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

(10) **DE 697 28 002 T2** 2004.09.16

(12) # Übersetzung der europäischen Patentschrift

(97) **EP 0 817 117 B1**
(21) Deutsches Aktenzeichen: **697 28 002.0**
(96) Europäisches Aktenzeichen: **97 110 798.2**
(96) Europäischer Anmeldetag: **01.07.1997**
(97) Erstveröffentlichung durch das EPA: **07.01.1998**
(97) Veröffentlichungstag der Patenterteilung beim EPA: **10.03.2004**
(47) Veröffentlichungstag im Patentblatt: **16.09.2004**

(51) Int Cl.[7]: **G06T 1/20**
G06T 9/00

(30) Unionspriorität:
673494 **01.07.1996** **US**

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

(73) Patentinhaber:
Sun Microsystems, Inc., Palo Alto, Calif., US

(72) Erfinder:
Deering, Michael F., Los Altos, US

(74) Vertreter:
HOFFMANN · EITLE, 81925 München

(54) Bezeichnung: **Steuerprozessor für einen drei-dimensionalen Beschleuniger, der die Fähigkeit geometrischer Dekompression besitzt und Verfahren zur Bearbeitung von geometrischen Daten in diesem Beschleuniger**

Beschreibung

[0001] Die vorliegende Erfindung bezieht sich auf einen Befehlsprozessor für einen Graphikbeschleuniger und ein Verfahren zum Verarbeiten von Geometriedaten in demselben, und insbesondere, auf eine verbesserte Befehlsprozessorarchitektur für einen 3-D Graphikbeschleuniger, der einen ersten Datenpfad für nicht-komprimierte Geometriedaten einschließt und einen zweiten Datenpfad, der eine Geometriedekomprimierungseinheit zum Empfangen und Dekomprimieren komprimierter Geometriedaten umfasst.

[0002] Ein dreidimensionaler (3-D) Graphikbeschleuniger ist ein spezialisiertes Graphikwiedergabeuntersystem für ein Computersystem, das ausgelegt ist, um die 3-D Wiedergabefunktionen von dem Host-Prozessor abzunehmen, um somit verbesserte Systemleistung bereitzustellen. In einem System mit einem 3-D Graphikbeschleuniger erzeugt ein Anwendungsprogramm, das auf dem Host-Prozessor des Computersystems ausgeführt wird, dreidimensionale Geometriedaten, die dreidimensionale Graphikelemente zum Anzeigen auf einer Anzeigevorrichtung definieren. Das Anwendungsprogramm veranlasst den Host-Prozessor, die Geometriedaten an den Graphikbeschleuniger zu übertragen. Der Graphikbeschleuniger empfängt die Geometriedaten und gibt die korrespondierenden Graphikelemente auf der Anzeigevorrichtung wieder.

[0003] Die Anzeigearchitektur eines dreidimensionalen Hochleistungsgraphiksystems verkörpert historisch einen Ausgleich zwischen ansteigender Systemleistung und Minimierung von Systemkosten. Frühere Graphiksysteme leiden jedoch gewöhnlich unter entweder einer begrenzten Leistung oder hohen Kosten infolge einer Vielfalt von Systembeschränkungen.

[0004] Anwendungen, die dreidimensionale Graphiken darstellen, erfordern einen gewaltigen Umfang von Verarbeitungsfähigkeiten. Zum Beispiel benötigt bei einem Computersystem zum Erzeugen eines geglätteten 3-D Bewegungsvideos das Computersystem das Aufrechterhalten einer Rahmenrate oder einer Update-Rate zwischen 20 bis 30 Rahmen pro Sekunde. Das erfordert einen 3-D Computergraphikbeschleuniger, der in der Lage ist, über eine Million Dreiecke pro Sekunde zu verarbeiten.

[0005] Im Allgemeinen haben 3-D Computergraphikbeschleuniger drei große Engpässe gehabt, die die Leistung begrenzen. Ein erster Engpass ist das Erforderniss, dass geometrische Wiedergabeprimitive, z. B. Linien und Dreiecke, von dem Hauptsystemspeicher in dem Host-Computer zu dem Graphikbeschleuniger übertragen werden müssen. Der Betrieb des Host-Prozessorspeichersystems und System-Bus', auf dem die Daten übertragen werden, kann die Übertragungsrate dieser geometrischen Wiedergabeprimitive von dem Host-Speicher an den 3-D Beschleuniger begrenzen. Ein zweiter Engpass sind die Scheitelpunktverarbeitungserfordernisse, einschließlich Transformation, Beleuchtung, Set-Up, etc. innerhalb des Beschleunigers. Ein dritter Engpass ist die Geschwindigkeit, mit der Pixel von Primitiven in den Rahmenspeicher gefüllt werden können.

[0006] Um eine 3-D Graphikarchitektur einer höheren Leistung zu bauen, muss der Durchsatz in allen der obigen drei Bereiche erhöht werden. Wie oben erwähnt, ist eine der Hauptengpässe in 3-D Graphikarchitekturen traditionell die Geschwindigkeit gewesen, bei der Pixel von Primitiven in den Rahmenpufferspeicher gefüllt werden. Die Systeme haben traditionell einen Dual-Port Video RAM (VRAM) oder einen verschachtelten DRAM verwendet, um zu versuchen, einen höheren Durchsatz zu erreichen. Ein neuer Typ Videospeicher, der als 3DRAM bezeichnet wird, erhöht die Pixeldurchsatzrate um eine Größenordnung. Mit dem Verwenden von 3DRAM in einem Graphikbeschleunigersystem befindet sich der 3-D Wiedergabeengpass nicht länger bei der Füllrate, mit der Pixel von Primitiven in den Rahmenspeicher gefüllt werden. Eher umfasst der Leistungsengpass typischerweise mit dem Verwenden von 3DRAM entweder die Transferrate von Geometriedaten auf dem System-Bus oder die 3-D Graphikbeschleunigerbearbeitung, einschließlich der Scheitelpunktverarbeitung. Daher ist eine neue 3-D Graphikbeschleunigerarchitektur gewünscht, die eine erhöhte Leistung bereitstellt.

[0007] US. Patent 5,408,605 von Deering, die zu Sun Microsystems gehören, offenbart einen Befehlsprozessor für einen 3-D Graphikbeschleuniger gemäß dem Stand der Technik. Wie gezeigt, schließt diese 3-D Graphikarchitektur aus dem Stand der Technik einen Befehls-Preprozessor ein, der einen oder mehrere Gleitkommaprozessoren koppelt. Jeder der Gleitkommaprozessoren wiederum koppelt eine Mehrzahl von Zeichnungsprozessoren.

[0008] In diesem System ist der Computersystemspeicher erforderlich, um einen großen Betrag von Geometriedaten zu dem Befehlsprozessor in dem 3-D Graphikbeschleuniger zu übertragen. Wie oben erwähnt, kann ein Betreiben des Host-Prozessorspeichersystems und System-Bus', auf dem Daten übertragen werden, die Geometriedatenübertragungsrate von dem Host-Speicher an den 3-D Graphikbeschleuniger begrenzen. Daher ist ein verbesserter 3-D Graphikbeschleuniger gewünscht, der verbesserte Geometriedatenübertragungsfähigkeiten einschließt.

