(19) Deutsches Patent- und Markenamt

(10) **DE 10 2014 117 465 B4** 2021.05.06

(12) # Patentschrift

(21) Aktenzeichen: **10 2014 117 465.8**
(22) Anmeldetag: **27.11.2014**
(43) Offenlegungstag: **02.07.2015**
(45) Veröffentlichungstag der Patenterteilung: **06.05.2021**

(51) Int Cl.: ***G06F 15/167*** (2006.01)
G06F 11/20 (2006.01)
G06F 12/00 (2006.01)

Innerhalb von neun Monaten nach Veröffentlichung der Patenterteilung kann nach § 59 Patentgesetz gegen das Patent Einspruch erhoben werden. Der Einspruch ist schriftlich zu erklären und zu begründen. Innerhalb der Einspruchsfrist ist eine Einspruchsgebühr in Höhe von 200 Euro zu entrichten (§ 6 Patentkostengesetz in Verbindung mit der Anlage zu § 2 Abs. 1 Patentkostengesetz).

(30) Unionspriorität:
14/142,726 **27.12.2013** **US**

(73) Patentinhaber:
Intel Corporation, Santa Clara, Calif., US

(74) Vertreter:
Maucher Jenkins Patentanwälte & Rechtsanwälte, 80538 München, DE

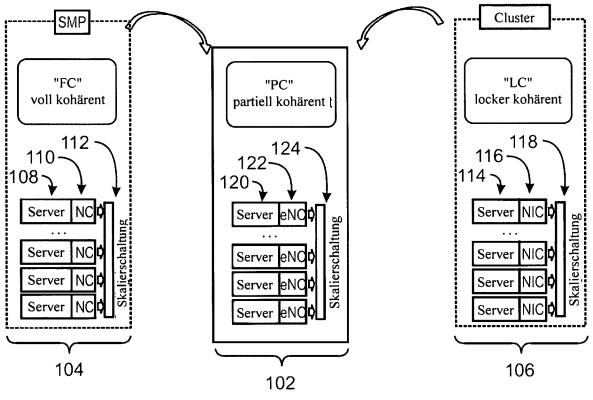
(72) Erfinder:
Das Sharma, Debendra, Saratoga, CA, US; Kumar, Mohan J., Aloha, Oreg., US; Fleischer, Balin T., Groton, Mass., US

(56) Ermittelter Stand der Technik:

FLEISCH, B.: Reliable distributed shared memory. In: IEEE Workshop on Experimental Distributed Systems, Huntsville, AL, USA, 1990, S. 102-105

(54) Bezeichnung: **Unterstützter kohärenter gemeinsamer Speicher**

(57) Hauptanspruch: Vorrichtung zur Bereitstellung eines kohärenten, über mehrere Cluster verteilten Speichers, umfassend:
einen Speicher-Controller;
einen oder mehreren Netzknoten, in denen der Speicher-Controller den Zugriff auf eine geteilte Speicherregion eines jeden Netzknotens regelt, sodass jede geteilte Speicherregion mit Load-Store-Semantik abrufbar ist, wobei der Speicher-Controller dafür ausgelegt ist, dass die geteilte Speicherregion jedes Netzknotens über den Speicher-Controller erreicht werden kann, auch wenn die CPUs des Netzknotens versagen; und
einen globalen Speicher, in dem jede geteilte Speicherregion vom Speicher-Controller im globalen Speicher verzeichnet ist.



Beschreibung

Technisches Gebiet

[0001] Diese Offenbarung bezieht sich generell auf ein System mit mehreren Netzknoten. Genauer bezieht diese Offenbarung sich auf gemeinsame Speicher innerhalb eines Systems mit mehreren Netzknoten.

Hintergrund

[0002] Ein System mit mehreren Netzknoten kann eine Vielzahl von Knoten enthalten. Diese Art von System enthält, ist jedoch nicht beschränkt auf, ein Netzwerk, ein Rack-Serversystem, einen Blade-Server und ähnliches. In einigen Fällen kann jeder Knoten ein großer, symmetrischer Multiprozessorknoten (SMP) sein, der einen wesentlichen Teil eines oder mehrerer Racks mit Hardware-Cache-Kohärenz zwischen dem Prozessor und Eingangs-/Ausgangsgeräten (I/O) im Knoten umfasst. Als Resultat der Cache-Kohärenz können große SMP-Systeme ausreichende Rechnerressourcen anwenden, um ein Problem mit fein abgestufter Auslastung zwischen den Computergeräten zu lösen, während sie eine große Speicherkapazität haben, um Anwendungsdaten zu speichern, die durch die Speicherauslastung wiederum jedem Computergerät direkt zugänglich sind. Das System kann auch ein lose verbundenes (LC) System sein, bestehend aus mehreren kleineren SMP-Systemen, wobei die Netzknoten die Aufgaben auf einem grob abgestuften Niveau koordinieren können.

[0003] Die wissenschaftliche Publikation „Reliable distributed shared memory“ von B.D. Fleisch, Proceedings, IEEE Workshop on Experimental Distributed Systems, 11. bis 12. Oktober 1990, Huntsville, AL, USA, Vol. 1, Seiten 102-105, doi: 10.1109/EDS.1990.138058, behandelt Anwendungen, welche einen gemeinsam genutzten Speicher in Single-Site-Systemen verwenden. Dabei besteht ein DSM-System (DSM - Distributed Memory System) aus einem gemeinsam genutzten Adressraum, auf den durch logische Operationen zugegriffen werden kann. Ein geteilter Speicher wird durch Verwendung eines Segments erzeugt, dessen Größe, Name und Zugriffsberechtigung beim Erzeugen festgelegt werden. Prozesse lokalisieren das Segment und fügen es ihrem Adressraum für virtuellen Speicher hinzu. Wenn ein das Segment betreffender Vorgang abgeschlossen ist, kann das Segment wieder abgetrennt werden, was zur Zerstörung von dessen Daten führt. Der geteilte Bereich des Adressraums eines Prozess ist dabei in Datenseiten unterteilt, die repliziert werden.

Figurenliste

[0004] Die folgende detaillierte Beschreibung ist im Zusammenhang mit den begleitenden Zeichnungen besser verständlich, die spezielle Beispiele vieler Objekte und Eigenschaften der Veröffentlichung enthalten.

Abb. 1 ist ein Blockdiagramm von Systemmodellen mit mehreren Netzknoten;

Abb. 2 ist eine Darstellung des partiell kohärenten Systems;

Abb. 3 ist eine Darstellung einer globalen Speicherkarte;

Abb. 4 ist ein Prozess-Flussdiagramm für kohärenten gemeinsamen Speicher über mehrere Cluster; und

Abb. 5 ist ein Blockdiagramm eines Netzknotens 500, der auf zusammengefasste Speicherressourcen zugreifen kann.

[0005] In der gesamten Offenbarung und in den Abbildungen werden dieselben Bezugsnummern verwendet, um auf gleiche Komponenten und Merkmale zu verweisen. Nummern der 100-Serie beziehen sich auf Merkmale, die ursprünglich in **Abb. 1** zu finden sind, Nummern der 200-Serie beziehen sich auf Merkmale, die ursprünglich in **Abb. 2** zu finden sind, usw.

Beschreibung der Ausführungsformen

[0006] Die vorliegende Erfindung ist im Hauptanspruch und in den nebengeordneten Ansprüchen definiert. Ein SMP-System umfasst eine Ein-Fehler-Domäne, bei der ein Fehler in einer Komponente oder in einem Software-Teil des Systems das Versagen des gesamten Systems zur Folge hat. Zum Beispiel versagt das ganze System, das den SMP-Netzknoten beinhaltet, wenn dieser SMP-Netzknoten versagt. Im Gegensatz dazu grenzt ein LC-System den Fehler einer jeden Komponente oder eines jeden Software-Teils durch unabhängige Fehler-Domänen ein. Demzufolge kann der betroffene Server oder die betroffene Komponente eines LC-Systems versagen, aber andere Server oder Komponenten werden weiter operieren, als wäre der Fehler nicht aufgetreten. Jedoch wird der Speicher in einem LC-System nicht durch Load-Store-Semantik geteilt. Vielmehr werden Nachrichten durch einen I/O-Treiber geschickt, um die Speicherteilung in einem LC-System zu bewerkstelligen. Die Verwendung des I/O-Treibers um die Speicherteilung zu ermöglichen kann aufgrund der mit I/O-Treibern einhergehenden höheren Wartezeit die Leistung des LC-Systems verglichen mit SMP-Systemen verringern.

[0007] Hier beschriebene Ausführungsformen beziehen sich auf kohärente gemeinsame Speicher über mehrere Cluster. In den Ausführungsformen ist ein Speicher-Controller mit einem oder mehreren Netzknoten gekoppelt. Der Speicher-Controller reguliert den Zugriff auf Speichermodule innerhalb je-

des Netzknotens mit Hilfe von Load-Store-Semantik. Das Speichermodul jedes Netzknotens kann in einen geteilten Speicher eines jeden Netzknotens eingeschlossen sein. Der geteilte Speicher ist zugänglich, auch wenn der Netzknoten versagt hat. Außerdem gewährleistet der Speicher-Controller einen globalen Speicher, und jeder geteilte Speicher einer Vielzahl von Netzknoten kann vom Speicher-Controller im globalen Speicher verzeichnet werden. Das Resultat ist ein cache-fähiger globaler Speicher. Der cache-fähige globale Speicher kann Datenkonstanz über mehrere Netzknoten und Cluster liefern, während er unabhängige Fehler-Domäne eines jeden Netzknotens oder Clusters aufrechterhält. Weiterhin ist der globale Speicher zugänglich und cache-fähig, weil er die Load-Store-Semantik als lokaler Speicher nutzt, während jedes Cluster seine separate Fehler-Domäne aufrechterhält. Zudem kann der geteilte Speicher Zuverlässigkeit, Erreichbarkeit und Service (RAS) bieten, einschließlich aller redundanten Anordnungen unabhängiger Festplatten (RAID). Die vorliegende Technik kann auch mit jeder beliebigen Rack-Scale-Architektur (RSA) mit hoher Speicherdichte genutzt werden.

