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

(10) **DE 102 97 430 T5** 2005.01.27

(12) # Veröffentlichung

der internationalen Anmeldung mit der
(87) Veröffentlichungs-Nr.: **WO 03/040932**
in deutscher Übersetzung (Art. III § 8 Abs. 2 IntPatÜG)
(21) Deutsches Aktenzeichen: **102 97 430.6**
(86) PCT-Aktenzeichen: **PCT/US02/35460**
(86) PCT-Anmeldetag: **04.11.2002**
(87) PCT-Veröffentlichungstag: **15.05.2003**
(43) Veröffentlichungstag der PCT Anmeldung
in deutscher Übersetzung: **27.01.2005**

(51) Int Cl.[7]: **G06F 12/16**
G06F 13/28

(30) Unionspriorität:
10/036749 08.11.2001 US

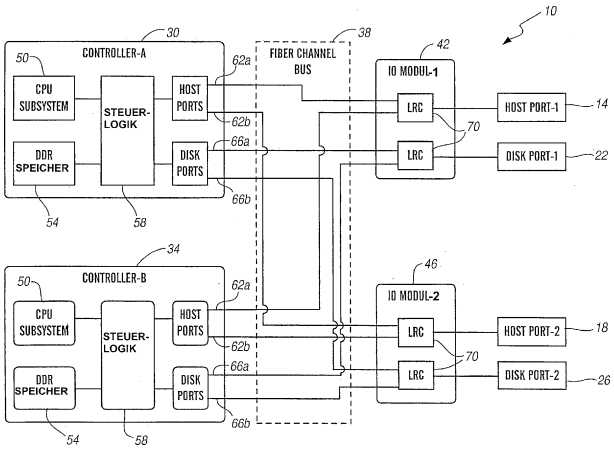
(74) Vertreter:
Grosse, Bockhorni, Schumacher, 81476 München

(71) Anmelder:
Chaparral Network Storage Inc., Longmont, Col., US

(72) Erfinder:
Busser, Richard W., Longmont, Col., US; Davies, Ian R., Longmont, Col., US

(54) Bezeichnung: **Datenspiegelung unter Anwendung von shared Bussen**

(57) Hauptanspruch: Verfahren zum Spiegeln von Daten in einem Speichersystem, das ein Speicher-Array enthält, das umfasst:
dass ein erstes Controllermanagementmodul zur Verfügung gestellt wird, das eine erste Direct-Memory-Access-Engine umfasst, wobei dieses erste Controllermanagementmodul zum Steuern von Lese-/Schreiboperationen den Speicher-Array einbezieht;
dass ein zweites Controllermanagementmodul zur Verfügung gestellt wird, das eine zweite Direct-Memory-Access-Engine umfasst, wobei das zweite Controllermanagementmodul zum Steuern von Lese-/Schreiboperationen den Speicher-Array einbezieht, wobei die erste Direct-Memory-Access-Engine benutzt wird zum Spiegeln von Daten zum zweiten Controllermanagementmodul und die zweite Direct-Memory-Access-Engine genutzt wird zum Spiegeln von Daten zum ersten Controllermanagementmodul;
und das Spiegeln der ersten Daten von dem ersten Controllermanagementmodul zum zweiten Controllermanagementmodul unter Anwendung der ersten Direct-Memory-Access-Engine erfolgt.



Beschreibung

Bereich der Erfindung

[0001] Die vorliegende Erfindung bezieht sich auf das Speichern von Daten in einer Mehrcontrollerkonfiguration, und insbesondere auf das Spiegeln von Daten, wobei Direct-Memory-Access-Engines genutzt werden.

Hintergrund der Erfindung

[0002] Netzwerkspeichercontroller werden typischerweise benutzt, um ein Host-Computersystem mit peripheren Speichereinheiten, wie z. B. Disk-Laufwerken oder Bandlaufwerken, zu verbinden. Der Netzwerkspeichercontroller arbeitet als eine Schnittstelle zwischen dem Host-Computer und den peripheren Speichereinheiten. Bei vielen Anwendungen führt der Netzwerkspeichercontroller das Abarbeiten von Funktionen auf Daten, die zwischen dem Host-Computer und den peripheren Einheiten übertragen werden, durch. Eine übliche Anwendung eines solchen Systems ist ein redundantes Array von billigen Disks (RAID). Ein RAID-System speichert Daten auf mehreren Disklaufwerken, um die Daten gegen einen Disklaufwerkausfall zu schützen. Wenn ein Disklaufwerk ausfällt, dann ist das RAID-System in der Regel in der Lage die Daten, die auf dem ausgefallenen Laufwerk gespeichert waren aus den verbleibenden Laufwerken in dem Array zu rekonstruieren. Ein RAID-System nutzt einen Netzwerkspeichercontroller, der in vielen Fällen einen RAID-Controller umfasst, als eine Schnittstelle zwischen dem Host-Computer und dem Array der Disklaufwerke.

[0003] Viele Anwendungen verlangen ein Speichersystem, um eine sehr hohe Verfügbarkeit zu haben. Diese hohe Verfügbarkeit ist ein Schlüsselanliegen bei vielen Anwendungen, wie z. B. Finanzinstituten und Airline-Reservierungssysteme, da die Anwender sich auf die Daten, die in den RAID-System gespeichert sind, voll und ganz verlassen. Bei dieser Art der Anwendungen kann eine Nichtverfügbarkeit von Daten, die auf dem RAID-System gespeichert sind, zu signifikanten Ertragsverlusten und/oder Kundenunzufriedenheit führen. Der Einsatz eines RAID-Systems bei einer solchen Anwendung erhöht die Verfügbarkeit der gespeicherten Daten, da falls ein einzelnes Diskettenlaufwerk ausfällt, die Daten immer noch abgespeichert werden können und aus dem System zurückerhalten werden. Zusätzlich zur Anwendung eines RAID-Systems, ist es üblich, redundante RAID-Controller zu nutzen, um die Verfügbarkeit des Speichersystems weiter zu erhöhen. In einer derartigen Situation werden zwei oder mehr Controller in einem RAID-System genutzt, wobei jeder Controller eine Ausfallssicherungseinrichtung hat, wobei falls einer der Controller ausfällt, der andere übrigbleibende Controller die Operationen für den ausgefallenen Controller übernimmt. Solch eine Plattform erweitert die Verfügbarkeit des RAID-Systems, jedoch kann es zu verschiedenen Nachteilen führen, die später diskutiert werden.

[0004] **Fig.** 1 zeigt eine Darstellung eines Blockdiagramms eines zweifachen Controller konfigurierten RAID-Netzwerkspeichercontrollers **10**, wobei eine Fibre-Channel zu Fibre-Channel Verbindung gezeigt wird. Bei diesem Beispiel kommunizieren beide, der Host-Computer und das Array der Disklaufwerke, mit dem Netzwerkspeichercontroller unter Ausnutzung von Fibre-Channel-Verbindungen. Während Fibre-Channel ein übliches Kanalmedium in solchen Systemen ist, sollte es klar sein, dass andere Kanäle ebenso genutzt werden können z.B. Small Computer System Interface (SCSI) oder Ethernet. Das RAID-System, das in **Fig.** 1 gezeigt wird, umfasst zwei Host-Ports, Host-Port-1 **14** und Host-Port-2 **18** und zwei Disk-Ports, Disk-Port-1 **22** und Disk-Port-2 **26**. Jeder Host-Port **14**, **18** kann mit verschiedenen Host-Computern gezoned werden, und jeder Disk-Port **22**, **26** kann mit verschiedenen Disk-Arrays gezoned werden, wie es bei RAID-Systemen üblich ist und wie es in der Technik wohl bekannt ist. Der Netzwerkspeichercontroller **10** umfasst zwei RAID-Controller, Controller-A **30** und Controller-B **34**. In einem System, das das Zonen von Controllern anwendet, kann Controller-A **30** mit dem Host-Port-1 **14** und Disk-Port-1 **22** gezont werden und Controller-B **34** kann mit dem Host-Port-2 **18** und Disk-Port-2 **26** gezont werden.

[0005] Wie es in der Technik bekannt ist, verlangen Systeme, die zweifache Controller einsetzen, die Spiegelung der Daten zwischen Controllern um Cache-Kohärenz zu erhalten. Jeder Controller **30**, **34** muss eine Kopie der Daten und des Status des anderen Controllers haben, um zwischen den Controllern Redundanz zu erhalten und um die Arbeit des RAID-Systems aufrecht zuhalten, falls einer der Controller ausfällt. Das Spiegeln der Daten zwischen den Controllern kann die Performance eines RAID-Systems verringern, da das Übertragen der Daten zwischen den Controllern die Prozessresourcen der Controller nutzt, genau wie die Kanalbandbreite, wie im Detail später näher erläutert wird.

[0006] Die Controller **30**, **34** sind mit einem Fibre-Channel Bus **38** verbunden, der mit zwei IO-Modulen verbunden ist, IO-Modul-1 **42** und IO-Modul-2 **46**. Jeder Controller **30**, **34** umfasst ein CPU-Subsystem **50**, einen Double Data Rate (DDR) Speicher **54**, Steuerlogik **58**, eine zweifache Port Fibre-Channel Verbindung mit zwei Host-Ports **62a**, **62b** und eine zweifache Port Fibre-Channel Verbindung mit zwei Disk-Ports **66a**, **66b**. Das CPU-Subsystem **58** führt Aufgaben aus, die erforderlich sind für die Speicherung der Daten auf einem Array von Disks, einschließlich dem striping der Daten, und dem Starten

und Ausführen der Lese- und Schreibkommandos. Der DDR-Speicher **54** ist ein nichtflüchtiges Speicher-Array für Daten und andere Informationen. Die Steuerlogik **58** führt verschiedene Funktionen aus, wie z.B. Herstellen von Schnittstellen mit dem CPU-Subsystem **50**, dem DDR-Speicher **54**, und den Host-Ports **62a**, **62b** und den Disk-Ports **66a**, **66b**. Die Steuerlogik **58** kann ebenso andere Funktionen enthalten, einschließlich einer Paritätserzeugungsfunlction, wie z.B. eine exklusiv OR (XOR)-Engine. Die Host-Ports **62a**, **62b** und Disk-Ports **66a**, **66b** ermöglichen die Kommunikation mit dem Fibre-Channel Backplane **38**. Die IO-Module **42**, **46** enthalten Link Resiliency Circuits (LRCs) **70**, ebenso bekannt als Port-Bypass-Circuits, dessen Funktion ist es jeden Host-Port **14**, **18** und jeden Disk-Port **22**, **26** mit jedem Controller **30**, **34** zu verbinden. Dies ermöglicht es beiden Controllern **30**, **34** Zugang zu beiden Host-Ports **14**, **18** und beiden Disk-Ports **22**, **26** zu erhalten.

[0007] Um volle Redundanz zu ermöglichen, muss jeder Controller eine Verbindung mit jedem Host-Port **14**, **18** und jedem Disk-Port **22**, **26** haben. Auf diese Weise, falls einer der Controller ausfällt, kann der andere Controller die Operationen fortführen. Jedoch, beim Gebrauch der Zoning-Techniken, um die Performance eines RAID-Systems zu erhöhen, sind die Hälfte dieser Ports passiv. Z. B. wenn Controller-A **30** mit einem Host-Port-1 **14** und ein Disk-Port-1 **22** gezoned wird, dann erhält Controller-A **30** die gesamte Kommunikation von Host-Port-1 **14** und steuert die Disk-Arrays) auf Disk-Port-1 **22**. Desgleichen, würde Controller-B **34** zum Host-Port-2 **18** und Disk-Port-2 **26** gezoned werden. Diese Zoning-Techniken sind in der Technik wohlbekannt und können sowohl die Performance eines RAID-Systems steigern als auch die Steuerung und Kommunikation der zwei Controller **30**, **34** vereinfachen. Bei dem Beispiel auf **Fig.** 1, sind auf dem Controller-A **30** die Host-Port Verbindung **62a** und die Disk-Port Verbindung **66a** mit dem Host-Port-1 **14** und Disk-Port-1 **22**, bzw. durch die LRCs **70** des IO-Moduls-1 **24** verbunden. Da Controller-A **30** mit Host-Port-1 **14** und Disk-Port-1 **22** gezoned ist, kommuniziert die Host-Port Verbindung **62a** und die Disk-Port Verbindung **66a** mit dem Host-Port-1 **14** und Disk-Port-1 **22** aktiv. Die verbleibende Host-Port Verbindung **62b** und Disk-Port Verbindung **66b** werden mit dem Host-Port-1 **18** und Disk-Port-2 **26**, bzw. durch die LRCs **70** des IO-Moduls-2 **46** verbunden. Diese Verbindungen sind üblicherweise passive Verbindungen, da Controller-A **30** nicht aktiv mit dem Host-Port-2 **18** und Disk-Port-2 **26** kommuniziert, so lange wie Controller-B **34** nicht ausfällt. Desgleichen würde Controller-B **34** mit dem Host-Port-2 **18** und Disk-Port-2 **26** gezoned werden. So dass auf Controller-B **34** die Host-Port Verbindung **62b** und Disk-Port Verbindung **66b** mit Host-Port-2 **18** und Disk-Port-2 **26** durch LRCs **70** des IO-Moduls-2 **46** kommunizieren würden. Die übrigbleibende Host-Port Verbindung **62a** und Disk-Port Verbindung **66a** würde mit dem Host-Port-1 **14** und dem Disk-Port-1 **22** durch LRCs **70** des IO-Moduls-1 **24** verbunden werden.

[0008] Wie oben erwähnt, wird bei typischen redundanten Controlleroperationen Daten zwischen den Controllern gespiegelt. Wenn Daten zwischen Controller-A **30** und Controller-B **34** gespiegelt werden, ist es üblich die gespiegelten Daten über die shared Disk-Port Verbindungen, insbesondere Disk-Port Verbindung **66b** des Controller-A **30**, und Disk-Port Verbindung **66a** des Controller-B, zu übertragen. Z. B. kann Controller-B **34** Daten über Host-Port-2 **18** erhalten, die auf ein Array von Disks über Disk-Port-2 geschrieben werden sollen. Controller-B **34** würde diese Daten erhalten und sie in Speicher **54** abspeichern. Um Cache-Kohärenz zu erhalten, muss Controller-B **34** diese Daten ebenso dem Controller-A **30** mitteilen, so dass beide Controller die Daten haben und falls einer ausfällt, der andere noch in der Lage ist, die Daten zu schreiben.

[0009] In einem traditionellen System, wird dieses Spiegeln über verschiedene Stufen erledigt. **Fig.** 12 ist eine Flow-Chart-Darstellung der Schritte, die erforderlich sind, um zwischen zwei Controllern bei einem aktiv/aktiv Controllerpaar Daten zu spiegeln. Zuerst erhält Controller-B **34** Daten, die auf das Disk-Array geschrieben werden sollen, wie durch Block **80** angezeigt. Um die Daten zu spiegeln, gibt Controller-B **34** ein erstes Spiegelungskommando heraus, das einen ersten Interrupt am Controller-A **30** erzeugt, der Controller-A **30** anzeigt, dass eine Botschaft gesendet wurde, wie durch Block **82** angezeigt wird. Ein Interrupt ist ein Signal, das automatisch durch Hardware auf einem Controller erzeugt wird, wenn eine Nachricht erhalten wurde, in diesem Beispiel Controller-B, zu einem Prozessor, in diesem Beispiel Hardware auf dem Controller-A, der dazu führt, dass der Prozessor aufhört mit dem was er tut und den Interrupt bedient. Wenn Controller-A den ersten Interrupt erhält, unterbricht er jede Prozessaktivität und führt den ersten Spiegelungsbefehl durch. Controller-B **34** gibt als nächstes einen zweiten Spiegelungsbefehl aus, der Metadaten umfasst, welche einen zweiten Interrupt erzeugen, wie durch den Block **84** angezeigt. Die Metadaten enthalten den aktuellen message body und Informationen, die dem Controller-A **30** den Speicherort anzeigen, an welchem die Benutzerdaten zu speichern sind. Als nächstes markiert Controller-A **30** die Inhalte seines nichtflüchtigen Speichers (NV-RAM) als ungültig für die Datenblöcke, was in den Metadaten spezifiziert ist, wie angezeigt durch Block **86**. Als nächstes gibt Controller-B **34** einen dritten Spiegelungsbefehl heraus, der Benutzerdaten umfasst, die dazu führen, dass ein dritter Interrupt ausgelöst wird, entsprechend dem Block **88**. Controller-A erhält die Benutzerdaten, speichert die Benutzerdaten in dem spezifizierten Ort in seinem NVRAM und

markiert die Inhalte des NVRAM als gültig für die spezifizierten Datenblöcke, wie durch Block **90** dargestellt. Wenn einmal der Controller-B **34** die dazu gehörende Schreiboperation komplettiert hat, gibt er einen vierten Spiegelungsbefehl heraus, der einen vierten Interrupt und eine Bekanntgabe, dass das Schreiben komplett ist, verursacht, wie durch den Block **94** angezeigt. Controller-A markiert dann, dass das Schreiben komplett ist, wie durch Block **94** angezeigt.