Detaillierte Beschreibung der Ausführungsformen

[0009] Gemäß einem Aspekt der Erfindung wird ein Befehlsprozessor für einen Graphikbeschleuniger bereitgestellt, umfassend:

einen oder mehr Eingangspuffer, die für ein Koppeln an einen Bus angepasst sind, zum Empfangen von Geometrieeingangsdaten, wobei die Geometrieeingangsdaten komprimierte Geometrieeingangsdaten

und nicht-komprimierte Geometrieeingangsdaten einschließen;
einen ersten Datenpfad, der mit dem einen oder den mehreren Eingangspuffern zum Übertragen der nicht-komprimierten Geometriedaten gekoppelt ist;
einen zweiten Datenpfad, der mit dem einen oder den mehreren Eingangspuffern zum Empfangen der komprimierten Geometriedaten gekoppelt ist, wobei der zweite Datenpfad eine Geometriedekomprimierungseinheit einschließt, die an einen Ausgang der Eingangspuffer zum Empfangen der komprimierten Geometrieeingangsdaten gekoppelt ist, wobei die Dekomprimierungseinheit zum Dekomprimieren der komprimierten Geometrieeingangsdaten zum Erstellen dekomprimierter Geometrieeingangsdaten betrieben wird;
einen oder mehrere Ausgangspuffer, die zum Empfangen von Daten von dem ersten Datenpfad und dem zweiten Datenpfad gekoppelt sind, wobei die Ausgangspuffer Ausgangsprimitivdaten bereitstellen.

[0010] Gemäß einem anderen Aspekt der Erfindung wird ein 3-D Graphikbeschleuniger bereitgestellt zum Durchführen dreidimensionaler Graphikbeschleunigungsfunktionen, umfassend:
einen Rahmenpufferspeicher;
den Befehlsprozessor zum Empfangen von Hochpegelzeichnungsbefehlen zum Zeichnen dreidimensionaler Objekte;
eine Mehrzahl von Gleitkommablöcken, die an den Befehlsprozessor zum Durchführen von Gleitkommaoperationen gekoppelt sind, wobei die Mehrzahl von Gleitkommablöcken die Hochpegelbefehle von dem Befehlsprozessor empfangen und geometriesche Gleitkommaoperationen in Antwort auf die Hochpegelbefehle durchführen, wobei die Mehrzahl von Gleitkommablöcken jeweils geometrieche Primitivdaten produzieren;
einen oder mehrere Zeichnungsblöcke, die an die Mehrzahl von Gleitkommablöcke gekoppelt sind, die geometrische Primitivdaten von der Mehrzahl von Gleitkommablöcken empfangen, wobei der eine oder mehrere Zeichnungsblöcke an den Rahmenpufferspeicher gekoppelt sind, zum Wiedergeben dreidimensionaler Objektpixeldaten in den Rahmenspeicherpuffer; und
einen Digital-Analogumwandler, der an den Rahmenpufferspeicher zum Empfangen von Pixeldaten von dem Rahmenpufferspeicher gekoppelt ist und einen analogen Ausgang an einen Videomonitor liefert;
wobei der Befehlsprozessor umfasst:
einen oder mehrere Eingangspuffer, die für ein Koppeln an einen Bus angepasst sind, zum Empfangen von Geometrieeingangsdaten, wobei die Geometrieeingangsdaten komprimierte Geometrieeingangsdaten und nicht-komprimierte Geometrieeingangsdaten einschließen,
eine Geometriedekomprimierungseinheit, die an einen Ausgang der Eingangspuffer zum Empfangen der komprimierten Geometrieeingangsdaten gekoppelt ist, wobei die Dekomprimierungseinheit zum Dekomprimieren der komprimierten Geometrieeingangsdaten arbeitet, um dekomprimierte Geometrieeingangsdaten herzustellen;
einen Multiplexer, der einen ersten Eingang einschließt, der einen Ausgang von den Eingangspuffern empfängt, und einen zweiten Eingang zum Empfangen eines Ausgangs von der Geometriedekomprimierungseinheit, wobei der erste Eingang des Multiplexers nicht-komprimierte Geometriedaten von dem einen oder den mehreren Eingangspuffern empfängt, wobei der zweite Eingang des Multiplexers dekomprimierte Geometrieeingangsdaten von der Geometriedekomprimierungseinheit empfängt, wobei der Multiplexer einen Ausgang einschließt, der ausgewählt entweder die nicht-komprimierten Geometriedaten oder die dekomprimierten Geometrieeingangsdaten bereitstellt;
einen oder mehrere Ausgangspuffer, die gekoppelt sind zum Empfangen von Daten von dem Multiplexer, wobei die Ausgangspuffer Ausgangsprimitivdaten bereitstellen.

[0011] Gemäß einem dritten Aspekt der Erfindung wird ein Verfahren zum Verarbeiten von Geometriedaten in einem Graphikbeschleuniger bereitgestellt, umfassend:
Empfangen von Geometrieeingangsdaten, wobei die Geometrieeingangsdaten komprimierte Geometrieeingangsdaten und nicht-komprimierte Geometrieeingangsdaten einschließen;
Übertragen der komprimierten Geometriedaten auf einen ersten Datenpfad an eine Geometriedekomprimierungseinheit;
die Dekomprimierungseinheit, die komprimierte Geometrieeingangsdaten zum Herstellen dekomprimierter Geometrieeingangsdaten dekomprimiert;
Übertragen der nicht-komprimierten Geometriedaten an einen zweiten Datenpfad, wobei der zweite Datenpfad eine Geometriedekomprimierungseinheit nicht einschließt; und
ausgewähltes Bereitstellen entweder der nicht-komprimierten Geometriedaten oder der dekomprimierten Geometriedaten an einen dritten Datenpfad.

[0012] Zum besseren Verständnis der Erfindung und zum Zeigen, wie dieselbe wirkungsvoll ausgeführt werden kann, wird nun mittels Beispielen auf die nachfolgenden Zeichnungen Bezug genommen, in denen:

[0013] **Fig.** 1 ein Computersystem darstellt, das einen dreidimensionalen (3-D) Graphikbeschleuniger gemäß der vorliegenden Erfindung einschließt;

[0014] **Fig.** 2 ein vereinfachtes Blockdiagramm des Computersystems von **Fig.** 1 ist;

[0015] **Fig.** 3 ein Blockdiagramm ist, das den 3-D Graphikbeschleuniger gemäß dem bevorzugten Ausführungsbeispiel der vorliegenden Erfindung darstellt;

[0016] **Fig.** 4 ein Blockdiagramm ist, das einen Teil des 3-D Graphikbeschleunigers der **Fig.** 3 darstellt;

[0017] **Fig.** 5 ein Blockdiagramm ist, das den Befehlsprozessor in dem 3-D Graphikbeschleuniger ge-

mäß dem bevorzugten Ausführungsbeispiel der vorliegenden Erfindung darstellt;

[0018] **Fig.** 6 ein Blockdiagramm ist, das einen der Gleitkommaprozessoren in dem 3-D Graphikbeschleuniger gemäß des bevorzugten Ausführungsbeispiels der vorliegenden Erfindung darstellt;

[0019] **Fig.** 7 ein Blockdiagramm ist, das einen der Zeichnungsprozessoren in dem 3-D Graphikbeschleuniger gemäß des bevorzugten Ausführungsbeispiels der vorliegenden Erfindung darstellt;

[0020] **Fig.** 8 ein Blockdiagramm ist, das den CF Bus darstellt, der den Befehls-Preprozessor mit jedem der Gleitkommaprozessoren verbindet;

[0021] **Fig.** 9 ein Blockdiagramm ist, das den FD Bus darstellt, der jeden der Gleitkommaprozessoren mit jedem der Zeichnungsprozessoren verbindet; und

[0022] **Fig.** 10 ein Blockdiagramm ist, das den CDC Bus darstellt, der den Befehls-Preprozessor mit jedem der Zeichnungsprozessoren verbindet.