[0008] In den Ausführungsformen umfasst jeder Netzknoten ein oder mehrere Datenverarbeitungsgeräte (z. B.CPUs), einen Speicher, der sowohl cache-fähig als auch nicht cache-fähig und unstetig oder nicht unstetig ist, und ein oder mehrere I/O-Geräte, auf denen sowohl ein BIOS-Image als auch ein Betriebssystem/Virtual Machine Monitor laufen. Auf diese Art ist jeder Netzknoten eine eingegrenzte Fehler-Domäne. Jeder Fehler in jeder Hardware-Komponente im Netzknoten oder der Software, die auf einem Netzknoten läuft, bringt im schlimmsten Fall nur diesen Netzknoten zum Erliegen.

[0009] In der folgenden Beschreibung und den Ansprüchen können die Begriffe „gekoppelt“ und „verbunden“ gemeinsam mit ihren Ableitungen verwendet sein. Es sollte selbstverständlich sein, dass diese Begriffe nicht als Synonyme füreinander zu verstehen sind. Vielmehr kann bei bestimmten Ausführungsformen „verbunden“ verwendet werden, um anzuzeigen, dass zwei oder mehr Elemente in direktem physischen oder elektrischen Kontakt miteinander stehen. „Gekoppelt“ kann bedeuten, dass zwei oder mehr Elemente in direktem physischen oder elektrischen Kontakt stehen. Jedoch kann „gekoppelt“ auch bedeuten, dass zwei oder mehr Elemente nicht in direktem Kontakt miteinander sind, trotzdem aber miteinander arbeiten oder interagieren. Der Begriff „lose gekoppelt“ bezieht sich auf ein System mit unabhängigen Fehler-Domänen. Deshalb ändert die Benutzung des Begriffs „gekoppelt“ nichts an dem, was als lose gekoppeltes System bekannt ist.

[0010] Einige Ausführungsformen können in einer oder einer Kombination von Hardware, Firmware und Software implementiert werden. Einige Ausführungsformen können auch als auf einem maschinenlesbaren Medium gespeicherte Anweisungen implementiert werden, die von einer Computerplattform gelesen und ausgeführt werden können, um die hier beschriebenen Operationen auszuführen. Ein maschinenlesbares Medium kann jeden Mechanismus für das Speichern oder Senden von Information in einer durch eine Maschine (z. B. ein Computer) lesbaren Form enthalten. Zum Beispiel kann ein maschinenlesbares Medium Read Only Memory (ROM), Random Access Memory (RAM), Magnetplattenspeichermedien, optische Speichermedien, Flash-Speicher-Geräte oder Ähnliches enthalten.

[0011] Eine Ausführungsform ist eine Implementierung oder ein Beispiel. Die Bezugnahme auf „eine Ausführungsform“, „einige Ausführungsformen“, „verschiedene Ausführungsformen“ oder „andere Ausführungsformen“ bedeutet, dass eine bestimmte Funktion, Struktur oder ein bestimmtes Merkmal, die/das im Zusammenhang mit den Ausführungsformen beschrieben wird, in mindestens einigen Ausführungsformen, aber nicht notwendigerweise allen Ausführungsformen der Erfindungen eingeschlossen ist. Die verschiedenen Aufführungen von „eine Ausführungsform“ oder „einige Ausführungsformen“ beziehen sich nicht notwendigerweise auf dieselben Ausführungsformen. Elemente oder Aspekte einer Ausführungsform können mit Elementen oder Aspekten einer anderen Ausführungsform kombiniert werden.

[0012] Nicht alle Komponenten, Funktionen, Strukturen, Merkmale usw., die hier beschrieben und gezeigt sind, müssen in einer bestimmten Ausführungsform oder Ausführungsformen enthalten sein. Wenn die Beschreibung einer Komponente, eines Merkmals, einer Struktur oder einer Charakteristik aussagt, dass sie/es enthalten sein „kann“ oder „könnte“, dann ist es zum Beispiel für diese bestimmte Komponente, ein bestimmtes Merkmal, eine bestimmte Struktur oder Charakteristik nicht erforderlich, enthalten zu sein. Wenn in der Beschreibung oder den Ansprüchen Bezug auf „ein“ Element genommen wird, bedeutet das nicht, dass es nur eines von diesem Element gibt. Wenn die Beschreibung oder die Ansprüche Bezug auf „ein zusätzliches“ Element nehmen, schließt das nicht aus, dass es dort mehr als ein zusätzliches Element gibt.

[0013] Es ist zu beachten, dass, obwohl einige Ausführungsformen in Bezugnahme auf bestimmte Implementierungen beschrieben wurden, sind andere Implementierungen gemäß einiger Ausführungsformen möglich. Zusätzlich brauchen die Anordnung und/oder Reihenfolge von Schaltelementen oder anderen Merkmalen, die in den Zeichnungen gezeigt und/oder hier beschrieben sind, nicht auf die bestimmte gezeigte und beschriebene Weise angeord-

net sein. Viele andere Anordnungen sind gemäß einiger Ausführungsformen möglich.

[0014] Bei jedem in einer Figur gezeigten System können die Elemente in einigen Fällen jeweils dieselbe Referenznummer oder eine unterschiedliche Referenznummer aufweisen, um anzudeuten, dass die repräsentierten Elemente unterschiedlich und/oder ähnlich sein könnten. Ein Element kann jedoch flexibel genug sein, um unterschiedliche Implementierungen zu haben und es kann mit einigen oder allen hier dargestellten oder beschriebenen Systemen funktionieren. Die verschiedenen in den Figuren dargestellten Elemente können dieselben Elemente oder sie können unterschiedlich sein. Welches ein erstes Element genannt wird und welches ein zweites Element, ist willkürlich.

[0015] **Abb. 1** ist ein Blockdiagramm **100** von Systemmodellen mit mehreren Netzknoten; Die Systemmodelle mit mehreren Netzknoten beinhalten ein partiell kohärentes System **102**, ein SMP-System **104** und ein LC-System **106**. Obwohl in jedem System verschiedene Server dargestellt sind, kann jedes System als ein Server betrachtet werden. Im SMP-System **104** ist jeder Netzknoten **108** mit einem Netzknoten-Controller (NC) **110** verbunden. Der NC **110** befähigt jeden Netzknoten **108**, sich mit einer Skalierungsschaltung **112** zu verbinden. Die Skalierungsschaltung **112** kann genutzt werden, um die Kommunikation zwischen jedem NC **110** des SMP-Systems **104** zu ermöglichen. Dementsprechend basiert das SMP-System **104** auf dem Netzknoten-Controller und hat einen geteilten Speicher. Das SMP-System ist vollständig kohärent und umfasst einen schnellen verteilten Sperrmanager. Jedoch ist das SMP-System **104** eine Ein-Fehler-Domäne. Mit anderen Worten wird ein einziger Fehler in irgendeinem Netzknoten **108** oder in einem Netzknoten-Controller **110** das ganze System zum Versagen bringen.

[0016] Im LC-System **106** ist jeder Netzknoten **114** mit einer Netzwerk-Schnittstellenkarte (NIC) **116** verbunden. In einigen Fällen ist die NIC **116** ein Ethernet-Gerät oder ein anderer I/O-Controller, das zu entfernt liegendem direktem Speicherzugriff (RDMA) fähig ist, wie z. B. ein InfiniBand Host Bus Adapter (IB HBA). Der NC **116** befähigt jeden Netzknoten **114**, sich mit einer RDMA-Schaltung **118** zu verbinden. Die RDMA-Schaltung **118** ermöglicht jeder NIC **116** die Weitergabe von Nachrichten, um eine Speicherteilung über das LC-System **106** zu gewährleisten. Folglich beinhaltet das LC-System **106** unabhängige Fehler-Domänen. Der Speicher wird jedoch nicht im LC-System **106** geteilt. Weiterhin ist es schwierig, die Auslastungen innerhalb des LC-Systems **106** auszugleichen, und das LC-System **106** hat eine verteilte Sperrmanager-Skalierbarkeit.

[0017] Das partiell kohärente System **102** beinhaltet eine Vielzahl von Netzknoten **120**, von denen jeder mit einem von einer Vielzahl von erweiterten Netzknoten-Controllern (eNC) 122 verbunden ist. Jeder eNC 122 verbindet seinen jeweiligen Netzknoten **120** mit einer Skalierungsschaltung **124**. Das partiell kohärente System **102** teilt den Speicher über das System mit mehreren Netzknoten mit unabhängigen Fehler-Domänen. Das partiell kohärente System **102** ist partiell kohärent durch die Nutzung einer Software wie unten beschrieben. Außerdem beinhaltet das partiell kohärente System **102** einen schnellen verteilten Sperrmanager.