[0010] Wie ersichtlich ist, kann diese Spiegelungstechnik, wenn diese Spiegelungstechnik beim Kopieren von Daten zwischen den Controllern erfolgreich ist, erhebliche Prozessressourcen in Beschlag nehmen. Jede Schreiboperation benötigt vier Interrupts, welche den erhaltenden Prozessor dazu bewegen, dass er alle Tasks unterbricht, die er aktuell durchführt und den Interrupt bedient. Deshalb würde es vorteilhaft sein, ein Netzwerkspeichercontroller zu haben, der weniger Prozessressourcen für das Spiegeln der Daten benötigt.

[0011] Zusätzlich wird das Spiegeln typischerweise unter Benutzung der Disk-Channels erledigt. In jeder der oben beschriebenen Spiegelungsbefehle, sendet der Controller-B **34** die Daten über die Disk-Port-Verbindung **66a** welche eine Verbindung mit dem LRC **70** herstellt, der mit dem Disk-Port-1 **22** verbunden ist. Die Daten werden durch den LRC **70** übertragen, wo sie dann an der Disk-Port-Verbindung **66a** auf dem Controller-A erhalten werden. Controller-A erhält dann die Daten und führt entsprechende Prozess- und Speicherschritte durch. Desgleichen wenn der Controller-A **30** Daten erhält, die in die Arrays der Disks am Disk-Port-1 **22** geschrieben werden sollen, sendet er die Daten dem Controller-B **34** unter Benutzung derselben Spiegelungstechnik. Beachte dass diese Technik keine dedizierten Disk-Ports benötigt, und mehr als ein Disk-Port genutzt werden kann.

[0012] Während dies den übrigbleibenden Disk-Port auf jedem Controller nutzt, bleibt der zweite Host-Port auf jedem Controller ungenutzt, darum passiv, während des normalen Betriebs des Systems. Die passiven Ports auf jedem Controller addieren sich zu einer signifikanten Menge an Hardware des Controllers und können signifikante Posten des Netzwerkspeichercontrollers **10** ergeben. Darum wäre es vorteilhaft, wenn ein redundanter Netzwerkspeichercontroller bereitgestellt werden würde, welcher eine hohe Verfügbarkeit bei reduzierten Kosten und Hardware verbunden mit passiven Ports, die auf dem Controller angeordnet sind, zur Verfügung stellen würde.

[0013] Zusätzlich resultiert aus dem Spiegeln von Daten in solch einem System, dass die gespiegelten und abgespeicherten Daten über den gleichen Port für den Controller gesendet werden, der die gespiegelten Daten erhalten oder genutzt hat, um die Daten zur Disk zu übertragen. Die Bandbreite zu und von der Disk-Array wird von den gespiegelten Daten benötigt, was die Performance des Netzwerkspeichercontrollers reduzieren kann. So dass es vorteilhaft wäre einen Netzwerkspeichercontroller zu haben, der wenig oder keine Disk-Channel-Bandbreite beim Spiegeln der Daten zwischen Controllern benötigt.

[0014] Ferner erfordern RAID-Controller mit der kontinuierlichen Steigerung des Bedarfs an Datenspeicherung oft Upgrades mit zusätzlichen Disk-Laufwerken oder schnelleren Bus-Schnittstellen. Jedoch kann ein RAID-Controller nicht konfiguriert werden, um zusätzliche Bus-Schnittstellen-Kapazität hinzuzufügen oder kann nicht einen neuen Typ einer Bus-Schnittstelle unterstützen. Solche Controller müssen normalerweise ersetzt werden, wenn ein Upgrade durchgeführt wird. Dieses Ersetzen der Controller kann die Kosten eines Upgrades eines RAID-Systems erhöhen. Das Ersetzen eines sich im Einsatz befindenden RAID-Controllers stellt einen Wertverlust dar, der die Entscheidung für ein Upgrade eines RAID-Systems blockieren kann. Darum wäre es vorteilhaft ein System zu haben, das Upgrades der Kapazität sowie neue Schnittstellentypen mit leichter Handhabung und reduzierten Kosten unterstützen könnte.

[0015] Entsprechend gibt es einen Bedarf eine Vorrichtung und Verfahren zu entwickeln zum Gebrauch in einem Netzwerkspeichercontroller, welcher: (1) Redundanz mit reduzierten Kosten für passive Komponenten bereitstellt, (2) die Menge an gespiegelten Daten welche über die Disk- oder Host-Ports gesendet werden, reduziert, (3) den Overhead an Prozessabläufen, der beim Spiegeln der Daten erzeugt wird, reduziert und (4) leicht ersetzbare und upgradefähige Komponenten bereitstellt.

Zusammenfassung der Erfindung

[0016] In Übereinstimmung mit der vorliegenden Erfindung, werden ein Verfahren und eine Vorrichtung zur Verfügung gestellt zum Spiegeln von Daten in einem Speichersystem, das ein Speicherarray umfasst. Die Vorrichtung umfasst ein erstes Controllermangagementmodul, das einen ersten Prozessor und eine erste Direct-Memory-Access-Engine enthält. Der erste Prozessor wird zum Steuern der Lese- und Schreibbefehle genutzt, die den Speicherarray einbeziehen. Die erste Direct-Memory-Access-Engine wird zum Speichern der Daten benutzt, die durch das erste Controllerspeichermodul erhalten werden. Die Vorrichtung enthält ebenso ein zweites Controllermanagementmodul, das einen zweiten Prozessor und eine zweite Direct-Memory-Access-Engine enthält. Der zweite Prozessor wird zum Steuern der Lese- und Schreibbefehle genutzt, die den Speicherarray einbeziehen. Die zweite Direct-Management-Access-Engine kann zum Übertragen von Daten von ei-

nem zweiten Controllermanagementmodul zu einem ersten Controllerspeichermodul genutzt werden. Daten werden von dem ersten Controllermanagementmodul zum zweiten Controllermanagementmodul gespiegelt, dabei wird die erste Direct-Memory-Access-Engine genutzt während ein Unterbrechen des zweiten Prozessors vermieden wird. Die erste Direct-Memory-Access-Engine ist separat, aber in Kommunikation mit dem ersten Prozessor und der erste Prozessor steuert das Spiegeln der Daten unter Ausnutzung der ersten Direct-Memory-Access-Engine. In einer Ausführungsform enthält das erste Controllermanagementmodul ein Field-Programmable-Gate-Array. Die erste Direct-Memory-Access-Engine ist in Kommunikation mit zumindest Teilen des Field-Programmable-Gate-Array und die erste Direct-Memory-Access-Engine kann ein Teil des Field-Programmable-Gate-Array sein.

[0017] In einer Ausführungsform enthält die Vorrichtung ein erstes Kanalschnittstellenmodul, welches einen ersten shared Path hat. Das erste Kanalschnittstellenmodul kommuniziert mit dem ersten Controllerspeichermodul und der erste shared Path wird zum übertragen von Daten zwischen dem ersten Controllermanagementmodul und dem zweiten Controllermanagementmodul genutzt. Eine passive Backplane verbindet das erste Kanalschnittstellenmodul mit dem ersten Controllermanagementmodul. Der zweite Prozessor steuert die Operationen, die mit dem zweiten Controllermanagementmodul verbunden sind, während die Daten zum zweiten Controllerspeichermodul gespiegelt werden. Die Daten werden zum zweiten Controllermanagementmodul gespiegelt unabhängig zur zweiten Direct-Memory-Access-Engine. Innerhalb des zweiten Controllermanagementmoduls gibt es nichtflüchtige Speicher und Daten können in dem nichtflüchtigen Speicher gespeichert werden unabhängig von dem zweiten Prozessor. Die erste Direct-Memory-Access-Engine markiert Teile des nichtflüchtigen Speichers, in dem die Daten gespeichert werden, als ungültig, und überträgt die Daten zum nichtflüchtigen Speicher. Die Teile des nichtflüchtigen Speichers, in denen die Daten gespeichert sind, werden dann als gültig gekennzeichnet.

[0018] Das Verfahren beinhaltet das Spiegeln der Daten von dem ersten Controllermanagementmodul zum zweiten Controllermanagementmodul unter Ausnutzung der ersten Direct-Memory-Access-Engine. Der erste Prozessor innerhalb des ersten Controllermanagementmoduls bestimmt, dass Datenspiegelung durchgeführt werden muss. Der zweite Prozessor innerhalb des zweiten Controllermanagementmoduls steuert Lese- und Schreiboperationen, die das Speicherarray einbeziehen und das Spiegeln der Daten wird durchgeführt, während ein Unterbrechen des zweiten Prozessors vermieden wird. Daher kann der zweite Prozessor seine eigenen Operationen fortsetzen, während der Zeit in der die Daten gespiegelt werden. Das Spiegeln der Daten wird unter Ausnutzung der ersten Direct-Memory-Access-Engine ausgeführt ohne dass die zweite Direct-Memory-Access-Engine benötigt wird. Während der Spiegelung werden die Daten in einem nichtflüchtigen Speicher in dem zweiten Controllermanagementmodul gespeichert. Wenn das Spiegeln durchgeführt wird, wird die erste Direct-Memory-Access-Engine genutzt, um die Inhalte des nichtflüchtigen Speichers, der Daten zu erhalten hat, als ungültig zu kennzeichnen und die Daten zum nichtflüchtigen Speicher zu übertragen. Die erste Direct-Memory-Access-Engine kennzeichnet dann die Inhalte des nichtflüchtigen Speichers, der die Daten erhalten hat, als gültig. In einer Ausführungsform wird die erste Direct-Memory-Access-Engine ebenso dazu genutzt, zum Bestimmen der Parität der Information, die auf dem Speicherarray abgespeichert ist unter Ausnutzung des ersten Controllermanagementmoduls.

Kurze Beschreibung der Zeichnungen

[0019] **Fig.** 1 Ist eine Blockdiagrammdarstellung eines konventionellen zweifachen Netzwerkspeichercontroller;

[0020] **Fig.** 2 Ist eine Blockdiagrammdarstellung einer Netzwerkspeichervorrichtung der vorliegenden Erfindung;

[0021] **Fig.** 3 Ist eine Blockdiagrammdarstellung eines Controllermanagementmoduls der vorliegenden Erfindung;

[0022] **Fig.** 4 Ist eine Blockdiagrammdarstellung eines Kanalschnittstellenmoduls der vorliegenden Erfindung;

[0023] **Fig.** 5 Ist eine Blockdiagrammdarstellung einer redundanten Netzwerkspeichervorrichtung der vorliegenden Erfindung;

[0024] **Fig.** 6 Ist eine Blockdiagrammdarstellung einer redundanten Netzwerkspeichervorrichtung, die ein ausgefallenes Controllermanagementmodul darstellt;

[0025] **Fig.** 7 Ist eine Blockdiagrammdarstellung einer redundanten Netzwerkspeichervorrichtung, die ein ausgefallenes Kanalschnittstellenmodul darstellt;

[0026] **Fig.** 8 Ist eine Blockdiagrammdarstellung einer redundanten Netzwerkspeichervorrichtung, die vier Kanalschnittstellenmodule darstellt;

[0027] **Fig.** 9 Ist eine Blockdiagrammndarstellung einer Netzwerkspeichervorrichtung, die einen Gigabit interconnect channel anwendet;

[0028] **Fig.** 10 Ist eine Blockdiagrammdarstellung

einer Netzwerkspeichervorrichtung die einen Ultra320 SCSI channel anwendet;

[0029] **Fig.** 11 Ist eine Blockdiagrammdarstellung einer Netzwerkspeichervorrichtung die ein Ethernet channel anwendet;

[0030] **Fig.** 12 Ist eine Flussdiagrammdarstellung der Schritte, die angewendet werden beim Spiegeln der Daten in einem aktiv/aktiv Controllerpaar, wobei Prozessorinterrupts über shared Disk-Channels genutzt werden;

[0031] **Fig.** 13 Ist eine Flussdiagrammdarstellung der Schritte, die angewendet werden beim Spiegeln der Daten zwischen Controllermanagementmodulen, wobei shared Busse und Directory Memory Access angewendet werden; und

[0032] **Fig.** 14 Ist eine Blockdiagrammdarstellung einer Netzwerkspeichervorrichtung der vorliegenden Erfindung, die besonders einen DMA-Path zwischen CMM-A und CMM-B hervorhebt.

Detaillierte Beschreibung

[0033] Mit Bezug nun auf die **Fig.** 2 wird ein Blockdiagramm der Netzwerkspeichervorrichtung **100** der vorliegenden Erfindung gezeigt. Die Netzwerkspeichervorrichtung **100** enthält eine oder mehr Controllermanagementmodule (CMMs). In der Ausführungsform die in der **Fig.** 2 gezeigt wird, gibt es zwei CMMs, CMM-A **104** und CMM-B **108**, obwohl eine einzelne CMM in Anwendungen genutzt werden kann in denen keine Redundanz erforderlich ist und zusätzliche CMMs in Anwendungen genutzt werden können, die zusätzliche Redundanz oder eine höhere Performance erfordern. Jede CMM **104**, **108** hat zwei Backplane-Schnittstellen **112**. Das System hat eine passive Bus-Backplane **116**, welche für jeden CMM zwei Busse hat. In der gezeigten Ausführungsform, nutzt die passive Bus-Backplane **116** next generation Peripheral Component Interconnect (PCIX) Busse, obwohl es offensichtlich ist, dass jede Bus-Technologie genutzt werden kann, einschließlich switched architectures wie z. B. Infiniband oder RapidIO, wie auch traditionelle Bus-Architekturen wie PCI local bus. Der passive Bus-Backplane **116** kann einen ersten Datenbus **112** haben, einen zweiten Datenbus **124**, einen dritten Datenbus **128** und einen vierten Datenbus **132**. Der erste Datenbus **120** und zweite Datenbus **124** sind verbunden mit der Backplane-Schnittstelle **112** auf CMM-A **104** über CMM-Busverbindung **134**, und der dritte Datenbus **128** und vierte Datenbus **132** ist verbunden mit der Backplane-Schnittstelle **112** auf CMM-B **108** über CMM Busverbindungen **134**.

[0034] In der in **Fig.** 2 gezeigten Ausführungsform werden PCIX-Busse in der passiven Backplane **116** genutzt. Die Anwendung der PCIX-Busse erlaubt das Verbinden von relatively high performance bus interconnection Komponenten mit dem passiven Backplane **116** mit wohlverstandenen und relativ einfachen Busprotokollen. PCIX-Technologie ist eine weiter entwickelte Technologie, die den traditionellen PCI-Bus in seiner Leistung steigert.

[0035] Die Netzwerkspeichervorrichtung **100** hat eine oder mehrere Kanalschnittstellenmodule (CIMs). In der in **Fig.** 2 dargestellten Ausführungsform gibt es zwei CIMs, CIM-1 **136** und CIM-2 **140**, obwohl es offensichtlich ist, dass diese Anzahl abhängig von der Konfiguration und Anwendung, in welcher die Netzwerkspeichervorrichtung **100** genutzt wird, variieren kann. Jede CIM **136**, **140** hat zwei CIM-Busschnittstellenports **144a**, **144b**. Auf jeder CIM **136**, **140** stellt ein CIM-Busschnittstellenport **144a** eine Verbindung mit einem Bus her, welcher mit dem CMM-A **104** verbunden ist und ein CIM-Busschnittstellenport **144b** stellt eine Verbindung mit einem Bus her, welcher mit dem CMM-B **108** über CIM-Busverbindung **146** verbunden ist. In der in **Fig.** 2 gezeigten Ausführungsform, stellt CIM-1 **136** eine Verbindung mit dem ersten Datenbus **120** und dritten Datenbus **128** her und CIM-2 **108** stellt eine Verbindung mit dem zweiten Datenbus **124** und vierten Datenbus **132** her. Jeder CIM **136**, **140** hat zwei Host-Ports **148**, welche Verbindungen zum Host-Channel **152** herstellen, welcher eine Verbindung mit dem Host-Computer (nicht gezeigt) herstellt. Jeder CIM **136**, **140** hat ebenso zwei Disk-Ports **156**, welche Verbindungen mit dem Disk-Channel **158** herstellen, welcher Verbindung zu einem oder mehr Speichereinheiten (nicht gezeigt) herstellt. Die Speichereinheiten können Speicherarrays sein, wie z.B. ein RAID-Array. In alternativen Ausführungsformen, wie detaillierter unten besprochen wird, kann eine CIM mehrere Host-Ports enthalten oder verschiedene Disk-Ports, abhängig von der Anwendung und den erforderlichen Kanalschnittstellen.