[0023] Das Folgende beschreibt ein System, das einen 3-D Graphikbeschleuniger für ein Computersystem umfasst, das eine verbesserte Leistung gegenüber den Auslegungen des Standes der Technik einschließt. Der 3-D Graphikbeschleuniger ist an einen System-Bus in dem Computersystem gekoppelt und empfängt komprimierte Geometriedaten von dem Speichersystem. Der 3-D Graphikbeschleuniger schließt einen Befehlsblock oder einen Preprozessor ein, eine Mehrzahl von Gleitprozessoren oder Blöcken, und einen oder mehrere Zeichnungsprozessoren oder Blöcke. Der Befehlsprozessor schließt einen ersten Datenpfad ein, der nicht-komprimierte Geometriedaten übermittelt und einen zweiten Datenpfad ein, der komprimierte Geometriedaten empfängt und die komprimierten Geometriedaten dekomprimiert.

[0024] In dem bevorzugten Ausführungsbeispiel schließt der Befehlsprozessor einen oder mehrere Eingangspuffer, eine Geometriedekomprimierungseinheit und eine Multiplexerlogik ein. Die Eingangspuffer sind an den System-Bus zum Empfangen von Geometrieeingangsdaten gekoppelt. Die Geometrieeingangsdaten schließen komprimierte Geometrieeingangsdaten und nicht-komprimierte Geometrieeingangsdaten ein. Die Geometriekomprimierungseinheit ist an den Ausgang der Eingangspuffer gekoppelt und empfängt und dekomprimiert die komprimierten Geometrieeingangsdaten. Der Multiplexer empfängt den nicht-komprimierten Ausgang von den Eingangspuffern und empfängt den dekomprimierten Ausgang von der Geometriedekomprimierungseinheit. Der Multiplexer schließt einen Ausgang ein, der ausgewählt entweder die nicht-komprimierten Geometriedaten oder den dekomprimierten Geometrieeingang bereitstellt.

[0025] Der Befehlsprozessor schließt einen Formatumwandler ein, der gekoppelt ist, um den Ausgang des Multiplexers zu empfangen, der zum Umwandeln empfangener Daten in ein oder mehrere Formattypen betrieben wird. Der Formatumwandler stellt einen Ausgang an einen Scheitelpunktakkumulationspuffer bereit, der eine Mehrzahl von Geometrieeingangsdaten speichert, die für einen oder mehr Scheitelpunkte erforderlich sind. Der Scheitelpunktakkumulationspuffer stellt einen Ausgang an Scheitelpunktpuffer bereit, die zum Konstruieren von geometrischen Primitiven verwendet werden. Der Befehlsprozessor schließt ebenso einen Sammlungspuffer ein, der an den Ausgang des Multiplexers zum Empfangen der nicht-geometrischen Daten gekoppelt ist. Jeder der Sammlungspuffer und der Scheitelpunktpuffer stellen Ausgänge an Ausgangspuffer bereit.

Fig. 1 – Computersystem

[0026] Bezugnehmend nun auf **Fig.** 1 ist ein Computersystem **80** gezeigt, das einen dreidimensionalen (3-D) Graphikbeschleuniger gemäß der vorliegenden Erfindung einschließt. Wie gezeigt, umfasst das Computersystem **80** eine Systemeinheit **82** und einen Videomonitor oder eine Anzeigevorrichtung **84**, die mit der Systemeinheit **82** gekoppelt ist. Die Anzeigevorrichtung **84** kann von jedem der verschiedenen Typen Anzeigemonitore oder Vorrichtungen sein. Verschiedene Eingangsvorrichtungen können mit dem Computersystem verbunden sein, einschließlich einer Tastatur **86** und/oder einer Maus **88**, oder einem anderen Eingang. Anwendungs-Software, dargestellt durch Floppy Disks **90**, können durch das Computersystem **80** ausgeführt werden, um zu bewirken, dass das System **80** graphische 3-D Objekte auf dem Videomonitor **84** darstellt. Wie weiter unten beschrieben, ermöglicht der 3-D Graphikbeschleuniger in dem Computersystem **80** die Anzeige von dreidimensionalen Graphikobjekten mit verbesserter Leistung.

Fig. 2 – Computersystemblockdiagramm

[0027] Bezugnehmend nun auf **Fig.** 2 ist ein vereinfachtes Blockdiagramm gezeigt, das das Computersystem von **Fig.** 1 darstellt. Elemente des Computersystems, die für ein Verstehen der vorliegenden Erfindung nicht notwendig sind, werden der Einfachheit halber nicht dargestellt. Wie gezeigt, schließt das Computersystem **80** eine zentrale Verarbeitungseinheit (CPU) **102** ein, die an einen Hochgeschwindigkeits-Bus oder einen System-Bus **104** gekoppelt ist. Ein Systemspeicher **106** ist vorzugsweise ebenso an den Hochgeschwindigkeits-Bus **104** gekoppelt.

[0028] Der Host-Prozessor **102** kann irgendeiner von verschiedenen Typen von Computerprozessoren, Multiprozessoren und CPUs sein. Der Systemspeicher **106** kann von irgendeinem von verschiedenen Typen von Speicheruntersystemen, einschließlich Zufallszugriffsspeichern (Random Access Memories) und Massenspeichervorrichtungen sein. Der System-Bus oder Host-Bus **104** kann von irgendeinem von verschiedenen Typen von Kommunikations- oder Host-Computer-Bussen für eine Kommunikation

zwischen Host-Prozessoren, CPUs und Speicheruntersystemen sein, ebenso wie spezialisierte Untersysteme. In dem bevorzugten Ausführungsbeispiel ist der Host-Bus **104** der UPA Bus, der ein 64-Bit-Bus ist, der bei 83 MHz betrieben wird.

[0029] Ein 3-D Graphikbeschleuniger **112** gemäß der vorliegenden Erfindung ist mit dem Hochgeschwindigkeitsspeicher-Bus **104** gekoppelt. Der 3-D Graphikbeschleuniger **112** kann mit dem Bus **104** gekoppelt sein durch zum Beispiel einen Kreuzschienenschaler (Cross Bar Switch) oder eine andere Busverbindungslogik. Es wird angenommen, dass verschiedene andere Peripherievorrichtungen oder andere Busse mit dem Hochgeschwindigkeitsspeicher-Bus **104** verbunden sein können, wie aus dem Stand der Technik bekannt ist. Wie gezeigt, ist der Videomonitor oder die Anzeigevorrichtung **84** mit dem 3-D Graphikbeschleuniger **112** verbunden.

[0030] Der Host-Prozessor **102** kann Information zu und von dem Graphikbeschleuniger **112** übertragen, gemäß einem programmierten Eingang/Ausgangs-(I/O)Protokoll über den Host-Bus **104**. In dem bevorzugten Ausführungsbeispiel werden Daten von dem Systemspeicher **106** an den Graphikbeschleuniger **112** übertragen durch Verwenden eines CPU Kopier(bcopy)befehls. In einer alternativen Ausführungsform greift der Graphikbeschleuniger **112** auf das Speichersubsystem **106** gemäß einem Direktspeicherzugriff(DMA)Protokoll zu.

