und fordert ein Burst-Schreiben auf eine nicht cachefähige Speicherstelle, indem sie in Kombination mit der Leitung CACHE# auf niedrigem Pegel, auf der Leitung M/IO# den niedrigen Pegel geltend macht. Wenn die Adresse auf einen cachefähigen Bereich des Speicheradressraums **30** zeigt, wird wiederum wie im Fall des Lesens ein Burst-Zugriff ausgeführt, wobei das Schreiben jedoch über den Cache erfolgt. In Reaktion auf diese Anforderung durch die CPU **5** gibt der Speichercontroller **10** (im Buszyklus WB2, nach einem einzigen Wartezustand) einen niedrigen Logikpegel auf der Leitung KEN# aus, um anzuzeigen, dass der Burst-Zugriff auf den Hauptspeicher **20** gewährt worden ist, in Kombination mit einem niedrigen Logikpegel auf der Leitung BRDY#, um anzuzeigen, dass am Endes des aktuellen Buszyklus WB2 gültige Daten erwartet werden. Die CPU **5** führt dann das Schreiben aus, indem sie am Ende des Buszyklus WB2 und der folgenden drei Buszyklen WB3 bis WB5 gültige Daten auf dem Datenbus DBUS platziert. Die Burst-Schreiboperation wird dann im letzten Buszyklus WB6 vom Speichercontroller **10**, der die Leitung BRDY# treibt, als abgeschlossen signalisiert.

[0043] Gemäß dieser Ausführungsform wird folglich ein burstfähiger Zugriff auf nicht cachefähige Bereiche des Hauptspeichers in einem mikroprozessorbasierenden Computersystem mit x86-Architektur zur Verfügung gestellt. Die Vorteile der Ausführung von Speicheroperationen mit Burst-Geschwindigkeiten werden folglich für einen weiteren Bereich von Speicheroperationen erzielt, insbesondere beim Zugriff auf Speicherstellen, die nicht cachefähig sind, wie etwa Video-RAM-Speicherstellen. Außerdem wird gemäß dieser Ausführungsform ein solcher Zugriff ermöglicht, ohne dass ein zusätzlicher Stift am Mikroprozessor erforderlich ist.

[0044] Wenn ein zusätzlicher Mikroprozessor-Anschlussstift und eine entsprechende Leiterbahn auf der Träger-Leiterplatte zur Verfügung stehen, könnten verschiedene erfinderische Aspekte gemäß einer zweiten Ausführungsform umgesetzt werden, wie nun mit Bezug auf **Fig.** 7 beschrieben wird. In **Fig.** 7 werden gleiche Bezugszeichen verwendet, um auf gleiche Elemente zu verweisen, die in **Fig.** 2 gezeigt sind.

[0045] **Fig.** 7 veranschaulicht eine CPU **50**, die ebenfalls ein Mikroprozessor der Pentium-Klasse ist, wie er weiter oben mit Bezug auf **Fig.** 2 beschrieben worden ist, wobei sie jedoch in diesem Fall so konfiguriert ist, dass bestimmte Speicherzugriffstypen eine Anforderung eines burstfähigen Zugriffs implizieren. Diese Anforderungen werden für eine Interpretation als Anforderungen eines Burst-Speicherzugriffs (entweder explizit oder implizit) an die Bussteuerlogik **55** im Speichercontroller **60** übertragen.

[0046] Beispielsweise könnte die Ausführung einer blockweisen Speicherzugriffsoperation durch die CPU **50**, wie etwa eines REP MOV Befehls, einen Steuerungsmerker in der Busschnittstelleneinheit **8** setzen, der an die Bussteuerlogik **55** übertragen wird. In diesem Fall, wenn der Steuerungsmerker gesetzt ist, könnte dann die Bussteuerlogik **55** alle nachfolgenden Speicherzugriffsanforderungen (Leitung M/IO# auf hohem Pegel) als Anforderungen von Burst-Zugriffen sowohl auf cachefähige als auch auf nicht cachefähige Bereiche des Speicheradressraums interpretieren. Als eine andere Möglichkeit könnten spezielle Befehle durch die CPU **50** ausführbar sein, die explizit für Burst-Speicherzugriffe sowohl auf cachefähige als auch auf nicht cachefähige Bereiche des Speichers gelten; es würde wiederum ein Merker oder ein Steuersignal von der CPU **50** gesetzt

werden, um dem Speichercontroller **60** zu signalisieren, dass ein Burst-Zugriff angefordert wird.

[0047] Da gemäß dieser zweiten Ausführungsform von der CPU **50** Burst-Speicherzugriffe sowohl für chachefähige als auch für nicht cachefähige Bereiche des Speichers angefordert werden könnten, werden vorzugsweise gesonderte Hinweise zur Cachefähigkeit und Burstfähigkeit vom Speichercontroller **60** an die CPU **50** abgegeben. Dies erfolgt, weil die CPU **50** eine implizierte Anforderung einer Burst-Zugriffsanforderung ausgeben könnte, ohne dass ihr bekannt ist, ob sich die Speicheradresse in einem cachefähigen Bereich des Speicheradressraums befindet; selbst wenn der Speicherzugriff nicht cachefähig ist, wird der Burst-Zugriff weiterhin angestrebt werden. Gemäß dieser Ausführungsform wird deshalb ein Burst-Freigabesignal BEN# geliefert, wobei dieses Signal und ein entsprechender CPU-Anschluss in dieser Ausführungsform zusätzlich zu den herkömmlichen Steuersignalen und Anschlussstiften der Pentiumklasse bereitgestellt werden. Wie in **Fig.** 6 gezeigt ist, gibt entsprechend dieser Ausführungsform der Erfindung die Bussteuerlogik **55** ein Signal auf der Leitung KEN# aus, um die Cachefähigkeit des Speicherzugriffs zu signalisieren (ähnlich wie weiter oben mit Bezug auf die erste Ausführungsform der Erfindung beschrieben worden ist) und gibt außerdem ein Signal auf der Leitung BEN# aus, um zu signalisieren, ob ein Burst-Zugriff gewährt worden ist oder nicht. Falls die CPU einen burstfähigen Zugriff auf eine nicht cachefähige Speicherstelle angefordert hat, wird die Leitung KEN# durch die Bussteuerlogik **55** (die einen nicht cachefähigen Zugriff anzeigt) auf hohem Pegel gehalten und die Leitung BEN# wird durch die Bussteuerlogik **55** auf einen niedrigen Pegel getrieben, um die Gewährung des burstfähigen Zugriffs anzuzeigen und somit die CPU **50** über das Vorliegen oder Erwarten von vier Vierfachwörtern Daten auf dem Datenbus DBUS zu unterrichten.

[0048] Nachdem nun verschiedene Aspekte der vorliegenden Ausführungsformen beschrieben worden sind, die auf nicht cachefähige Daten gerichtet sind, veranschaulicht **Fig.** 7 einen detaillierten Plan noch einer weiteren Ausführungsform, die in der CPU **5** von **Fig.** 1 oder in anderen CPUs enthalten sein könnte und die auf eine Konfiguration gerichtet ist, in welcher die CPU nicht cachefähige Daten für einen bestimmten Zeitraum vorübergehend speichern und modifizieren könnte. Beispielsweise könnte es bei den vorliegenden Ausführungsformen erstrebenswert sein, nicht cachefähige Daten während eines Zeitraums zu speichern und zu modifizieren, der länger als jener ist, der für die Verarbeitung eines einzelnen Befehls gebraucht wird. Um ein weiteres Beispiel zu geben: Einige Befehle, wie etwa der weiter obenerwähnte Befehl REP MOV schlagen naturgemäß eine wiederholte Operation an einem Block von Speicherzellen vor, jedoch könnten andere Befehle in einer Schleife sein, wobei Einzelbefehle in einer solchen Schleife nicht selbst eine Schleife signalisieren und folglich keine Operation an einem Block von Speicherzellen darstellen. Für diese letzteren Befehlstypen könnte es auch erstrebenswert sein, nicht cachefähige Daten für die Dauer eines Teils der Schleife oder während der gesamten Schleife, die diese Befehlstypen enthält, zu speichern und zu modifizieren. So bietet die folgende Ausführungsform Schaltungen, Systeme und Verfahren, um eine derartige Operation zu erzielen.

[0049] In **Fig.** 7, der sich dann zugewandt wird, ist die weiter oben erörterte Busschnittstelleneinheit **8** veranschaulicht, die wiederum sowohl mit den drei Bussen DBUS, ABUS und CBUS gekoppelt ist als auch das Signal CLK von der Taktgeberschaltung **16** empfängt. Unterhalb der Busschnittstelleneinheit **8** sind jedoch zahlreiche zusätzliche Schaltungen, die bisher noch nicht erörtert worden sind (vom L2-Cache **6** verschieden). Was diese zusätzlichen Schaltungen anbetrifft, so ist als Erstes zu beachten, dass zwecks Vereinfachung jeder der Busse DBUS, ABUS und CBUS über die Busschnittstelleneinheit **8** an einen gemeinsamen Bus B gekoppelt gezeigt ist. So ist beabsichtigt, dass der Bus B innerhalb des Mikroprozessors **5** Daten, Adressen oder Steuerinformationen transportieren könnte. Der Bus B ist so angeschlossen, dass er eine Datentransaktionseinheit an ein erstes Register **62** liefert oder eine Datentransaktionseinheit von diesem empfängt.

[0050] In der vorliegenden Erfindung beträgt die Transaktionsbreite des Busses B 8 Bytes. Des Weiteren ist für unten erörterte Zwecke in der bevorzugten Ausführungsform das Register **62** so bemessen, dass bis zu einer Burst-Zeile Daten gespeichert werden; folglich könnte dann das Register **62**, wenn die Burst Size des Busses **32** Bytes beträgt, bis zu 32 Datenbytes speichern. Des Weiteren ist jedoch zu beachten, dass in alternativen Ausführungsformen das Register **62** größer oder kleiner als eine einzelne Burst Size von Daten sein könnte. Der Bus B ist des Weiteren so gekoppelt, dass er eine Adresse an ein zweites Register **64** liefert oder eine Adresse von diesem empfängt, wobei diese Adresse den Daten entspricht, die in das Register **62** einzulesen sind oder aus dem Register **62** zu schreiben sind.

[0051] Außerdem sind den Registern **62** und **64** vorzugsweise drei Anzeiger **66**, **67** und **68** zugeordnet. Der Anzeiger **66** gibt an, ob die Daten im Register **62** aktiv (live) oder abgelaufen (expired) sind, wobei diese Ausdrücke im Folgenden verständlich werden; daher wird der Anzeiger **66** nachfolgend als L/E-Anzeiger **66** bezeichnet. Der Anzeiger **67** gibt an, ob die Daten im Register **62** sauber (clean) oder schmutzig (dirty) sind, und wird daher im Folgenden als C/D-Anzeiger **67** bezeichnet. Die Ausdrücke sauber (clean) und schmutzig (dirty)

werden hier in gleicher Weise wie für Caches verwendet, wobei angegeben wird, ob die Daten modifiziert worden sind, nachdem sie im Register **62** empfangen worden sind. Außerdem sollte aus dem oben Dargestellten in Erinnerung gebracht werden, dass das Datenregister **62** vorzugsweise so bemessen ist, dass 32 Bytes Informationen gespeichert werden. In alternativen Ausführungsformen könnte daher der C/D-Anzeiger **67** ein einzelner Anzeiger für den gesamten Datensatz sein, der im Register **62** gespeichert ist, oder als eine andere Möglichkeit könnte er für jedes Byte gesonderte Angaben enthalten und wird folglich insgesamt 32 Anzeiger, jeweils einen für jedes der 32 potenziell im Register **62** gespeicherten Datenbytes, umfassen. Des Weiteren könnte der C/D-Anzeiger **67** gesonderte Anzeigen für weitere Byte-Gruppierungen wie etwa die Bustransaktionseinheit mit einem Umfang von 8 Bytes oder Gruppen von 4 Bytes wie auch andere Gruppen enthalten. Der Anzeiger **68** gibt an, ob Abschnitte der Daten gültig (valid) oder ungültig (invalid) sind und wird deshalb nachfolgend als V/I-Anzeiger **68** bezeichnet. Die Ausdrücke "gültig" und "ungültig" werden ebenfalls im Sinne von Caches verwendet, d. h., um anzugeben, ob ein bestimmter Abschnitt (z. B. ein Byte) der Daten von einer Schaltung, die diese Informationen liest, als zuverlässig angesehen werden kann. Es wird noch einmal aus dem oben Dargestellten in Erinnerung gerufen, dass das Datenregister **62** vorzugsweise so bemessen ist, dass es 32 Bytes Informationen speichert. In diesem Zusammenhang enthält in der bevorzugten Ausführungsform der V/I-Anzeiger **68** vorzugsweise gesonderte Angaben für jeden Bustransaktionsumfang und wird daher insgesamt vier Anzeiger enthalten, jeweils einen für jede der 8-Byte-Transaktionseinheiten Daten, die im Register **62** gespeichert sind. Jedoch könnte wie im Fall des C/D-Anzeigers **67** der Anzeiger **68** in alternativen Ausführungsformen gesonderte Anzeigen für andere Byte-Gruppierungen, wie etwa für Gruppen von 4 Bytes, einzelne Bytes oder auch für andere Gruppen enthalten. Jeder der Anzeiger **66**, **67** und **68** könnte, wie Fachleuten bekannt ist, auf unterschiedliche Art und Weise dargestellt sein, etwa als ein Bit in gesonderten oder gemeinsamen Registern, als ein Signal auf einer Leiterbahn oder als ein Zustand in einer Zustandsmaschine. Die Definition dieser Anzeigen, d. h. aktiv (live) oder abgelaufen (expired), gültig (valid) oder ungültig (invalid) und schmutzig (dirty) oder sauber (clean) ist weiter unten mit Bezug auf die Funktionsweise der Schaltungen von **Fig.** 7 näher beschrieben.