[0018] **Abb. 2** ist eine Darstellung des partiell kohärenten Systems **102**. Das partiell kohärente System **102** beinhaltet einen Netzknoten **202** und einen Netzknoten **204**. Der Netzknoten **202** beinhaltet einen Speicher-Controller (FMC) **206**, und der Netzknoten **204** beinhaltet einen FMC **208**. Zusätzlich beinhaltet der Netzknoten **202** einen Netzknoten-Speicher **214** und einen lokalen Speicher **218**. Der Netzknoten **204** beinhaltet einen Netzknoten-Speicher **216** und einen lokalen Speicher **220**. Jeder FMC **206** und **208** kann mit dem ihm zugeordneten Netzknoten eine eigenständige Komponente bilden, wie in **Abb. 2** dargestellt. In einigen Ausführungsformen können die FMCs **206** und **208** in die CPU(s) innerhalb jedes Netzknotens des Systems mit mehreren Netzknoten integriert sein. Folglich kann der FMC **206** in einigen Ausführungsformen in den CPU **210A** und den CPU **210B** des Netzknotens **202** integriert sein, und der FMC **208** kann in den CPU **212A** und den CPU **212B** des Netzknotens **204** integriert sein. Die CPUs **210A**, **210B**, **212A** und **212B** greifen alle auf einen globalen Speicher zu, indem sie ein PLM- (Plattsmouth) Protokoll benutzen, das Store-Semantik (für die Speicherkarte des Systems) ähnlich des SMI3 und das I/O-Protokoll (wie PCIe) für Block-Speicherzugriff kombiniert. Der globale Speicher beinhaltet den Netzknotenspeicher **214** und den Netzknotenspeicher **216**. In den Ausführungsformen kann auf den globalen Speicher wie auf einen geteilten Speicher oder einen Blockspeicher zugegriffen werden. Der globale Speicher kann in mehrere Regionen unterteilt sein. Zudem umfassen der FMC **206** und der FMC **208** eine Fehler-Isolations-Begrenzung **207A** bzw. eine Fehler-Isolations-Begrenzung **207B**, wobei durch andere Netzknoten auf den globalen Speicher zugegriffen werden kann, auch wenn der lokale Netzknoten nicht funktioniert.

[0019] Ein Plattsmouth- (PLM) Link kann benutzt werden, um jeden CPU mit dem FMC zu verbinden. Entsprechend beinhaltet der Netzknoten **202** ein Paar PLM-Verbindungen **222**, um den CPU **210A** und den CPU **210B** mit dem FMC **206** zu verbinden. Gleichermaßen beinhaltet der Netzknoten **204** ein Paar PLM-Verbindungen **224**, um den CPU **212A** und den CPU **212B** mit dem FMC **208** zu verbin-

den. Eine PLM-Verbindung **226A** und eine PLM-Verbindung **226B** können ebenfalls benutzt werden, um den Netzknoten **202**, bzw. den Netzknoten **204**, mit dem Schalter **228** zu verbinden. Jede PLM-Verbindung kann beide Store-Semantiken mit optionaler Datenverzeichnisinformation wie SMI3 und einem I/O-Protokoll mit Auslastungs- und Speicherfunktionalität, wie ein Peripheral Component Interconnect Express-Protokoll (PCIe) unterstützen. In den Ausführungsformen kann jede Verbindung, die Store-Semantik und ein I/O-Protokoll mit einem gewöhnlichen Pin-Set unterstützt, genutzt werden, um einen Netzknoten mit einem SMC zu verbinden. Darüber hinaus kann jede Verbindung, die Store-Semantik und ein I/O-Protokoll mit einem gewöhnlichen Pin-Set unterstützt, genutzt werden, um einen CPU mit einem FMC zu verbinden. Zudem können PLM-Verbindungen unter Nutzung der physikalischen Ebene der PCIe-Architektur implementiert werden.

[0020] Auf den globalen Speicher kann über den Schalter **228** zugegriffen werden. Der Schalter **228** kann genutzt werden, um mehrere FMCs aus einer Vielzahl von Netzknoten innerhalb eines Systems mit mehreren Netzknoten zu verbinden. In einigen Fällen kann der Schalter **228** ein Stormlake- (STL) Schalter, ein anderer FMC, der als Schalter genutzt wird, oder ein direktes Anschlussgerät sein. Der Schalter kann genutzt werden, um Anfragen für globale Daten zwischen einem oder mehreren Netzknoten zu lenken. In jedem Fall wird der Schalter **228** genutzt, um Nachrichten mit kurzer Wartezeit über den globalen Speicher zu senden. In den Ausführungsformen sind die vielen FMCs entweder direkt mit PLM-Links oder über einen anderen FMC-Schalter miteinander verbunden. Darüber hinaus können in den Ausführungsformen mehrere FMCs durch Tunnelung des PLM-Protokolls über ein Netzwerk-Stack wie STL über einen STL-Schalter verbunden sein.

[0021] Als Resultat der FMCs einer Vielzahl von Netzknoten, die über einen Schalter und PLM-Links miteinander verbunden sind, ist der globale Speicher geteilt und kann über Load-Store-Semantik abgerufen werden. Bei für einen Netzknoten lokalen Berechnungen kann der Netzknoten seinen eigenen Reservespeicher für diese Berechnungen abrufen. Der globale Speicher, der sich auf einer Vielzahl von Netzknoten befindet, kann denselben Speicher haben, und jeder Netzknoten kann Operationen auf diesem Speicher ausführen. Zudem können Netzknoten durch Strategien speziellen Stellen des globalen Speichers zugeordnet werden, und die Strategien können von jedem Knoten oder dem Schalter, der die FMCs der Vielzahl von Netzknoten verbindet, aufrechterhalten werden.

[0022] Anstatt Nachrichten durch ein RMDA zu leiten, werden Load-Store-Semantiken genutzt, um zwischen Netzknoten durch den FMC zu kommunizieren. Jeder FMC umfasst eine Fehler-Isolations-Begrenzung, die dafür sorgt, dass der globale Speicher jedes Netzknotens über den FMC erreicht werden kann, auch wenn die CPUs des Netzknotens versagen. Wie oben diskutiert, kann der geteilte Speicher über ein STL-Netzwerk-Stack oder den PLM-Link abgerufen werden. Jeder FMC der Vielzahl der Netzknoten kann unter der Verwendung von Load-Store-Semantik Nachrichten zwischen den Netzknoten leiten, blockiert jedoch nicht den Datenverkehr der Vielzahl von Netzknoten.

[0023] Die Fehler-Isolations-Begrenzungen eines FMC können unter Verwendung verschiedener Techniken implementiert sein. Bei manchen Ausführungsformen wird Hardware genutzt, um sicherzustellen, dass jeder CPU unabhängig von anderen CPUs innerhalb desselben Netzknotens und Systems ist. Auf diese Art wirkt sich das Versagen unabhängiger CPUs nicht auf die Arbeit anderer CPUs aus. In anderen Ausführungsformen kann das Versagen eines CPU bei anderen CPUs ebenfalls ein Versagen auslösen, jedoch kann der globale Speicher innerhalb des gescheiterten Netzknotens angeschaltet und aktiv sein, sodass der Netzknoten versagen kann, ohne sich auf die Arbeit anderer Netzknoten auszuwirken, und der Speicher des gescheiterten Netzknotens bleibt abrufbar.

[0024] Abb. 3 ist eine Darstellung einer globalen Speicherkarte **300**. Die globale Speicherkarte **300** ist so dargestellt, wie sie von einem oder mehreren FMCs erkannt wird, die als Router oder Schalter agieren, um den Zugriff auf den globalen Speicher über die Netzknoten zu koordinieren. Teile der globalen Speicherkarte können auf einem Netzknoten **302** und einem Netzknoten **306** gespeichert werden. Der globale Speicher kann in mehrere geteilte Regionen **306** unterteilt sein. Der globale Speicher kann von einem FMC geregelt werden wie in **Abb. 2** dargestellt. Demzufolge sind jeder Netzknoten **302** und **306** des globalen Speichers in der globalen Speicherkarte verzeichnet wie in der globalen Speicherkarte **300** vom FMC dargestellt. Im Besonderen kann eine geteilte Speicherregion **308** des Netzknotens **302** eine beliebige Anzahl (1 bis n) von geteilten Speicherregionen umfassen. Eine geteilte Speicherregion **310** des Netzknotens **304** kann eine weitere beliebige Anzahl (1 bis p) von geteilten Speicherregionen umfassen. Der globale Speicher umfasst dann die geteilte Speicherregion **308** (1 bis n) und die geteilte Speicherregion **310** (1 bis p). Jede geteilte Speicherregion kann physisch an ein FMC angeschlossen oder über mehrere FMCs verteilt sein. Darüber hinaus kann die Größe der Speicherregionen variabel oder fix sein. In den Ausführungsformen kann jede Region auf einer pagelevel Abstufung gehalten werden, sodass eine komplette Speicherregion als Teil eines Speichermanagement-Schemas abgerufen werden kann. Wie in **Abb. 2** dargestellt, kann jeder Netzknoten einen lo-

kalen Speicher umfassen, auf den der FMC nicht zugreifen kann und der nicht auf der globalen Speicherkarte **300** verzeichnet ist. Die globale Cluster-Speicherkarte **300** umfasst einen Teil **312**, erkennt eine lokale, kohärente Speicherregion **314** und eine lokale, kohärente Speicherregion **316** als Privatspeicher eines jeden einzelnen Netzknotens, der nicht über die Auslastungs-/Speicherstruktur abrufbar ist.

[0025] Die lokalen kohärenten Speicherregionen **314** und **316** können als eine Nachrichtenregion genutzt werden. Demzufolge umfasst jede der lokalen kohärenten Speicherregionen **314** und **316** eine Nachrichtenregion **318** bzw. eine Nachrichtenregion **320**. Während die lokale Nachrichtenregion **318** und die Nachrichtenregion **320** nicht direkt von einem FMC zugänglich sind, der als Schalter oder Router agiert, um den Speicher über die Netzknoten aufzuteilen, kann der FMC indirekt auf die Nachrichtenregion **322** zugreifen.