[0036] Wenn der Host-Computer Daten sendet, werden diese über den Host-Channel **152** gesendet und beim Host-Port **128** auf den CIMs **136**, **140** erhalten. Diese Daten werden zu den CMMs **104**, **108** über den passiven Backplane **116** gesendet. Die CMMs **104**, **108** enthalten Speicher und Prozesseinheiten, wie unten detaillierter beschrieben wird, die die Daten in einer geeigneten Form anordnen zur Speicherung auf den Speichereinheiten. Z.B. wenn das System in einem RAID **5** Disk-Array-System genutzt wird, werden die CMMs **104**, **108** die Daten in geeignete Stripes anordnen, die auf die Disks geschrieben werden können und einen Paritätsblock für die Datenstreifen berechnen. So dass die CMMs **104**, **108** die Daten zur Speicherung verarbeiten und formatieren. Wenn dies einmal beendet ist, übertragen die CMMs **104**, **108** die Daten, die fertig zum Speichern sind, an die CIMs **136**, **140** über den passiven

Backplane **116**. Die CIMs **136**, **140** senden dann die Daten an die Speichereinheiten, die an den Disk-Port **156** angeschlossen sind. Wie unten detaillierter besprochen wird, können Daten zwischen den CMMs **104**, **108** übertragen werden unter Ausnutzung der CIMs **136**, **140** und dem passiven Backplane **116**. Zusätzlich wie ebenso unten beschrieben wird, können die CMMs **104**, **108** und CIMs **136**, **140** mit spezifischen Laufwerken oder Hosts verbunden werden.

[0037] Diese Konfiguration stellt eine modulare und redundante Architektur zur Verfügung in welcher der Host-Channel **152** und der Disk-Channel **158** nicht notwendigerweise das gleiche Kanalmedium benötigen. Die Modularität der CMMs **104**, **108** und CIMs **136**, **140** erlaubt ebenso zu relativ geringen Kosten Upgrades und einfaches Ersetzen der ausgefallenen Einheiten. Der Gebrauch eines passiven Backplane **116**, um Daten zwischen CMMs **104**, **108** auszutauschen, vermeidet ebenso den Gebrauch der Kanal-Bandbreite des Disk-Channel **158** oder des Host-Channel **152** wie es für das Datenspiegeln in einer traditionellen redundanten Controllerumgebung notwendig wäre, wie es unten beschrieben wird.

[0038] Mit Bezug nun auf **Fig.** 3 ist eine Blockdiagrammdarstellung eines CMM **104** dargestellt. Der CMM **104** enthält verschiedene Komponenten, einschließlich eines CPU-Subsystems **160**, eines Speichers **164**, und einer Schnittstelle FPGA **168**. Das CPU-Subsystem **160** kann ein Standardtyp-CPU sein, wie z. B. ein weithin genutzter Mikroprozessor, oder ein anwendungsspezifischer Prozessor. In einer Ausführungsform ist das CPU-Subsystem **160** ein Intel Pentium (TM)-Klasse Mikroprozessor. Das CPU-Subsystem **160** kommuniziert mit dem Schnittstellen FPGA **168**, wobei ein Standardbus wie z.B. ein PCI-Bus genutzt wird. Der Speicher **164** ermöglicht das temporäre Speichern der Daten innerhalb des CMM **104**. Diese Speicherung wird während normalen Lese- und Schreiboperationen für verschiedene Zwecke genutzt, wie z. B. Speichern von sich in einer Warteschlange befindlichen Daten, die darauf warten in eine Disk-Array geschrieben zu werden. In einer Ausführungsform, wird ein DDR-Speicher DIMM genutzt, welcher mit der Schnittstelle FPGA **168** kommuniziert, wobei eine Bus-Schnittstelle genutzt wird.

[0039] Die Schnittstelle FPGA **168** enthält eine Anzahl von Komponenten. Es ist offensichtlich, dass diese Komponenten zu einem einzelnen FPGA kombiniert werden können oder auf verschiedenen Komponenten innerhalb der CMM **104** vorhanden sein können. In einer Ausführungsform, die in der **Fig.** 3 gezeigt wird, enthält die Schnittstelle FPGA **168** eine PCI-Schnittstelle **172**, eine Speicherschnittstelle **176**, eine XOR-Engine **180**, eine Bridge-Core **184**, eine DMA-Engine **188**, Daten-FIFOs **192** und zwei Backplane-Schnittstellen **112**. Die PCI-Schnittstelle **172** arbeitet als eine Schnittstelle zwischen dem CPU-Subsystem **160** und den anderen Abschnitten der Schnittstelle FPGA **168**. In der gezeigten Ausführungsform, nutzt diese Schnittstelle eine Standard PCI-Busverbindung. Die PCI-Schnittstelle **172** stellt eine Verbindung mit dem Bridge-Core **184** her, welches im Gegenzug eine Verbindung mit der Backplaneschnittstelle **112** herstellt, welche eine Schnittstelle schafft mit dem ersten Datenbus **120** und dem zweiten Datenbus **124**, die auf dem passiven Backplane **116** angeordnet sind.

[0040] Die Speicherschnittstelle **176** arbeitet als eine Schnittstelle zwischen dem Speicher **164** und der Schnittstelle FPGA **168**. Die XOR-Engine **180** dient dazu XOR-Operationen auf die Daten, die zu speichern sind, auszuführen, um Paritätsinformationen aus den Daten, welche zu schreiben sind, zu erhalten. Die XOR-Engine **180** wird ebenso in Situationen genutzt, wo der Gebrauch der Paritätsinformation erforderlich ist, um Daten von einem ausgefallenen Laufwerk in einer Disk-Array wieder zu gewinnen. Die XOR-Engine **180** stellt eine Verbindung mit dem CPU-Subsystem **160** durch die PCI-Schnittstelle **172** her. Die Daten-FIFOs **192** stellen eine Verbindung mit der Speicherschnittstelle **176** und der Bridge-Core **184** her, die im Gegenzug eine Verbindung mit der PCIX-Schnittstelle **196** herstellen.

[0041] Die Daten-FIFOs dienen als eine Warteschlange, welche von dem CMM **104** genutzt wird, um Lese- und Schreiboperationen zu verwalten. Die DMA-Engine **188** dient dazu DMA-Daten einem anderen CMM bereitzustellen, wenn die CMMs arbeiten, um Redundanz bereitzustellen, wie im Detail unten erläutert wird. Die DMA-Engine **188** wird in einer Ausführungsform ebenso in Verbindung mit der XOR-Engine **180** genutzt, um XOR-Operationen durchführen zu können sowie zum Lesen der Daten aus zwei Bereichen innerhalb des Speichers **164** und zum Bereitstellen der Daten der XOR-Engine **180**, sowie zum Schreiben der Outputs der XOR-Engine in einen dritten Bereich innerhalb des Speichers **164**.

[0042] Bezugnehmend nun auf die **Fig.** 4 die eine Blockdiagrammdarstellung eines CIM **136** zeigt. Die CIM **136** enthält eine PCIX-Bridge **200** und zwei Kanalschnittstellen **204**. Die PCIX-Bridge **200** steht zur Verfügung, um ein erstes switched Path **208** und ein zweites switched Path **212** zu verbinden. Jeder switched Path **208**, **212** stellt eine Verbindung mit einem Busschnittstellen-Port **144** her, welcher im Gegenzug eine Verbindung mit einem PCIX-Bus auf dem passiven Backplane **116** über eine CIM-Busverbindung **146** hergestellt. Die PCIX-Bridge **200** steht zur Verfügung, um Daten, welche über die switched PCIX-Paths **208**, **212** gesendet werden, darzustellen und zu bestimmen ob die Daten zwischen den switched PCIX-Paths **208**, **212** geroutet werden müssen. Dieses Darstellen und Routen der Daten zwischen

den switched PCIX-Paths **208**, **212** wird vorrangig dazu genutzt die Spiegelungsoperationen zwischen CMMs zu ermöglichen und wird unten detaillierter beschrieben.

[0043] Die Kanalschnittstellen **204** stellen eine Verbindung her zwischen den switched PCIX-Paths **208**, **212** zu dem Host-Port **148** und dem Disk-Port **156**. Die Kanalschnittstellen **204** sind betriebsbereit, um Daten, welche über die switched PCIX-Paths **208**, **212** gesendet werden, darzustellen und zu bestimmen, ob die Daten zu dem Host-Port **148** oder dem Disk-Port **156** geroutet werden müssen. Durch dieses Darstellen und Routen werden Daten zu dem geeigneten Disk- oder Host-Ort geleitet und dadurch werden gespiegelte Daten nicht durch den Host- oder die Disk-Ports **148**, **156** geschickt. Die Kanalschnittstellen **204** erlauben Kommunikation über das geeignete Kanalmedium für die Anwendung. Z. B. wenn die Host-Kanäle **152** und die Disk-Kanäle **156** Fibre-Channel nutzen, wurden die Kanalschnittstellen **204** als Schnittstelle zwischen den switched PCIX-Paths **208**, **212** und dem Fibre-Channel arbeiten. Desgleichen wenn die Host-Kanäle **152** und die Disk-Kanäle **158** einen SCSI-Kanal benutzen, würden die Kanalschnittstellen **204** als Schnittstelle zwischen den switched PCIX-Paths **208**, **212** und dem SCSI-Kanal arbeiten. Wenn beide, die Host-Kanäle **152** und die Disk-Kanäle **158**, das gleiche Kanalmedium nutzen, kann der CIM **136** zur Kommunikation mit beiden, den Host-Kanälen **152** und den Disk-Kanälen **158**, durch den Gebrauch der Host-Ports **148** und Disk-Ports **146** genutzt werden.

[0044] In einer Ausführungsform gebrauchen die Disk-Kanäle **158** und die Host-Kanäle **152** nicht das gleiche Kanalmedium. In dieser Ausführungsform wird eine verschiedene CIM für jedes verschiedene Kanalmedium genutzt. Z.B. falls der Host-Computer ein Fibre-Channel nutzt und das Disk-Array einen SCSI-Kanal benutzt, würde der Host-Computer eine Verbindung zu einer CIM unter Ausnutzung einer Fibre-Channel-Schnittstelle herstellen und das Disk-Array würde eine Verbindung zu einer anderen CIM unter Ausnutzung einer SCSI-Kanalschnittsstelle herstellen. Falls Redundanz erforderlich ist, könnten zwei oder mehr CIMs an jedes Kanalmedium angeschlossen werden.

[0045] In der Ausführungsform, die in der **Fig.** 4 gezeigt wird, kommuniziert der erste switched PCIX-Path **208** mit dem ersten Datenbus **120** und der zweite switched PCIX-Path **212** kommuniziert mit dem dritten Datenbus **128** durch den Busschnittstellen-Port **144** und die CIM-Busverbindung **146**. Die PCIX-Bridge **200** kann als ein Kommunikations-Path genutzt werden für eine CMM, um mit einer anderen CMM zu kommunizieren, wie es unten detaillierter besprochen wird. Es ist offensichtlich, dass eine ähnliche Konfiguration für die übrigen CIMs genutzt wird, die auf dem Netzwerk-Controller vorhanden sind. Z.B. ist in der Ausführungsform, die in **Fig.** 2 gezeigt wird, CIM-2 **140** verbunden mit dem dritten Datenbus **128** und dem vierten Datenbus **132** und darum würde CIM-2 **140** switched PCIX-Paths **208**, **212** haben, die mit jeweils dem zweiten Datenbus **124** und dem vierten Datenbus **132** kommunizieren. Desgleichen, falls mehr als zwei CIMs vorhanden sind, werden sie konfiguriert, um mit den geeigneten Bussen auf der passiven Backplane **116** zu kommunizieren, wie es von der Anwendung gefordert wird.

[0046] Bezugnehmend wieder auf die **Fig.** 2 bis 4, sind in einer Ausführungsform CMM-A **104** und CMM-B **108** jeweils mit einem Abschnitt jedes CIM **136**, **140** zugeordnet. In solch einem Fall hat eine CMM **104** oder **108** exklusives Besitzrecht an einem PCIX-Path **208** oder **212** durch das Ermöglichen eines Zugangs zu dem PCIX-Path **208** oder **212** von dem Bus-Segment, mit dem der PCIX-Path **208** oder **212** verbunden ist. Z.B. wendet in einer Ausführungsform CMM-A **104** PCIX-Path **208** in jedem CIM **136**, **140** an, um mit dem Host-Kanal **152** und Disk-Kanal **158** zu kommunizieren. Diese Verbindung wird erreicht durch Verbinden des Busschnittstellenports **144**, der mit dem ersten switched PCIX-Path **208** in CIM-1 **136** mit dem ersten Datenbus **120** auf dem Backplane **116** verbunden ist. Desgleichen ist der Busschnittstellenport **144**, der verbunden ist mit dem ersten switched PCIX-Path **208** in CIM-2 **140**, verbunden mit dem zweiten Datenbus **124** auf dem passiven Backplane **116**. CMM-A **104** kommuniziert darum mit dem Host und den Disk-Kanälen **152**, **158** durch den ersten switched PCIX-Path auf jedem CIM **136**, **140**. Ähnlich sind die Busschnittstellenports **144**, die verbunden sind mit den zweiten switched PCIX-Paths **212** in jedem CIM **136**, **140**, jeweils mit den dritten und vierten Datenbussen **128**, **132** verbunden. CMM-B **108** kommuniziert darum mit den Host- und Disk-Kanälen **152**, **158** durch den zweiten switched PCIX-Path **212** auf jeder CIM **136**, **140**.

[0047] Bezugnehmend nun auf die **Fig.** 5 ist eine Blockdiagrammdarstellung einer Netzwerkspeichervorrichtung **100a**, die redundante Komponenten enthält, gezeigt. In dieser Ausführungsform werden zwei CMMs genutzt, CMM-A **104** und CMM-B **108**. Zwei CIMs werden genutzt, CIM-1 **136** und CIM-2 **140**. CMM-A **104** und CIM-1 **136** sind beide mit dem ersten Datenbus **120** in der passiven Backplane **116** verbunden. CMM-A **104** und CIM-2 **104** sind beide mit dem zweiten Datenbus **124** in der passiven Backplane **116** verbunden. CMM-B **108** und CIM-1 **136** sind beide mit dem dritten Datenbus **128** auf der passiven Backplane **116** verbunden. CMM-B **108** und CIM-2 **140** sind beide mit dem vierten Datenbus **132** auf dem passiven Backplane **116** verbunden.

[0048] Wie dem Fachmann bekannt ist, erfordern redundante Controller das Spiegeln der Daten zwi-

schen den zwei Controllern, die dem Speicher-Subsystem angeschlossen sind. Dies geschieht auf Grund der Verwendung eines Zurückschreibe-Cache, wo der Controller von dem Host-Computer Daten erhält, die Daten cached und eine Botschaft an den Host-Computer sendet, dass die Daten geschrieben wurden. So dass der Host-Computer bestimmt, dass die Daten geschrieben werden, wenn sie tatsächlich im Controller gespeichert sind und dort darauf warten in die Laufwerke des Disk-Arrays geschrieben zu werden. Um zu unterstützen, dass sichergestellt ist, dass diese Daten nicht verloren gehen im Falle eines Ausfalls spiegeln redundante Controller diese Daten an den anderen Controller, so dass eine andere Kopie der Daten auf dem anderen Controller vorhanden ist. Dies ist bekannt als Cache-Kohärenz. In einer Ausführungsform spiegeln die CMMs **104**, **108** Daten, um Cache-Kohärenz in der Netzwerkspeichervorrichtung **100a** zu ermöglichen. Dies kann durchgeführt werden durch die Implementierung eines DMA-Path zwischen CMM-A **104** und CMM-B **108**. Dies kann erledigt werden durch das Bereitstellen einer DMA-Engine **188** in der Schnittstelle FPGA **186**, wie oben mit Bezug auf die **Fig.** 3 erläutert wurde, und eine shared Path **216**, die eine PCIX-Bridge **200** in jeder CIM **136**, **140** anwendet, wie oben mit Bezug auf **Fig.** 4 besprochen wurde. Jede CMM **104**, **108** nutzt diese DMA-Path, um Daten zum anderen CMM zu senden. Durch Anwendung des DMA-Paths, können die zwei CMMs **104**, **108** Daten spiegeln ohne die Hilfe des Host-Kanal **152** zu benötigen oder des Disk-Kanals **158**, so dass Kanalbandbreite in dem Disk-Kanal **158** oder Host-Kanal **152** von den CMMs **104**, **108** nicht benötigt wird beim Spiegeln der Daten.