[0052] Der L/E-Anzeiger **66** ist an eine Aktivität/Abgelaufen-Steuer- und Reaktionsschaltung **70** angeschlossen. Vor allem könnte die Aktivität/Abgelaufen-Steuer- und Reaktionsschaltung **70** in der bevorzugten Ausführungsform den Zustand in den L/E-Anzeiger **66** schreiben und danach denselben Wert für weiter unten erläuterte Zwecke lesen. Außerdem könnte die Aktivität/Abgelaufen-Steuer- und Reaktionsschaltung **70** sowohl eine Adresse vom Bus B als auch die Adresse aus dem Register **64** empfangen. Des Weiteren ist die Aktivität/Abgelaufen-Steuer- und Reaktionsschaltung **70** so angeschlossen, dass sie ein Steuersignal von einem Zähler **72** empfängt. Dieses Steuersignal wird vom Zähler **72** geltend gemacht, wenn er von einem ersten Wert, der von einem Anfangswertregister **74** geliefert wird, bis zu einem Endwert, der von einem Schwellenwertregister **76** geliefert wird, zählt. Es ist zu beachten, dass die Werte in den Registern **74** und **76** auf unterschiedliche Art und Weise gebildet sein könnten, etwa als feste Werte oder indem sie programmierbar gehalten werden. Wenn diese Werte programmierbar sind, könnten sie auf verschiedene Ebenen gesetzt werden, etwa durch den Anwender oder auf einer Ebene unterhalb der Anwenderebene (z. B. durch das Betriebssystem). Was den Zähler **72** betrifft, so könnte er entsprechend verschiedenen bekannten Techniken derart beschaffen sein, dass er von seinem Anfangswert zu seinem Schwellenwert vorrückt. Des Weiteren ist zu beachten, dass angegeben ist, dass der Zähler "vorrückt", womit sowohl ein Zähler, der von einem ersten Wert zu einem zweiten Wert inkrementiert, als auch ein Zähler, der von einem ersten Wert zu einem zweiten Wert dekrementiert, eingeschlossen sein soll. Wie weiter unten ausführlich dargestellt ist, kann die Aktivität/Abgelaufen-Steuer- und Reaktionsschaltung **70** so betrieben werden, dass sie ein HIT/MISS-Signal ausgibt, das sich auf die gespeicherten Daten, ihre Adresse, ihre L/E-Anzeige und gegebenenfalls auf den Zählstand vom Zähler **72** bezieht. Schließlich wird angemerkt, dass die Aktivität/Abgelaufen-Steuer- und Reaktionsschaltung **70** von einem Fachmann unter Verwendung verschiedener bekannter Methoden aufgebaut werden könnte, wenn ihre Funktionalität und die Schaltungsbeziehungen, die weiter unten erörtert werden, verstanden worden sind.

[0053] Die schematische Darstellung von **Fig.** 7 enthält des Weiteren eine C/D-Steuer- und Reaktionsschaltung **78**. Insbesondere könnte die C/D-Steuer- und Reaktionsschaltung **78** in der bevorzugten Ausführungsform den Zustand in den C/D-Anzeiger **67** und in den V/I-Anzeiger **68** schreiben und danach dieselben Werte für weiter unten erläuterte Zwecke lesen. Außerdem könnte die C/D-Steuer- und Reaktionsschaltung **78** die Daten im Register **62** lesen und auch Steuersignale an die beiden Register **62** und **64** liefern, um unter weiter unten beschriebenen Bedingungen zu veranlassen, dass die Daten und die Adresse dieser Register auf den Bus B geschrieben werden. Des Weiteren ist die C/D-Steuer- und Reaktionsschaltung **78** so gekoppelt, dass sie das weiter oben erörterte Steuersignal vom Zähler **72** empfängt, wobei sie optional auch so gekoppelt sein könnte, dass sie sowohl den Zählstand des Zählers **72** als auch den Anfangs- und den Schwellenwert aus dem Register **74** bzw. **76** liest. Schließlich wird angemerkt, dass die C/D-Steuer- und Reaktionsschaltung **78** von einem Fachmann unter Verwendung verschiedener bekannter Methoden aufgebaut werden könnte, wenn ihre

Funktionalität und die Schaltungsbeziehungen, die weiter unten erörtert werden, verstanden worden sind.

[0054] Schließlich veranschaulicht **Fig.** 7 eine Level-2-Cache-Schaltung **6**. Aus Gründen, die weiter unten klarer werden, ist die Level-2-Cache-Schaltung **6** einbezogen, um den Unterschied der Speichertechniken der Schaltungen von **Fig.** 7 hervorzuheben. Kurz, aus dem weiter oben Dargestellten sollte in Erinnerung gebracht werden, dass der Hauptspeicher (siehe z. B. **Fig.** 3) Speicherraum besitzen könnte, der sich in cachefähige und nicht cachefähige Daten unterteilt. Mit Bezug auf **Fig.** 7 könnten Teile der cachefähigen Daten in der Level-2-Cache-Schaltung **6** (oder einer anderen Cache-Struktur) gespeichert sein. Hingegen könnten Teile der nicht cachefähigen Daten im Datenregister **62** gespeichert sein. Sowohl diese Techniken als auch die Vorteile aus der letzteren Speicherung sind weiter unten ausführlich dargestellt.

[0055] **Fig.** 8 veranschaulicht einen Ablaufplan für ein Verfahren **80** mit den verschiedenen praktisch ausführbaren Schritte der Schaltungen von **Fig.** 7. Bevor diese Schritte genau beschrieben werden, wird zunächst angemerkt, dass die Reihenfolge der verschiedenen Schritte im Verfahren **80** nur beispielhaft gegeben ist und ein Fachmann eine solche Reihenfolge leicht verändern könnte, während er sich gleichzeitig noch verschiedene erfinderische Aspekte, die von den vorliegenden Schaltungsanordnungen geboten werden, zu Nutze macht. Außerdem könnten verschiedene der Schritte von **Fig.** 8 auch gleichzeitig während eines gemeinsamen Taktzyklus statt sequenziell ausgeführt werden; es ist jedoch ein sequenzieller Ablauf gezeigt, um die gebotene Erörterung zu vereinfachen.

[0056] In der **Fig.** 8, der sich nun zugewandt wird, beginnt das Verfahren **80** mit dem Schritt **82**, in dem die CPU **5** Daten und eine entsprechende Adresse in das Register **62** bzw. **64** einliest. In der bevorzugten Ausführungsform sind die Daten, die gelesen werden, nicht cachefähige Daten. Außerdem sind diese nicht cachefähigen Daten vorzugsweise eine Burst-Sequenz von Daten. Es sollte aus dem oben Dargestellten, beispielsweise in Verbindung mit **Fig.** 4, in Erinnerung gebracht werden, dass ein Verfahren zum Lesen einer Sequenz von burstfähigen, nicht cachefähigen Daten dargestellt wird. Folglich wird dieses oder ein alternatives Verfahren ausgeführt, um aufeinander folgende Mengen von nicht cachefähigen Daten zu erhalten. Aus der weiter obenbeschriebenen **Fig.** 4 sollte in Erinnerung gebracht werden, dass das Beispiel vier Vierfachwörter (jeweils von 64 Bits oder 8 Bytes) lieferte und folglich der gesamte Burst **32** Bytes umfasst. Wie weiter oben dargelegt ist das Datenregister **62** vorzugsweise so bemessen, dass es bis zu einer Burst-Sequenz Daten speichern kann. Folglich speichert das Register **62** nach dem Schritt **82** alle 32 Bytes Daten. Außerdem sollte in Verbindung mit **Fig.** 4 in Erinnerung gebracht werden, dass auf den Busleitungen A31–A3 eine Adresse ausgegeben wurde, um die Burst-Sequenz Daten zu adressieren. Da diese Adresse den Daten entspricht, wird sie im Register **64** gespeichert. Außerdem ist zu beachten, dass, obwohl die obige Beschreibung eine Burst-Sequenz von Daten bevorzugt, in einer alternativen Operation weniger als ein Burst Daten im Register **62** gespeichert werden könnte (oder mehr als ein Burst, wenn das Register so bemessen ist, dass es mehr als einen einzigen Burst fasst). Wenn weiter oben angegeben worden ist, dass das Register **62** so betrieben werden kann, dass es bis zu einer Burst-Zeile Daten speichern kann, könnte es folglich jetzt klar weniger als einen Burst, etwa nur eine einzige Dateneinheit (z. B. 8 Bytes) speichern. Schließlich ist anzumerken, dass dann, wenn das erste Datum im Register **62** ankommt, der Zähler **72** vorzurücken beginnt, wie sich weiter unten bestätigen wird. Wenn diese Daten als ein Burst ankommen, beginnt der Zähler zudem vorzugsweise beim Empfang des ersten Teils des Bursts, etwa der ersten acht Bytes, vorzurücken.

[0057] Der Schritt **84** initialisiert die Werte der Anzeiger **66**, **67** und **68**. Hier gilt wiederum, dass dieser Schritt in **Fig.** 8 einfach nach dem Schritt **82** gezeigt worden ist, um die gebotene Erörterung zu untergliedern. Folglich könnten die folgenden Aktionen während des gleichen Taktzyklus wie der Schritt **82** ausgeführt werden, wobei die Anzeiger in Reaktion auf die neuen Daten, in vom Register **62** empfangen werden, gesetzt werden. Nun zu den Anzeigern: Der L/E-Anzeiger **66** wird so initialisiert, dass er angibt, dass die im Register jüngst empfangenen Daten aktiv (live) sind, wie wiederum weiter unten ausführlicher erörtert wird. Außerdem wird in einer den Bedeutungen für Caches ähnlichen Art und Weise der C/D-Anzeiger **67** so initialisiert, dass er angibt, dass die jüngst empfangenen Daten im Register **62** sauber (clean) sind, und der V/I-Anzeiger **68** wird so initialisiert, dass er angibt, dass die jüngst empfangenen Daten im Register **62** gültig (valid) sind. Des Weiteren ist zu beachten, dass diese Einstellungen von der Aktivität/Abgelaufen-Steuer- und Reaktionsschaltung **70**, die den neuen Wert in den L/E-Anzeiger **66** schreibt, und von der C/D-Steuer- und Reaktionsschaltung **78**, die den neuen Wert in den CD-Anzeiger **67** und den V/I-Anzeiger **68** schreibt, vorgenommen werden könnten. Diese zwei letzteren Angaben haben wiederum die Bedeutung, die typisch im Zusammenhang mit Caches verwendet wird. Folglich bedeutet die Angabe "sauber" (clean) mit Bezug auf den C/D-Anzeiger **67**, dass die Daten im Register **62** nicht verändert worden sind, seitdem sie in das Register **62** eingelesen worden sind, während umgekehrt eine Angabe, dass die Daten "schmutzig" (dirty) sind, bedeutet, dass sie verändert worden sind, seitdem sie in das Register **62** eingelesen worden sind, und daher für Kohärenzzwecke die aktuellste Form dieser

Daten sind (d. h. die Datenzeile ist in einer Fachleuten bekannten "Zurückschreibbetriebsart" wirksam). Was den V/I-Anzeiger **68** anbetrifft, so bedeutet die Angabe "gültig" (valid), dass die Daten im Register **62** von Schaltungen, die Zugriff auf diese Daten haben, gelesen werden könnten, während umgekehrt dieselben Daten nicht gelesen werden sollten, wenn sie als "ungültig" (invalid) gekennzeichnet sind.

[0058] Der Schritt **86** zielt auf die Anlaufprozedur für den Zähler **72** ab, wobei dieser Schritt wiederum vorzugsweise als Paralleloperation zu den oben angeführten Schritten **82** und **84** ausgeführt wird. Im Besonderen liest der Zähler **72** während des Schritts **86** einen Anfangszählstand aus dem Register **74** aus. Beispielsweise könnte der Wert im Register **74** null sein, wenn der Zähler **72** ein inkrementierender Zähler ist. Tatsächlich ist jedoch weiter zu beachten, dass als eine andere Möglichkeit der Zähler **72** eine Rücksetz-Hardware enthalten könnte, die, wenn sie geltend gemacht wird, den Zähler auf einen vorgegebenen Wert initialisiert, ohne einen Anfangswert aus einem Register oder aus einer anderen Speichervorrichtung auszulesen. Nachdem der Zähler **72** initialisiert worden ist, beginnt der Schritt **86** außerdem das Vorrücken des Zählstandes (wiederum indem das Vorrücken begonnen wird, nachdem das Register **62** die Daten oder einen ersten Teil dieser Daten empfangen hat). Wie weiter oben erwähnt worden ist, wird der Ausdruck "Vorrücken" verwendet, um anzudeuten, dass gemäß den vorliegenden Ausführungsformen der Zähler seinen Zählstand erhöhen oder verringern könnte. Beispielsweise könnte, wenn der Anfangswert null war, wie weiter oben vorgeschlagen worden ist, das Vorrücken des Zählstandes von null aus entweder aufwärts (d. h. inkrementierend) oder abwärts (d. h. dekrementierend) sein. Andererseits könnte das Vorrücken des Zählstandes, wenn der Anfangswert irgendeine andere Zahl wäre, ebenso gut von diesem Wert aus aufwärts oder abwärts erfolgen. Des Weiteren ist zu beachten, dass auch der Schwellenwert fest sein könnte, während der Anfangswert variabel wäre, wie etwa in dem Fall, in dem ein Anfangswert eine positive Zahl in einem Register wäre und der Zähler **72** auf einen Schwellenwert von null dekrementiert werden würde. In diesem Fall könnte wiederum der Schwellenwert im Zähler **72** fest sein, ohne dass eine gesonderte Speichervorrichtung wie etwa das Register **76** erforderlich ist. In der Tat könnten von einem Fachmann noch weitere Methoden in Erwägung gezogen werden, wobei jedoch solche Alternativen schwieriger umzusetzen sein könnten. Ungeachtet des Vorrückverfahrens des Zählers **72** ist weiter zu beachten, dass der Zählstand auf eine zeitliche Aktivität reagiert. Beispielsweise rückt in der obenbeschriebenen Ausführungsform der Zähler **72** in Reaktion auf jeden Taktzyklus vor, wobei dieser Taktzyklus von einem Taktgeber **16** geliefert wird oder von diesem abgeleitet wird. Insbesondere enthält die CPU **5**, wie Fachleuten für Mikroprozessoren bekannt ist, typisch einen Kern, der mit einer Taktgeschwindigkeit arbeitet, und einen Hauptbus, der mit einer anderen Taktgeschwindigkeit arbeiten könnte. Diese Taktzyklen stimmen entweder mit den Taktzyklen vom Taktgeber **16** überein oder sind (entweder direkt oder als Vielfache) von diesem abgeleitet. Auf jeden Fall rückt der Zähler **72** bei einer gegebenen Mikroprozessortaktzeit der Ausführungsstufe vorzugsweise in Reaktion auf jeden Taktzyklus vor. Als eine andere Möglichkeit der Ausführung könnte jedoch der Zähler **72** in Reaktion auf irgendein anderes Signal vorrücken, das während einer Zeit, nachdem die Daten in das Register **62** eingelesen worden sind und der Zähler initialisiert worden ist, auftritt. Beispielsweise könnte der Zähler **72** jedes Mal dann vorrücken, wenn eine Transaktion über den DBUS (der in **Fig.** 7 im Bus B eingeschlossen ist) stattfindet. Folglich rückt der Zähler **72** für jene Taktzyklen, in denen keine Daten auf dem DBUS vorliegen, nicht vor, während er hingegen für jene Taktzyklen, in denen Daten auf dem DBUS vorliegen, in Reaktion auf jeden derartigen Zyklus, einen Schritt vorrückt.