[0031] Ein Graphikanwendungsprogramm, das auf dem Host-Prozessor **102** ausgeführt wird, erzeugt Geometriedaten-Arrays, die dreidimensionale Geometrieinformation enthalten, die ein Bild zum Anzeigen auf der Anzeigevorrichtung **84** definieren. Der Host-Prozessor **102** überträgt die Geometriedaten-Arrays an das Speicheruntersystem **106**. Danach wird der Host-Prozessor **102** betrieben, um Daten an den Graphikbeschleuniger **112** über den Host-Bus **104** zu übertragen, vorzugsweise durch Verwenden des bcopy-Befehls. Alternativ liest der Graphikbeschleuniger **112** in Geometriedaten-Arrays unter Verwendung von DMA Zugriffszyklen über den Host-Bus **104**. In einem anderen Ausführungsbeispiel ist der Graphikbeschleuniger **112** an den Systemspeicher **106** durch einen direkten Port gekoppelt, wie der Advanced Graphics Port (AGP) der durch die Intel Corporation veröffentlicht wurde.

[0032] Die dreidimensionale Geometrieinformation in den Geometriedaten-Arrays umfasst einen Strom von Eingangsscheitelpunktpaketen, die Scheitelpunktkoordinaten (Scheitelpunkte), Scheitelpunktpositionen und andere Information enthält, die Dreiecke, Vektoren, und Punkte in einem dreidimensionalen Raum definiert, der allgemein als Modellraum bezeichnet wird. Jedes Eingangsscheitelpunktpaket kann jede Kombination von dreidimensionaler Scheitelpunktinformation, einschließlich Scheitelpunktposition, Scheitelpunkt-Normal, Scheitelpunktfarbe, Facetten-Normal, Facetten-Farbe, Texturtafelkoordinaten, Pick-ID's, Header und andere Information enthalten.

Fig. 3 – Graphikbeschleuniger

[0033] Bezugnehmend nun auf **Fig.** 3 ist ein Blockdiagramm gezeigt, das den 3-D Graphikbeschleuniger **112** gemäß dem bevorzugten Ausführungsbeispiel der vorliegenden Erfindung dargestellt. **Fig.** 4 ist ein detaillierteres Diagramm, das einen Teil des 3-D Graphikbeschleunigers **112** darstellt. Wie gezeigt umfasst der 3-D Graphikbeschleuniger **112** prinzipiell einen Befehls-Preprozessor oder Befehlsblock **142**, einen Satz Gleitkommaprozessoren, oder Gleitkommablöcke **152A**–**152F**, einen Satz Zeichnungsprozessoren oder Zeichnungsblöcke **172A** und **172B**, einen Rahmenpuffer, der 3DRAM umfasst und einen Zufallszugriffsspeicher/Digital-Analogumwandler (RAMDAC) **196**.

[0034] Wie gezeigt, schließt der 3-D Graphikbeschleuniger **112** einen Befehlsblock **142** ein, der die Schnittstelle zu dem Speicherbus **104** darstellt. Der Befehlsblock **142** stellt eine Schnittstelle zwischen dem Graphikbeschleuniger **112** und dem Host-Bus **104** dar und steuert die Übertragung der Daten zwischen andern Blöcken oder Chips in dem Graphikbeschleuniger **112**. Der Befehlsblock **142** vorverarbeitet ebenso Dreiecke und Vektordaten und führt eine Geometriedatendekomprimieren durch, wie weiter unten beschrieben wird.

[0035] Der Befehlsblock **142** stellt eine Schnittstelle zwischen einer Mehrzahl von Gleitkommablöcken **152** dar. Der 3-D Graphikbeschleuniger **112** schließt vorzugsweise bis zu sechs Gleitkommablöcke an, bezeichnet als **152A**–**152F**, wie gezeigt ist. Die Gleitkommablöcke **152A**–**152F** empfangen Hochpegelzeichnungsbefehle und erzeugen Graphikprimitive, wie Dreiecke, Linien etc. zum Wiedergeben dreidimensionaler Objekte auf dem Bildschirm. Die Gleitkommablöcke **152A**–**152F** führen eine Transformation, ein Anhaften bzw. Abschneiden (Clipping), Beleuchtung und Set-Up-Operationen an empfangenen Geometriedaten durch. Jeder der Gleitkommablöcke **152A**–**152F** verbindet einen entsprechenden Speicher **153A**–**153F**. Die Speicher **153A**–**153F** sind vorzugsweise 32k × 36-Bit SRAM und werden für Mikrocode und Datenspeicherung verwendet.

[0036] Der Befehlsblock **142** stellt eine Schnittstelle zwischen den Gleitkommablöcken **152A**–**152F** durch eine Mehrzahl von Punkt-zu-Punkt-Bussen oder Direktdatenkanälen dar, bezeichnet als **154A**–**154F**. Somit schließt der Befehlsblock **142** einen Direktkanal zu jedem der entsprechenden Gleitkommablöcken **152A**–**152F** ein. Die Mehrzahl von Punkt-zu-Punkt-Bussen oder Direktdatenkanälen **154A**–**154F** sind jeweils vorzugsweise unidirektionale 8-Bit-Busse, die bei 100 MHz betrieben werden. Die Direktdatenkanäle **154A**–**154F** umfassen kollektiv 48 Bits, und die Direktdatenkanäle **154A**–**154F** werden kollektiv als der CF-Bus (Befehls/Gleitbus) bezeichnet. Datenübertragungen über den CF-Bus

umfassen 48 Bit Übertragungen, die über sechs Zyklen durchgeführt werden, wobei der Start der Übertragung unter den sechs separaten Bussen synchronisiert wird.

[0037] Wie weiter unten diskutiert wird, schließt der CF-Bus ebenso neun zusätzliche Bits ein, die mit drei der 8-Bit-Busse kombiniert werden, um einen 33-Bit-Bus zu bilden, der als der CD-Bus bezeichnet wird (**Fig.** 8–10). Wie in den **Fig.** 3 und 4 gezeigt, umfassen die Busse **154A**, **154B**, und **154C** kollektiv den CD-Bus und sind 11-Bit-Busse, wobei jeder einen 8-Bit-Bus plus drei zusätzliche Bits umfasst. Der CD-Bus ist ein direkter unidirektionaler Bus von dem Befehlsblock **142** zu den Zeichnungsblöcken **172A** und **172B**. Der CD-Bus "leiht" (borrows) Zyklen und Datenleitungen von dem CF-Bus **154**, um schnell 32-Bit-Daten von dem Befehlsblock **142** zu den Zeichnungsblöcken **172A** und **172B** zu senden, unter Verwenden eines Datenpfads in drei der Gleitkommablöcke **152A**–**152F** als eine Leitung.

[0038] Wie gezeigt, schließt der Befehlsblock **142** separate FIFO-Puffer **144A**–**F** ein, die mit jedem der entsprechenden Kanäle **154A**–**F** korrespondieren. Diese FIFO-Puffer **144** werden verwendet, um Daten zu speichern oder zu puffern, bevor die Daten auf dem entsprechenden Kanal **154A**–**F** zu den entsprechenden Gleitkommablöcken **152A**–**F** übermittelt werden. Wie gezeigt, schließt jeder Gleitkommablock **152A**–**F** einen entsprechenden Eingangs-FIFO-Puffer **155A**–**155F** ein, der zum Empfangen von Daten von dem entsprechenden Kanal **154A**–**F** gekoppelt ist.