[0049] Zusätzlich zu dem reduzierten Bedarf an Kanalbandbreite wie es in traditionellen Spiegelungskonfigurationen genutzt wird, kann die Netzwerkspeichervorrichtung **100a** ebenso konfiguriert werden, um weniger Prozessressourcen als traditionelle Techniken zu nutzen. **Fig.** 13 zeigt eine Flussdiagrammdarstellung eines DMA-Spiegelungsverfahrens, das in einer Ausführungsform der vorliegenden Erfindung genutzt wird. Wie dem Fachmann bekannt ist, ist DMA die Fähigkeit, die von einigen Computerbus-Architekturen zur Verfügung gestellt wird, die ermöglicht, dass Daten von einer angeschlossenen Einheit in den Computerspeicher gesendet werden, ohne die Computerprozessressourcen zu benutzen. In einer Ausführungsform haben CMM-A und CMM-B einen spezifischen Abschnitt des DDR nichtflüchtigen Random-Access-Memory (NVRAM) als einen Bereich entwickelt, der genutzt wird zum Direct-Memory-Access durch den anderen CMM. Wenn CMM-A **104** Daten hat, welche zum CMM-B **108** gespiegelt werden müssen, startet CMM-A **104** zuerst eine erste DMA-Transaktion, um die Inhalte eines Abschnitts des CMM-B DDR-Speichers **164** als ungültig für die Datenblöcke zu kennzeichnen, die betroffen sind mit der Schreiboperation, wie gezeigt durch Block **500**. Ebenso in der ersten DMA-Transaktion eingeschlossen ist, dass die Anwenderdaten gespiegelt werden, welche in den spezifizierten Datenblöcken gespeichert sind, wie dargestellt durch Block **500**.

[0050] Mit Bezug auf **Fig.** 14 wird die Hardware, die mit einer DMA-Transaktion verbunden ist, detaillierter beschrieben. Wenn die DMA-Transaktionen durchgeführt werden, steuert das CPU-Subsystem **160** des CMM-A **104** die Operation des DMA-Engine **188** auf CMM-A **104**. Die DMA-Engine **188** auf CMM-A **104** greift auf den DDR-Speicher **164** des CMM-B **108** durch die PCIX-Bridge **200** des CIM-1 **136** und die Schnittstelle FPGA **168** des CMM-B **108** zu. So dass, wenn der Befehl gegeben wurde, die Inhalte des CMM-B **108** DDR-Speichers **164** als ungültig zu kennzeichnen, die DMA-Engine **188** auf der CMM-A **104** auf den DMA-Path durch die Daten-FIFOs **192** und Backplane-Schnittstelle **112** der CMM-A Schnittstelle FPGA **168** zugreift und den Befehl durch den DMA-Path zu der CMM-B **108** Schnittstelle FPGA **168** überträgt. Der Befehl wird durch die CMM-B **108** Schnittstelle FPGA **168** durch die Backplane-Schnittstelle **112** und Daten-FIFOs **192** zur Speicherschnittstelle **176** und dann zu dem DDR-Speicher **164** geroutet. Dieser DMA-Path ist gekennzeichnet durch die gestrichelten Linien in der **Fig.** 14. So dass das CMM-B CPU-Subsystem **160** und die CMM-B **108** DMA-Engine **188** nicht in die DMA-Transaktion einbezogen sind, die von der CMM-A **104** initiiert wurde. Wenn die Benutzerdaten übertragen werden, führt das CMM-A **104** CPU-Subsystem **160** den ersten DMA-Befehl fort. Der DMA-Befehl zeigt die Quelladresse innerhalb des CMM-A **104** DDR-Speichers **164**, die Zieladresse innerhalb des CMM-B **108** NV-RAM **164** und die Länge des Datenblocks, der zu übertragen ist, an. Die CMM-A **104** DMA-Engine **188** erhält den DMA-Befehl und greift auf den CMM-A **104** DDR-Speicher **164** durch die Daten-FIFOs **192** und Speicherschnittstelle **176** zu. Die CMM-A **104** DMA-Engine **188** überträgt dann die geeigneten Daten von dem CMM-A **104** NVRAM **164** durch die CMM-A **104** Schnittstelle FPGA **168** über den DMA-Path, durch die CMM-B **108** Schnittstelle FPGA **168** und in den geeigneten Ort in der CMM-B **108** DDR-Speicher **164**. Bezugnehmend auf den Block **504**, CMM-A **104** initiiert dann eine zweite DMA-Transaktion, um Abschnitte der Inhalte des CMM-B **108** DDR-Speichers **164** als gültig für die spezifizierten Datenblöcke zu kennzeichnen. Da die Transaktionen DMA-Übertragungen sind, erfordern sie nicht Prozessressourcen vom CMM-B **108** oder unterbrechen den CMM-B **108**, was die Performance des Systems erhöht, weil CMM-B in der Lage ist, andere Prozessfunktionen unabhängig von der Spiegelungsoperation durchzuführen.

[0051] In einer anderen Ausführungsform sind die

beiden DMA-Transaktionen, die in der **Fig.** 13 gezeigt werden, kombiniert in einer einzelnen geordneten DMA-Transaktion. In dieser Ausführungsform hat der DDR-Speicher **164** in jeder CMM **108**, **108** zwei Speicherregionen zum Speichern von Metadaten, welche mit den Benutzerdaten verbunden sind. Wenn eine DMA-Übertragung initiiert wurde, wird ein erster einzelner String in der ersten Speicherregion abgespeichert, gefolgt von den Daten, die zu übertragen sind. Am Ende der DMA-Übertragung ist der erste einzelne String in der zweiten Speicherregion abgespeichert. Falls ein CMM **104**, **108** wegen eines Ausfalls wiedergewonnen werden muss, werden die Strings, die in der ersten und zweiten Speicherregion abgespeichert wurden, verglichen. Der Vergleich wird nur bei Eintreten eines CMM-Ausfalls durchgeführt, so dass es kein Verlust der Performance während der regulären Tätigkeit gibt. Wenn die Strings zueinander passen, zeigt das, dass die gespiegelten Daten gültig sind.

[0052] Bezugnehmend wieder auf die **Fig.** 5 ist ebenso eine Ausfallsicherungs-Resetverbindung **240** zwischen CMM-A **104** und CMM-B **108** vorhanden. Die Ausfallsicherungs-Resetverbindung **240** wird zum Mitteilen eines Ausfalls eines der CMMs **104**, **108** genutzt. In einer Ausführungsform ist die Ausfallsicherungs-Resetverbindung **240** eine serielle Verbindung zwischen CMM-A **104** und CMM-B **108**. In dieser Ausführungsform erhält jeder CMM **104**, **108** ein Herzschlagsignal aufrecht, welches über die Ausfallsicherungs-Resetverbindung **240** kommuniziert wird und durch den anderen CMM beobachtet wird. Wenn ein Problem bei dem Herzschlagsignal erkannt wird, kann ein CMM **104**, **108** ein Signal über die Ausfallsicherungs-Resetverbindung **240** senden, um die Tätigkeit des anderen CMM zu beenden. Zum Beispiel, falls CMM-B **108** einen Ausfall hat oder nicht sauber arbeitet, erkennt der CMM-A **104**, dass das Herzschlagsignal von CMM-B **108** nicht länger aktiv ist. Nach einer voreingestellten Zeitperiode, in welcher kein Herzschlagssignal erhalten wird, sendet der CMM-A **104** ein Beendigungssignal an den CMM-B **108**. Wenn der CMM-B **108** das Beendigungssignal erhält, bricht er seine Tätigkeit ab. Der CMM-A übernimmt dann die Steuerung aller Lese- und Schreiboperationen. Dies ist notwendig, da der CMM-B in einer ungewöhnlichen Weise arbeiten könnte. Desgleichen falls der CMM-A **104** ausfällt, würde der CMM-B **108** den Hinweis erhalten über die Ausfallsicherungs-Resetverbindung **240** und die Kontrolle übernehmen über alle Lese- und Schreiboperationen. So dass das System redundant ist und fortführt zu arbeiten, wenn ein CMM **104** oder **108** ausfällt.

[0053] Bezugnehmend nun auf die **Fig.** 6 wird die Tätigkeit des Systems, wenn ein CMM ausfällt, beschrieben. Wie in **Fig.** 6 dargestellt, haben die Netzwerkspeichervorrichtungen **100a** den CMM-A **104** und CMM-B **108**, einen passiven PCIX-Backplane **116** und einen CIM-1 **136** und einen CIM-2 **140**. Wenn CMM-B **108** ausfällt, erkennt CMM-A **104** den Ausfall über die Ausfallsicherungs-Resetverbindung **240**, wie oben beschrieben, und beendet die Operationen auf dem CMM-B **108**. CMM-A **104** übernimmt dann die Kontrolle über alle Speicher- und Steuerungsoperationen, die zuvor von dem CMM-B **108** durchgeführt wurden. Wenn dies passiert, sendet CMM-A **104** einen Befehl an CIM-1 **136** und CIM-2 **140**, um Kommunikationen mit CMM-B **108** zu verhindern. In diesem Fall würde CIM-1 **136** diesen Befehl erhalten und den ersten switched Path **208**, der mit dem vierten Datenbus **132** verbunden ist, inaktiv setzen und die PCIX-Bridge **200** und die Kanalschnittstelle **204**, die mit dem CMM-B **108** verbunden ist, zurücksetzen. CIM-2 **140** erhält ebenso den Befehl von CMM-A **104** und führt dieselbe Funktion zum Abschalten des zweiten switched Path **212**, der mit dem dritten Datenbus **128** verbunden ist, aus und zum Zurücksetzen der PCIX-Bridge **200** und der Kanal-Schnittstelle **204**, die mit dem CMM-B **108** verbunden ist. In einer Ausführungsform enthält die passive Backplane **116** Steuerlogikverbindungen, welche eine Verbindung herstellen mit den Busschnittstellenports **144** auf den CIMs **136**, **140** und welche mit den CMMs **104**, **108** verbunden sind. Die CMMs **104**, **108** können diese Steuerlogikverbindungen nutzen, um die Busschnittstellenports **144** auf den CIMs **136**, **140** zu aktivieren und zu deaktivieren. Alternativ können andere Ausführungsformen genutzt werden zum Aktivieren und Deaktivieren der switched Paths **208**, **212** sowie Steuerlogik innerhalb der CIM, welche Befehlsinformationen z.B. über die PCIX-Busse auf der passiven Backplane **116** erhält.

[0054] Bezugnehmend nun auf die **Fig.** 7 wird nun die Operation des Systems beschrieben, falls ein CIM ausfällt. Die CMMs **104**, **108** beobachten die CIMs **136**, **140** und im Falle eines Fehlers oder eines Ausfalls, kommunizieren sie einen Befehl über die Steuerlogikverbindungen, um die Operation des CIM **136** oder **140**, welcher ausgefallen ist, zu beenden. Wie in **Fig.** 7 bildlich dargestellt, hat der CIM-1 **136** einen Ausfall. CMM-A **104** bestimmt dass CIM-1 **136** einen Ausfall gehabt hat und deaktiviert CIM-1 **136**. CMM-A **104** teilt diese Information dann dem CMM-B **108** über die PCIX-Bridge **200** auf CIM-2 **140** mit.

[0055] Bezugnehmend nun auf die **Fig.** 8 wird ein Blockdiagramm einer Netzwerkspeichervorrichtung **100b** dargestellt, in welcher vierfache CIM-Module vorhanden sind. In dieser Ausführungsform kommunizieren zwei CMMs, CMM-A **104** und CMM-B **108** mit vier CIMs, CIM-1 **136**, CIM-2 **140**, CIM-3 **300** und CIM-4 **304**. In dieser Ausführungsform ist CMM-A **104** mit dem ersten und zweiten Datenbus **120**, **124** in der passiven Backplane **116** verbunden. Desgleichen ist CMM-B mit dem dritten und vierten Datenbus **128**, **132** in der passiven Backplane **116** verbunden.

CIM-1 ist mit dem zweiten und vierten Datenbus **124**, **132** verbunden, so dass jedem CMM **104**, **108** ein Zugriff auf den CIM-1 **136** ermöglicht wird. CIM-2 **140** ist mit dem ersten und dritten Datenbus **120**, **128** verbunden, CIM-3 **300** ist mit dem zweiten und vierten Datenbus **124**, **132** verbunden, und CIM-4 ist mit dem ersten und dritten Datenbus **120**, **128** verbunden. So dass jeder CMM **104**, **108** mit jedem CIM kommunizieren kann. Eine Netzwerkspeichervorrichtung dieser Ausführungsform ist in verschiedenen Fällen nützlich, einschließlich z. B. wenn mehrere Hosts vorhanden sind. In dieser Ausführungsform können CIM-1 **136** und CIM-2 **140** Kommunikationen mit einem ersten Host zur Verfügung stellen, und CIM-3 **300** und CIM-4 **304** können Kommunikationen mit einem zweiten Host zur Verfügung stellen. Die gleiche Anordnung kann für mehrere Disk-Nodes aufgebaut werden, wie z. B. zwei separate RAID-Arrays. Wie es offensichtlich ist, stellt diese Konfiguration ein skalierbares System dar, welche Kommunikationen zwischen einem oder mehr Host-Nodes und einem oder mehreren Disk-Nodes bereit stellen können, während ebenso redundante Operationen bereit gestellt werden. Zusätzlich kann solch eine Ausführungsform nützlich sein, um Hosts und/oder Disk-Arrays zu verbinden, welche ein unterschiedliches Kanalmedium nutzen. Z. B. kann ein bestehendes System zwei CIMs haben und Fibre-Channelverbindungen für beide der Host- und Diskkanäle benutzen. Wenn ein Anwender das System upgraden möchte, um eine andere Disk-Array hinzuzufügen, welche eine SCSI-Verbindung nutzt, könnten zusätzliche CIMs hinzugefügt werden, welche die Kommunikation mit einem SCSI-Kanal ermöglichen, wobei das Upgrade des bestehenden Systems ermöglicht wird, ohne existierende Hardware ersetzen zu müssen.

[0056] Bezugnehmend nun auf die **Fig.** 9 bis 11 werden verschiedene alternative Ausführungsformen eines CIM gezeigt, um ein Beispiel bereitzustellen für die verschiedenen Konfigurationen, die ein CIM haben kann, und der verschiedenen Kanalmedien, mit welchem ein CIM verbunden werden kann. **Fig.** 9 zeigt eine Blockdiagramm-Darstellung eines PCIX mit einer Gigabit Interconnect (GBIC) konfigurierten CIM **136a**. Innerhalb der CIM **136a** stellt die PCIX-Bridge **200a** eine Verbindung mit einer zweifachen Port-GBIC-Schnittstelle **400** her. Jeder Port der zweifachen Port-GBIC-Schnittstelle **400** stellt mit einem Serializer/Deserializer (SERDES) **404a**, **404b** eine Verbindung her. Jeder SERDES **404a**, **404b** stellt eine Verbindung mit dem Kanalmedium her, wobei eine Kanalverbindung benutzt wird. In der Ausführungsform, die in **Fig.** 9 gezeigt wird, stellt eine SERDES **404a** mit einem GBIC-Host-Kanal **152a** eine Verbindung her und der andere SERDES **404b** stellt eine Verbindung mit einem GBIC-Disk-Kanals **158a** her.

[0057] **Fig.** 10 zeigt eine Blockdiagrammdarstellung des PCIX zum SCSI CIM **136b**. Innerhalb des CIM **136b**, stellt die PCIX-Bridge **200b** mit einer zweifachen Port Ultra320 SCSI-Schnittstelle **408** eine Verbindung her. Jeder Port der zweifachen Port U1tra320-SCSI-Schnittstelle **408** stellt eine Verbindung mit dem Host- oder Disk-Kanal her und hat ebenso eine Beendigungs-412-Schnittstelle, wie es für SCSI-Systeme erforderlich ist. In der Ausführungsform, die in der **Fig.** 10 gezeigt wird, stellt ein Port der zweifachen Port U1tra320-SCSI-Schnittstelle **408** mit einem very high density interconnect (VHDIC) Host-channel **152b** eine Verbindung her und ein Port der zweifachen Port Ultra320-SCSI-Schnittstelle **408** stellt eine Verbindung mit einem VHDIC-Disk-Channel **158b** her.

[0058] **Fig.** 11 zeigt eine Blockdiagrammdarstellung eines PCIX zum Ethernet CIM **136c**, welcher quick Switch-Connections **416a**, **416b** zum Gebrauch in den switched Paths einsetzt. Die quick Switch-Connections **416a**, **416b** sind Busrelais, welche Enable-Eingänge besitzen, welche die quick Switch-Connections **416a**, **416b** aktivieren und deaktivieren. Jede quick Switch-Connection **416a**, **416b** stellt eine Verbindung mit einer Schnittstellenverbindung **420** her, welche ein Acceleration-FPGA und Daten-FIFOs enthält. Die Schnittstellenverbindung **420** stellt eine Verbindung mit einem Gigabit-Ethernet-ASIC **424** her, welches saubere Funktionen mit den Daten ermöglicht, um die Daten über eine Ethernetverbindung zu kommunizieren. Der Gigabit-Ethernet-ASIC **424** stellt eine Verbindung mit einem MAC/Physical-Converter **428** her, welcher das Signal in physische Signale umwandelt, welche dann zu dem Transformer **432** geroutet werden, um das Signal mit der richtigen Spannung auszugeben. In einer Ausführungsform stellt der Transformer **432** mit der GBIC-Verbindung zu dem Disk-Channel **158c** eine Verbindung her. In dieser Ausführungsform der **Fig.** 11 könnten, falls ein redundantes System erforderlich wäre, shared Paths auf anderen CIMs zur Verfügung gestellt werden. Es wird bevorzugt, dass verschiedene Kanalmedien in einem einzelnen System genutzt werden können unter Anwendung einer Kombination von verschiedenen Schnittstellenmodulen, wie solche die in den **Fig.** 9 bis 11 gezeigt werden. Z. B. kann ein Host-Computer mit dem Netzwerkspeichercontroller verbunden werden unter Anwendung eines Fibre-Kanalmediums und der Netzwerksspeichercontroller kann mit einer Disk-Array verbunden werden unter Anwendung eines SCSI-Kanalmediums.