[0059] Der Schritt **88** veranlasst ein Warten auf der Grundlage des vorrückenden Zählstandes des Zählers **72**. Insbesondere bewertet der Schritt **88** ständig, ob der Zählstand des Zählers **72** niedriger als der Schwellenwert im Register **76** ist. Wenn nicht, kehrt der Ablauf zum Schritt **88** zurück, um die Bewertung fortzusetzen. Wenn der Zählstand den Schwellenwert erreicht, wird das Verfahren **80** jedoch mit dem Schritt **90** fortgesetzt. Es ist außerdem zu beachten, dass der Schritt **88** angibt, dass die Anfrage eine "kleiner als"-Bedingung zur Grundlage hat, die selbstverständlich in "kleiner als oder gleich" verändert werden könnte, so dass der Ablauf nur dann mit dem Schritt **90** fortgesetzt wird, wenn der Zählstand den Schwellenwert übersteigt. Zum anderen könnte festgelegt werden, dass der Ablauf mit dem Schritt **90** fortgesetzt wird, wenn der Zählstand einen Schwellenwert erreicht, wobei der Schwellenwert der Wert im Register **76** ist, wenn die "kleiner als"-Bedingung verwendet wird, oder der Schwellenwert der Wert im Register **76** plus eins ist, wenn die "kleiner als oder gleich"-Bedingung verwendet wird.

[0060] Der Schritt **90**, der erreicht worden ist, nachdem der Zählstand des Zählers **76** den Schwellenwert im Register **76** erreicht hat, ändert den Zustand des L/E-Anzeigers **66** von aktiv (live) in abgelaufen (expired). Diese Operation könnte wiederum durch die Aktivität/Abgelaufen-Steuer- und Reaktionsschaltung **70** ausgeführt werden, die den neuen Wert in den L/E-Anzeiger **66** schreibt. Bei den so weit gegebenen Operationen ist weiter zu beachten, dass zwischen den Schritten **84** und **90** die Speicherung der Daten und Adresse für eine cacheartige Speicherung von ansonsten nicht cachefähigen Daten sorgt. Insbesondere signalisiert der Anzeiger **66**, bevor der Zählstand die Schwelle erreicht, dass die Daten aktiv sind. Diese Angabe signalisiert jeder an-

fordernden Schaltung, dass es während des aktiven Zustands zulässig ist, die Daten im Register **62** entweder zu lesen oder zu schreiben. Selbstverständlich ist diese Angabe auch unter Berücksichtigung der beiden anderen Anzeiger **67** und **68** zu sehen. Es sei beispielsweise angenommen, dass ein vollständiger Burst von Daten im Register **62** gespeichert ist (d. h. dass der Anzeiger **68** gültig signalisiert), dass der Zählstand seinen Schwellenwert nicht erreicht hat (d. h. dass der Anzeiger **66** aktiv signalisiert) und dass die Daten seit ihrem Empfang nicht modifiziert worden sind (d. h. dass der Anzeiger **67** sauber signalisiert). Folglich könnte wie bei einer Cache-Konfiguration eine auf dem Bus B platzierte Adresse, womit versucht wird, die entsprechenden Daten zu lesen, die Daten im Register **62** adressieren. Insbesondere wird eine solche Adresse von der Aktivität/Abgelaufen-Steuer- und Reaktionsschaltung **70** empfangen. Die Schaltung **70** ermittelt dann, ob die Adresse auf dem Bus B mit der Adresse im Register **64** übereinstimmt und ob die Daten aktiv und gültig sind; wenn diese Bedingungen erfüllt sind, dann gibt die Schaltung **70** in der An einer Cacheschaltung ein Treffersignal (HIT) aus. Folglich könnte die Schaltung, welche die Adresse ausgegeben hat, dann in Reaktion auf das HIT-Signal die Daten aus dem Register **62** auslesen. Außerdem ist in einem ähnlichen Zusammenhang ebenfalls zu beachten, dass von der Schaltung **70** auch eine Snoop-Adresse in ähnlicher Weise empfangen werden könnte. Wie Fachleuten bekannt ist, ermöglicht ein Snooping Cachestrukturen, eine ordnungsgemäße Speicherkohärenz aufrechtzuerhalten. Folglich könnte die Schaltung **70** die Snoop-Adresse mit der Adresse im Register **64** vergleichen und ein HIT-Signal ausgeben, wenn die entsprechenden Daten aktiv und gültig sind. Des Weiteren könnten dann, wenn die Snoop-Anforderung außerdem eine Konvertierungsoperation erfordert, die entsprechenden Daten auf den Bus B geschrieben werden und folglich in Reaktion auf die Snoop-Anforderung verfügbar sein. In diesem Fall würde der CD-Anzeiger **67** ebenfalls auf sauber gesetzt werden, da die Kopie der Daten im Register **62** folglich nicht länger die einzige Kopie dieser Informationen ist. In noch einem weiteren Beispiel könnte die Snoop-Anforderung zusätzlich zu einer Konvertierung der Daten auch ein Ungültigmachen erfordern. Folglich würde zusätzlich zur Ausgabe der Daten der V/I-Anzeiger **68** auf ungültig gesetzt werden. Sowohl diese als auch weitere Snoop-Verfahren, wie sie Fachleuten beispielsweise aus dem Abschnitt 8.3 des Dokuments von Hennessy und Patterson: "Computer Architecture A Quantitative Approach", 2. Aufl., 1996, Verlag Morgan Kaufmann Publishers Inc., bekannt sind, wobei dieser Abschnitt den Titel "Centralized Shared-Memory Architectures" trägt, könnten daher eingeschlossen sein.

[0061] Zusätzlich zu dem weiter oben Dargestellten ist auf Grund der beschriebenen cacheartigen Operation weiter zu beachten, dass die Daten im Register **62**, während sie aktiv sind, auch wieder in der cacheartigen Weise modifiziert werden könnten. Folglich bestimmt dann, wenn eine Adresse, die Informationen zu schreiben versucht, auf dem Bus B platziert ist, wiederum die Schaltung **70**, ob diese Adresse mit der Adresse im Register **64** übereinstimmt und ermittelt dann, wenn das der Fall ist, weiter, ob die Daten im Register **62** aktiv sind. Wenn ja, könnten die Daten im Register **62** modifiziert werden. Außerdem erfasst in einem solchen Fall die Schaltung **78** die Veränderung an den Daten und ändert den Zustand des C/D-Anzeigers **67** von sauber auf schmutzig. Die vorliegenden Ausführungsformen könnten sowohl diese als auch weitere cacheartige Operationen einschließen.

[0062] Die zwei unmittelbar vorausgehenden Abschnitte haben zwar eine cacheartige Operation für ausgewählte Schaltungen der **Fig.** 7 beschrieben, trotzdem ist weiter zu beachten, dass diese Operation eine "Aktivitätsdauer" aufweist, die durch den Zählstand im Zähler **72** definiert ist. Insbesondere werden während des Schritts **90** die Daten als abgelaufen gekennzeichnet, und daher wird nach dieser Aktion die Schaltung **70** immer einen Fehltreffer signalisieren. Mit anderen Worten: Selbst wenn eine Adresse (entweder eine Busadresse für ein Lesen oder Schreiben vom Bus B oder eine Snoop-Adresse) an die Schaltung **70** ausgegeben wird und diese mit der Adresse im Register **64** übereinstimmt, wird die Schaltung **70** erfassen, dass die Daten abgelaufen sind, da der Zählstand im Zähler **72** seinen entsprechenden Schwellenwert erreicht hat. Folglich sind die Daten nur während der "Aktivitätsdauer", zwischen dem Anfangszählstand und dem Schwellenwertzählstand, in einer cacheartigen An und Weise verfügbar. Es ist jedoch weiter zu beachten, dass diese Operation nur deshalb als cacheartig dargestellt worden ist, weil sie jenen Operationen typischer Cache-Strukturen ähnlich ist. Da die Daten jedoch definitionsgemäß nicht cachefähig sind (wie vom Hauptspeicher signalisiert wird), sind sie nicht in weiteren Cache-Strukturen wie etwa dem Level-2-Cache **6** oder anderen Caches innerhalb oder außerhalb der CPU **5** vorhanden.

[0063] Im Schritt **92**, dem sich nun zugewandt wird, wird ermittelt, ob der C/D-Anzeiger **67** entsprechend der Gültigkeit der Daten sauber oder schmutzig ist. Wenn der Anzeiger angibt, dass die Daten im Register **62** sauber (und auch gültig) sind, wird das Verfahren **80** mit dem Schritt **96** fortgesetzt. Andererseits gibt dann, wenn der C/D-Anzeiger **67** angibt, dass die Daten im Register **62** schmutzig sind, die Schaltung **78** im Schritt **94** entsprechende Steuersignale aus, so dass die Daten im Register **62** entsprechend der im Register **64** gespeicherten Adresse in den Hauptspeicher (oder irgendeine andere Speicherstruktur höherer Ebene) geschrieben werden, wodurch eine ordnungsgemäße Speicherkohärenz sichergestellt wird. Es ist weiter zu beachten, dass der

Schritt **92** in alternativen Ausführungsformen verändert oder weggelassen werden könnte. Während beispielsweise das Verfahren **80** veranschaulicht, dass schmutzige Daten, nur nachdem der Zählstand seinen Schwellenwert erreicht hat, in den Hauptspeicher geschrieben werden, ist weiter zu beachten, dass die C/D-Steuer- und Reaktionsschaltung **78** auch zu einem beliebigen Zeitpunkt gleichzeitig den Anfangswert, den Schwellenwert und den Zählerwert lesen könnte. Folglich könnte die Schaltung **78** auf der Grundlage dieser Werte den C/D-Anzeiger **67** bewerten, bevor der Zählstand den Schwellenwert erreicht, und wenn die Daten schmutzig sind, entsprechende Steuersignale ausgeben, um zu bewirken, dass die Daten in den Hauptspeicher geschrieben werden, bevor der Zählstand den Schwellenwert erreicht. Folglich könnten auf der Grundlage einer gewissen zeitlichen Mittelung oder dergleichen zwischen dem Anfangswert und dem Schwellenwert des Zählers periodische Aktualisierungen des Hauptspeichers vorgenommen werden. Als ein weiteres Beispiel für eine Modifikation des oben Dargestellten ergibt sich die Idee des Zurückschreibens von schmutzigen Informationen in einen Speicher höherer Ebene im Kontext der von Fachleuten als Zurückschreiben bezeichneten Methode, d. h. einer Methode, bei welcher die Daten zu Anfang in eine Cachestruktur (oder eine cacheartige Struktur) geschrieben werden und später ausgegeben werden, um eine Speicherstruktur höherer Ebene zu aktualisieren. In einer alternativen Ausführungsform könnte deshalb die bekannte Durchschreib-Methode verwendet werden. In einem solchen Fall wird jedes Mal, wenn eine Modifikation an den schon im Register **62** gespeicherten Daten erfolgt, diese auch in den Hauptspeicher geschrieben. In einem solchen Fall ist es nicht erforderlich, einen schmutzig versus sauber angebenden Anzeiger zu haben, da die Daten definitionsgemäß immer sauber sind (denn es gibt immer eine Kopie der gleichen Informationen, die in einem Speicher höherer Ebene zur Verfügung steht). Tatsächlich besteht, nachdem die Alternative des Durchschreibens und Zurückschreibens gegeben worden ist, noch eine weitere Methode der vorliegenden Ausführungsform darin, die Auswahl dynamisch veränderbar zu machen. Beispielsweise könnte ein Freigabesignal verwendet werden. In diesem Fall werden dann, wenn das Signal in einem ersten Zustand ist, die Daten in einer Zurückschreib-Betriebsart geschrieben, weshalb die Beibehaltung des C/D-Anzeigers **67** und seine Signalisierung in Bezug auf diese Daten erforderlich ist. Im umgekehrten Fall, wenn sich das Signal in einem zweiten Zustand befindet, werden die Daten in einer Durchschreib-Betriebsart geschrieben, wodurch die Notwendigkeit der Beibehaltung des C/D-Anzeigers **67** oder der Reaktion auf die Signalisierung dieses in Bezug auf diese Daten aufgehoben wird. Wenn eine geeignete Methode verwendet wird, um die Kohärenz des Speichers infolge einer Modifikation der Daten im Register **62** zu gewährleisten, wird schließlich das Verfahren **80** mit dem Schritt **96** fortgesetzt.

[0064] Der Schritt **96** gibt lediglich ein Ende des Ablaufplans für das besondere Beispiel von Daten und ihrer entsprechenden Adresse an. Jedoch könnte nach dem Schritt **96** das Verfahren **80** mit Bezug auf andere nicht cachefähige, in das Register **62** geschriebene Daten wiederholt werden, wobei dann die Schritte des Verfahrens **80** in der obenbeschriebenen An und Weise wiederholt werden.

[0065] **Fig.** 9 veranschaulicht eine zur **Fig.** 7 alternative Ausführungsform, bei der viele gleichartige Konzepte übertragen und reproduziert worden sind, um zahlreiche voneinander unabhängige Sätze von Daten und Adressen zu speichern. Um die Aspekte, die mit den **Fig.** 6 und 7 gemeinsam sind, zu vereinfachen wurden folglich auf die meisten Beispiele die gleichen Bezugszeichen übertragen, wobei jedoch untere Indizes hinzugefügt oder die Bezugszeichen verändert wurden, um diese veränderten Merkmale hervorzuheben, die jeweils weiter unten erörtert werden.

[0066] Die Ausführung von **Fig.** 9 ermöglicht die Speicherung von drei verschiedenen Datenzeilen in den entsprechenden Registern $\mathbf{62_1}$, $\mathbf{62_2}$ und $\mathbf{62_3}$. Jede dieser Zeilen kann wiederum bevorzugt so betrieben werden, dass sie bis zu einer Burst-Zeile Daten (z. B. 32 Bytes) speichert. Außerdem hat jede Zeile Daten eine entsprechende Adresse, die in einem entsprechenden Register $\mathbf{64_1}$, $\mathbf{64_2}$ und $\mathbf{64_3}$ gespeichert ist. In der Tat ist zu beachten, dass dort, wo mehrere Speichereinheiten verwendet werden, wie in **Fig.** 9 gezeigt ist, eine Methode die Verwendung der vorhandenen Mikroprozessor-Speicherschaltungen ist. Beispielsweise enthält ein Mikroprozessor oftmals einen Vorverarbeitungspuffer, der in der Lage ist, Zeilen von Daten und die entsprechenden Adressen zu speichern. Folglich ist ein Verfahren zur Implementierung der Mehrfachdaten- und Adressregister von **Fig.** 9, die Verantwortlichkeit des Befehlsvorverarbeitungspuffers zusammen mit der in diesem Dokument beschriebenen Operation zu verwenden. Mit anderen Worten: Jeder Speicherzeile könnte ein Freigabesignal zugeordnet sein. Wenn das Freigabesignal in einem ersten Zustand ist, wird die Zeile gemäß der im Voraus festgelegten Vorverarbeitungsfunktion des Mikroprozessors betrieben. Wenn jedoch das Freigabesignal in einem zweiten Zustand ist, wird die Zeile gemäß den vorliegenden Ausführungsformen betrieben. Für mehr Informationen über eine erfinderische Vorverarbeitungs-Pufferanordnung könnte der Leser die vorläufige US-Patentanmeldung Nr. 60/024,860 (Anwalts-Aktenzeichen TI-18851P) mit dem Titel "Microprocessor Circuit, Systems, And Methods Using A Combined Writeback Queue And Victim Cache", eingereicht am 28. August 1996, einsehen, die hiermit durch die Bezugnahme Bestandteil dieses Patents wird.