[0026] Die geteilte Speicherregion **308** und die geteilte Speicherregion **310** sind für jeden der Netzknoten sichtbar, der denselben Adressbereich hat wie die globale Cluster-Speicherkarte **300**. Jede geteilte Speicherregion kann andere Zugriffsrechte auf jedes Set von Netzknoten haben. Die Zugriffsrechte können auf einem Set von Strategien basieren. Darüber hinaus sind sowohl der Adressbereich einer jeden geteilten Speicherregion als auch jegliche Zugriffsrechte durch ein Set von Bereichsregistern festgelegt. In einigen Fällen können der Adressbereich einer jeden geteilten Speicherregion und die Zugriffsrechte durch eine Seiten-Tabelle, die im Speicher angesiedelt ist, implementiert werden, wenn die Regionen (Super-) Seiten der FMC(s) sind. Der globale Speicher ist in jedem Netzknoten cache-fähig, wenn der Netzknoten die geeigneten Zugriffsrechte besitzt. Jedoch können ein oder mehrere FMCs, die den globalen Speicher regeln, keinen Hardware-basierten Cachekohärenten Mechanismus zwischen den Netzknoten erzwingen. Anstatt dessen ist die Datenkohärenz durch eine Software erzwungen, die auf jedem der Netzknoten läuft.

[0027] Die Nachrichtenregion **318** und die Nachrichtenregion **320** können genutzt werden, um eine Datenkohärenz über die Netzknoten **302** und **304** sicherzustellen. Jeder Netzknoten kann eine Nachricht zu den anderen Netzknoten schicken, die einen Zugang zu einem bestimmten Teil des Speichers haben, und Information hinsichtlich des Status des bestimmten Teils des Speichers anfordern. Beispielsweise kann ein erster Netzknoten anfordern, dass irgendein Netzknoten mit Daten, die zu einer bestimmten Speicherregion gehören, diese Speicherregion aktualisiert, wenn er Daten besitzt, die zu dieser Speicherregion gehören. Jeder Netzknoten, der diesen Teil des Speichers hat, kann auf die Nachricht antworten und den anfordernden ersten Netzknoten darüber informieren, dass die Speicherregion aktualisiert und ausgetauscht wurde. In einigen Fällen ist das Schicken von Nachrichten, um auf den globalen Speicher zuzugreifen, ein Software-basierter Handshake, der ein direkter Speicherzugriff ist und keinen I/O-Stack nutzt, um auf die Daten zuzugreifen.

[0028] Der globale Speicher kann eine Vereinbarung darüber enthalten, welche Netzknoten die Daten im globalen Speicher aktualisieren können; es existiert ein Clustering-Speichermodell mit Handshaking zwischen den Netzknoten. Darüber hinaus können die FMCs die geeigneten Zugriffsrechte für jeden Netzknoten sicherstellen sowie Zugriff auf die Daten eines jeden Netzknotens gewähren, der nicht mehr funktioniert. Dieser Zugriff erfolgt unter Verwendung von Load-Store-Semantik und Hardware ohne die Verzögerung eines I/O-software-Stacks. Darüber hinaus kann der Speicher wie ein flacher Speicher in einer linearen Art per Bytes abgefragt werden, eher als ein Block-Zugriff. In einigen Fällen sind die geteilten Speicherregionen cache-fähig. Weiterhin können die Nachrichtenregionen in manchen Fällen benutzt werden, um Daten zwischen Netzknoten zu leiten, anstatt die FMCs zu nutzen, um Nachrichten zu schicken, die die auf den Netzknoten gespeicherten Daten betreffen.

[0029] **Abb. 4** ist ein Prozess-Flussdiagramm **400** für kohärenten gemeinsamen Speicher über mehrere Cluster. Am Block **402** wird ein cache-fähiger globaler Speicher gebildet. In einigen Fällen ist der cache-fähige globale Speicher befähigt, geteilte Speicherregionen über mehrere Cluster zu nutzen, wobei die geteilten Speicherregionen durch die Nutzung von Load-Store-Semantik abrufbar sind. Am Block **404** sichert die Nutzung eines Software-Mechanismus die Datenkohärenz über mehrere Cluster. Am Block **406** erhält die Nutzung eines Speicher-Controllers die unabhängigen Fehler-Domänen für jedes Cluster aufrecht.

[0030] In manchen Ausführungsformen wird der Speicher-Controller benutzt, um Reliability-Availability-Serviceability (RAS) -Eigenschaften über das System mit mehreren Netzknoten zu gewährleisten. Um unternehmensfähig zu sein, unterstützt der FMC Speicherreplikation, so wie verschiedene Formen von RAIDs über andere FMCs. Auf diese Art wird die Rekonstruktion von Inhalten eines replizierten Speichers ermöglicht, wenn ein FMC oder der ihm zugeordnete globale Speicher versagen. Die Replikation kann eine K-Aryl-Replikation sein, wobei jeder Eintrag in (k-1) Kopien repliziert wird. Das Adressbereichsregister (oder der Page Table) speichert den primären Ort zusammen mit den Backup-Orten. Im Fall von RAID-Schemata erhält der Host-FMC die anderen Adressen und FMCs aufrecht, die zusammen durchsucht werden. Der FMC, der die primären Orte hostet, repliziert den Eintrag in jedem FMC, das den Backup-Ort hostet. Im Fall von RAIDed-Konfiguratio-

nen sendet der hostende FMC die Information zu den RAID-Orten, die die Parität speichern.

[0031] Bei einem Eintrag sendet der FMC, der der primäre für die geschriebene Adresse ist, die Einträge zu den Backup-Orten. In einigen Ausführungsformen sendet der FMC die Einträge zum RAID-Ort, damit der/die FMC(s) die Parität speichern können. Die Backup-FMCs senden den Eintrag zurück zum primären FMC. Auch wenn die Einträge gepostet sind, gilt der Eintrag im primären FMC nicht als vollständig, bis alle Einträge vollständig sind. Der primäre FMC unterhält einen Timer für jedes der anderen FMCs, dem es den Eintrag schickt. Wenn die Beendigung nicht von jedem Ziel-FMC bestätigt wird, kann der primäre FMC sich wegen Zeitüberschreitung abschalten. Darüber hinaus kann der primäre FMC versuchen, die Transaktion zu wiederholen und dabei einen alternativen Pfad zu benutzen, oder er informiert die Systemsoftware, die notwendige Wiederherstellungsaktion durchzuführen.

[0032] Ein Read kann entweder vom primären FMC oder von einem Backup-FMC bereitgestellt werden, wenn die Replikation ermöglicht ist. Das FMC, das dem Netzknoten zugeordnet ist, der die Read-Abfrage generiert, enthält einen Timer. Wenn die Beendigung nicht vor Ablauf der Zeit bestätigt wird, kann eine vorher festgelegte Anzahl von Versuchen, einen alternativen Pfad zum selben FMC oder zu einem Backup-FMC zu finden, durchgeführt werden. Wenn die Transaktion wieder wegen Zeitüberschreitung abgebrochen wird, kann sie den Datenrückfluss verhindern. Der FMC kann ebenfalls den Zeitüberschreitungsfehler dem Softwaresystem melden, um die nötige Korrekturmaßnahme zu bekommen oder einfach, um den Fehler zu verzeichnen. In den Ausführungsformen können die Inhalte zu einem anderen FMC mit freien Kapazitäten transferiert werden, wenn ein FMC oder ein Speichermodul eines FMCs versagt, und die Bereichsregister (oder Page Table Entries) werden entsprechend aktualisiert.

[0033] **Abb. 5** ist ein Blockdiagramm eines Netzknotens **500**, der auf zusammengefasste Speicherressourcen zugreifen kann. Der Netzknoten **500** kann beispielsweise ein Laptop, ein Desktop-Computer, ein Tablet, ein Mobilgerät, ein Server oder Blade-Server oder Ähnliches sein. Der Netzknoten **500** kann auch ein Netzknoten innerhalb einer beliebigen Rack-Scale-Architektur (RSA) mit hoher Speicherdichte sein. In einigen Beispielen ist ein Netzknoten ein beliebiges Gerät, das fähig ist, über das System mit mehreren Netzknoten mit einem anderen Netzknoten zu kommunizieren. Entsprechend ist das System mit mehreren Netzknoten in einigen Beispielen ein Netzwerk von Netzknoten, bei dem jeder Netzknoten ein beliebiges Gerät ist, das über das Netzwerk kommunizieren kann. Zudem ist das System mit mehreren Netzknoten in einigen Beispielen ein Server in einem Rack-Server-System.

[0034] Der Netzknoten **500** kann einen zentralen Prozessor (central processing unit, CPU) 502 beinhalten, die konfiguriert ist, um gespeicherte Befehle auszuführen. Der CPU **502** kann ein Einkernprozessor, ein Mehrkernprozessor, ein Computercluster oder jede Zahl anderer Konfigurationen sein. In einigen Fällen können der CPU **502** und andere Komponenten des Netzknotens **500** als System On Chip (SOC) implementiert sein. Außerdem kann der Netzknoten **500** mehr als ein CPU **502** umfassen. Die Befehle, die vom CPU **502** ausgeführt werden, können genutzt werden, um Speicherressourcen über mehrere Netzknoten zu sammeln.

[0035] Der Netzknoten **500** kann außerdem einen Grafikprozessor (graphics processing unit, GPU) 504 enthalten. Wie dargestellt, kann der CPU **502** über ein Bus **506** mit dem GPU **504** gekoppelt sein. Jedoch können der CPU **502** und der GPU **504** in manchen Ausführungsformen auf demselben Chip lokalisiert sein. Der GPU **504** kann konfiguriert sein, um jede beliebige Zahl von Grafikoperationen innerhalb des Netzknotens **500** auszuführen. Der GPU **504** kann zum Beispiel grafische Bilder, grafische Frames, Videos oder ähnliches rendern oder manipulieren, die einem Benutzer des Netzknotens **500** angezeigt werden. In einigen Fällen beinhaltet der Netzknoten **500** jedoch keinen GPU **504**.