[0059] Die obige Diskussion der Erfindung wurde dargestellt, zum Zwecke der Illustration und Beschreibung. Ferner ist mit der Beschreibung nicht beabsichtigt, die Erfindung auf die hier offenbarte Form zu begrenzen. Konsequenterweise befinden sich Variationen und Modifikationen entsprechend der obigen Lehre, innerhalb der Fähigkeiten und Wissen der

relevanten Technik, im Rahmen der vorliegenden Erfindung. Die Ausführungsformen, die hier beschrieben wurden, sind ferner zum Erklären der besten derzeit bekannten Ausführungsformen bei der Umsetzung der Erfindungen und um anderen Fachleuten zu ermöglichen, diese Erfindungen in solchen oder in anderen Ausführungsformen und mit den verschiedenen Modifikationen, die von ihrer speziellen Anwendung oder Gebrauch der Erfindung erforderlich sind, anzuwenden. Es ist beabsichtigt, dass die beigefügten Ansprüche derart ausgestaltet sind, um alternative Ausführungsformen in dem Maße in dem dies durch den Stand der Technik ermöglicht ist, mit einschließen.

Zusammenfassung

[0060] Ein Netzwerkspeichercontroller zum Übertragen von Daten zwischen einem Host-Computer und einer Speichereinheit, wie ein redundantes Array von kostengünstigen Disks (RAID), wird offenbart. Der Netzwerkspeichercontroller umfasst zumindest ein Kanalschnittstellenmodul, welches derart eingerichtet ist, dass es mit dem Host-Computer und der Speichereinheit verbunden werden kann. Das Kanalschnittstellenmodul ist mit einer passiven Backplane verbunden und überträgt selektiv Daten zwischen dem Host-Computer, der Speichereinheit und dem passiven Backplane. Der Netzwerkspeichercontroller umfasst ebenso zumindest ein Controllermanagementmodul, das an dem passiven Backplane angeschlossen ist. Das Controllermanagementmodul kommuniziert mit dem Kanalschnittstellenmodul über die passive Backplane und bearbeitet und speichert vorübergehend Daten, die von dem Host-Computer oder der Speichereinheit erhalten wurden. In Anwendungen in denen Redundanz erforderlich ist, können zumindest zwei Controllermanagementmodule und zumindest zwei Kanalschnittstellenmodule genutzt werden. Die Controllermanagementmodule können Daten zwischen sich spiegeln unter Verwendung der passiven Backplane und eines shared communication Path auf den Kanalschnittstellenmodulen, unter substantieller Vermeidung des Gebrauchs des Host- oder Diskkanals, um Daten zu spiegeln. Die Kanalschnittstellenmodule sind betriebsbereit, um den Host-Computer oder die Speichereinheit mit einem oder mehreren Controllerspeichermodule zu verbinden. Die Controllerspeichermodule können eine DMA-Engine enthalten, um die Übertragung der gespiegelten Daten zu erleichtern.

Patentansprüche

1. Verfahren zum Spiegeln von Daten in einem Speichersystem, das ein Speicher-Array enthält, das umfasst:
dass ein erstes Controllermanagementmodul zur Verfügung gestellt wird, das eine erste Direct-Memory-Access-Engine umfasst, wobei dieses erste Controllermanagementmodul zum Steuern von Lese-/Schreiboperationen den Speicher-Array einbezieht;
dass ein zweites Controllermanagementmodul zur Verfügung gestellt wird, das eine zweite Direct-Memory-Access-Engine umfasst, wobei das zweite Controllermanagementmodul zum Steuern von Lese-/Schreiboperationen den Speicher-Array einbezieht, wobei die erste Direct-Memory-Access-Engine benutzt wird zum Spiegeln von Daten zum zweiten Controllermanagementmodul und die zweite Direct-Memory-Access-Engine genutzt wird zum Spiegeln von Daten zum ersten Controllermanagementmodul;
und das Spiegeln der ersten Daten von dem ersten Controllermanagementmodul zum zweiten Controllermanagementmodul unter Anwendung der ersten Direct-Memory-Access-Engine erfolgt.

2. Verfahren nach Anspruch 1, wobei das erste Controllermanagementmodul einen ersten Prozessor umfasst und der Spiegelungsschritt das Bestimmen umfasst, dass dieser Spiegelungsschritt durchgeführt werden soll, wobei der erste Prozessor genutzt wird.

3. Verfahren, nach Anspruch 1, wobei das zweite Controllermanagementmodul einen zweiten Prozessor umfasst und der Spiegelungsschritt Spiegelung der ersten Daten unabhängig vom zweiten Prozessor beinhaltet.

4. Verfahren nach Anspruch 1, wobei das zweite Controllermanagementmodul einen zweiten Prozessor umfasst, in welchem der zweite Prozessor genutzt wird zum Steuern der Lese-/Schreiboperationen, wobei der Speicher-Array einbezogen ist und der Spiegelungsschritt durchgeführt wird, während eine Unterbrechung des zweiten Prozessors vermieden wird.

5. Verfahren nach Anspruch 1, wobei das zweite Controllermanagementmodul einen zweiten Prozessor umfasst und in welchem der zweite Prozessor betriebsbereit ist, um zum Steuern von wenigstens einer ersten Leseoperation und einer ersten Schreiboperation, während des Spiegelungsschritts, genutzt zu werden.

6. Verfahren nach Anspruch 1, wobei das zweite Controllermanagementmodul einen zweiten Prozessor umfasst und der Spiegelungsschritt durchgeführt wird, wobei die erste Direct-Memory-Access-Engine genutzt wird, während die zweite Direct-Memory-Access-Engine und der zweite Prozessor nicht genutzt wird.

7. Verfahren nach Anspruch 1, wobei das zweite Controllermanagementmodul einen zweiten Prozessor umfasst und der Spiegelungsschritt die Speicherung der ersten Daten in einem nichtflüchtigen Spei-

cher des zweiten Controllermanagementmoduls umfasst, ohne den zweiten Prozessor zu nutzen.

8. Verfahren nach Anspruch 1, wobei das zweite Controllermanagementmodul einen nichtflüchtigen Speicher umfasst, und der Spiegelungsschritt: markieren zuerst eines Bereichs des Inhalts des nichtflüchtigen Speichers, der diese ersten Daten als ungültig empfängt und übertragen dieser ersten Daten zum nichtflüchtigen Speicher in einer ersten DMA-Transaktion; und markieren zweitens der Menge der Inhalte des nichtflüchtigen Speichers als gültig in einer zweiten DMA-Transaktion, umfasst.

9. Verfahren nach Anspruch 1, wobei das zweite Controllermanagementmodul einen nichtflüchtigen Speicher umfasst und der Spiegelungsschritt: speichern eines ersten String in einem ersten Speicher-Bereich des nichtflüchtigen Speichers, übertragen der ersten Daten zum nichtflüchtigen Speicher und speichern des ersten String in einem zweiten Bereich des nichtflüchtigen Speichers, umfasst.

10. Verfahren nach Anspruch 9, wobei der Spiegelungsschritt ausgeführt wird unter Verwendung einer einzelnen DMA-Transaktion.

11. Verfahren nach Anspruch 1, wobei das erste Controllermanagementmodul einen ersten Prozessor umfasst und ferner den Schritt umfasst: speichern von Informationen in dem Speicher-Array unter Benutzung des ersten Prozessors und Benutzung einer XOR-Engine, um Parität zu bestimmen.

12. Vorrichtung zum Spiegeln von Daten in einem Speichersystem, das einen Speicher-Array umfasst, bestehend aus: einem ersten Controllermanagementmodul, das einen ersten Prozessor und eine erste Direct-Memory-Access-Engine umfasst, wobei der erste Prozessor genutzt wird zum Steuern der Leseoperationen und Schreiboperationen unter Einbezug des Speicher-Arrays und der ersten Direct-Memory-Access-Engine, die genutzt wird zum Speichern von Daten, die durch das erste Controllermanagementmodul erhalten werden;
ein zweites Controllermanagementmodul, das einen zweiten Prozessor und eine zweite Direct-Memory-Access-Engine umfasst, wobei der zweite Prozessor genutzt wird zum Steuern von Leseoperationen und Schreiboperationen unter Einbezug des Speicher-Arrays und der zweiten Direct-Memory-Access-Engine, die genutzt wird zum Speichern von Daten, die durch das zweite Controllermanagementmodul erhalten werden;
wobei die ersten Daten vom ersten Controllermanagementmodul von einem Host erhalten werden und die ersten Daten gespiegelt werden von dem ersten Controllermanagementmodul zum zweiten Controllermanagementmodul, wobei die erste Direct-Memory-Access-Engine genutzt wird, während eine Unterbrechung des zweiten Prozessors vermieden wird.

13. Vorrichtung nach Anspruch 12, wobei das erste Controllermanagementmodul nichtflüchtige Speicher umfasst und die ersten Daten in dem nichtflüchtigen Speicher gespeichert werden.

14. Vorrichtung nach Anspruch 12, wobei die erste Direct-Memory-Access-Engine separat ist von aber in Kommunikation mit dem ersten Prozessor und der erste Prozessor die Spiegelung der ersten Daten startet, wobei die erste Direct-Memory-Access-Engine genutzt wird.

15. Vorrichtung nach Anspruch 12, wobei das erste Controllermanagementmodul ein Field-Programmable-Gate-Array umfasst und die erste Direct-Memory-Access-Engine in Kommunikation ist mit zumindest Teilen davon.

16. Vorrichtung nach Anspruch 12, die weiter umfasst: ein erstes Kanalschnittstellenmodul, das ein erstes shared Path hat, das erste Kanalschnittstellenmodul, das mit dem ersten Controllermanagementmodul kommuniziert und in welchem der erste shared Path genutzt wird zum Übertragen der ersten Daten zwischen dem ersten Controllermanagementmodul und dem zweiten Controllermanagementmodul.

17. Vorrichtung nach Anspruch 16, die weiter umfasst: ein passives Backplane, das das erste Kanalschnittstellenmodul und das erste Controllermanagementmodul verbindet.

18. Vorrichtung nach Anspruch 12, wobei der zweite Prozessor die Operationen steuert, die dem zweiten Controllermanagementmodul zugeordnet sind, während die ersten Daten zum zweiten Controllermanagementmodul gespiegelt werden.

19. Vorrichtung nach Anspruch 12, wobei die ersten Daten zum zweiten Controllermanagementmodul gespiegelt werden unabhängig von der zweiten Direct-Memory-Access-Engine.

20. Vorrichtung nach Anspruch 12, wobei das zweite Controllermanagementmodul einen nichtflüchtigen Speicher umfasst und die ersten Daten in dem nichtflüchtigen Speicher gespeichert werden unabhängig von dem zweiten Prozessor.

21. Vorrichtung nach Anspruch 12, wobei das zweite Controllermanagementmodul einen nichtflüchtigen Speicher umfasst und die erste Direct-Memory-Access-Engine genutzt wird zum Bereitstellen eines Hinweises, dass zumindest für Teile des nichtflüchtigen Speichers, die die ersten Daten erhalten sollen, diese Teile ungültig sind und nachdem die ersten Daten erhalten werden vom nichtflüchtigen Spei-

cher, die erste Direct-Memory-Access-Engine genutzt wird, um diese Abschnitte als gültig zu markieren.

22. Vorrichtung nach Anspruch 12, wobei das zweite Controllermanagementmodul einen nichtflüchtigen Speicher umfasst, der zumindest eine erste Speicherregion und eine zweite Speicherregion aufweist und diese erste Direct-Memory-Access-Engine genutzt wird zum Bereitstellen eines ersten String zur Speicherung in der ersten Speicherregion bevor die ersten Daten erhalten werden von dem nichtflüchtigen Speicher und zum Bereitstellen des ersten Strings zur Speicherung in der zweiten Speicherregion nachdem die ersten Daten erhalten werden vom nichtflüchtigen Speicher.

23. Vorrichtung nach Anspruch 21, wobei die erste Direct-Memory-Access-Engine betriebsbereit ist, um den ersten String zur Verfügung zu stellen, um die ersten Daten zu übertragen und um den zweiten String in einer einzelnen Direct-Memory-Access-Transaktion bereitzustellen.

Es folgen 13 Blatt Zeichnungen

Anhängende Zeichnungen

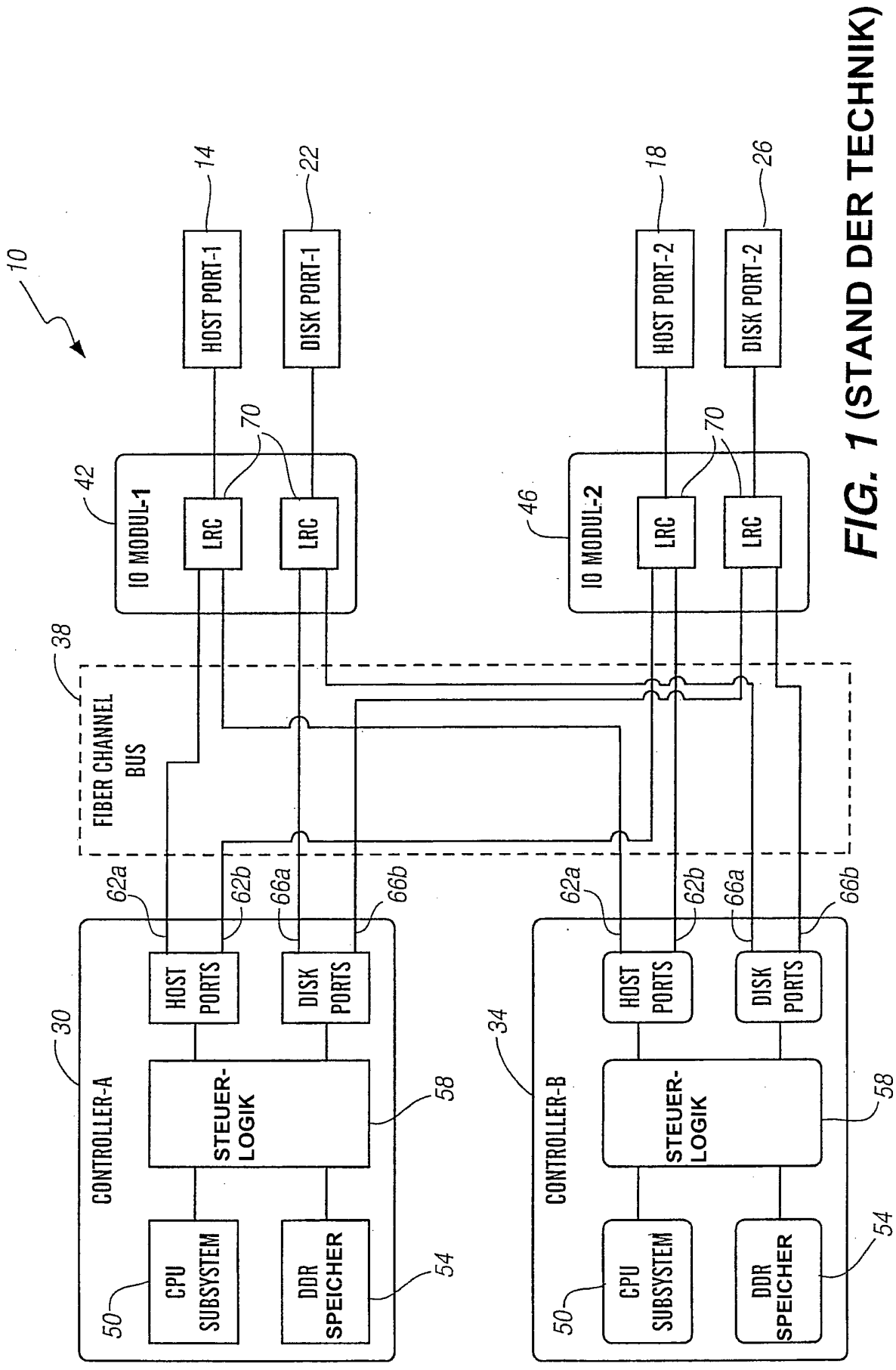


FIG. 1 (STAND DER TECHNIK)