[0067] Zurück zu jeder Zeile von Informationen, die in **Fig.** 9 im Sinne der vorliegenden Ausführungsformen bereitgestellt wird: Jede solche Zeile enthält weiter einen entsprechenden L/E-Anzeiger $\mathbf{66}_1$, $\mathbf{66}_2$ und $\mathbf{66}_3$, einen entsprechenden C/D-Anzeiger $\mathbf{67}_1$, $\mathbf{67}_2$ und $\mathbf{67}_3$ sowie eine entsprechenden V/I-Anzeiger $\mathbf{68}_1$, $\mathbf{68}_2$ und $\mathbf{68}_3$. Jedes der obigen Register und jeder der obigen Zeiger ist wiederum an Steuerschaltungen gekoppelt, wobei diese eine Aktivität/Abgelaufen-Steuer- und Reaktionsschaltung **70** und eine C/D-Steuer- und Reaktionsschaltung **78** einschließen. In diesem Beispiel umfassen die Schaltungen **70** und **78** jedoch zusätzliche Hardware, um sowohl die mehreren voneinander unabhängigen Daten- und Adressleitungen als auch ihre entsprechenden Anzeiger anzupassen. Schließlich hat jede Leitung auch einen entsprechenden Zähler $\mathbf{72}_1$, $\mathbf{72}_2$, bzw. $\mathbf{72}_3$. Was die Zähler anbetrifft, so ist jedoch zu beachten, dass die Zähler statt auf ein Anfangswert- und ein Schwellenwertregister **74** und **76** auf eine Nachschlagtabelle **98** Zugriff haben. Jeder Eintrag in der Nachschlagtabelle schließt eine Adresse und sowohl einen entsprechenden Anfangswert als auch einen Schwellenwert ein. Die Funktionalität der Nachschlagtabelle **98** ist weiter unten ausführlicher dargestellt.

[0068] Die Funktionsweise der Schaltung von **Fig.** 9 ist im Großen und Ganzen dem weiter oben erörterten Verfahren **80** von **Fig.** 8 gleich. Jedoch werden die gleichen Schritte, die weiter oben mit Bezug auf das Verfahren **80** dargelegt worden sind, vorzugsweise unabhängig in Bezug auf jede der verschiedenen Leitungen (d. h. sowohl für Daten als auch für ihre entsprechende Adresse, ihre entsprechenden Anzeiger und ihren entsprechenden Zähler) ausgeführt. Folglich wird der Leser für eine umfassende Beschreibung auf die oben gegebene Erörterung von **Fig.** 8 verwiesen. An diesem Punkt wird für einen Fachmann verständlich sein, dass jeder Zeile Daten und der entsprechenden Adresse eine Aktivitätsdauer auf Grund des entsprechenden Zählers zugeordnet ist und die Daten während der Aktivitätsdauer gelesen oder geschrieben werden könnten, wobei die Speicherkohärenz entweder während dieser Aktivitätsdauer oder am Ende dieser Aktivitätsdauer sichergestellt wird. Der einzige weitere Unterschied tritt im Zusammenhang mit der Nachschlagtabelle **98** auf, weshalb dieses Konzept unmittelbar folgend angesprochen wird.

[0069] Auf die Nachschlagtabelle **98** wird in Verbindung mit dem Schritt **84** von **Fig.** 8 zugegriffen. Es sollte in Erinnerung gebracht werden, dass der Schritt **84** die Werte der Anzeiger **66** und **67** initialisiert, wobei weiter oben gezeigt worden ist, wie ein solches Verfahren in Verbindung mit den Registern **74** und **76** vorkommen könnte. In der alternativen Ausführungsform von **Fig.** 10, die auch für eine Implementierung einer einzelnen Zeile umgesetzt werden könnte, wie etwa in **Fig.** 7, liefert die Nachschlagtabelle **98** die Anfangs- und Schwellenwerte für jeden der entsprechenden Zähler $\mathbf{72}_1$ bis $\mathbf{72}_3$. Insbesondere wird jedes Mal, wenn eine Adresse in einem Register **64** empfangen wird, diese Adresse in der Nachschlagtabelle **98** gesucht. Vorausgesetzt, die Adresse wird dann in der Tabelle **98** ausfindig gemacht, wird auch ein entsprechender Eintrag sowohl für den Anfangs- als auch für den Schwellenwert angetroffen, die den entsprechenden Daten in einem Register **62** zuzuordnen sind. Falls die Adresse nicht gefunden wird, könnten sowohl für den Anfangs- als auch für den Schwellenwert vorgegebene Standardwerte verwendet werden oder es könnte ein Fehler auftreten. Bei dieser Ausführung könnten deshalb die Werte in der Nachschlagtabelle **98** fest, programmiert und/oder dynamisch verändert sein, so dass sie von einer Gruppe adressierbarer Daten zur nächsten verschieden sind. Des Weiteren könnte, obwohl die Nachschlagtabelle **98** als sowohl einen Anfangswert als auch einen Schwellenwert für jede Adresse besitzend dargestellt wird, wie weiter oben angegeben ist, jeder dieser Werte festverdrahtet programmiert sein, um auf einen Anfangswert oder einen festen Schwellenwert von null zurückzusetzen und auf diesen festen Schwellenwert zu dekrementieren. Folglich braucht, wenn einer der beiden Werte fest ist, dieser Wert nicht in der Nachschlagtabelle **98** enthalten zu sein; in diesem Fall würde folglich die Tabelle **98** nur den jeder Adresse entsprechenden nicht festen Wert speichern.

[0070] Die oben abgeführten Aspekte der **Fig.** 7 bis 9 vorausgesetzt, ist es außerdem aufschlussreich, ein Anwendungsbeispiel für diese Aspekte in einem Systemkontext zu zeigen. Diesbezüglich ist anzumerken, dass bei der Graphikverarbeitung oftmals von einer von der CPU unabhängigen Schaltungsanordnung ein autonomer Prozess an Punktrasterdaten, die in einem Bildwiederholspeicher oder dergleichen gespeichert sind, ausgeführt wird. Folglich ändert diese Graphikverarbeitung häufig die Daten, weshalb diese Daten als nicht cachefähig gekennzeichnet sind, weil ansonsten die Gefahr bestünde, dass die Daten von dem unabhängigen Graphik-Beschleuniger verändert werden könnten, während sie auch in einen Cache der CPU **5** gespeichert werden. Angesichts der obigen Ausführungsformen der **Fig.** 6 bis 8 ist jedoch nun zu beachten, dass die Aktivitätsdauer eines Datums begrenzt werden könnte, um eine solche Gefahr zu minimieren. Angenommen, es ist empirisch ermittelt worden, dass der autonome Prozess die Daten nur alle 64 Taktzyklen verändern wird. In diesem Fall könnte der Schwellenwert, der in einem Schwellenwertregister **76** gespeichert wird (oder ein Eintrag in einer Nachschlagtabelle **98**) auf einen Wert festgelegt werden, der weniger als 64 Taktzyklen entspricht. Das hat zur Folge, dass, obwohl diese nicht cachefähigen Daten des Punktrasterdaten-Bildwiederholspeichers im Register **62** gespeichert werden würden, jede Veränderung an ihnen während ihrer Aktivitätsdauer erfolgen müsste und folglich wahrscheinlich stattfinden würde, bevor dieselben Daten durch die autonome Graphikver-

arbeitung eine Änderung erfahren würden. Außerdem sind während dieser Aktivitätsdauer die Daten auch einfach für Leseoperationen durch die CPU **5** ohne Zugriff auf den Hauptspeicher und daher ohne die Verzögerung zu erfahren, die für einen derartigen Zugriff erforderlich wäre, verfügbar. Folglich werden dort, wo derartige Zugriffe erforderlich wären, klare Vorteile gegenüber dem Stand der Technik geboten.

[0071] **Fig.** 10 veranschaulicht eine zu **Fig.** 7 alternative Ausführungsform, auf welche verschiedene der weiter oben erörterten Konzepte mit zusätzlichen Modifikationen innerhalb des Geltungsbereichs der Erfindung übertragen worden sind. Folglich wird wiederum in allgemeiner Weise ein Mikroprozessor **5** spezifiziert, der eine Busschnittstelleneinheit **8** aufweist, die sowohl drei externe Bussignale (d. h. DBUS, ABUS und CBUS) als auch ein Signal CLK empfängt. Wiederum zur Vereinfachung ist jeder der Busse DBUS, ABUS und CBUS über die Busschnittstelleneinheit **8** an einen gemeinsamen Bus B angeschlossen dargestellt. Einführend sei angemerkt, dass die Ausführungsform von **Fig.** 10 zeigt, wie mehrere der vorliegenden erfinderischen Konzepte weiter kombiniert werden können, wobei eine vorhandene Cache-Struktur modifiziert wird. Beispielsweise könnten die in **Fig.** 10 unter dem Bus B gezeigten Schaltungen in einer Cache-Struktur, etwa in dem in verschiedenen der oben angeführten Figuren gezeigten L2-Cache **6**, enthalten sein, oder dieser Struktur zugeordnet sein. Durch das Zuordnen der vorliegenden Ausführungsform zu einem vorhandenen Cache verringert sich bei einer solchen Methode die Komplexität der Ausführung, etwa der Datenpfad, da kein gesonderter Datenpfad zu einer Struktur ausgeführt werden muss, die von einem vorhandenen Cache vollkommen unabhängig ist. Nachdem diese Einführung gegeben worden ist, stellt die folgende Erörterung sowohl die verschiedenen Schaltungen als auch ihre Beziehung zu den vorliegenden Ausführungsformen wie auch zu derzeitigen Cache-Strukturen dar.

[0072] Die Schaltungen von **Fig.** 10 enthalten einen Tag-Speicher **100**, der einem Datenspeicher **102** zugeordnet ist. Im Allgemeinen ist die Zuordnung eines Tag-Speichers zu einem Datenspeicher Fachleuten wohl bekannt, wobei der Tag-Speicher Informationen speichert, die einen entsprechenden Eintrag in den Datenspeicher betreffen. Der Tag-Speicher **100** und der Datenspeicher **102** weisen eine übereinstimmende Anzahl von Reihen auf, wie in **Fig.** 10 durch eine ganze Zahl N veranschaulicht ist. Was den Tag-Speicher **100** anbetrifft, so enthält jede seiner Reihen drei Abschnitte, wie Fachleuten bekannt ist, und könnte auch weitere enthalten. Was die drei bekannten Abschnitte anbelangt, die gezeigt sind, so enthält der Tag-Speicher **100** eine Adresse der entsprechenden Daten im Datenspeicher **102**, einen Anzeiger, der angibt, ob die entsprechenden Daten sauber oder schmutzig sind (d. h. als ein C/D-Anzeiger dargestellt), und einen Anzeiger, der angibt, ob die entsprechenden Daten gültig oder ungültig sind (d. h. als ein V/I-Anzeiger dargestellt). Wie bei den obigen Ausführungsformen könnten die Anzeiger durch ein Signal oder Bit repräsentiert sein und könnten für den gesamten Umfang eines Dateneintrags gelten oder es könnten mehrere Anzeiger für Teile des Dateneintrags enthalten sein (z. B. pro Byte, für mehrere Bytes, die Burst Size und so fort). Schließlich könnten, um ein entsprechendes HIT/MISS-Signal zu liefern, wie weiter unten gezeigt ist, die Adressen aus dem Tag-Speicher **100** gelesen werden, wobei sowohl jeder der C/D-Anzeiger als jeder der V/I-Anzeiger durch eine C/D- und V/I-Steuer- und Reaktionsschaltung **104** sowohl gelesen als auch geschrieben werden könnte.

[0073] Zusätzlich zu den bekannten Elementen umfasst der Tag-Speicher **100** auch wenigstens zwei weitere Aspekte, wodurch ein System im Rahmen der vorliegenden erfinderischen Ausführungsformen geschaffen wird. Als ein erster Aspekt könnte zusätzlich zu Anzeige der Gültigkeit für Zwecke, die Fachleuten bekannt sind, jeder V/I-Anzeiger auch in Reaktion auf eine Zählerbewertung und die L/E-Steuerschaltung **106** auf ungültig gesetzt werden. Wie im Folgenden besser verständlich wird, wird deshalb, wenn die Aktivitätsdauer, falls es eine für die Daten, die dem Tag-Speicher-Eintrag entsprechen, gibt, ihren Schwellenwert erreicht, der V/I-Anzeiger auf ungültig gesetzt. Als ein zweiter Aspekt enthält eine Anzahl von Reihen im Tag-Speicher **100**, die in der bevorzugten Ausführungsform alle N Reihen umfasst, zusätzlich zu den drei übrigen Abschnitten, die weiter oben beschrieben worden sind, einen Zähleridentifikationsabschnitt (CTR ID abgekürzt). Die Funktionalität der CTR ID-Anzeiger ist weiter unten ausführlich beschrieben.