[0039] Jeder der Gleitkommablöcke **152A**–**F** ist mit jedem der zwei Zeichnungsblöcke **172A** und **172B** verbunden. Der 3-D Graphikbeschleuniger **112** schließt vorzugsweise zwei Zeichnungsblöcke **172A** und **172B** ein, obwohl eine größere oder kleinere Anzahl verwendet werden kann. Die Zeichnungs- oder Wiedergabeblöcke **172A** und **172B** führen eine Bildschirmraumwiedergabe der verschiedenen Graphikprimitive durch und werden betrieben, um die vervollständigten Pixel in den 3DRAM-Array aneinander zu reihen oder zu füllen. Die Zeichnungs- oder Wiedergabeblöcke **172A** und **172B** fungieren ebenso als DRAM-Steuerchips für den Rahmenpuffer. Die Zeichnungsprozessoren **172A** und **172B** geben fortlaufend ein Bild in den Rahmenpuffer **100** gemäß eines Zeichnungspakets, das von einem der Gleitkommaprozessoren **152A**–**152F** empfangen wird, oder gemäß einem direkten Port-Paket, das von dem Befehls-Preprozessor **142** empfangen wird.

[0040] Jeder der Gleitkommablöcke **152A**–**F** ist mit den zwei Zeichnungsblöcken **172A** und **172B** durch entsprechende Punk-zu-Punkt-Busse oder direkte Datenkanäle **162A**–**162F** und **164A**– **164F** verbunden. Wie gezeigt, schließt jeder der Gleitkommablöcke **152A**–**F** einen entsprechenden ersten direkten Kanal **162A**–**F** zu dem Zeichnungsblock **172A** ein, und jeder der Gleitkommablöcke **152A**–**F** schließt einen entsprechenden zweiten Kanal **164A**–**F** zu dem anderen Zeichnungsblock **172B** ein. Somit schließt jeder der Gleitkommablöcke **152A**–**F** einen direkten Kanal zu jedem der Zeichnungsblöcke **172A** und **172B** ein. Die Mehrzahl von Punkt-zu-Punkt-Bussen oder Direktdatenkanälen **162A**–**162F** und **164A**–**164F** sind jeweils unidirektionale 11-Bit-Busse, die bei 100 MHz betrieben werden.

[0041] Somit schließt der Graphikbeschleuniger **112** zwei Sätze von sechs 11-Bit-Bussen ein, die unabhängige Pfade von jedem der Gleitkommablöcke **152A**–**F** an jedem Zeichnungsprozessor **172A** und **1728** bereitstellen. Die direkten Datenkanäle **154A**–**154F** umfassen kollektiv 48 Bits und der direkte Datenkanal **162A**–**F** und **164A**–**F** werden kollektiv als die FD-Busse (Gleit/Zeichnungsbus) bezeichnet.

[0042] Jeder der Gleitkommablöcke **152A**–**F** wird vorzugsweise betrieben, um die gleichen Daten an die zwei Zeichnungsblöcken **172A** und **172B** zu senden. Mit anderen Worten, sind die gleichen Daten immer auf beiden Sätzen von Datenleitungen, die von jedem der Gleitkommablöcke **152A**–**F** kommen. Somit überträgt, wenn der Gleitkommablock **152A** Daten überträgt, der Gleitkommablock **152A** die gleichen Daten über beide Kanäle **162A** und **164A** zu den Zeichnungsprozessoren **172A** und **172B**.

[0043] Daten werden auf dem FD-Bus gleichzeitig mit 32 Bit übertragen unter Verwenden von drei Zyklen, ohne Synchronisierung zwischen den sechs separaten Bussen. Das 33ste Bit jeder Übertragung ist ein Steuerungs-Bit, das auf 1 gesetzt wird, um anzuzeigen, dass das letzte Wort des Primitivs übertragen wurde. In einigen Fällen werden die Ausgänge von drei der Gleitkommablöcke **152A**–**152C** für ein 33 Bit (32 Daten, 1 Steuerung) CD-Buszyklus "geliehen", wie oben beschrieben.

[0044] Wie in **Fig.** 4 gezeigt, schließt jeder der Gleitkommablöcke **152A**–**F** Ausgangs-FIFO-Puffer **158A**–**F** ein, die mit jedem der entsprechenden Kanäle **162A**–**F** und **164A**–**F** gekoppelt sind. Entsprechend schließt jeder der entsprechenden Zeichnungsblöcke **172A** und **172B** Eingangs-FIFO-Puffer **182** und **184** entsprechend ein. Wie in **Fig.** 9 gezeigt, schließt der Zeichnungsblock **172A** Eingangs-FIFO-Puffer **182A**–**F** zum Koppeln der entsprechenden Kanäle **162A**–**F** ein. Entsprechend schließt der Zeichnungsblock **172B** entsprechende FIFO-Puffer **184A**–**F** (nicht gezeigt) ein, zum Koppeln der entsprechenden Kanäle **164A**–**F**.

[0045] Der Graphikbeschleuniger **112** schließt zwei unidirektionale Busse ein, die als der CD-Bus (**Fig.** 10) und der DC-Bus **173** bezeichnet werden, für Datenübertragungen zwischen dem Befehlsprozessor **142** und den Zeichnungsprozessoren **172A** und **172B**. Der CD/Bus ist ein unidirektionaler Bus zum Übertragen von dem Befehlsprozessor **142** an die Zeichnungsprozessoren **172A** und **172B**. Wie oben diskutiert, wird der CD-Bus teilweise von den drei der entsprechenden Gleitkommablöcke **152A**–**152C** umfasst. Der CD-Bus verwendet oder "leiht" Zyklen und Adern von dem CF-Bus, den drei Gleitkommablö-

cken **152A**–**152C** und dem FD-Bus. Der DC-Bus **173** ist ein unidirektionaler Bus zum Übertragen von den Zeichnungsprozessoren **172A** und **172B** zu dem Befehlsprozessor **142**, wie in den **Fig.** 3 und 4 gezeigt ist. Der CD-Bus und der DC-Bus sind deutlicher in **Fig.** 10 dargestellt.

[0046] Jeder der entsprechenden Zeichnungsblöcke **172A** und **172B** ist an einen Rahmenpuffer gekoppelt, wobei der Rahmenpuffer vier Bänke 3DRAM Speicher **192A**–**B** und **194A**–**B** umfasst. Der Zeichnungsblock **172A** ist mit den zwei 3DRAM Bänken **192A** und **192B** gekoppelt, bzw. der Zeichnungsblock **172B** ist mit den zwei 3DRAM Bänken **194A** und **194B** gekoppelt. Jede der Bänke umfasst 3 3DRAM Chips, wie gezeigt. Die 3DRAM Speicher oder Bänke **192A**–**B** und **194A**–**B** bilden kollektiv den Rahmenspeicher, der 1280 × 1024 auf 96 Bit tief ist. Der Rahmenpuffer speichert Pixel korrespondierend zu 3-D-Objekten, die durch die Zeichnungsblöcke **172A** und **172B** wiedergegeben werden.