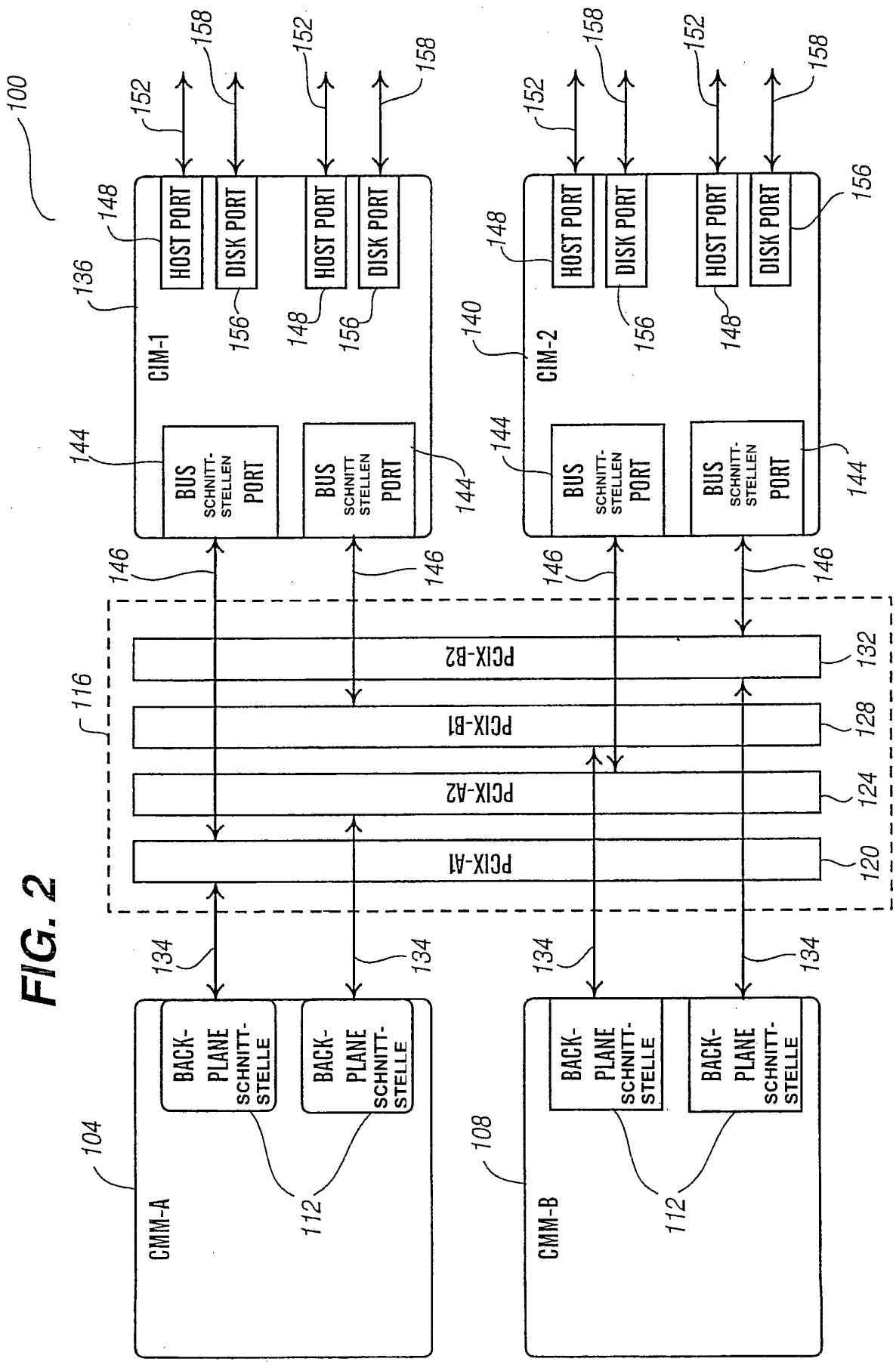

100
CIM-1
CIM-2
HOST PORT
DISK PORT
BUS SCHNITT-STELLEN PORT
PCIX-B2
PCIX-B1
PCIX-A2
PCIX-A1
CMM-A
CMM-B
BACK-PLANE SCHNITT-STELLE
104
108
112
116
120
124
128
132
134
136
140
144
146
148
152
156
158

FIG. 2

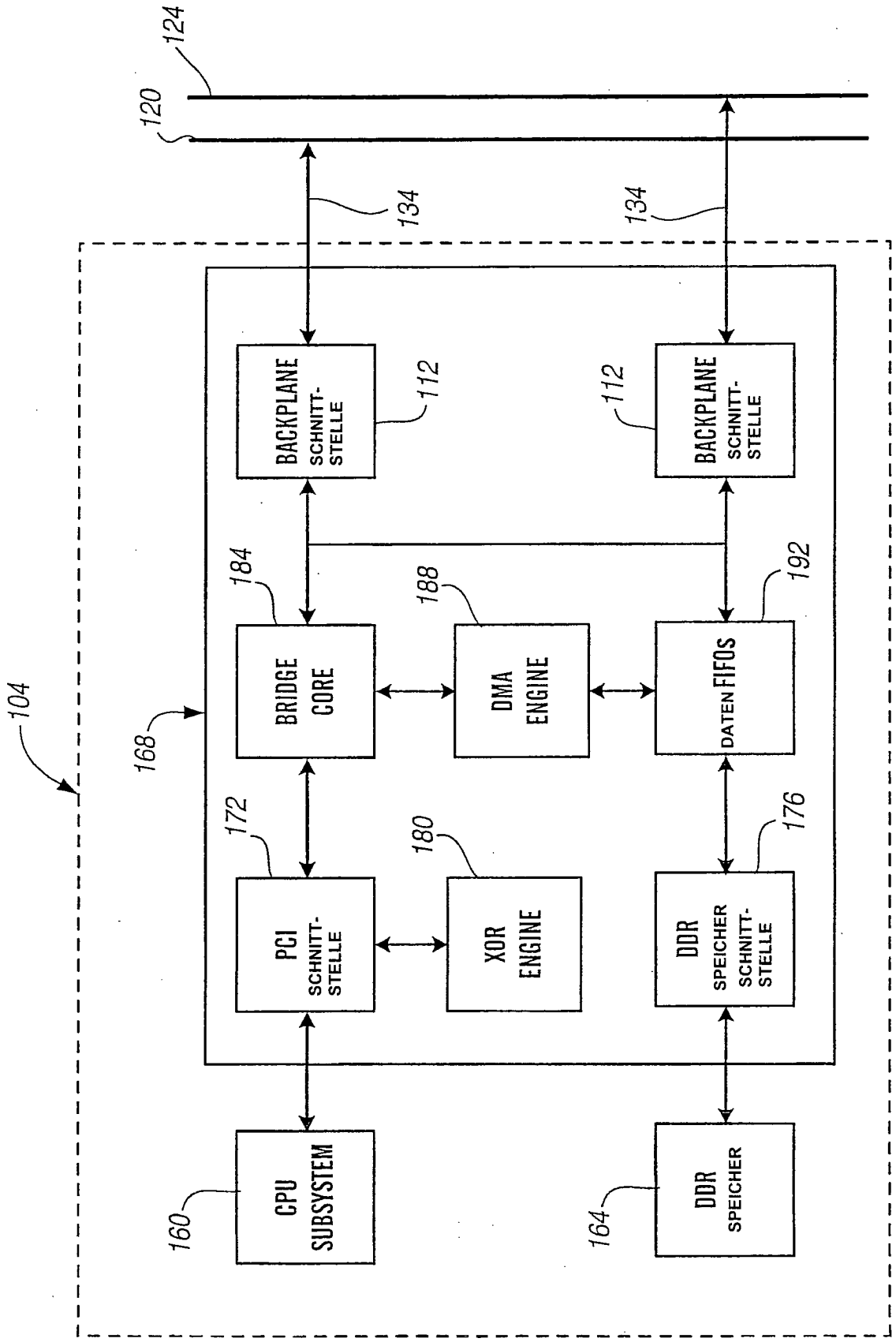

124
120
134
134
104
168
BACKPLANE SCHNITT-STELLE
112
112
BACKPLANE SCHNITT-STELLE
184
188
192
BRIDGE CORE
DMA ENGINE
DATEN FIFOS
172
180
176
PCI SCHNITT-STELLE
XOR ENGINE
DDR SPEICHER SCHNITT-STELLE
160
CPU SUBSYSTEM
164
DDR SPEICHER

FIG. 3

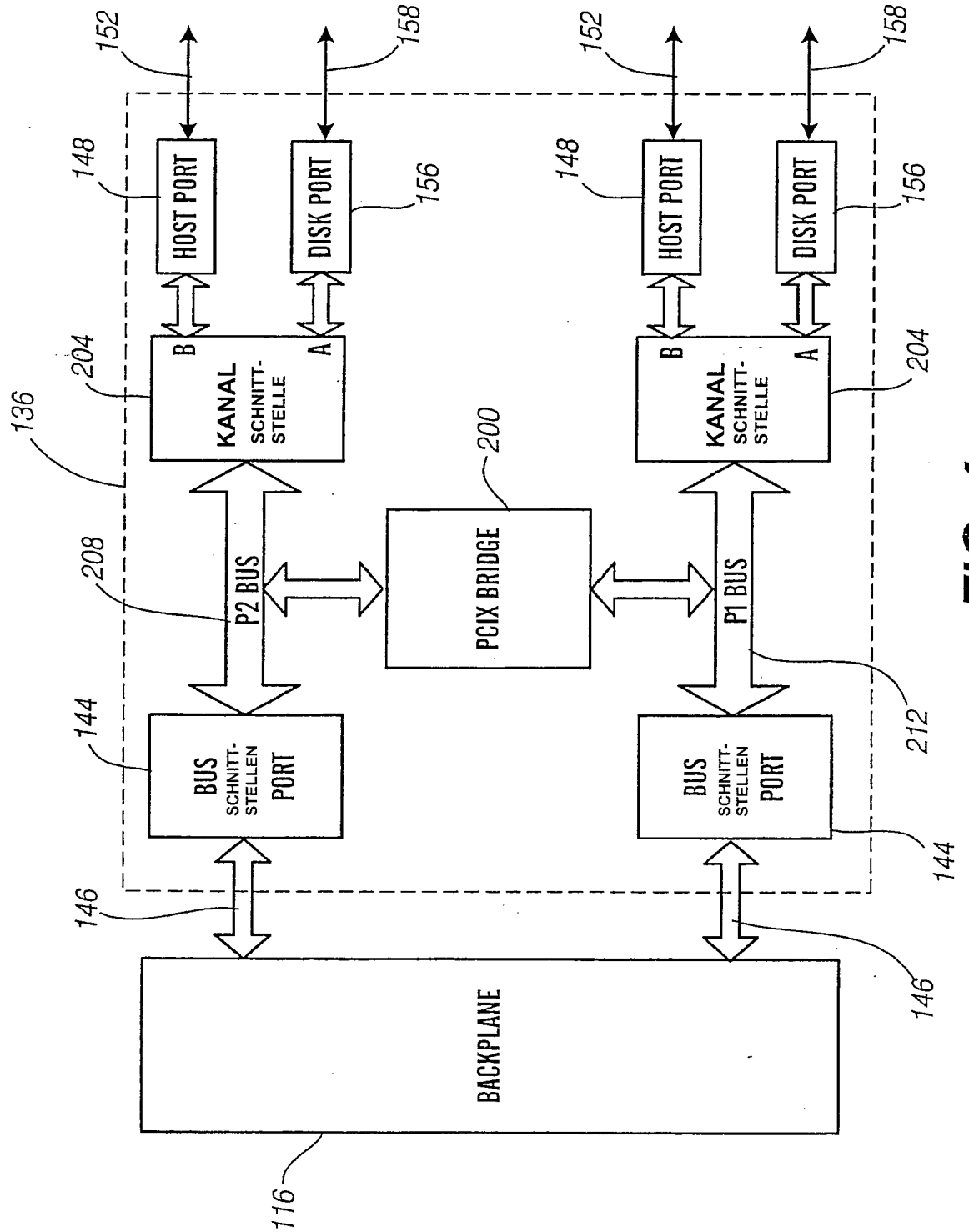

152
158
148
HOST PORT
DISK PORT
156
204
B
A
KANAL SCHNITT-STELLE
136
200
PCIX BRIDGE
208
P2 BUS
P1 BUS
212
144
BUS SCHNITT-STELLEN PORT
146
BACKPLANE
116

FIG. 4

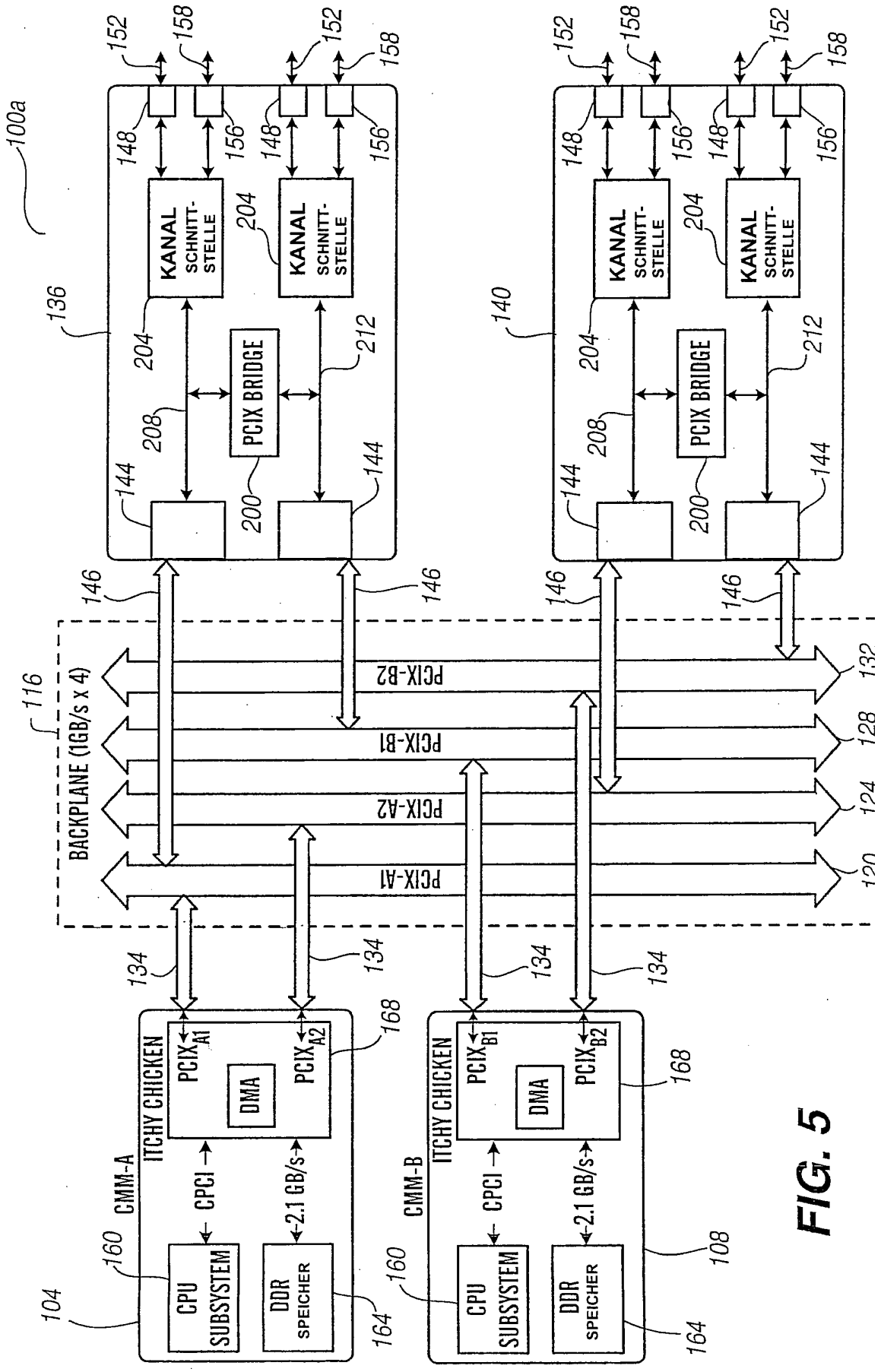

100a
136
140
KANAL SCHNITT-STELLE
PCIX BRIDGE
BACKPLANE (1GB/s x 4)
116
PCIX-B2
PCIX-B1
PCIX-A2
PCIX-A1
CMM-A
CMM-B
ITCHY CHICKEN
PCIX A1
PCIX A2
PCIX B1
PCIX B2
DMA
CPCI
2.1 GB/s
CPU SUBSYSTEM
DDR SPEICHER
104
108
132
128
124
120
134
144
146
148
152
156
158
160
164
168
200
204
208
212