[0074] Um die verbleibenden Verbindungen von **Fig.** 10 zu vervollständigen: In dem Beispiel von **Fig.** 10 sind mehrere Zähler enthalten, wobei drei Zähler als Zähler **1** bis Zähler **3** bezeichnet sind. Es ist zu beachten, dass die Anzahl der Zähler verändert werden könnte, wobei sie jedoch in der vorliegenden Ausführungsform bevorzugt erheblich kleiner als die Anzahl N der Reihen im Datenspeicher **100** ist. Jeder der Zähler ist so angeschlossen, dass er das Signal CLK empfängt, so dass er, wie in den Ausführungsformen der **Fig.** 7 bis 8, weiter oben, aufeinander folgende Taktzyklen zählt. Als eine andere Möglichkeit könnte jedoch, wie weiter oben erwähnt worden ist, wenigstens einer der Zähler so gekoppelt sein, dass er andere aufeinander folgende Ereignisse zählt, etwa Bustransaktionen oder andere von einem Fachmann bestimmbare Ereignisse. Jeder der Zähler ist außerdem an eine Nachschlagtabelle **108** angeschlossen, welche die gleiche Funktionalität erfüllt, wie sie weiter oben in Verbindung mit der Nachschlagtabelle **98** von **Fig.** 9 beschrieben worden ist. Folglich wird

für einen Fachmann nachzuvollziehen sein, dass die Nachschlagtabelle **108** in Reaktion auf die Adresse der Daten einen Anfangswert und/oder einen Schwellenwert an jeden entsprechenden Zähler liefert, wodurch die Aktivitätsdauer der dem Zähler zugeordneten Daten definiert wird, wie weiter unten beschrieben ist. Außerdem sollte aus den vorhergehenden Erörterungen in Erinnerung gebracht werden, dass die vorliegenden Ausführungsformen Alternativen zu einer Nachschlagtabelleneingabe in jeden Zähler vorsehen, wie etwa einen festen Wert für den Anfangs- und/oder Schwellenwert, wobei dieser feste Wert zu dem Zähler festverdrahtet programmiert sein könnte oder von einem Register oder dergleichen geliefert werden könnte.

[0075] Die Wirkungsweise der Schaltungen der **Fig.** 10 hat einige Gemeinsamkeiten mit verschiedenen Ausführungsformen, die im Zusammenhang mit vorhergehenden Figuren beschrieben worden sind, weshalb angenommen wird, dass der Leser die an früherer Stelle beschriebenen Ausführungsformen versteht, so dass einige der Einzelheiten weiter unten nicht erneut dargelegt werden müssen. In einem ersten Sinn wirken der Tag-Speicher **100** und der Datenspeicher **102** gemäß bekannter Verfahren. Deshalb wird eine Adresse auf dem Bus B zum Tag-Speicher **100** gekoppelt und unter der Voraussetzung, dass die Adresse mit einer Adresse im Tag-Speicher **100** übereinstimmt, die gültigen Daten entspricht, gibt die C/D- und V/I-Steuer- und Reaktionsschaltung **104** ein HIT-Signal aus, von dem ab die entsprechende Aktion durchgeführt werden könnte (d. h. in Abhängigkeit von der Anforderung, die der Adresse entspricht, wie etwa das Lesen der Daten, das Aktualisieren der Daten oder das Reagieren auf eine Art Snoop).

[0076] Es ist zu beachten, dass zusätzlich zu der bekannten Funktionalität des Tag-Speichers **100** die Ausführungsformen von **Fig.** 10 weiter eine erfinderische Operation enthalten, die ermöglicht, den Daten im Datenspeicher **102** eine Aktivitätsdauer zuzuordnen. Es sollte insbesondere in Erinnerung gebracht werden, dass jede Reihe im Tag-Speicher **100** eine Zähleridentifikation CTR ID enthält. Es ist nun zu beachten, dass eine CTR ID für eine bestimmte Reihe so genutzt werden kann, dass sie eine Identifikation für jeden der Zähler **1** bis **3** speichert, wobei eine solche Identifikation zur Folge hat, dass der identifizierte Zähler den Daten für diese Zeile entspricht. Beispielsweise sei angenommen, dass die CTR ID_1, die den $Daten_1$ entspricht, den Zähler **2** identifiziert. Deshalb sorgt der Zähler **2** für eine Aktivitätsdauer wie weiter oben eingeführt in Bezug auf die $Daten_1$. Folglich sind, solange der Zählstand des Zählers **2** seinen Schwellenwert noch nicht erreicht hat, die $Daten_1$ nicht abgelaufen. Außerdem ist zu beachten, dass die Ausführungsform der CTR ID von 0 auch ermöglicht, demselben Zähler mehr als eine Reihe Daten zuzuordnen. Folglich wird unter der Voraussetzung des gleichen Beispiels, das unmittelbar vorhergehend geliefert worden ist (d. h. CTR ID_1 identifiziert den Zähler **2**), angenommen, dass die CTR ID_2 ebenfalls den Zähler **2** identifiziert. Folglich sind, solange der Zähler **2** seinen Schwellenwert nicht erreicht hat, weder die $Daten_1$ noch die $Daten_1$ abgelaufen. Des Weiteren ist zu beachten, dass auf Grund der Tatsache, dass mehr als eine Reihe demselben Zähler zugeordnet werden könnte, wie weiter oben erwähnt worden ist, die Ausführungsform von **Fig.** 10 ermöglicht, dass die Anzahl der Zähler geringer als die Anzahl N der Reihen im Tag-Speicher **100** ist.

[0077] Des Weiteren ist zu beachten, dass zusätzlich zur Funktionalität der Aktivitätsdauer der Ausführungsformen von **Fig.** 10 die CTR ID oder irgendeine alternative Angabe, wie etwa ein gesondertes Bit oder ein Signal, das von der CTR ID verschieden ist, einer bestimmten Reihe erlaubt anzuzeigen, dass die Reihe keinem Zähler zugeordnet ist. Mit anderen Worten: Wenn eine Reihe als keinem Zähler zugeordnet gekennzeichnet ist, dann werden die Daten für diese Reihe entsprechend den ansonsten vorliegenden Cache-Grundsätzen behandelt und nicht mittels einer Aktivitätsdauer verwaltet. Beispielsweise könnte in der bevorzugten Ausführungsform die Codierung jeder CTR ID derart sein, dass für einen bestimmten Code angegeben wird, dass der entsprechenden Zeile kein Zähler zugeordnet ist, während jeder andere Code einen der Zähler identifiziert. Im vorliegenden Beispiel wird angenommen, dass die CTR ID ein 2-Bit-Signal ist. Folglich könnte das Signal wie in der folgenden Tabelle 2 gezeigt codiert sein:

CTR ID-Wert	identifizierter Zähler
00	kein Zähler

01	Zähler 1
10	Zähler 2
11	Zähler 3

Tabelle 2

[0078] Folglich wird anhand der Tabelle 2 für einen Fachmann nachzuvollziehen sein, dass eine CTR ID von 00 angibt, dass die entsprechende Zeile Daten im Datenspeicher **102** gemäß der gegebenen Cache-Architektur und den gegebenen Cache-Grundsätzen zu behandeln ist, ohne eine Aktivitätsdauer der entsprechenden Daten zu berücksichtigen. Folglich wird eine solche Zeile oder werden solche Zeilen eher cachefähige Daten als nicht cachefähige Daten speichern. Andererseits ist dann, wenn eine CTR ID einer Kombination von zwei Bits ungleich null entspricht, den Daten ein Zähler zugeordnet, so dass die erforderliche Bedingung vorliegt, um zu bestimmen, ob die Daten aktiv oder abgelaufen sind. Schließlich ist weiter zu beachten, dass, obwohl die Tabelle 2 zeigt, dass ein Wert ungleich null in der CTR ID eine Zeile Daten mit einem einzigen Zähler in Verbindung bringt, in einer alternativen Ausführungsform ein Hinweis geliefert werden könnte, um eine Bedingung auf der Grundlage von mehr als einem Zähler anzugeben. Beispielsweise könnten dann, wenn statt zwei Bits drei für die CTR ID verwendet werden, eine oder mehrere der 3-Bit-Kombinationen eine Bedingung auf der Grundlage mehrerer Zähler angeben. Beispielsweise könnte die Bedingung derart sein, dass die Schaltung **104** nicht eher angibt, dass die entsprechenden Daten abgelaufen sind, als beide identifizierten Zähler ihre entsprechenden Schwellenwerte erreicht haben.

[0079] Von einem Fachmann werden noch weitere Beispiele bestimmbar sein.

[0080] Es ist zu beachten, dass sowohl die Konfiguration von **Fig.** 10 als auch ihre Funktionsweise noch einen weiteren alternativen Aspekt zur Folge haben, nämlich in Verbindung mit der Angabe, ob Daten aktiv oder abgelaufen sind. Es sollte in Erinnerung gebracht werden, dass die Ausführungsformen der **Fig.** 7 und 9 einen L/E-Anzeiger enthalten, der von einem V/I-Anzeiger für jede Zeile gesondert und unabhängig ist. Bei der Ausführungsform von **Fig.** 10 ist jedoch zu beachten, dass als eine andere Möglichkeit nicht für jede Zeile ein gesonderter und unabhängiger V/I-Anzeiger bereitgestellt wird. Stattdessen ist die Zählerbewertungs- und L/E-Steuereinheit **106** so angeschlossen, dass sie den V/I-Anzeiger für jede Zeile des Tag-Speichers **102** modifizieren könnte. Das hat zur Folge, dass dann, wenn der Zählstand in einem Zähler abläuft, nicht ein gesonderter L/E-Anzeiger auf abgelaufen gesetzt wird, sondern die Zählerbewertungs- und L/E-Steuereinheit **106** den V/I-Zeiger für diese Zeile auf ungültig setzt. Wie bei den oben angegebenen Ausführungsformen wird auf dieses Ereignis hin auch bestimmt, ob die entsprechenden Daten schmutzig sind (d. h. mittels ihres entsprechenden C/D-Anzeigers), und wenn ja, werden die schmutzigen Daten in eine höhere Speicherebene geschrieben, um eine korrekte Speicherkohärenz sicherzustellen. Da es keinen gesonderten L/E-Anzeiger gibt, gibt der V/I-Anzeiger in jedem Fall nicht nur an, ob die entsprechenden Daten gemäß bekannter Cache-Grundsätze gültig sind, sondern könnte des Weiteren die Daten im Hinblick auf einen abgelaufenen Zählstand als ungültig spezifizieren. Weil ein ungültiger Zustand eines V/I-Anzeigers in der bekannten Cache-Technologie angibt, dass die Daten nicht länger gültig sind, ist folglich zu beachten, dass deshalb die abgelaufenen Daten im Folgenden als ungültig behandelt werden und demnach eine Schaltung, die diese Informationen im Datenspeicher **102** sucht, sich nicht auf die abgelaufenen Daten verlassen wird. Ist diese Funktionalität gegeben, wird für einen Fachmann weiter nachzuvollziehen sein, dass die Zählerbewertungs- und L/E-Steuerschaltung **106** genügend Schaltungsanordnungen umfasst, um sowohl jeden der Zähler als auch die Werte ungleich null jeder CTR ID zu überwachen. Folglich werden dann, wenn ein Zähler seinen Schwellenwert erreicht, die Daten jeder Zeile, die über ihre CTR ID diesen Zähler identifiziert, ungültig gemacht, d. h. der V/I-Anzeiger, der jeder derartigen Zeile entspricht, wird von der Schaltung **106** auf ungültig gesetzt. Zur Veranschaulichung sollte von weiter oben das Beispiel in Erinnerung gebracht werden, bei dem sowohl die CTR ID_1 als auch die CTR ID_2 jeweils den Zähler **2** identifizieren. Bei diesem Beispiel wird die Zählerbewertungs- und L/E-Steuerschaltung **106** den Zeitpunkt erfassen, zu dem der Zähler **2** seinen Schwellenwert erreicht. In Reaktion auf dieses Ereignis wird die Zählerbewertungs- und L/E-Steuereinheit **106** sowohl V/I_1 als auch V/I_2 auf ungültig setzen. Schließlich ist zu beachten, dass diese andere Möglichkeit der Beseitigung eines gesonderten und unabhängigen L/E-Anzeigers ermöglicht, Daten, die einen noch aktiven Zählstand besitzen, anderweitig als ungültig zu kennzeichnen. Wenn beispielsweise eine bestimmte Zeile von Daten einem Zähler entspricht, der auf einen Schwellenwert vorrückt, diesen jedoch noch nicht erreicht hat, während ein Snoop und ein Ungültig vom

Tag-Speicher **100** empfangen wird, dann wird die Zeile von der C/D- und V/I-Steuer- und Reaktionsschaltung **104** auf ungültig gesetzt, obwohl der Zähler seinen Schwellenwert noch nicht erreicht hat. Hingegen wird eine gesonderte L/E-Anzeige, wie in den **Fig.** 7 und 9, einen solchen Fall nicht ermöglichen und deshalb eine gesonderte Anzeige eines Zählers liefern, der seinen Schwellenwert noch nicht erreicht hat. Folglich könnte jede der hier dargestellten Ausführungsformen entweder alternativ, d. h. entweder als ein L/E-Anzeiger gesondert und unabhängig vom V/I-Anzeiger, oder als ein einzelner V/I-Anzeiger, der gemäß bekannten Cache-Grundsätzen betrieben wird, jedoch zusätzlich Daten als ungültig identifiziert, wenn die Aktivitätsdauer der Daten ihren Schwellenwert erreicht hat, verwendet werden.

[0081] Aus dem bisher Dargestellten sollte nachvollzogen werden können, dass die oben angegebenen Ausführungsformen die Speicherung von nicht cachefähigen Daten in einer cacheartigen Struktur während einer bestimmten Aktivitätsdauer dieser Daten ermöglichen und dass diese Struktur unabhängig von einer vorhandenen Cache-Struktur oder mit dieser verschmolzen sein könnte. In jedem Fall könnten die mit einer Aktivitätsdauer versehenen Daten während dieser Zeit gelesen und modifiziert werden, ohne dass sie im Hauptspeicher gesucht werden müssen. Außerdem könnte der Schwellenwert der Aktivitätsdauer eingestellt werden, um ihn an verschiedene Bedingungen anzupassen, um die Möglichkeit von aktiven Operationen an zwei verschiedenen Versionen der Daten, die derselben Adresse entsprechen, zu minimieren oder zu beseitigen.