[0047] Jeder der 3DRAM Speicher **192A**–**B** und **194A**–**B** ist an einen RAMDAC (Zufallszugriffspeicher Digital/Analogumwandler) **196** gekoppelt. Der RAMDAC **196** umfasst einen programmierbaren Video-Timing-Generator und einen programmierbaren Pixeltaktsynthesizer, entsprechend mit Kreuzschienenfunktionen, ebenso wie traditionelle Farb-Look-Up-Tafeln und Dreifachvideo DAC-Schaltkreisen. Der RAMDAC wiederum ist an den Videomonitor **84** gekoppelt.

[0048] Der Graphikbeschleuniger **112** schließt weiterhin einen bidirektionalen Bus **195** ein, der als der CM-Bus bezeichnet wird, zum Verbinden des Befehlsblocks **142** und des RAMDAC **196**. Wie gezeigt, ist ein Boot PROM **197** und ein Audioblock **198** an den CM-Bus **195** gekoppelt. Der CM-Bus **195** wird vorzugsweise bei 25 MHz betrieben.

[0049] Der Befehlsblock ist vorzugsweise als ein Einzelchip implementiert. Alle der "Gleitkommablöcke" **152** sind vorzugsweise als separate Chips implementiert. In dem bevorzugten Ausführungsbeispiel können bis zu sechs Gleitkommablöcken oder Chips **152A**–**F** eingeschlossen sein.

[0050] Jeder der Zeichnungsblöcke oder Prozessoren **172A** und **172B** umfasst vorzugsweise separate Chips.

Direkte Datenkanäle

[0051] Wie oben diskutiert, schließt die 3-D Graphikbeschleunigerarchitektur der vorliegenden Erfindung eine Mehrzahl von direkten Kanälen zwischen dem Befehlsblock **142** und jedem der Gleitkommablöcke **152A**–**F** ein, ebenso wie eine Mehrzahl von direkten Kanälen zwischen jedem der Gleitkommablöcke **152A**–**F** und den entsprechenden Zeichnungsblöcken **172A** und **172B**.

[0052] Wie im Hintergrundabschnitt diskutiert, haben die Architekturen aus dem Stand der Technik einen gemeinsamen Bus eingeschlossen, der diese Elemente verbindet. Jedoch wird der Befehlsblock **142** allgemein betrieben, um separate Daten zu jedem der Gleitkommablöcke **152A**–**152F** zu senden, im allgemeinen in einer Round-Robin-Art. Mit anderen Worten arbeitet die Befehlslogik **142** im Allgemeinen, um eine Burst-Übertragung von Daten an nur einem der Gleitkommablöcke **152** bereitzustellen, wie Gleitkommablock **152A**, und dann eine Burst-Datenübertragung zu einem anderen der Gleitkommablöcke **152** bereitzustellen, wie dem **152B**, und so fort. Diese Burst-Natur der Datenübertragung tritt ebenso zwischen jedem der Gleitkommablöcke **152A**–**F** und den zwei Zeichnungsblöcken **172A** und **172B** auf. Mit anderen Worten stellt jeder der entsprechenden Gleitkommablöcke **152A**–**152F** im Allgemeinen entsprechende individuelle Burst-Datenübertragungen an jeden der Zeichnungsblöcke **172A** und **172B** bereit.

[0053] Die Mehrzahl von direkten Datenkanälen oder Punkt-zu-Punkt-Bussen führt die Burst-Datenübertragungen zwischen dem Befehlsblock **142** und jedem der Gleitkommablöcke **152A**–**152F** durch. Die Mehrzahl von direkten Datenkanälen oder Punkt-zu-Punkt-Bussen führt ebenso die Burst-Datenübertragung zwischen jedem der Gleitkommablöcke **152A**–**152F** und den Zeichnungsprozessoren **172A** und **172B** durch. Die Verwendung von direkten Datenpfaden anstelle eines geteilten Busses ermöglicht die Verwendung einer Anzahl von kleineren Datenpfaden, z. B. 8-Bit Datenpfaden, während eine ähnliche Bandbreite zu Auslegungen aus dem Stand der Technik bereitgestellt wird. Die Verwendung dieser kleineren direkten Datenpfade stellt ebenso bessere elektrische Charakteristiken für die Graphikarchitektur bereit. Zuerst sind die Direktdatenkanalausgangspins auf dem Befehlschip nur erforderlich, um eine einzelne Vorrichtung anzutreiben, im Gegensatz zum Antreiben einer Mehrzahl von Vorrichtungen in einer geteilten Busarchitektur. Ebenso weist jeder der Gleitkommaprozessoren **152A** bis **152F** eine reduzierte Anzahl von Pins auf, da jeder nur mit einem 8-Bit Bus verbunden ist. Weiterhin stellen die direkten Datenpfade eine verbesserte Konnexität zwischen einer Mehrzahl von Boards bereit. Die verbesserten elektrischen Charakteristiken ermöglichen ebenso die Verwendung einer höheren Taktgeschwindigkeit, somit eine erhöhte Übertragungsbandbreite bereitstellend.

[0054] In einigen Fällen ist der Befehlsblock **142** erforderlich, um die gleichen Daten zu jedem der Gleitkommablöcke **152A**–**152F** zu senden. Wenn zum Beispiel der Befehlsblock **142** erforderlich ist zum Senden von Matrixdaten, gefolgt durch eine Mehrzahl von Dreiecksdaten und jedes der nachfolgenden Dreiecke die Verwendung der Matrixdaten erfordert, dann ist es erforderlich, die Matrixdaten zuerst zu jedem der Gleitkommablöcke **152A**–**152F** zu übertragen, bevor irgendeins der nachfolgenden Dreiecke zu irgendeinem der entsprechenden Gleitkommaeinheiten gesendet wird. Mit anderen Worten kann es ei-

nem Gleitkommablock **152** nicht erlaubt sein, eines dieser nachfolgenden Dreiecke zu empfangen, bis die entsprechende Matrix schon empfangen wurde, die erforderlich ist, um die Dreiecke zu verarbeiten.

[0055] Wenn der Befehlsblock **142** erforderlich ist, um die gleichen Daten zu jedem der Gleitkommablöcke **152A**–**152F** zu senden, dann ist der Befehlsblock **142** erforderlich, um zu warten, dass alle der FIFOs **144A**–**144F** leer sind und/oder bis ein ausreichender Raum der entsprechenden FIFOs für diese gemeinsame Übertragung erscheint. Somit ist, wenn der Befehlsblock **142** erforderlich ist, um die gleichen Daten zu senden, d. h. parallele Sendedaten, zu jedem der Gleitkommablöcke **152A**–**152F**, der Befehlsblock **142** erforderlich, um auf jeden der FIFOs **144A**–**144F** zu warten, um einen ausreichenden Raum in ihren FIFOs aufzuweisen, und es ist erforderlich, die gleichen Daten zu jedem der FIFOs **144A**– **144F** zu übertragen. Es wird bemerkt, dass diese Sendeübertragung bei einer verminderten Übertragungsrate eines Systems aus dem Stand der Technik auftreten kann, das einen gemeinsamen Bus verwendet. Diese gemeinsamen Übertragungen sind im Allgemeinen selten und bewirken keine nachteilige Systemleistung.