FIG. 5

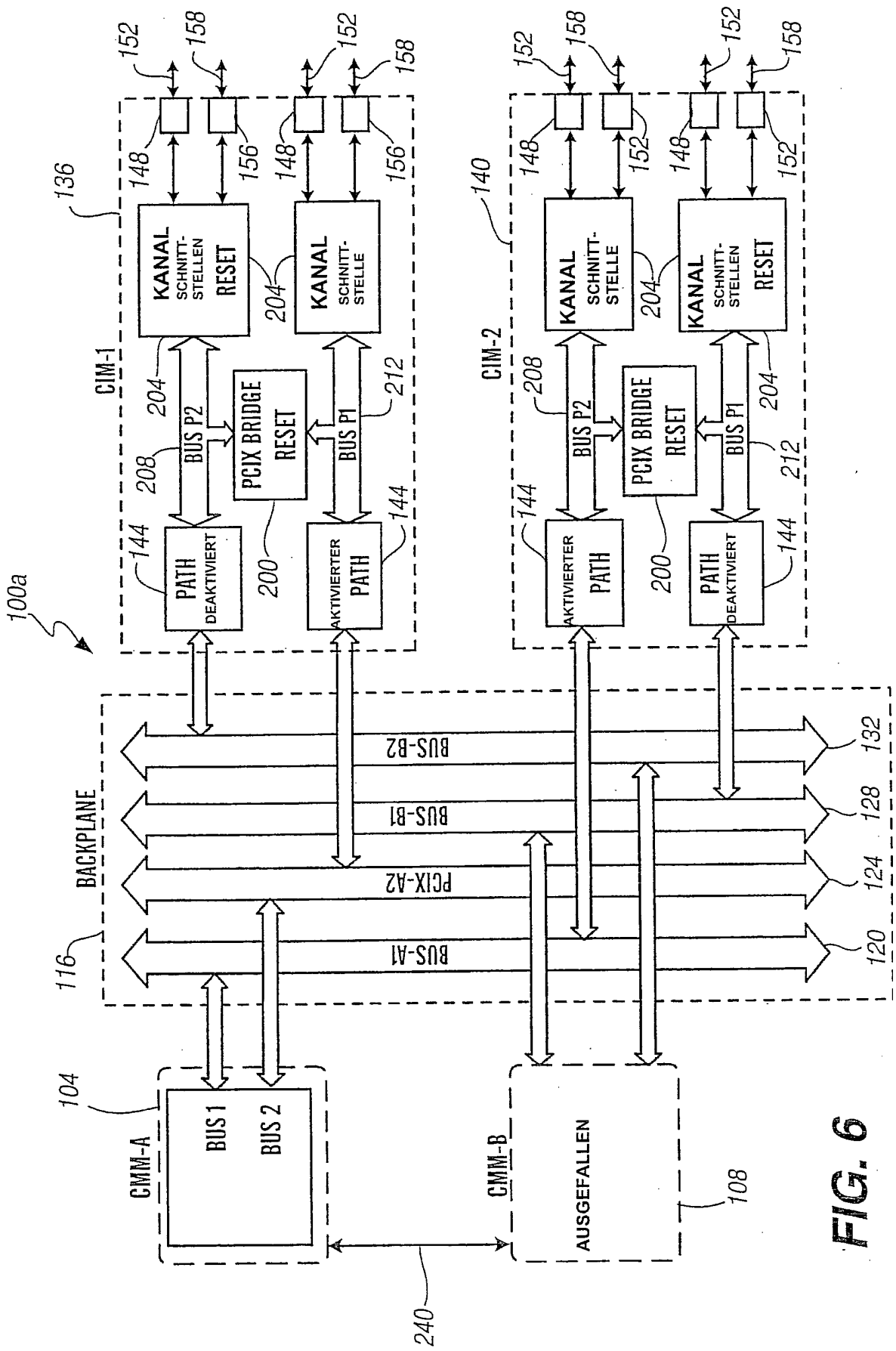

100a
CIM-1
CIM-2
136
140
KANAL SCHNITT-STELLEN RESET
KANAL SCHNITT-STELLE
PCIX BRIDGE RESET
BUS P1
BUS P2
PATH DEAKTIVIERT
AKTIVIERTER PATH
BACKPLANE
116
BUS-B2
BUS-B1
PCIX-A2
BUS-A1
132
128
124
120
CMM-A
CMM-B
104
BUS 1
BUS 2
AUSGEFALLEN
108
240
148
152
156
158
144
200
204
208
212

FIG. 6

FIG. 7

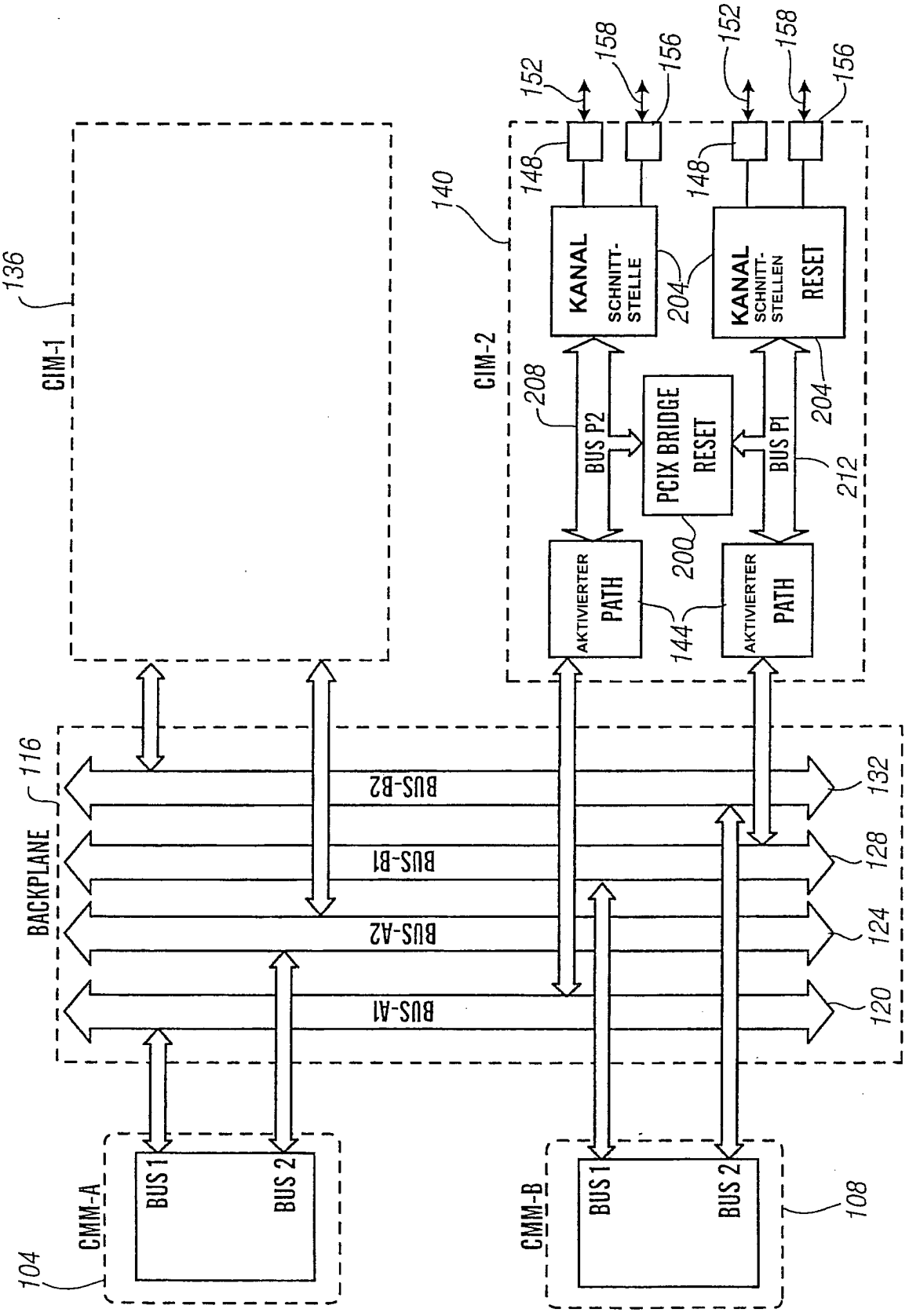
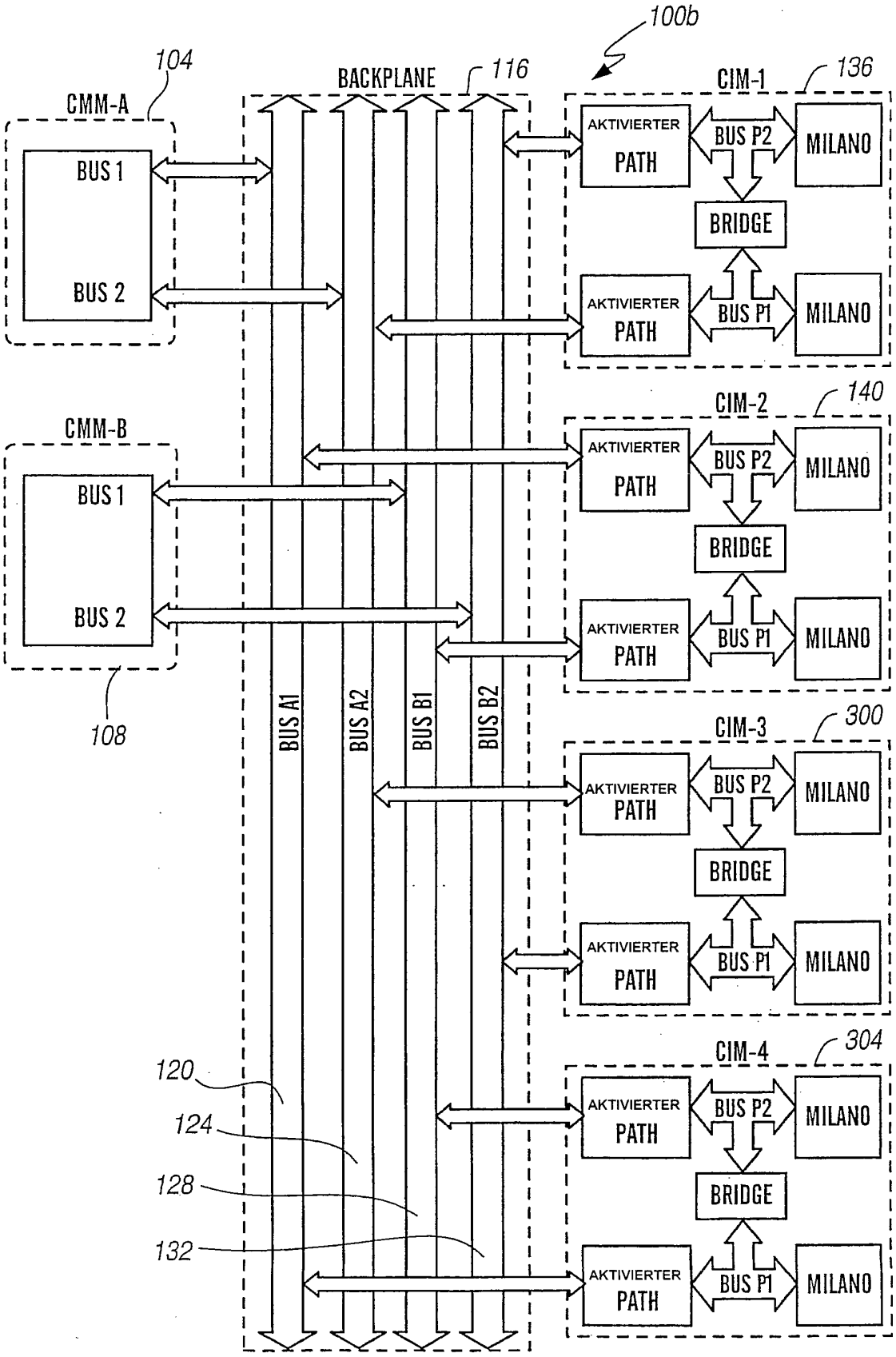

136
CIM-1
140
CIM-2
152
158
156
148
KANAL
SCHNITT-
STELLE
204
KANAL
SCHNITT-
STELLEN
RESET
208
BUS P2
PCIX BRIDGE
RESET
BUS P1
212
200
AKTIVIERTER
PATH
144
116
BACKPLANE
BUS-B2
BUS-B1
BUS-A2
BUS-A1
132
128
124
120
CMM-A
BUS 1
BUS 2
104
CMM-B
108

FIG. 8


100b
104
CMM-A
BUS 1
BUS 2
BACKPLANE
116
CIM-1
136
AKTIVIERTER PATH
BUS P2
MILANO
BRIDGE
AKTIVIERTER PATH
BUS P1
MILANO
CIM-2
140
CMM-B
BUS 1
BUS 2
108
BUS A1
BUS A2
BUS B1
BUS B2
CIM-3
300
CIM-4
304
120
124
128
132

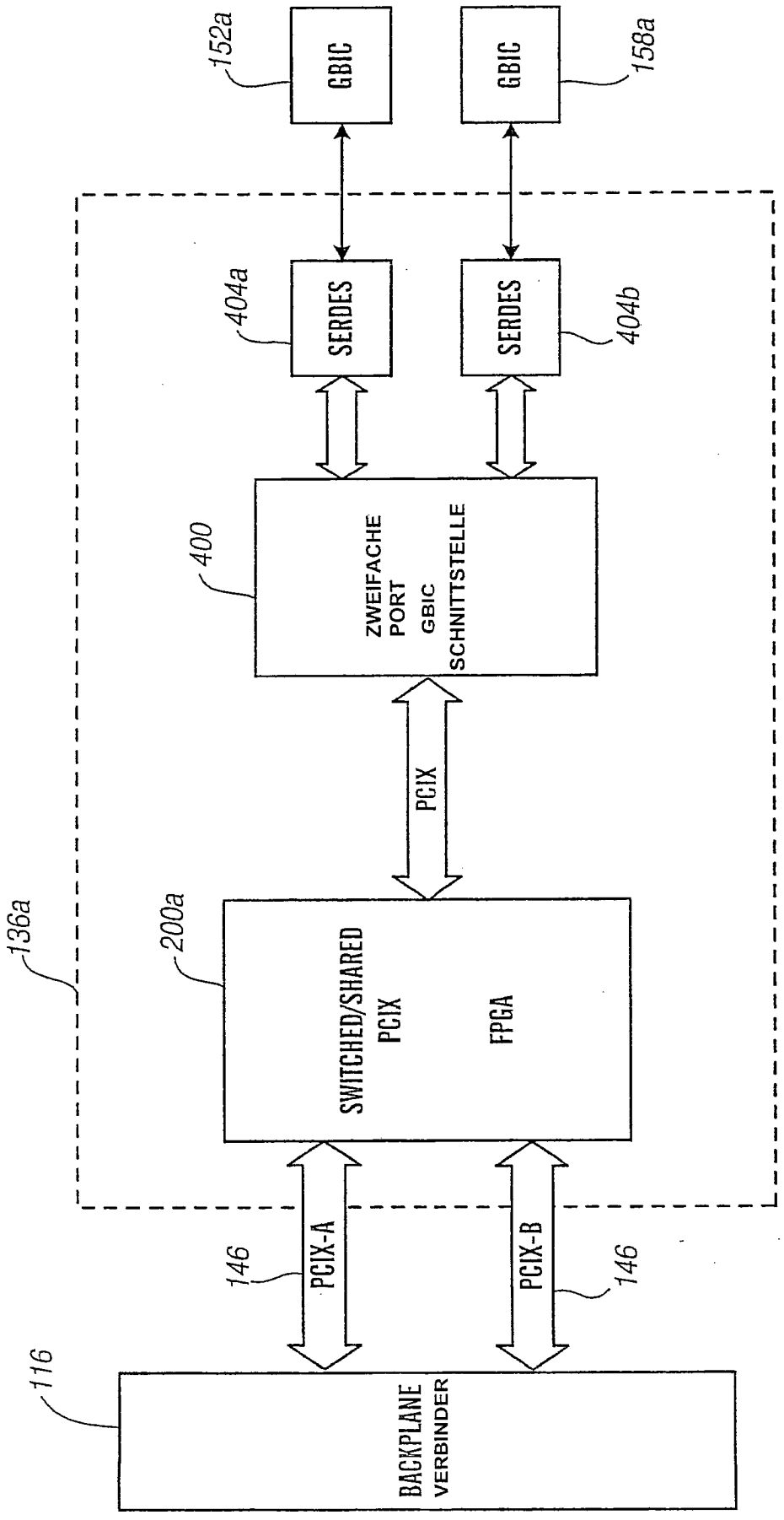

152a
GBIC
158a
GBIC
404a
SERDES
404b
SERDES
400
ZWEIFACHE PORT GBIC SCHNITTSTELLE
PCIX
136a
200a
SWITCHED/SHARED PCIX FPGA
146
PCIX-A
PCIX-B
146
116
BACKPLANE VERBINDER