[0036] Der CPU **502** kann auch über den Bus **506** mit einem Eingangs-/Ausgangs- (I/O)-CPU verbunden sein. In den Ausführungsformen wird der I/O-CPU 508 so eingesetzt, dass der CPU **502** auf einen gesammelten Speicher in einem System mit mehreren Netzknoten zugreifen kann. Der CPU **502** kann den gesammelten Speicher abrufen, ohne dabei einen bestimmte Speicher innerhalb des Netzknotens **500** einzuschließen. Weiterhin kann der I/O-CPU 508 einen gesammelten Speicher innerhalb des Systems mit mehreren Netzknoten abrufen, ohne Kommunikations- und Netzwerkprotokolle zu benutzen wie zum Beispiel Transmission Control Protocol und Internet Protocol (TCP/IP) und InfiniBand (IB). In den Ausführungsformen wird eine Verbindung, wie zum Beispiel ein Plattsmouth-(PLM)-Link 510, benutzt, um jeden Netzknoten mit einem gemeinsamen Speicher-Controller zu verbinden, wobei Store-Semantik-basierte Protokolle benutzt werden, die auf einem seriellen Link laufen. Ein Peripheral Component Interconnect Express (PCIe) Link **512** kann benutzt werden, um den CPU **502** mit einem Netzwerk zu verbinden.

[0037] Der CPU **502** kann außerdem über ein Bus **506** mit einer Eingangs-/Ausgangsschnittstelle (I/O) 514 gekoppelt werden, um den Netzknoten **500** mit einem oder mehreren I/O-Geräten 516 zu verbinden. Die I/O-Geräte 516 können z. B. eine Tastatur und

ein Zeigegerät umfassen, wobei das Zeigegerät u. a. ein Touchpad oder einen Touchscreen umfassen kann. Die I/O-Geräte 516 können integrierte Komponenten des Netzknotens **500** sein oder sie können Geräte sein, die extern mit dem Netzknoten **500** verbunden sind. Der CPU **502** kann auch über das Bus **506** mit einer Displayschnittstelle **518** gekoppelt werden, um den Netzknoten **500** mit einem Anzeigegerät **520** zu verbinden. Das Anzeigegerät **520** kann einen Anzeigebildschirm enthalten, der eine integrierte Komponente des Netzknotens **500** ist. Das Anzeigegerät **520** kann außerdem u. a. einen Computermonitor, ein Fernsehgerät oder einen Projektor enthalten, der extern mit dem Netzknoten **500** verbunden ist.

[0038] Das Blockdiagramm von **Abb. 5** soll nicht angeben, dass der Netzknoten **500** alle von den in **Abb. 5** dargestellten Bestandteilen umfassen muss. Außerdem kann der Netzknoten **500** je nach den Details der konkreten Implementierung eine beliebige Anzahl von zusätzlichen, in **Abb. 5** nicht dargestellten Bestandteilen umfassen. Darüber hinaus kann der Netzknoten **500** weniger Komponenten beinhalten als in **Abb. 5** dargestellt. Zum Beispiel muss der Netzknoten **500** kein GPU **504**, keine Schnittstelle für I/O-Geräte 514 und keine Schnittstelle für Displays **518** enthalten.

[0039] Die vorliegenden Techniken ermöglichen einen cache-fähigen, globalen Speicher mit unabhängigen Fehlerdomänen. Der globale Speicher kann von verschiedenen Netzknoten geteilte Daten speichern (z. B. Datenbanken) und zudem für die schnelle Kommunikation zwischen Netzknoten benutzt werden. Wenn der geteilte Speicher persistiert (d. h. in nicht-unstetigen Speichern (NVM)), dauert es nicht lang, bis Operationen nach einem geplanten oder nicht geplanten Ausfall von Netzknoten und einer Verteilung von Aufgaben zwischen Netzknoten wieder aufgenommen werden, weil die Daten schon im Speicher sind. Darüber hinaus gibt es durch eine Software-erzwungene Datenkonsistenz eine explizite Abnahme von modifizierten, cache-fähigen Daten, mit denen Kontrollpunkte zur Wiederherstellung geschaffen werden können, falls ein Netzknoten ausfällt.

[0040] Die vorliegenden Techniken bieten zudem RAS-Features, um das Speicherlevel elastisch zu halten. Darüber hinaus kann der Speicher in einigen Ausführungsformen ein Ersatz für eine Ablage sein. Wenn der Speicher ein nicht-unstetiger Speicher ist, kann eine ganze Datenbank von diesem Speicher aus erstellt werden, sodass Teile der Datenbank nicht von einer CD oder einem Solid-State-Laufwerk (SSD) hochgeladen werden. Auf diese Art wird die Zugriffszeit auf die Datenbank reduziert. In einigen Fällen hat ein nicht-unstetiger Speicher der nächsten Generation eine große Kapazität, die als Ablage dienen, aber mit Store-Semantik abgerufen werden kann. Darüber hinaus gewährleistet der in den vorliegenden Techniken beschriebene nicht-unstetige Speicher dieselbe Speicherelastizität. Der nicht-stetige Speicher kann sehr häufig repliziert werden. Auf diese Art kann jedes beliebige RAID-Schema implementiert werden, um ein hohes Level an Zuverlässigkeit und Fehlerisolation zu bieten.

BEISPIEL 1

[0041] Hier wird ein Apparat für kohärenten, geteilten Speicher über mehrere Cluster beschrieben. Der Apparat besitzt einen Speicher-Controller, einen oder mehrere Netzknoten und einen globalen Speicher. Der Speicher-Controller regelt den Zugriff auf eine geteilte Speicherregion eines jeden Netzknotens, sodass jede geteilte Speicherregion mit Load-Store-Semantik abrufbar ist, auch als Antwort auf ein Versagen des Netzknotens. Jede geteilte Speicherregion wird vom Speicher-Controller im globalen Speicher verzeichnet.

[0042] Der Speicher-Controller kann innerhalb eines oder mehrerer Netzknoten lokalisiert sein. Zudem ermöglicht die Load-Store-Semantik die Kommunikation zwischen diesem einen oder mehreren Netzknoten. Der Speicher-Controller kann zudem die Speicherreplikation unterstützen, sodass der globale Speicher unabhängig vom Status der Netzknoten abrufbar ist. Weiterhin kann der Speicher-Controller alle redundanten Anordnungen unabhängiger Festplatten (RAID) über den globalen Speicher unterstützen, sodass jeder Teil des globalen Speichers im Fall eines Fehlers rekonstruiert werden kann. Der Apparat kann einen Backup-Speicher-Controller umfassen, wobei der Backup-Speicher-Controller im Fall eines Versagens des ersten Speicher-Controllers zum Einsatz kommt. Als Antwort auf ein Versagen des Speicher-Controllers können die Inhalte des gescheiterten Speicher-Controllers auf einen anderen Speicher-Controller übertragen werden. Zudem können die Inhalte des gescheiterten Speichermoduls auf einen anderen Speicher-Controller oder ein anderes Speichermodul übertragen werden, wenn ein einem Speicher-Controller zugeordnetes Speichermodul ausfällt.

BEISPIEL 2

[0043] Hier wird ein System für unterstützte, kohärente geteilte Speicher beschrieben. Das System umfasst einen partiell kohärenten Speicher und einen Speicher-Controller. Der partiell kohärente Speicher umfasst eine Vielzahl von geteilten Speicherregionen einer Vielzahl von Clustern, während eine unabhängige Fehlerdomäne gewährleistet wird, und der Speicher-Controller ermöglicht den Zugriff auf den partiell kohärenten Speicher durch Load-Store-Semantik.

[0044] Die Vielzahl geteilter Speicherregionen kann über einen Plattsmouth-Link, einen Netzwerk-Stack, einen I/O-Stack oder eine beliebige Kombination dieser abgerufen werden. Weiterhin kann die Vielzahl von Clustern auf Daten zugreifen, die in geteilten Speicherregionen abgelegt sind, und die Daten aus den geteilten Speicherregionen in einem lokalen Cache aufbewahren. Die Cluster des partiell kohärenten Speichers können über ein oder mehrere erweiterte Netzwerkschnittstellen-Controller verbunden werden. Weiterhin kann jeder Netzknoten einen lokalen Speicher haben, der von anderen Netzknoten nicht direkt abgerufen werden kann. Die geteilte Speicherregion kann zentralisiert sein, und die unabhängige Fehlerdomäne eines jeden Clusters kann durch eine Fehler-Isolations-Begrenzung gewährleistet werden, die durch den Speicher-Controller implementiert wird.

BEISPIEL 3

[0045] Hier wird eine Methode kohärenter geteilter Speicher über mehrere Cluster beschrieben. Die Methode umfasst die Befähigung eines cache-fähigen globalen Speichers, geteilte Speicherregionen über mehrere Cluster zu nutzen, wobei die geteilten Speicherregionen durch die Nutzung von Load-Store-Semantik abrufbar sind. Die Methode umfasst zudem die Sicherstellung der Datenkohärenz über mehrere Cluster durch einen Software-unterstützten Mechanismus. Weiterhin umfasst die Methode die Aufrechterhaltung einer unabhängigen Fehler-Domäne für jedes Cluster durch einen Speicher-Controller.

[0046] Der Speicher-Controller kann über die Cluster verteilt sein. Zudem kann die Load-Store-Semantik jedes Cluster befähigen, direkt mit einem anderen Cluster zu kommunizieren. Weiterhin ermöglicht eine Fehler-Isolations-Begrenzung die unabhängigen Fehlerdomänen für jedes Cluster.

[0047] In der vorangehenden Beschreibung sind verschiedene Ausführungsformen des offenbarten Gegenstandes beschrieben worden. Zum Zwecke der Erklärung wurden spezifische Anzahlen, Systeme und Konfigurationen dargelegt, um ein gründliches Verständnis des Gegenstandes bereitzustellen. Es ist jedoch für einen Fachmann, der den Vorteil dieser Offenbarung nutzt, selbstverständlich, dass der Gegenstand ohne die spezifischen Details umgesetzt werden kann. In anderen Fällen wurden wohl bekannte Merkmale, Komponenten oder Module weggelassen, vereinfacht, kombiniert oder aufgeteilt, um den offenbarten Gegenstand nicht in den Hintergrund rücken zu lassen.