[0056] Die Gleitkommablöcke **152A**–**152F** können nicht notwendigerweise Dreiecke in der exakten Reihenfolge ausgeben, in der diese Dreiecke durch den Befehlsblock **142** empfangen werden. Es wird bemerkt, dass es im Allgemeinen nicht notwendig ist, die exakte serielle Reihenfolge der empfangenen Dreiecke aufrecht zu erhalten. In dem bevorzugten Ausführungsbeispiel schließt die 3-D Graphikbeschleunigerarchitektur einen ersten Modus ein, bei dem eine exakte serielle Reihenfolge der empfangenen Dreiecke nicht aufrecht erhalten wird. Das System beinhaltet ebenso einen zweiten Modus, bei dem die Gleitkommablöcke **152A**–**152F** zum Ausgeben wiedergegebener Dreiecke in der exakten Reihenfolge konfiguriert sind, in der diese Dreiecke durch den Befehlsblock **12** empfangen werden.

[0057] Daher stellt das System und das Verfahren der vorliegenden Erfindung eine Mehrzahl von direkten Kanälen oder Punkt-zu-Punkt-Bussen zwischen dem Befehlsblock **142** und jedem der Gleitkommablöcke **152A**–**F** bereit. Das System und Verfahren der vorliegenden Erfindung stellt ebenfalls eine Mehrzahl von direkten Kanälen oder Punkt-zu-Punkt-Bussen zwischen den Gleitkommablöcke **152A**–**152F** und jedem der Zeichnungsblöcke **172A** und **172B** bereit. Mit anderen Worten stellt die vorliegende Erfindung eine Mehrzahl von gewidmeten schmalen Bussen, vorzugsweise 8-Bit Datenbussen bereit, die den Befehlsblock **142** mit jedem der Gleitkommablöcke **152A**–**152F** verbindet, ebenso wie eine Mehrzahl von schmalen Bussen, vorzugsweise 8-Bit Bussen, die jeden der Gleitkommablöcke **152A**–**152F** mit jedem der Zeichnungsblöcke **172A** und **172B** verbindet. Somit schließt die vorliegende Erfindung nicht eine gemeinsame Bus- oder geteilte Busarchitektur für eine Konnexität ein, sondern beinhaltet eher direkte Zwischenverbindungen zwischen jedem der logischen Elemente. Das stellt verbesserte elektrische Charakteristiken bereit und reduziert die Pinerforderlichkeiten und erleichtert ebenso höhere Taktgeschwindigkeit und stellt somit eine verbesserte Leistung gegenüber den Auslegungen aus dem Stand der Technik dar.

Fig. 5 – Befehlsblock

[0058] Wie oben beschrieben, ist der Befehls-Preprozessor oder Befehlsblock **142** zur Kommunikation über den Host-Bus **104** gekoppelt. Der Befehls-Preprozessor **142** empfängt Geometriedaten-Arrays, die von dem Speicheruntersystem **106** über den Host-bus **82** durch den Host-Prozessor **102** übertragen werden. In dem bevorzugten Ausführungsbeispiel empfängt der Befehls-Preprozessor **142** Daten, die von dem Speicheruntersystem **106** übertragen werden, einschließlich sowohl komprimierter als auch nicht-komprimierter Geometriedaten. Wenn der Befehls-Preprozessor **142** komprimierte Geometriedaten empfängt, wird der Befehls-Preprozessor **142** zum Dekomprimieren der Geometriedaten betrieben, um dekomprimierte Geometriedaten herzustellen.

[0059] Der Befehls-Preprozessor **142** implementiert vorzugsweise zwei Daten-Pipelines, dieses sind eine 3-D Geometrie-Pipeline und eine Direkt-Port-Pipeline. In der Direkt-Port-Pipeline empfängt der Befehls-Preprozessor **142** direkte Port-Daten über den Host-Bus **104** und überträgt die direkten Portdaten über den Befehl-zum-Zeichnen (CD) Bus zu dem Zeichnungsprozessor **172A**–**172B**. Wie oben erwähnt, verwendet oder "leiht" der CD-Bus Teile von anderen Bussen, um einen direkten Datenpfad von dem Befehlsprozessor **142** zu den Zeichnungsprozessoren **172A**– **172B** zu bilden. die direkten Port-Daten werden optional durch den Befehls-Preprozessor **142** verarbeitet, um X11 Funktionen durchzuführen, wie Buchstaben schreiben, Bildschirmrollen und Blockbewegungen in Übereinstimmung mit den Zeichnungsprozessoren **172A**–**172B**. Die direkten Port-Daten können ebenso Registerschreiben an den Zeichnungsprozessoren **172A**–**172B** einschließen und individuelles Pixelschreiben an die Rahmenpuffer 3DRAM **192** und **194**.

[0060] In der 3-D Geometrie-Pipeline greift der Befehls-Preprozessor **142** auf einen Strom von eingegebenen Scheitelpunktpaketen von den Geometriedaten-Arrays zu.

[0061] Wenn der Befehls-Preprozessor **142** einen Strom von Eingangsscheitelpunktpaketen von den Geometriedaten-Arrays empfängt, wird der Befehls-Preprozessor **142** zum erneuten Bestellen der Information betrieben, die in den eingegebenen Scheitelpunktpaketen enthalten ist und optional Information in den eingegebenen Scheitelpunktpaketen löschen. Der Befehls-Preprozessor **142** konvertiert vorzugsweise die empfangenen Daten in ein Stan-

dardformat. Der Befehls-Preprozessor **142** wandelt die Information in jedem der eingegebenen Scheitelpunktpakete von einem unterschiedlichen Zahlenformat in das 32 Bit IEEE Gleitkommazahlformat. Der Befehlsprozessor **142** wandelt 8-Bit Festkommazahlen, 16-Bit Festkommazahlen und 32-Bit oder 64-Bit IEEE Gleitkommazahlen um. Für Normal- und Farbwerte kann der Befehls-Preprozessor **142** die Daten in einen Festkommawert umwandeln.

[0062] Der Befehls-Preprozessor **142** kann ebenso betrieben werden, um eingegebene Scheitelpunktinformation zu akkumulieren, bis ein vollständiges Primitiv empfangen wurde. Der Befehls-Preprozessor **142** überträgt dann Ausgangsgeometriepakete oder Primitivdaten über den Befehl-zum-Gleitkomma (CF)-Bus an einer der Gleitkommaprozessoren **152A**–**152F**. Die ausgegebenen Geometriepakete umfassen die erneut formatierten Scheitelpunktpakete mit optionalen Modifikationen und Datenersetzungen.

[0063] Bezugnehmend nun auf **Fig.** 5 wird ein Blockdiagramm gezeigt, dass den Befehlsprozessor oder Befehlsblock **142** darstellt. Wie gezeigt, schließt der Befehlsblock **142** Eingangspuffer **302** und Ausgangspuffer **304** als Schnittstelle zu dem Host-Bus **104** ein. Die Eingangspuffer **302** sind mit an einen globalen Datenausgeber **306** und eine Adressdecodierungslogik **308** gekoppelt. Der globale Datenausgeber **306** ist mit den Ausgangspuffern **304** und dem CM-Bus verbunden und führt eine Datenübertragung aus. Die Adressdecodierungslogik **308** empfängt einen Eingang von dem DC-Bus, wie gezeigt. Die Adressdecodierungslogik **308** ist ebenso gekoppelt, um einen Ausgang an einen Eingangs-FIFO-Puffer **312** bereitzustellen.