FIG. 9

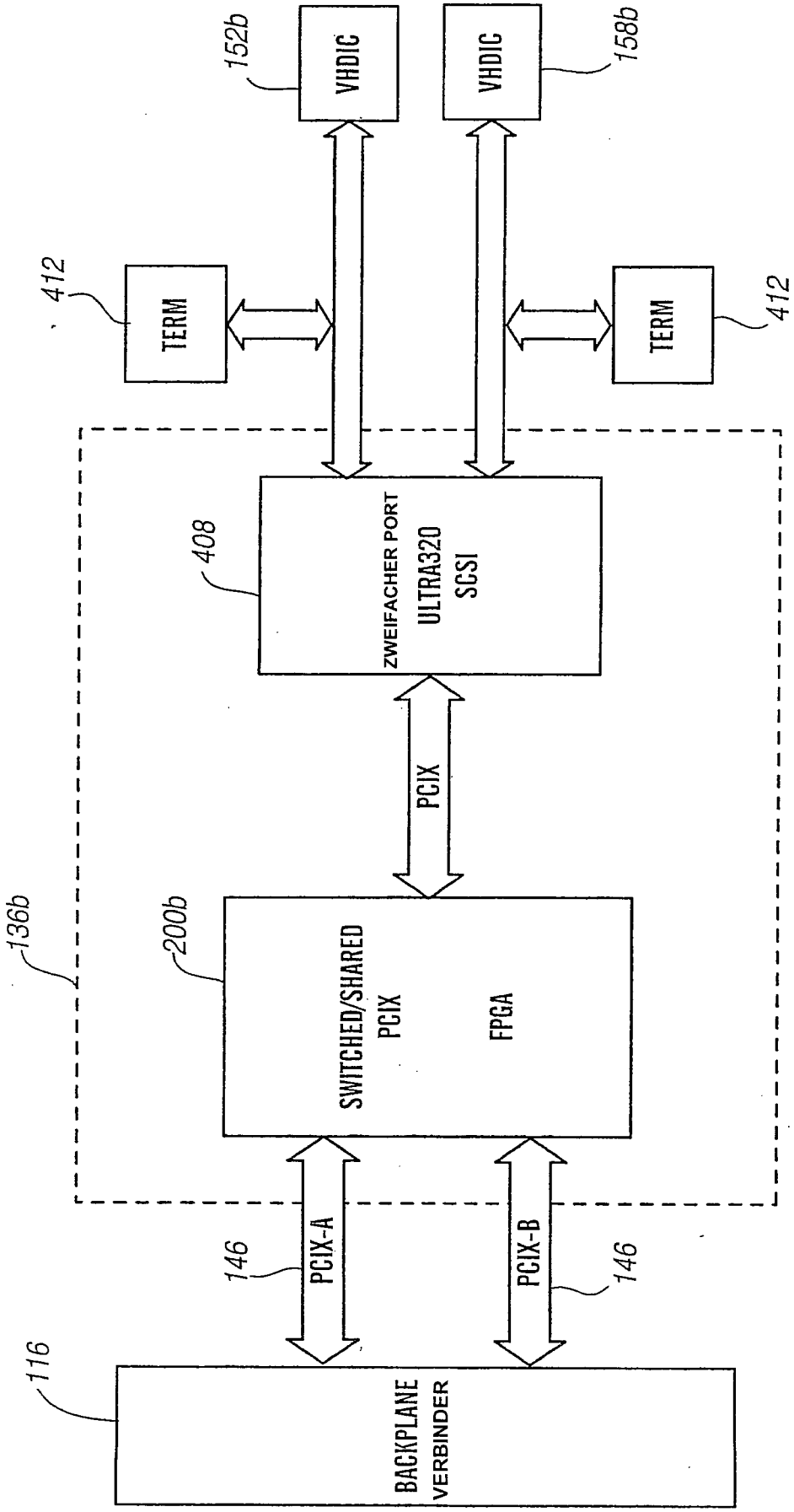

152b
VHDIC
158b
VHDIC
412
TERM
TERM
412
408
ZWEIFACHER PORT
ULTRA320
SCSI
PCIX
136b
200b
SWITCHED/SHARED
PCIX
FPGA
146
PCIX-A
PCIX-B
146
116
BACKPLANE
VERBINDER

FIG. 10

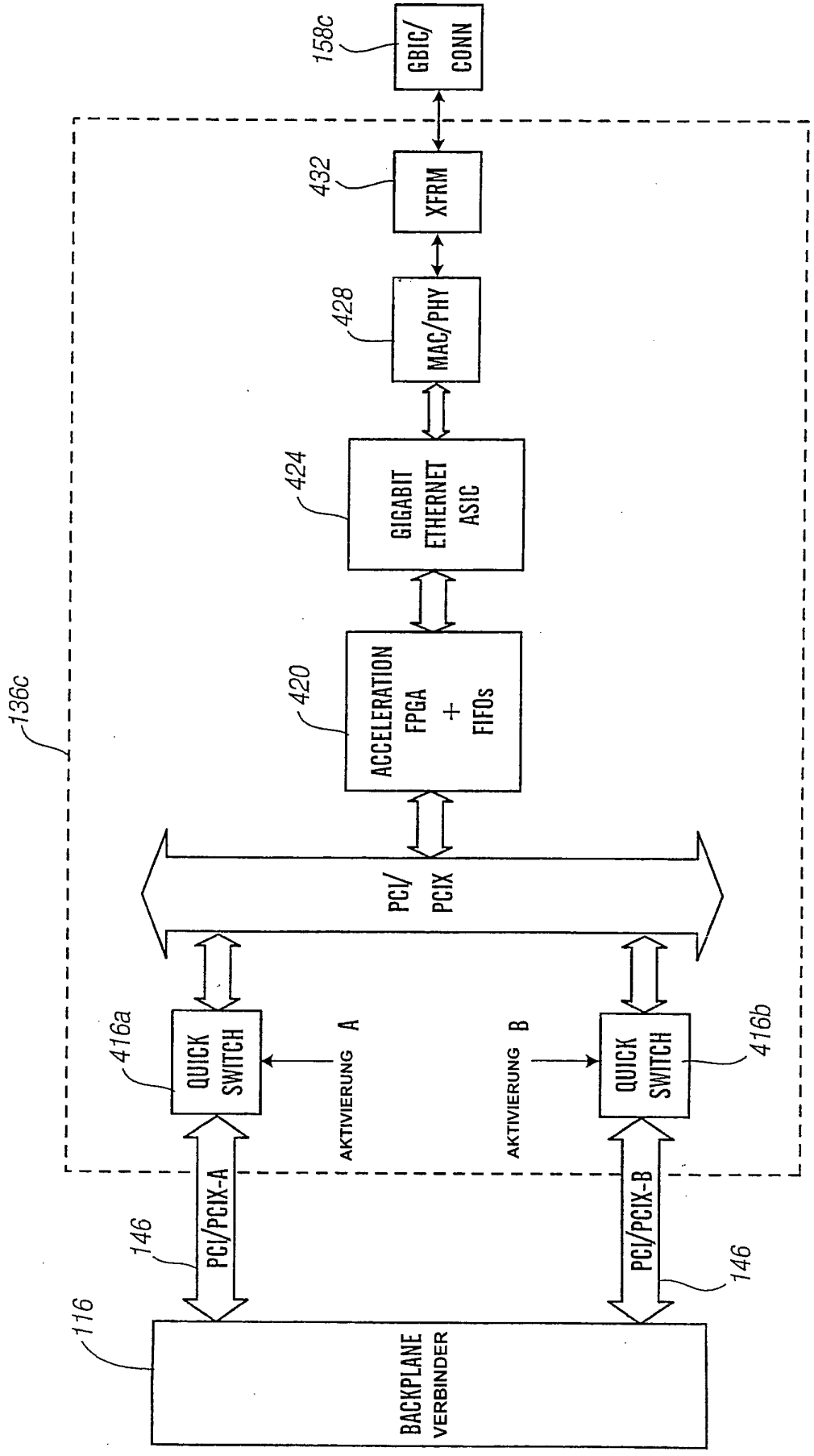

158c
GBIC/
CONN
432
XFRM
428
MAC/PHY
424
GIGABIT
ETHERNET
ASIC
420
ACCELERATION
FPGA
+
FIFOs
136c
PCI/
PCIX
416a
QUICK
SWITCH
AKTIVIERUNG A
AKTIVIERUNG B
QUICK
SWITCH
416b
146
PCI/PCIX-A
PCI/PCIX-B
146
116
BACKPLANE
VERBINDER

FIG. 11

FIG. 12 (STAND DER TECHNIK)

SPIEGELUNG VON DATEN BEI EINEM AKTIV/AKTIV CONTROLLERPAAR UNTER VERWENDUNG VON PROZESSOR-INTERRUPTS ÜBER SHARED DISK CHANNELS

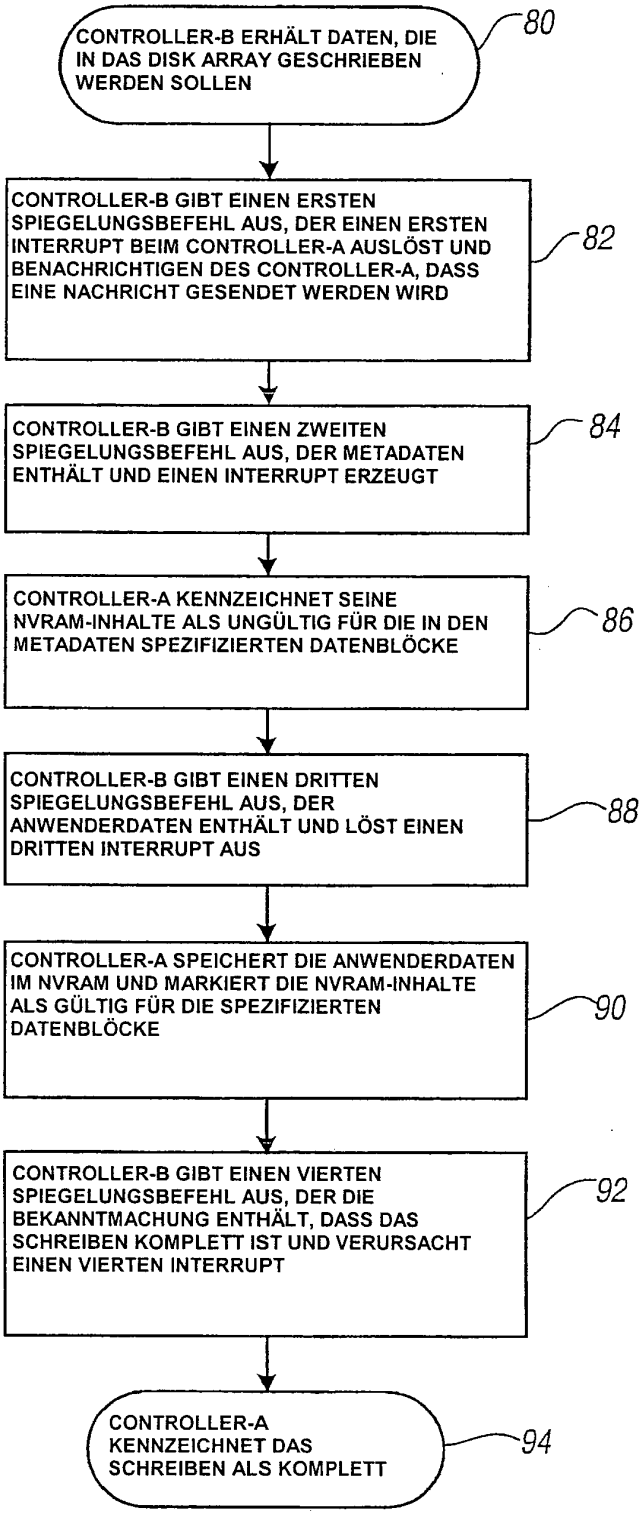
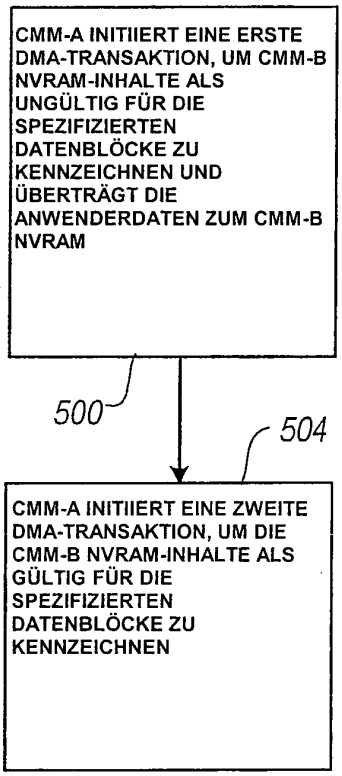


FIG. 13



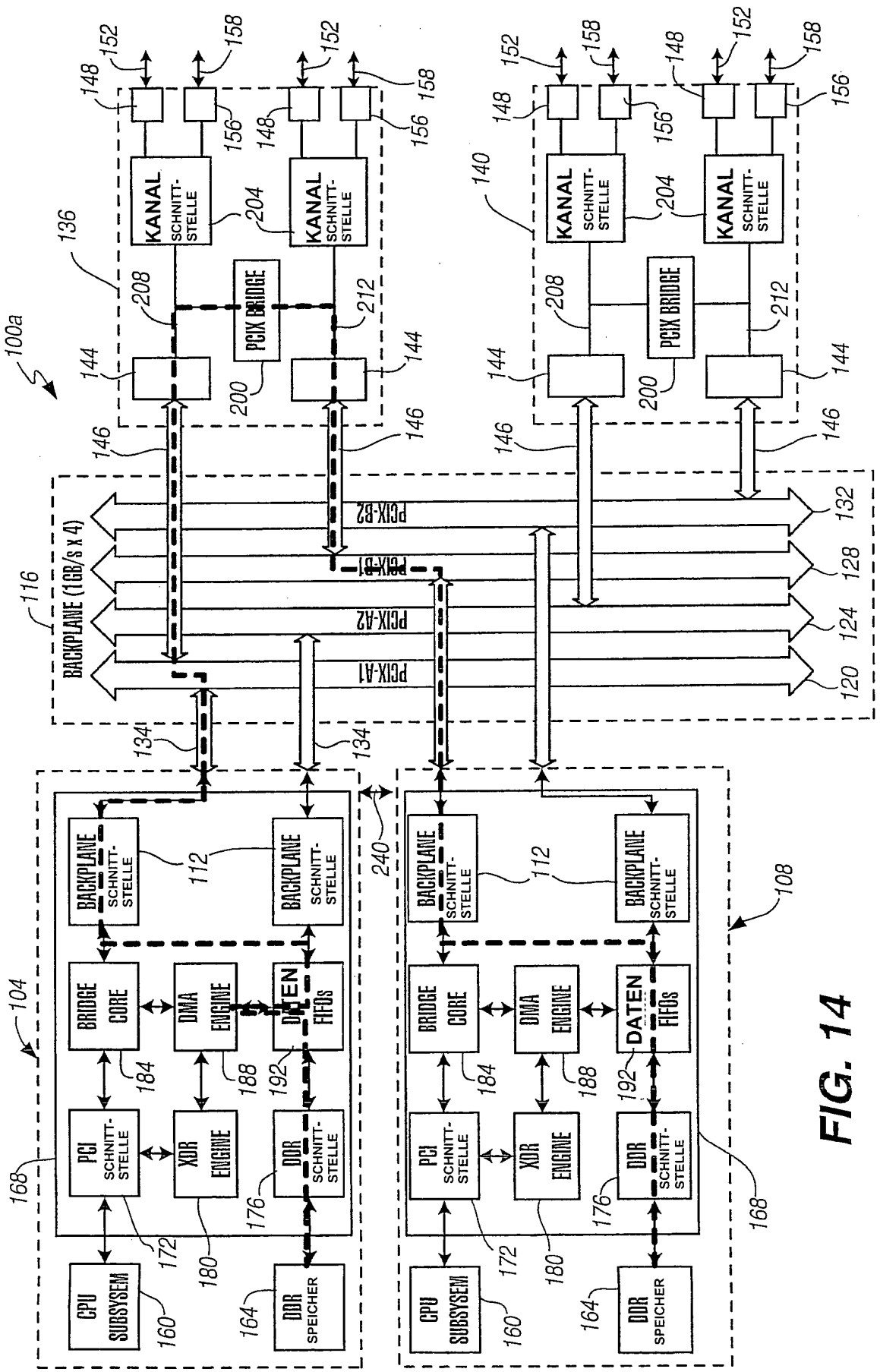

100a
116
BACKPLANE (10GB/s x 4)
PCIX-B2
PCIX-B1
PCIX-A2
PCIX-A1
132
128
124
120
136
140
KANAL SCHNITT-STELLE
PCIX BRIDGE
204
208
212
200
144
146
148
152
156
158
134
240
108
104
BACKPLANE SCHNITT-STELLE
112
BRIDGE CORE
DMA ENGINE
DATEN FIFOs
184
188
192
PCI SCHNITT-STELLE
XOR ENGINE
DDR SCHNITT-STELLE
168
172
176
180
CPU SUBSYSEM
160
DDR SPEICHER
164

FIG. 14