[0048] Verschiedene Ausführungsformen des offenbarten Gegenstandes können in Hardware, Firmware, Software oder Kombinationen davon implementiert sein, und können unter Bezugnahme auf oder in Verbindung mit Programmcode, wie z. B. Befehlen, Funktionen, Verfahrensweisen, Datenstrukturen, Logik, Anwendungsprogrammen, Designdarstellungen oder Formaten zur Simulation, Emulation und Herstellung eines Designs beschrieben werden, die, wenn von einer Maschine darauf zugegriffen wird, dazu führen, dass die Maschine Aufgaben ausführt, wobei abstrakte Datentypen oder hardwarenaher Zusammenhang definiert wird oder ein Ergebnis erzeugt wird.

[0049] Für Simulationen kann ein Programmcode für Hardware mit einer Hardwarebeschreibungssprache oder einer anderen funktionalen Beschreibungssprache stehen, die ein Modell dessen darstellt, wie die designte Hardware funktionieren sollte. Der Programmcode ist eine Aufbau- oder Maschinensprache, oder Daten, die kompiliert oder interpretiert werden. Weiterhin ist es in diesem Bereich üblich, von Software in einer Form zu sprechen, die die Annahme einer Aktion oder das Auslösen eines Ergebnisses beinhaltet. Solche Ausdrücke sind nur ein kurzer Weg, die Ausführung des Programmcodes durch ein laufendes System zu erklären, was einen Prozessor dazu veranlasst, eine Aktion durchzuführen oder ein Ergebnis zu liefern.

[0050] Programmcode kann beispielsweise in unstetigem und/oder nichtunstetigem Speicher abgelegt werden, wie auf Speichergeräten und/oder einem angegliederten maschinenlesbaren oder maschinenabrufbaren Medium mit Solid-State-Speicher, Festplatten, Floppy-Disks, optischen Speichern, Tapes, Flash-Speichern, Speichersticks, digitalen Videodisks, digitalen versatilen Disks (DVDs), etc., und auch auf exotischeren Medien wie maschinenabrufbaren biologischen statusbewahrenden Speichern. Ein maschinenlesbares Medium kann einen beliebigen konkreten Speichermechanismus umfassen, der maschinenlesbare Informationen sendet oder empfängt, wie Antennen, optische Fasern, Kommunikationsschnittstellen, etc. Programmcode kann in Form von Paketen, seriellen Daten, parallelen Daten etc. übertragen und in komprimiertem oder verschlüsseltem Format genutzt werden.

[0051] Programmcode kann in Programme implementiert werden, die auf programmierbaren Maschinen laufen, wie mobile oder stationäre Computer, Personal Digital Assistants, Set Top Boxen, Mobiltelefone und Pager, und andere elektronische Geräte, von denen jedes einen Prozessor enthält, einen von diesem Prozessor lesbaren unsteten oder nicht unsteten Speicher, mindestens einem Eingabegerät und/oder einem oder mehreren Ausgangsgeräten. Programmcode kann auf die vom Eingabegerät kommenden Daten angewendet werden, um die beschriebenen Ausführungsformen umzusetzen und Ausgabeinformation zu generieren. Die Ausgabeinformation kann auf ein oder mehrere Ausgabegeräte angewendet werden. Ein Fachmann weiß

zu schätzen, dass Ausführungsformen der Offenbarung mit verschiedenen Computersystemkonfigurationen benutzt werden können, einschließlich Multiprozessor- oder Mehrkernprozessorsysteme, Minicomputer, Mainframe-Computer, ebenso wie pervasive oder Miniaturcomputer oder Prozessoren, die in nahezu jedes Gerät eingebettet werden können. Ausführungsformen der Offenbarung können auch in verteilten Computer-Umgebungen praktiziert werden, wobei Aufgaben durch entfernt liegende Datenverarbeitungsgeräte ausgeführt werden, die über ein Kommunikationsnetzwerk verbunden sind.

[0052] Obwohl Operationen als aufeinander folgende Prozesse beschrieben sein können, können einige von ihnen tatsächlich parallel, gleichzeitig und/oder in einer verteilten Umgebung, und mit einem Programmcode, der von Single- oder Multiprozessormaschinen lokal und/oder aus der Ferne abgelegt werden kann, ablaufen. Zusätzlich kann die Abfolge der Operationen in einigen Ausführungsformen neu geordnet werden, ohne vom Gedanken des offenbarten Gegenstands abzuweichen. Der Programmcode kann von oder in Verbindung mit eingebetteten Controllern benutzt werden.

[0053] Während der offenbarte Gegenstand unter Bezugnahme auf veranschaulichende Ausführungsformen beschrieben wurde, soll diese Beschreibung nicht einschränkend ausgelegt werden. Verschiedene Modifikationen der veranschaulichenden Ausführungsformen sowie anderer Ausführungsformen des Gegenstandes, der Fachleuten, die der offenbarte Gegenstand betrifft, offensichtlich ist, sollen als im Umfang des offenbarten Gegenstandes einbegriffen erachtet werden.

Patentansprüche

1. Vorrichtung zur Bereitstellung eines kohärenten, über mehrere Cluster verteilten Speichers, umfassend:
einen Speicher-Controller;
einen oder mehreren Netzknoten, in denen der Speicher-Controller den Zugriff auf eine geteilte Speicherregion eines jeden Netzknotens regelt, sodass jede geteilte Speicherregion mit Load-Store-Semantik abrufbar ist, wobei der Speicher-Controller dafür ausgelegt ist, dass die geteilte Speicherregion jedes Netzknotens über den Speicher-Controller erreicht werden kann, auch wenn die CPUs des Netzknotens versagen; und
einen globalen Speicher, in dem jede geteilte Speicherregion vom Speicher-Controller im globalen Speicher verzeichnet ist.

2. Vorrichtung nach Anspruch 1, in welcher der Speicher-Controller innerhalb des einen oder mehrerer Netzknoten sitzt.

3. Vorrichtung nach Anspruch 1, in welcher die Load-Store-Semantik die Kommunikation zwischen dem einen oder mehreren Netzknoten ermöglicht.

4. Vorrichtung nach Anspruch 1, in welcher der Speicher-Controller die Speicherreplikation unterstützt, sodass der globale Speicher unabhängig vom Status der Netzknoten abrufbar ist.

5. Vorrichtung nach Anspruch 1, in welcher der Speicher-Controller alle redundanten Anordnungen unabhängiger Festplatten (RAID) über den globalen Speicher unterstützt, sodass jeder Teil des globalen Speichers im Fall eines Fehlers rekonstruiert werden kann.

6. Vorrichtung nach Anspruch 1, in welcher dieser Apparat einen Backup-Speicher-Controller umfasst, wobei der Backup-Speicher-Controller im Fall eines Versagens des ersten Speicher-Controllers zum Einsatz kommt.

7. Vorrichtung nach Anspruch 1, in welcher als Antwort auf ein Versagen des Speicher-Controllers die Inhalte des gescheiterten Speicher-Controllers auf einen anderen Speicher-Controller übertragen werden.

8. Vorrichtung nach Anspruch 1, in welcher die Inhalte des gescheiterten Speichermoduls auf einen anderen Speicher-Controller oder ein anderes Speichermodul übertragen werden, wenn ein einem Speicher-Controller zugeordnetes Speichermodul ausfällt.

9. System für unterstützten kohärenten gemeinsamen Speicher, umfassend:
einem partiell kohärenten Speicher, wobei der partiell kohärente Speicher eine Vielzahl von geteilten Speicherregionen einer Vielzahl von Clustern umfasst, während eine unabhängige Fehlerdomäne für jedes Cluster aufrechterhalten wird; und
einem Speicher-Controller, wobei der Speicher-Controller den Zugriff auf den partiell kohärenten Speicher durch Load-Store-Semantik ermöglicht, und wobei der Speicher-Controller dafür ausgelegt ist, dass die geteilte Speicherregion jedes Netzknotens über den Speicher-Controller erreicht werden kann, auch wenn die CPUs des Netzknotens versagen.

10. System nach Anspruch 9, in dem die Vielzahl geteilter Speicherregionen über einen Plattsmouth-Link, einen Netzwerk-Stack, einen I/O-Stack oder eine beliebige Kombination dieser abgerufen werden kann.

11. System nach Anspruch 9, in dem die Vielzahl von Clustern auf Daten zugreifen, die in geteilten Speicherregionen abgelegt sind, und die Daten aus den geteilten Speicherregionen in einem lokalen Cache aufbewahren kann.

12. System nach Anspruch 9, in dem die Cluster des partiell kohärenten Speichers über ein oder mehrere erweiterte Netzwerkschnittstellen-Controller verbunden werden.

13. Verfahren zur Bereitstellung eines kohärenten, über mehrere Cluster verteilten Speichers, dieses Verfahren umfassend:
Befähigen eines cache-fähigen globalen Speichers, geteilte Speicherregionen über mehrere Cluster zu nutzen, wobei die geteilten Speicherregionen durch die Nutzung von Load-Store-Semantik abrufbar sind;
Sicherstellen der Datenkohärenz über mehrere Cluster durch einen Software-unterstützten Mechanismus; und
Aufrechterhalten unabhängiger Fehlerdomänen für jedes Cluster durch die Nutzung eines Speicher-Controllers, wobei der Speicher-Controller dafür ausgelegt ist, dass die geteilte Speicherregion jedes Netzknotens über den Speicher-Controller erreicht werden kann, auch wenn die CPUs des Netzknotens versagen.