Patentansprüche

1. Mikroprozessor (**2**), der mit einem externen Schreib-Lese-Speicher (**20**) gekoppelt werden kann, der einen adressierbaren Speicherraum zum Speichern von cachefähigen und nicht cachefähigen Daten besitzt, wobei der Mikroprozessor eine Datenspeicherschaltung (**62**) zum Speichern eines Abschnitts der nicht cachefähigen Daten, eine Adressenspeicherschaltung (**64**) zum Speichern einer Adresse, die dem Abschnitt der nicht cachefähigen Daten entspricht, sowie einen Adressen/Daten-Bus (B), der mit der Datenspeicherschaltung (**62**) verbunden ist, Datenübertragungen ermöglicht und eine Adresse für die Datenübertragung transportiert, umfasst,
wobei die in der Datenspeicherschaltung (**62**) gespeicherten Daten eine Cache-Kopie von Daten sind, die durch einen autonomen Prozess geändert werden können, der von einer von der CPU (**5**) des Mikroprozessors unabhängigen Schaltungsanordnung ausgeführt wird, gekennzeichnet durch
einen Zähler (**72**), der einen Zählstand in Reaktion auf ein Taktsignal von einem Anfangswert zu einem Schwellenwert vorrückt, und einen Anzeiger (**66**), der anzeigt, wenn die Gültigkeit des Abschnitts der nicht cachefähigen Daten in der Datenspeicherschaltung (**62**) abgelaufen ist, weil der Zählstand den Schwellenwert erreicht;
wobei der Schwellenwert auf einen Wert eingestellt ist, um die Aktivitätsdauer der in der Datenspeicherschaltung (**62**) gespeicherten Daten zu begrenzen;
eine Aktivität/Abgelaufen-Steuer- und -Reaktionsschaltung (**70**), die von dem Adressen/Daten-Bus (B) und von dem Adressenspeicherregister (**64**) jeweils eine Adresse empfängt und so angeschlossen ist, dass sie von dem Zähler (**72**) ein Steuersignal empfängt, wobei die Aktivität/Abgelaufen-Steuer- und -Reaktionsschaltung (**70**) so betreibbar ist, dass sie ein HIT-Signal ausgibt, wenn der Anzeiger (**66**) anzeigt, dass Daten in der Datenspeicherschaltung (**62**) aktiv sind und die von dem Adressen/Daten-Bus (B) empfangene Adresse mit der Adresse von dem Adressenspeicherregister (**64**) übereinstimmt, und andernfalls ein MISS-Signal ausgibt;
wobei das HIT-Signal das Lesen von Daten von der Datenspeicherschaltung (**62**) freigibt.

2. Mikroprozessor nach Anspruch 1, bei dem die Anzeige des Anzeigers (**66**), dass Daten aktiv sind, jeder anfragenden Schaltung anzeigt, dass es während des Aktivitätsstatus zulässig ist, Daten aus der Datenspeicherschaltung (**62**) zu lesen oder Daten in diese zu schreiben.

3. Mikroprozessor nach Anspruch 1 oder Anspruch 2, ferner mit:
einem Clean/Dirty-Anzeiger (**67**), der anzeigt, dass der Abschnitt der nicht cachefähigen Daten in der Datenspeicherschaltung (**62**) durch eine Datenschreiboperation modifiziert worden ist, nachdem er in der Datenspeicherschaltung (**62**) gespeichert worden ist.

4. Mikroprozessor nach Anspruch 3, ferner mit:
einer Schaltungsanordnung (**78**), die den Clean/Dirty-Anzeiger (**67**) bewertet;
und einer Schaltungsanordnung, die die in der Datenspeicherschaltung (**62**) gespeicherten nicht cachefähigen Daten in Reaktion darauf, dass die Bewertungsschaltungsanordnung (**78**) erfasst, dass sich der Clean/Dirty-Anzeiger (**67**) von einem sauberen Zustand zu einem schmutzigen Zustand geändert hat, in den adressierbaren Speicherraum (**20**) schreibt.

5. Mikroprozessor nach einem vorhergehenden Anspruch, ferner mit einem Registerspeicherplatz (**76**)

zum Speichern des Schwellenwertes.

6. Mikroprozessor (**2**) nach einem vorhergehenden Anspruch, ferner mit einem Registerspeicherplatz (**74**) zum Speichern des Anfangswertes.

7. Mikroprozessor (**2**) nach einem der Ansprüche 1 bis 6, ferner mit:
einer Nachschlagtabelle (**98**) zum Speichern mehrerer Werte;
einer Schaltungsanordnung, die die Nachschlagtabelle in Reaktion auf die Adresse in der Adressenspeicherschaltung abfragt, wobei die Adresse einem der mehreren Werte entspricht; und
einer Schaltungsanordnung zum Auswählen des einen der mehreren Werte als den Schwellenwert.

8. Mikroprozessor (**2**) nach einem der Ansprüche 1 bis 6, ferner mit:
einer Nachschlagtabelle (**98**) zum Speichern mehrerer Werte;
einer Schaltungsanordnung, die in Reaktion auf die Adresse in der Adressenspeicherschaltung die Nachschlagtabelle abfragt, wobei die Adresse einem der mehreren Werte entspricht; und
einer Schaltungsanordnung zum Auswählen des einen der mehreren Werte als den Anfangswert.

9. Mikroprozessor (**2**) nach einem vorhergehenden Anspruch, der ferner einen CPU-Kern (**5**) umfasst, der in Reaktion auf ein Prozessortaktsignal arbeitet, wobei:
das Taktsignal, das den Zähler (**72**) vorrückt, das Prozessortaktsignal ist.

10. Mikroprozessor (**2**) nach einem der Ansprüche 1 bis 8, ferner mit einer Busschnittstelleneinheit (**8**), die so betreibbar ist, dass sie eine Bustransaktion während eines Buszyklus ausführt und der Zähler (**72**) in Reaktion auf jede der Bustransaktionen vorrückt.

11. Mikroprozessor nach einem vorhergehenden Anspruch, bei dem:
die Datenspeicherschaltung (**62**) mehrere Datenspeicherleitungen (**62_1**, **62_2**, **62_3**) umfasst, wovon jede dem Speichern des Abschnitts von nicht cachefähigen Daten dient;
die Adressenspeicherschaltung (**64**) mehrere Adressenspeicherleitungen (**64_1**, **64_2**, **64_3**) umfasst, wovon jede dem Speichern einer Adressen von Daten in einer entsprechenden der mehreren Datenspeicherleitungen (**62_1**, **62_2**, **62_3**) dient;
der Zähler (**72**) mehrere Zähler (**72_1**, **72_2**, **72_3**) umfasst, wovon jeder dazu dient, einen Zählstand in Reaktion auf eine vorübergehende Aktivität von einem Anfangswert zu einem Schwellenwert vorzurücken, wobei jeder der mehreren Zähler (**72_1**, **72_2**, **72_3**) für zugeordnete Daten eine Aktivitätsdauer vorsieht.

Es folgen 8 Blatt Zeichnungen

Anhängende Zeichnungen

FIG. 1

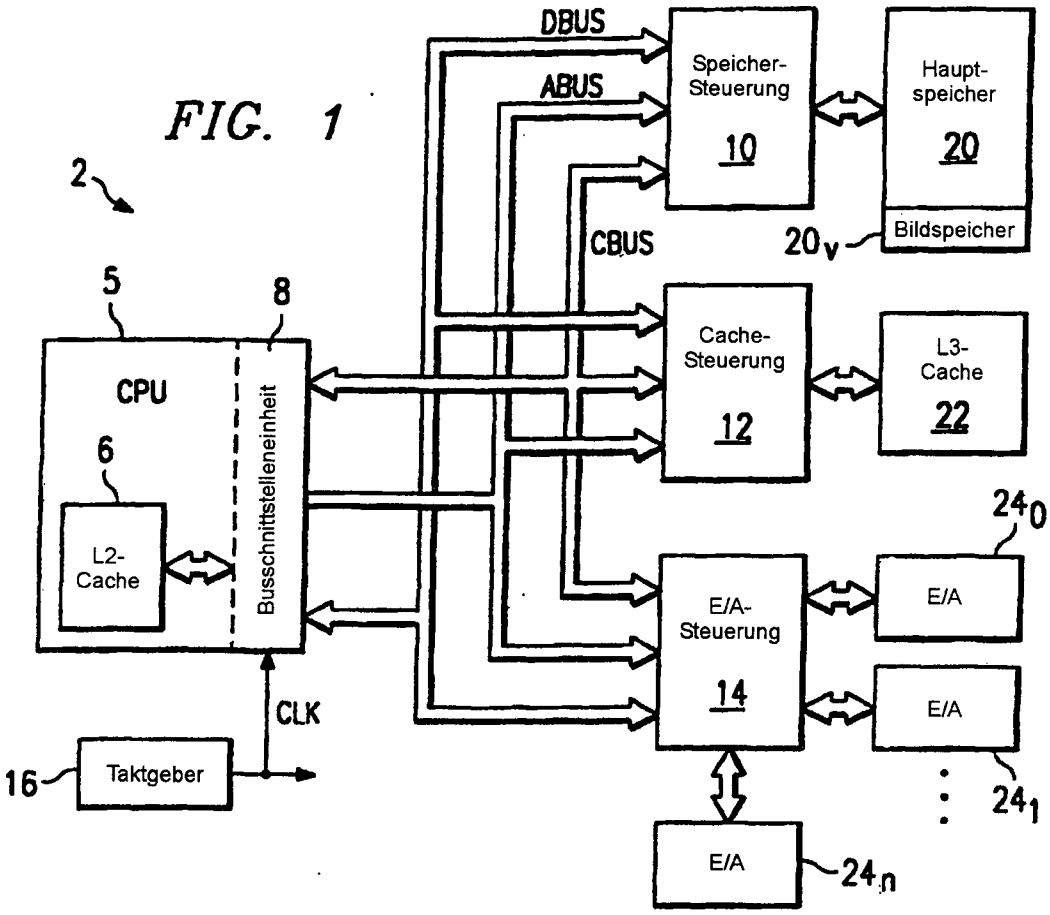
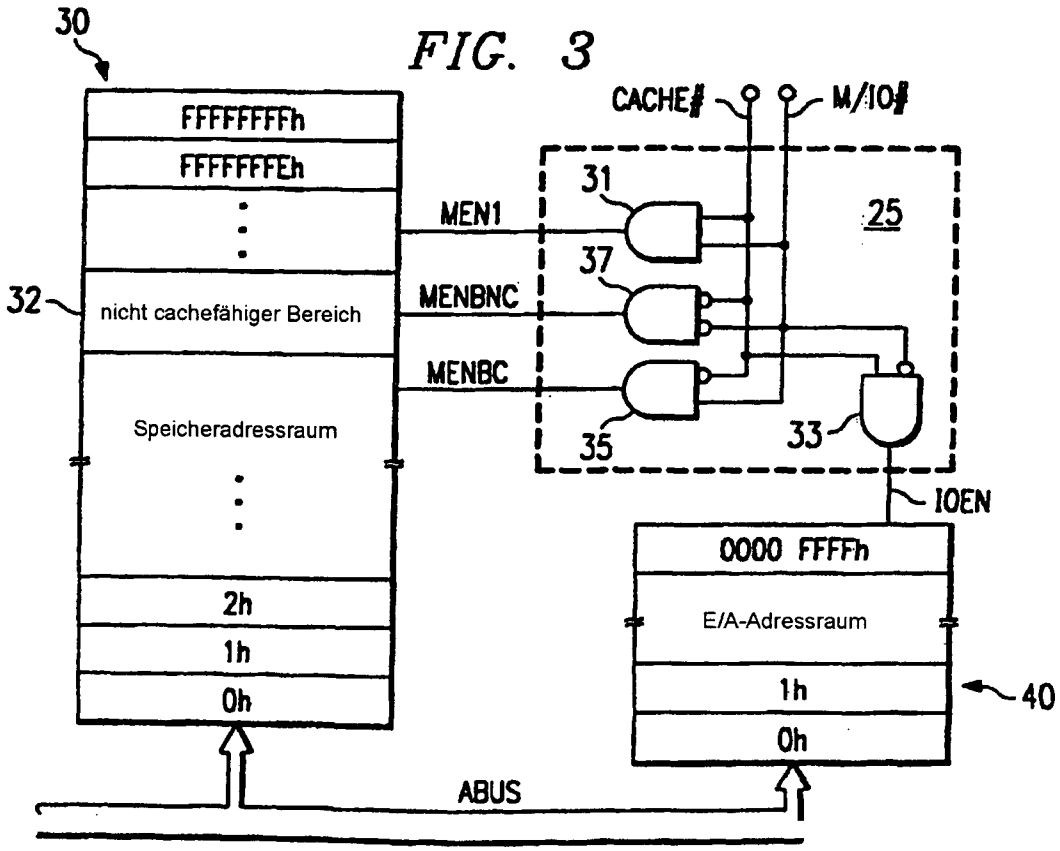


FIG. 3



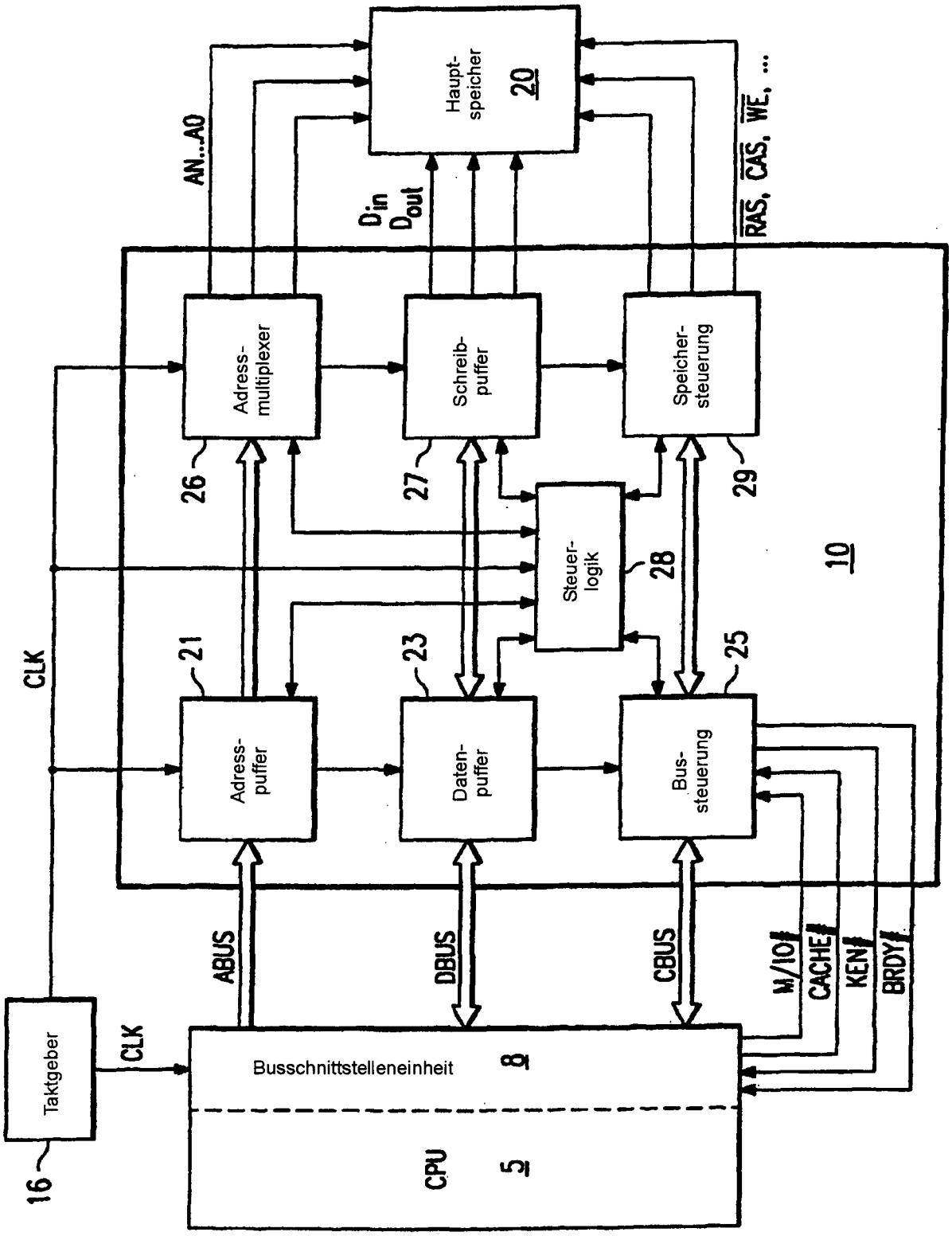

Haupt-
speicher
20
AN..A0
D_{in}
D_{out}
RAS, CAS, WE, ...
Adress-
multiplexer
Schreib-
puffer
Speicher-
steuerung
26
27
29
Steuer-
logik
28
10
CLK
21
23
25
Adress-
puffer
Daten-
puffer
Bus-
steuerung
ABUS
DBUS
CBUS
W/IO
CACHE
KEN
BRDY
Taktgeber
CLK
Busschnittstelleneinheit
8
CPU
5
16