14. Verfahren nach Anspruch 13, in welchem der Speicher-Controller über die Cluster verteilt ist.

15. Verfahren nach Anspruch 13, in welchem die Load-Store-Semantik jeden Cluster befähigen, direkt mit einem anderen Cluster zu kommunizieren.

16. Verfahren nach Anspruch 13, in welchem eine Fehler-Isolations-Begrenzung unabhängige Fehlerdomänen für jedes Cluster ermöglicht.

17. Vorrichtung zur Bereitstellung eines kohärenten, über mehrere Cluster verteilten Speichers, umfassend:
ein Mittel, das den Zugriff auf die Speichermodule eines jeden Netzknotens eines Clusters durch Load-Store-Semantik regelt;
ein Mittel, um geteilte Speicherregionen der Speichermodule in einem globalen Speicher zu verzeichnen,
wobei das Mittel, das den Zugriff auf die Speichermodule eines jeden Netzknotens eines Clusters durch Load-Store-Semantik regelt, dafür ausgelegt ist, dass die geteilte Speicherregion jedes Netzknotens über das Mittel erreicht werden kann, auch wenn die CPUs des Netzknotens versagen.

18. Vorrichtung nach Anspruch 17, in welcher die Regelung des Zugriffs auf Speichermodule innerhalb des einen oder mehrerer Netzknoten lokalisiert ist.

19. Vorrichtung nach Anspruch 17, in welcher die Load-Store-Semantik die Kommunikation zwischen dem einen oder mehreren Netzknoten ermöglicht.

20. Vorrichtung nach Anspruch 17, umfassend ein Mittel, das die Speicherreplikation ermöglicht, sodass der globale Speicher unabhängig vom Status der Netzknoten abrufbar ist.

21. Vorrichtung nach Anspruch 17, umfassend ein Mittel für alle redundanten Anordnungen unabhängiger Festplatten (RAID) über den globalen Speicher, sodass jeder Teil des globalen Speichers im Fall eines Fehlers rekonstruiert werden kann.

22. Konkretes, maschinenlesbares Medium, welches Code umfasst, der bei seiner Ausführung einen Prozessor veranlasst:
einen cache-fähigen globalen Speicher zu ermöglichen, der geteilte Speicherregionen über mehrere Cluster nutzt, wobei die geteilten Speicherregionen durch die Nutzung von Load-Store-Semantik abrufbar sind;
die Datenkohärenz über mehrere Cluster durch einen Software-unterstützten Mechanismus sicherzustellen; und
unabhängige Fehlerdomänen für jedes Cluster durch die Nutzung eines Speicher-Controllers aufrechtzuerhalten, wobei der Speicher-Controller dafür ausgelegt ist, dass die geteilte Speicherregion jedes Netzknotens über den Speicher-Controller erreicht werden kann, auch wenn die CPUs des Netzknotens versagen.

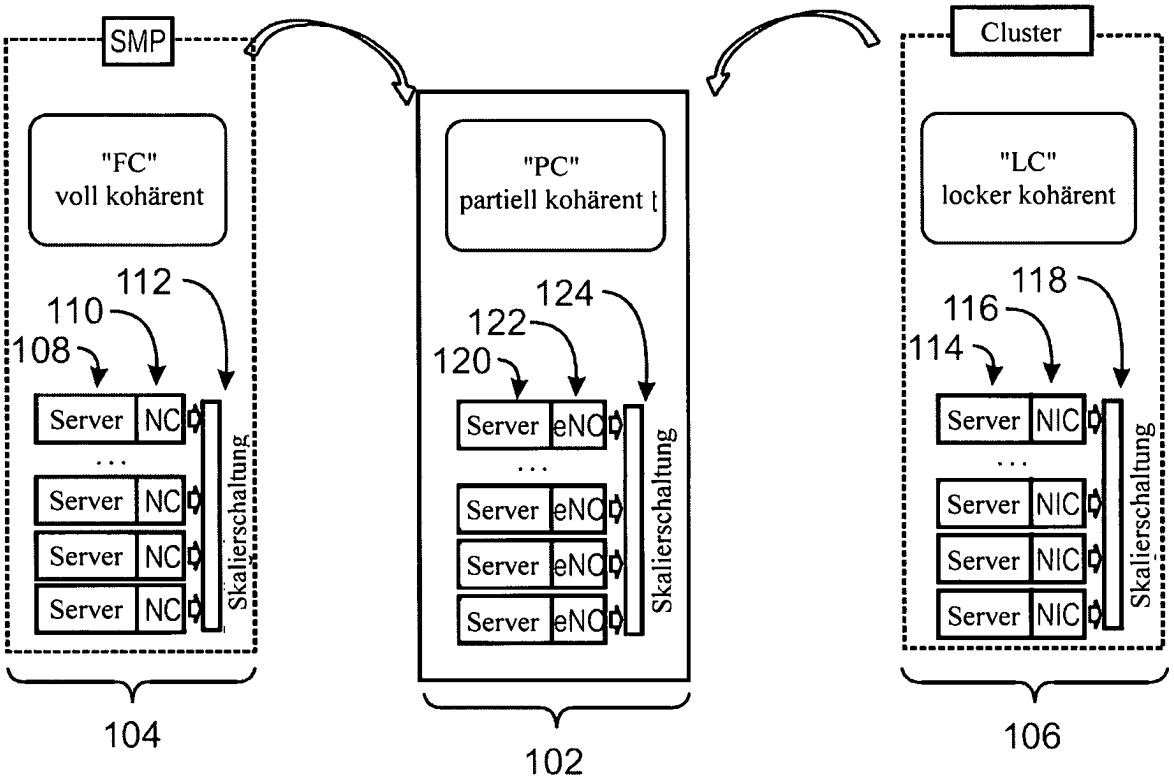
23. Konkretes, maschinenlesbares Medium nach Anspruch 22, wobei der Speicher-Controller über die Cluster verteilt ist.

24. Konkretes, maschinenlesbares Medium nach Anspruch 22, wobei die Load-Store-Semantik jedes Cluster befähigt, direkt mit einem anderen Cluster zu kommunizieren.

25. Konkretes, maschinenlesbares Medium nach Anspruch 22, wobei eine Fehler-Isolations-Begrenzung unabhängige Fehlerdomänen für jedes Cluster ermöglicht.

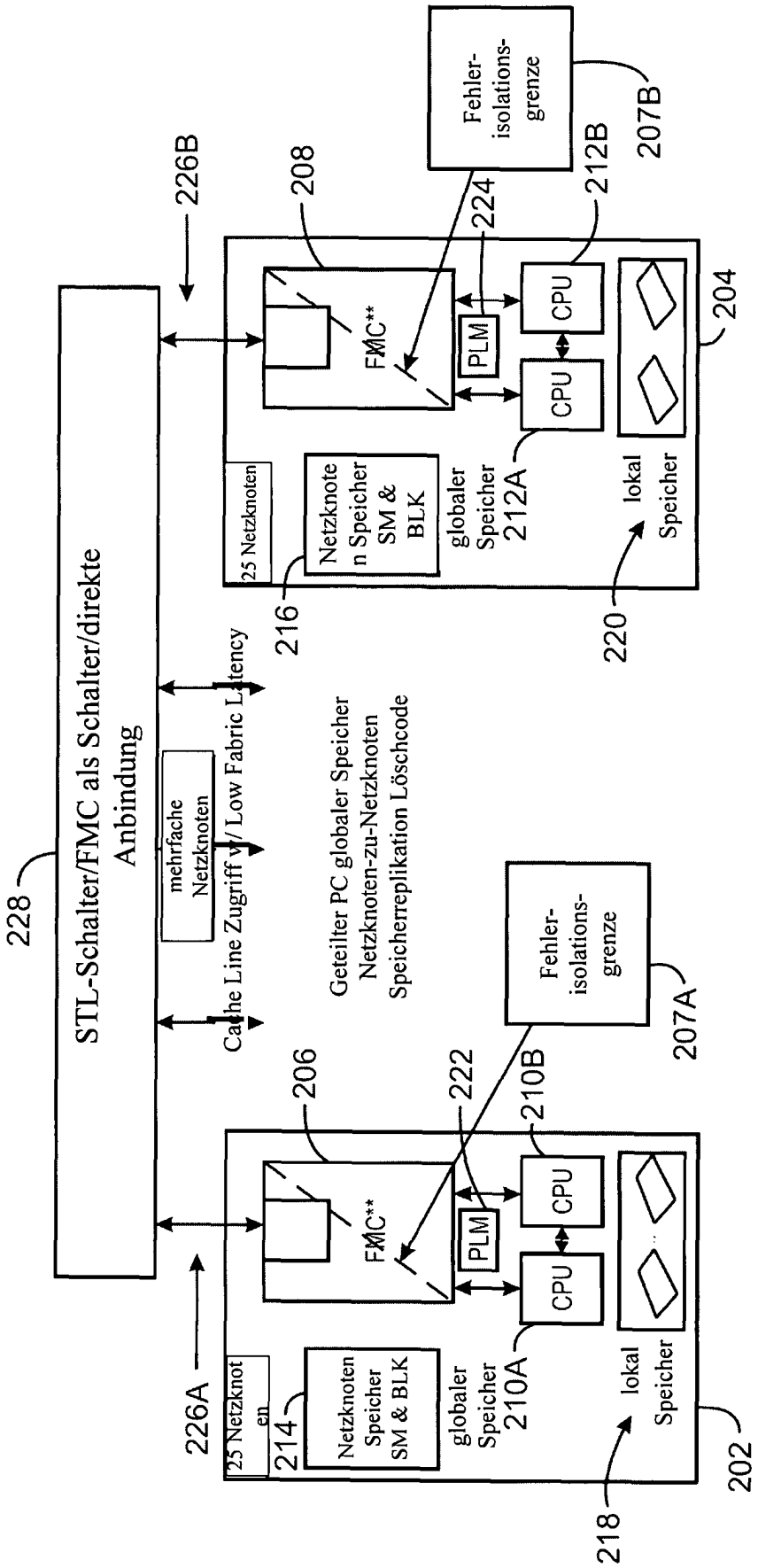
Es folgen 5 Seiten Zeichnungen

Anhängende Zeichnungen



100

FIG. 1


STL-Schalter/FMC als Schalter/direkte Anbindung
228
226B
226A
mehrfache Netzknoten
Cache Line Zugriff w/ Low Fabric Latency
Geteilter PC globaler Speicher
Netzknoten-zu-Netzknoten
Speicherreplikation Löschcode
208
206
FMC**
PLM
CPU
CPU
224
222
212B
210B
210A
212A
Fehler-isolations-grenze
207B
207A
25 Netzknoten
214
216
Netzknoten Speicher SM & BLK
globaler Speicher
lokal Speicher
218
220
202
204