FIG. 2

FIG. 4

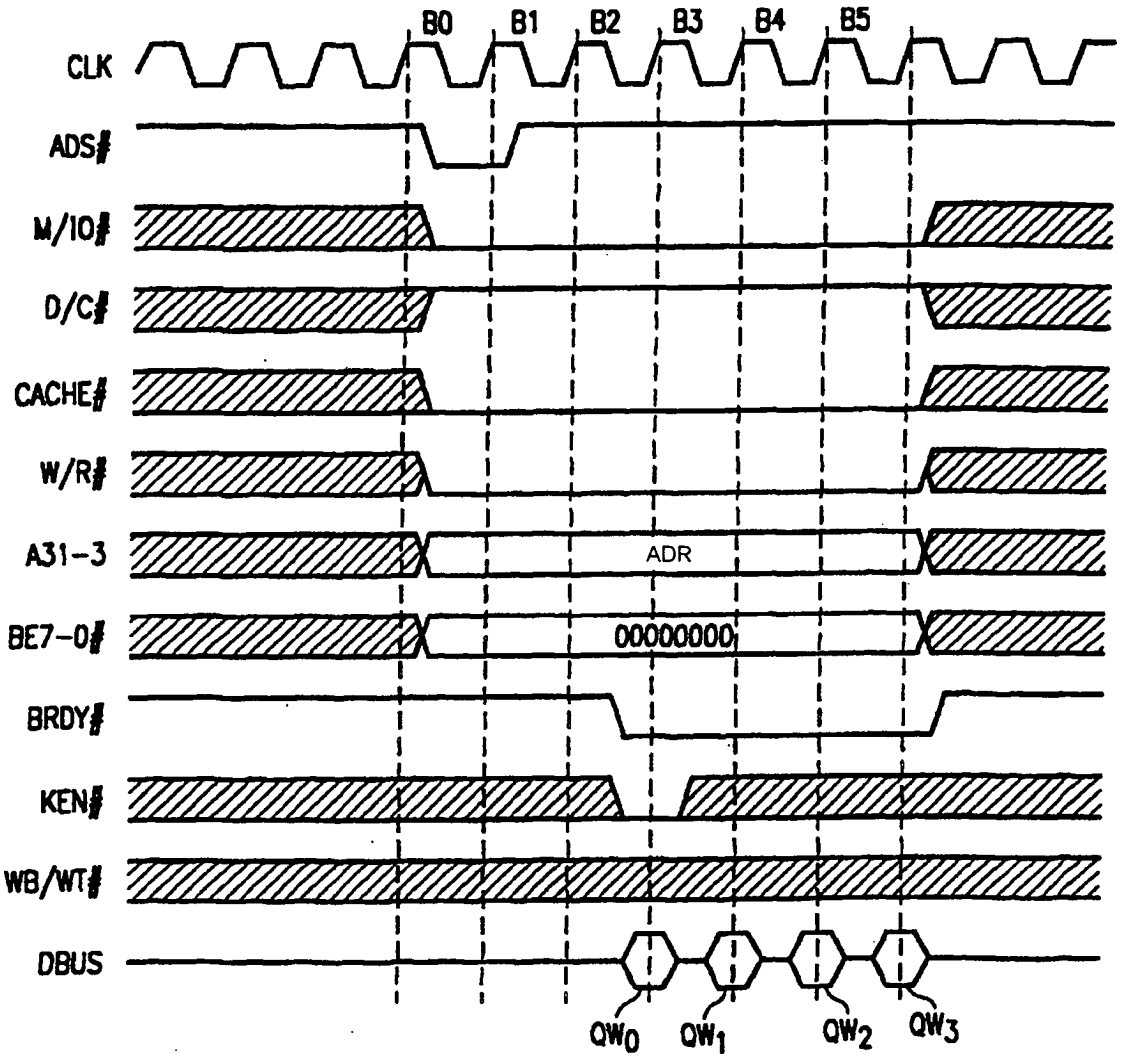
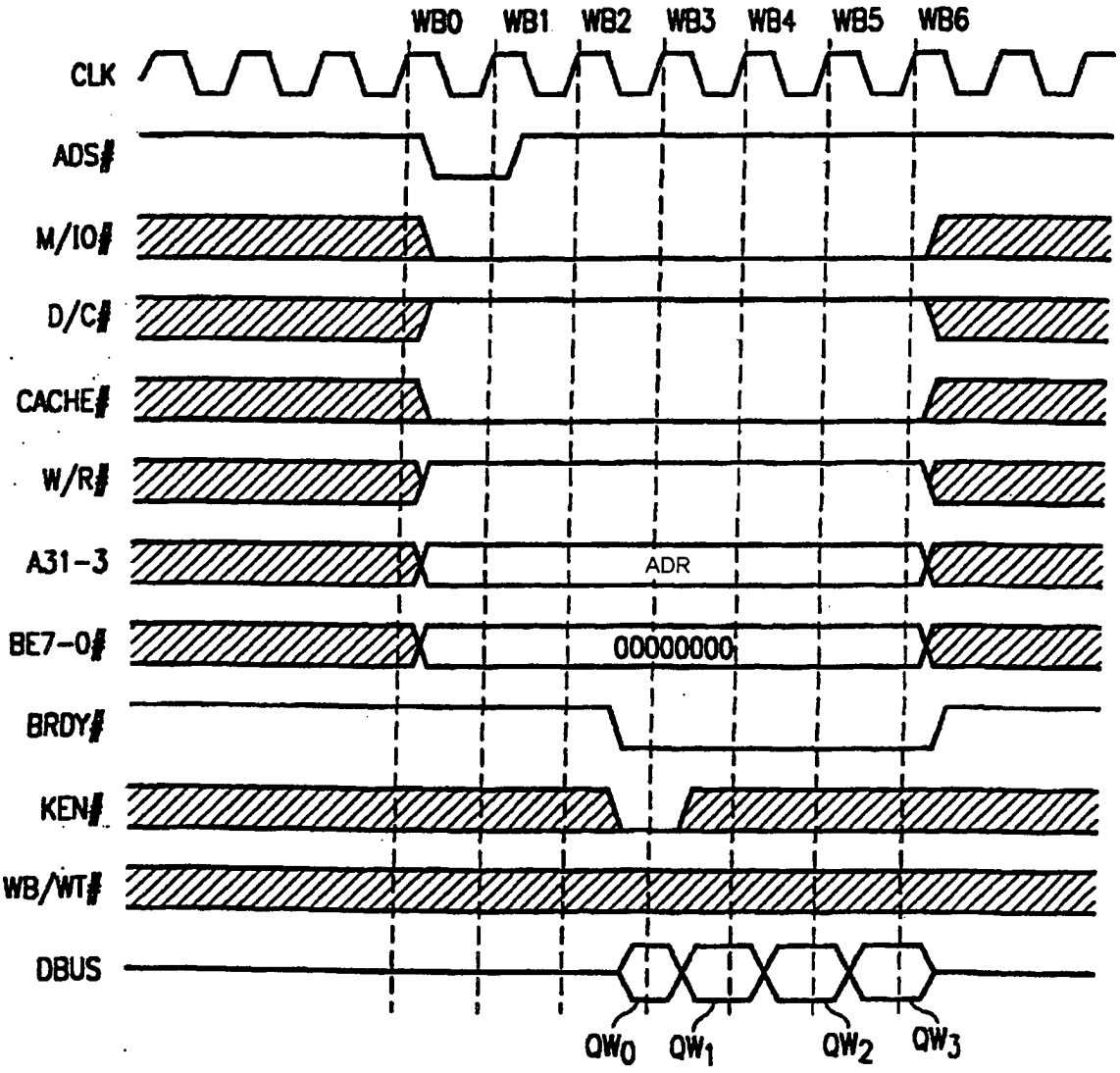

B0
B1
B2
B3
B4
B5
CLK
ADS#
M/IO#
D/C#
CACHE#
W/R#
A31-3
ADR
BE7-0#
00000000
BRDY#
KEN#
WB/WT#
DBUS
QW0
QW1
QW2
QW3

FIG. 5


WB0
WB1
WB2
WB3
WB4
WB5
WB6
CLK
ADS#
M/IO#
D/C#
CACHE#
W/R#
A31-3
ADR
BE7-0#
00000000
BRDY#
KEN#
WB/WT#
DBUS
QW0
QW1
QW2
QW3

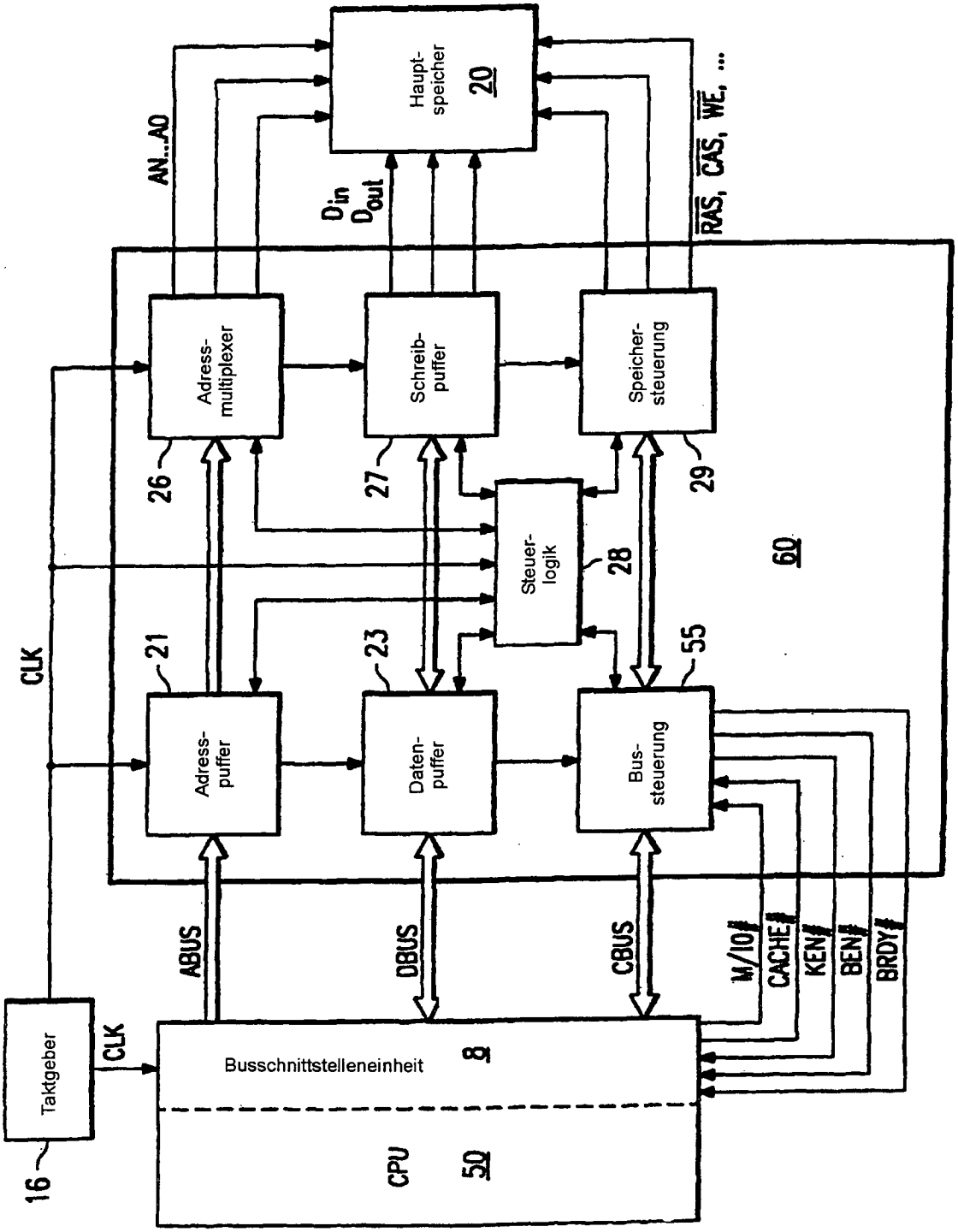
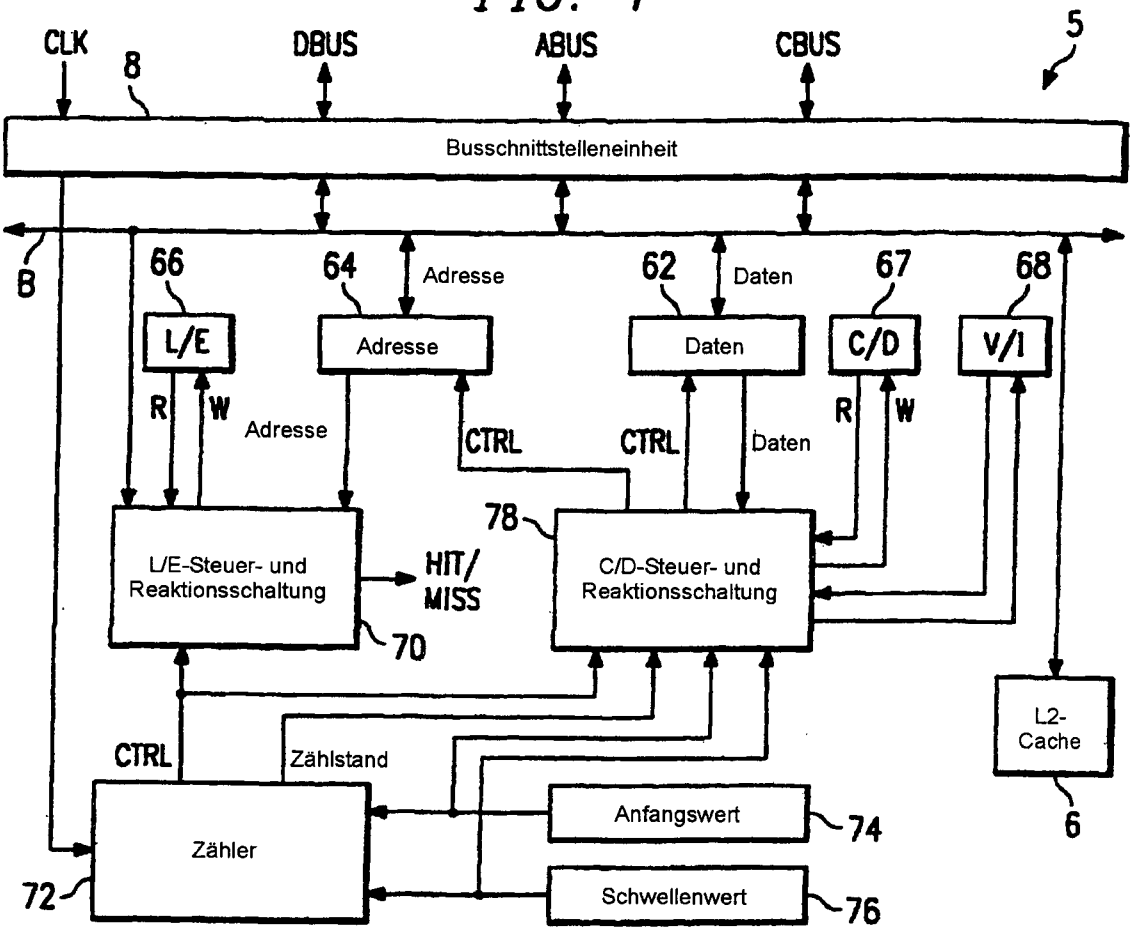