200
FIG. 2

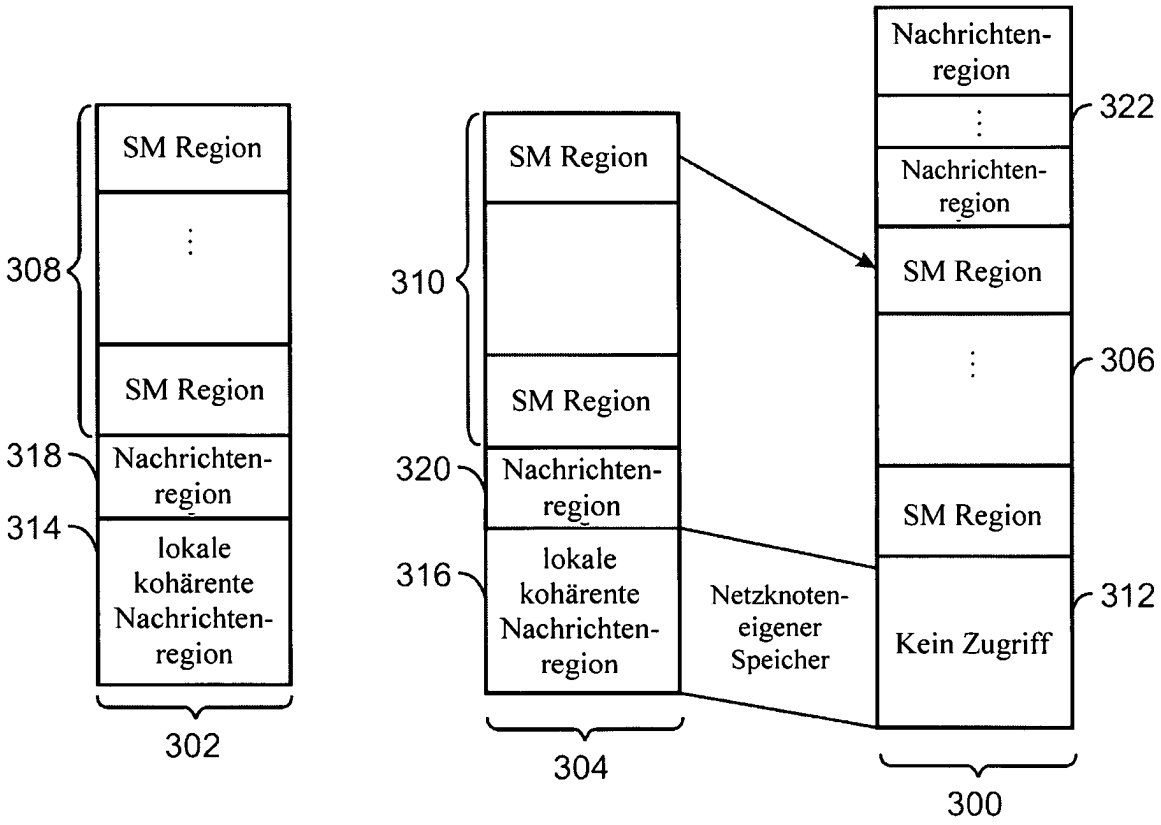
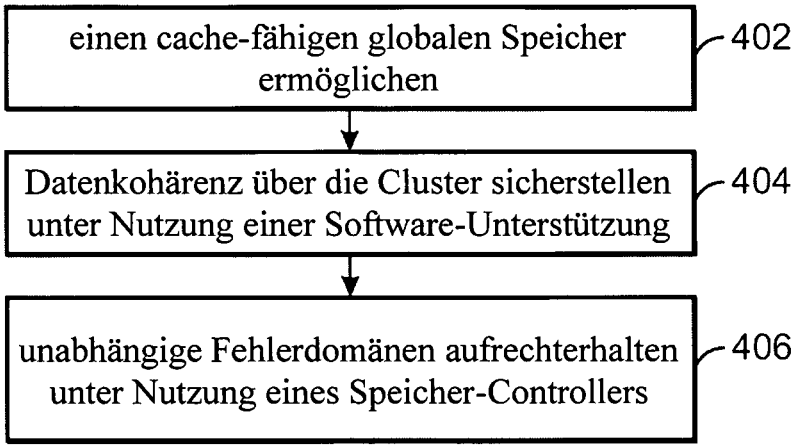

SM Region
SM Region
Nachrichten-region
lokale kohärente Nachrichten-region
308
318
314
302
SM Region
SM Region
Nachrichten-region
lokale kohärente Nachrichten-region
310
320
316
304
Netzknoten-eigener Speicher
Nachrichten-region
Nachrichten-region
SM Region
SM Region
Kein Zugriff
322
306
312
300

FIG. 3


einen cache-fähigen globalen Speicher ermöglichen
402
Datenkohärenz über die Cluster sicherstellen unter Nutzung einer Software-Unterstützung
404
unabhängige Fehlerdomänen aufrechterhalten unter Nutzung eines Speicher-Controllers
406

400

FIG. 4

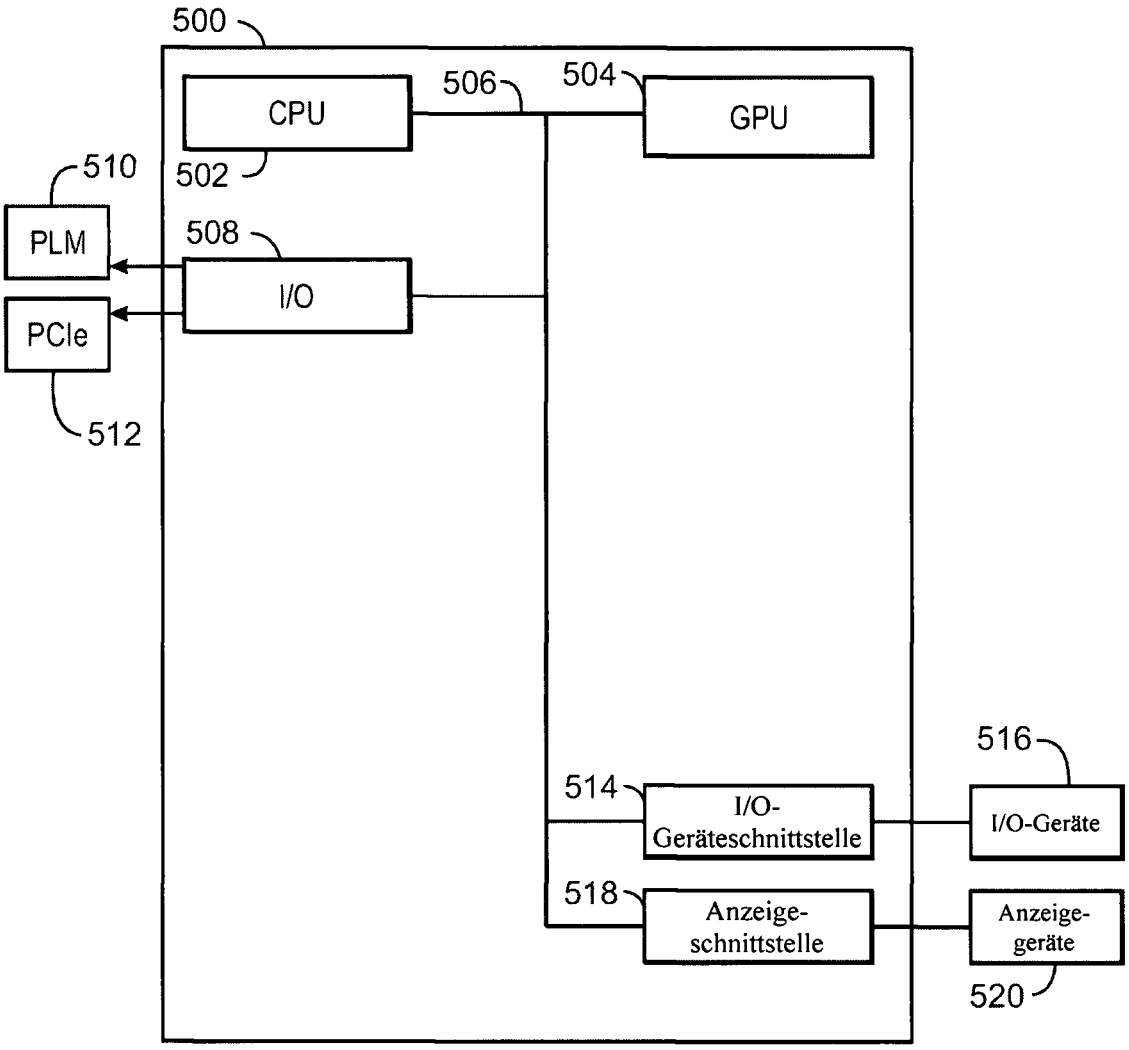


FIG. 5