Haupt-
speicher
20
AN...A0
D_{in}
D_{out}
RAS, CAS, WE, ...
Adress-
multiplexer
Schreib-
puffer
Speicher-
steuerung
26
27
29
Steuer-
logik
28
60
CLK
21
23
55
Adress-
puffer
Daten-
puffer
Bus-
steuerung
ABUS
DBUS
CBUS
M/IO#
CACHE#
KEN#
BEN#
BRDY#
Taktgeber
CLK
Busschnittstelleneinheit
8
CPU
50
16

FIG. 6

FIG. 7


CLK
8
DBUS
ABUS
CBUS
5
Busschnittstelleneinheit
B
66
64
Adresse
62
Daten
67
68
L/E
Adresse
Daten
C/D
V/I
R
W
Adresse
CTRL
CTRL
Daten
R
W
78
L/E-Steuer- und Reaktionsschaltung
HIT/ MISS
70
C/D-Steuer- und Reaktionsschaltung
L2- Cache
6
CTRL
Zählstand
Zähler
72
Anfangswert
74
Schwellenwert
76

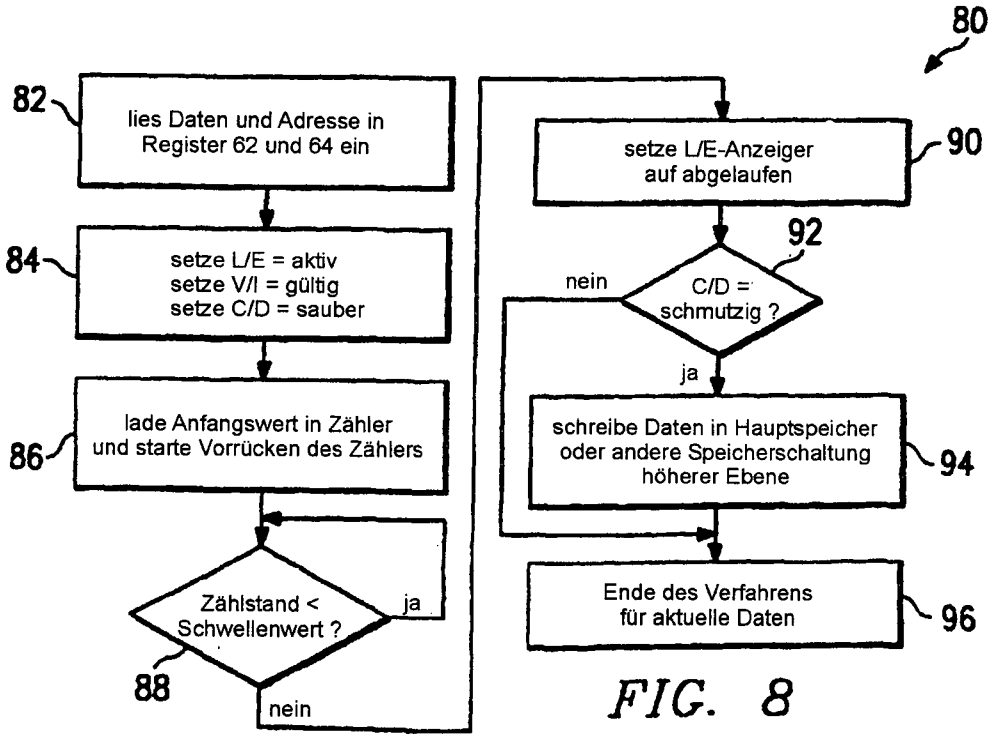

80
82
lies Daten und Adresse in Register 62 und 64 ein
84
setze L/E = aktiv
setze V/I = gültig
setze C/D = sauber
86
lade Anfangswert in Zähler und starte Vorrücken des Zählers
Zählstand < Schwellenwert ?
ja
88
nein
setze L/E-Anzeiger auf abgelaufen
90
92
nein
C/D = schmutzig ?
ja
schreibe Daten in Hauptspeicher oder andere Speicherschaltung höherer Ebene
94
Ende des Verfahrens für aktuelle Daten
96

FIG. 8

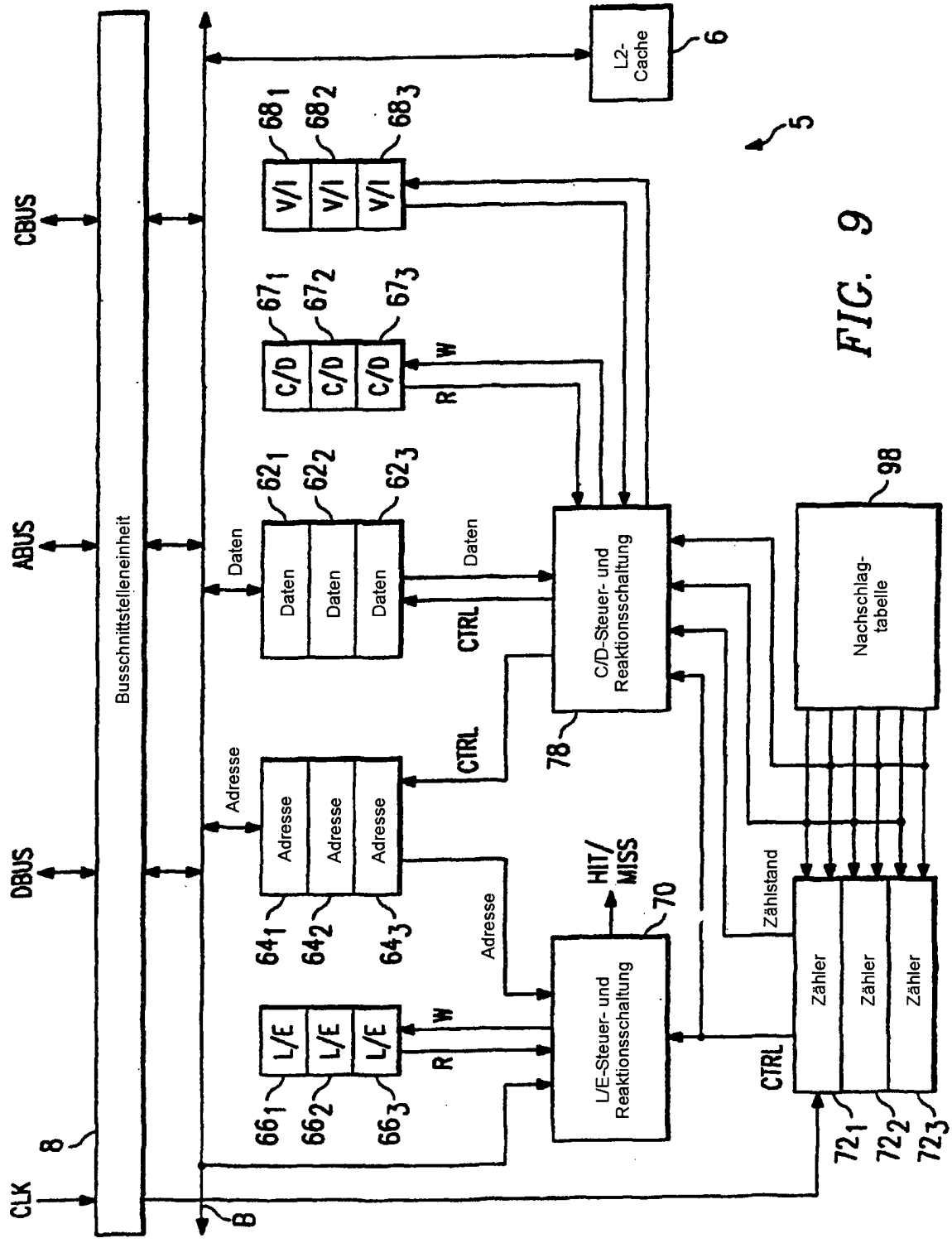
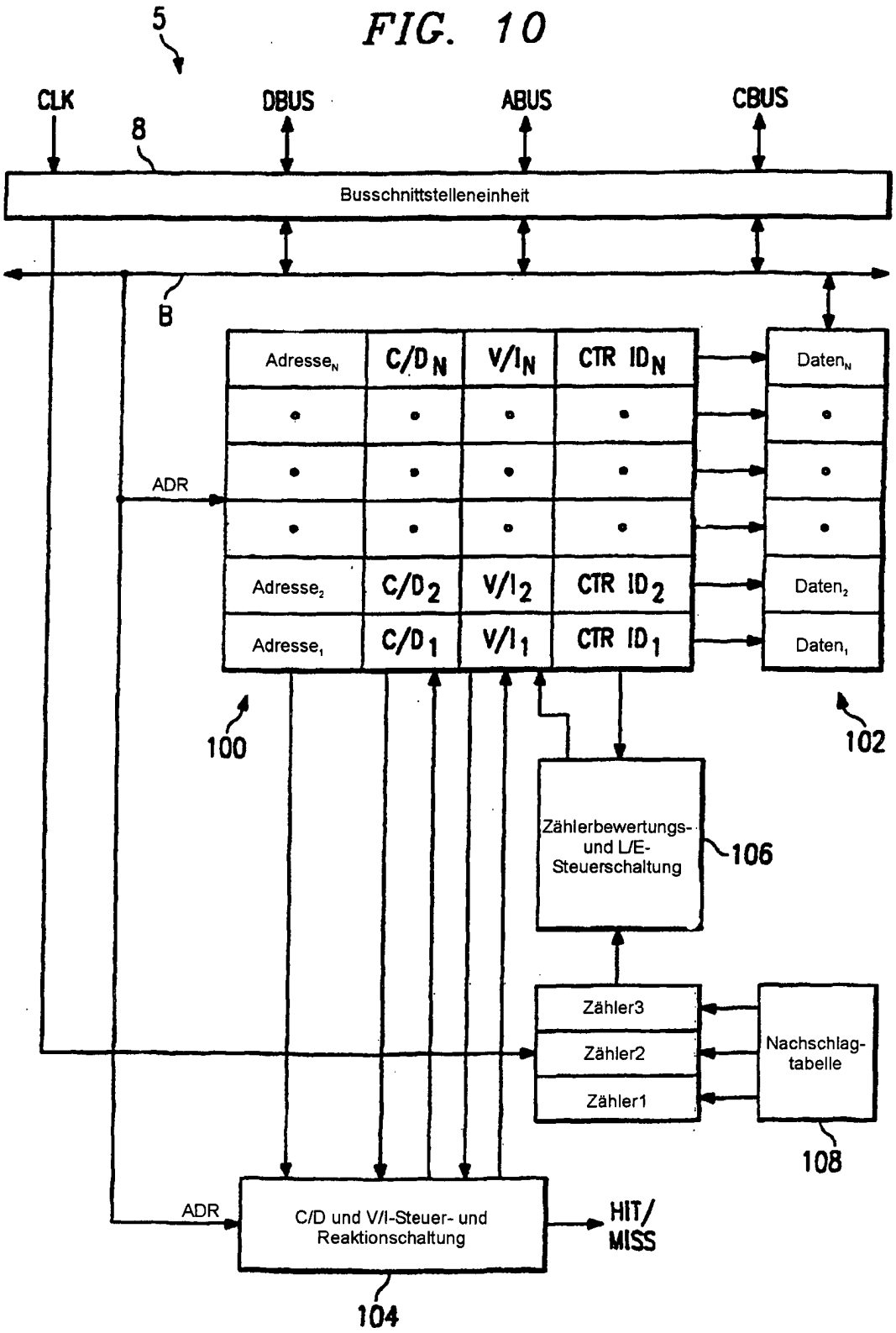

CBUS
ABUS
DBUS
CLK
8
B
Busschnittstelleneinheit
L2-Cache
6
5
V/I
C/D
L/E
Daten
Adresse
CTRL
R
W
C/D-Steuer- und Reaktionsschaltung
78
L/E-Steuer- und Reaktionsschaltung
70
HIT/MISS
Nachschlag-tabelle
98
Zähler
Zählstand

FIG. 9

FIG. 10


5
CLK
DBUS
ABUS
CBUS
8
Busschnittstelleneinheit
B
Adresse$_N$
C/D$_N$
V/I$_N$
CTR ID$_N$
Daten$_N$
ADR
Adresse$_2$
C/D$_2$
V/I$_2$
CTR ID$_2$
Daten$_2$
Adresse$_1$
C/D$_1$
V/I$_1$
CTR ID$_1$
Daten$_1$
100
102
Zählerbewertungs- und L/E-Steuerschaltung
106
Zähler3
Zähler2
Zähler1
Nachschlag-tabelle
108
ADR
C/D und V/I-Steuer- und Reaktionschaltung
HIT/ MISS
104