[0064] Im Allgemeinen, weist der Rahmenpuffer eine Mehrzahl von Kartographien auf, einschließlich eines 8-Bit Modus für rote, grüne und blaue Ebenen, einen 32-Bit Modus für einen individuellen Pixelzugriff und einen 64-Bit Modus für einen Zugriff auf die Pixelfarbe zusammen mit den Z-Puffer-Werten. Der Boot-Prom **197**, Audiochip **198** und RAMDAC **196** weisen ebenso einen Adressraum innerhalb des Rahmenpuffers auf. Der Rahmenpuffer schließt ebenso einen Registeradressraum für unter anderem einen Befehlsblock und Zeichnungsprozessorregister ein. Die Adressdecodierungslogik **308** wird zum Erstellen von Tags für den Eingangs-FIFO **312** betrieben, die spezifizieren, welche Logikeinheit Daten empfangen soll und wie die Daten konvertiert werden. Der Eingangs-FIFO-Puffer **312** hält 128 64-Bit Worte, plus einen 12-Bit Tag, der das Ziel von Daten spezifiziert und wie die Daten verarbeitet werden sollen.

[0065] Der Eingangs-FIFO **312** ist über einen 64-Bit Bus mit einem Multiplexer **314** gekoppelt. Der Eingangs-FIFO **312** stellt ebenso einen Ausgang an eine Geometriedekomprimierungseinheit **316** bereit. Wie oben diskutiert, empfängt der Befehlsblock **142** sowohl komprimierte als auch nicht-komprimierte Geometriedaten. Die Dekomprimierungseinheit **316** empfängt die komprimierten Geometriedaten und wird zum Dekomprimieren dieser komprimierten Geometriedaten betrieben, um dekomprimierte Geometriedaten herzustellen. Die Dekomprimierungseinheit **316** empfängt einen Strom von 32-Bit Worten und stellt dekomprimiert Geometriedaten oder Primitivdaten her. Dann werden dekomprimierte Geometriedaten, die von der Dekomprimierungseinheit **316** ausgegeben werden, an einen Eingang des Multiplexers **314** bereitgestellt. Der Ausgang des Multiplexer **314** wird an einen Formatumwandler **322**, einen Sammelpuffer **324** und eine Registerlogik **326** bereitgestellt. Im Allgemeinen werden die dekomprimierten Geometriedaten, die von der Dekomprimierungseinheit ausgegeben werden an entweder den Formatumwandler **322** oder den Sammelpuffer **324** bereitgestellt.

[0066] Im Wesentlichen kann die Geometriedekomprimierungseinheit **316** betrachtet werden als eine Umleitung auf den Datenpfad zwischen dem Eingangs-FIFO **312** und der nächsten Stufe der Verarbeitung, die entweder der Formatumwandler **322** oder der Sammelpuffer **324** ist. Für Daten, die durch den Befehlsprozessor **142** empfangen werden, die keine komprimierten Geometriedaten sind, d. h. nicht komprimierte Daten, werden diese Daten von dem Eingangs-FIFO **312** direkt durch den Multiplexer **314** zu entweder dem Formatumwandler **322**, dem Sammelpuffer **324** oder der Registerlogik **326** bereitgestellt. Wenn der Befehlsprozessor **142** komprimierte Geometriedaten empfängt, müssen diese Daten zuerst von dem Eingangs-FIFO **312** an die Geometriedekomprimierungseinheit **316** bereitgestellt werden, um dekomprimiert zu werden, bevor sie an eine andere Logik bereitgestellt werden.

[0067] Somit schließt der Befehlsblock **142** einen ersten Datenpfad ein der mit den Eingangspuffern **302** oder einem Eingangs-FIFO **312** zum Übertragen der nicht-komprimierten Geometriedaten gekoppelt ist, direkt durch den Multiplexer **314** an entweder den Formatumwandler **322** oder den Sammelpuffer **324**. Der Befehlsblock **142** schließt ebenfalls einen zweiten Datenpfad ein, der mit den Eingangspuffern **302** oder einem Eingangs-FIFO **312** zum Empfangen komprimierter Geometriedaten gekoppelt ist. Der zweite Datenpfad schließt eine Geometriedekomprimierungseinheit ein, die an einen Ausgang des Eingangs-FIFO **312** zum Empfangen und Dekomprimieren der komprimierten Geometriedaten gekoppelt ist, um dekomprimierte Geometriedaten herstellen.

[0068] Der Formatumwandler **322** empfängt ganze und/oder Gleitkommadaten, und gibt entweder Gleitkomma- oder Festkommadaten aus. Der Formatumwandler **322** stellt dem Befehlsprozessor **142** die Flexibilität bereit, um eine Mehrzahl von verschiedenen Datentypen zu empfangen, während jede der Gleitblockeinheiten **152A**–**152F** mit nur einem einzelnen Datentyp für ein besonderes Wort bereitgestellt wird.

[0069] Der Formatumwandler **322** stellt einen 48-Bit

Ausgang an einen Scheitelpunktakkumulationspuffer **322** bereit. Die Scheitelpunktakkumulation **332** wiederum stellt einen Ausgang an die Scheitelpunktpuffer **334** bereit. Der Scheitelpunktakkumulationspuffer **322** und die Scheitelpunktpuffer **334** stellen Ausgänge an den Sammelpuffer **324** bereit, der wiederum einen Ausgang zurück an die Ausgangspuffer **304** bereitstellt.

[0070] Der Scheitelpunktakkumulationspuffer **332** wird verwendet, um Daten zu speichern oder akkumulieren, die für ein Primitiv erforderlich sind, das von dem Formatumwandler **322** empfangen wird. Der Scheitelpunktakkumulationspuffer **332** umfasst tatsächlich zwei Sätze Register, d. h. ist doppeltgepuffert. Der erste Satz von Registern wird zum Erstellen eines Scheitelpunktes verwendet, und der zweite Satz von Registern wird zum Kopieren der Daten in einen der Scheitelpunktpuffer **334** verwendet. Wie weiter unten diskutiert erlauben diese zwei Sätze Register einen effizienteren Betrieb. Datenworte werden eins nach dem anderen in den ersten oder oberen Puffer des Scheitelpunktakkumulationspuffers **332** geschrieben und diese Werte verbleiben unverändert, bis ein neuer Wert das entsprechende Wort überschreibt. Daten werden von dem ersten Satz Register an den zweiten Satz Register in einem Zyklus übertragen, wenn eine zutreffende Bedingung auftritt.

[0071] Die Scheitelpunktpuffer **334** werden zum Konstruieren oder "Aufbauen" ("Building-Up") von geometrischen Primitiven verwendet, wie Linien, Dreiecke, etc. Linien und Dreiecke erfordern zwei bzw. drei Scheitelpunkte, um ein Primitiv zu vervollständigen. Gemäß einem Ausführungsbeispiel der Erfindung können neue Primitive durch Ersetzen eines Scheitelpunktes eines bestehenden Primitivs erstellt werden, wenn das erstellte Primitiv einen oder mehr Scheitelpunkte mit dem zuvor erstellten Primitiv teilt. Mit anderen Worten erinnern oder aufrechterhalten die Scheitelpunktpuffer **334** Scheitelpunktwerte und verwenden diese Scheitelpunktwerte intelligent wieder, wenn ein Primitiv oder Dreieck einen oder mehrere Scheitelpunkte oder andere Information mit einem benachbarten Primitiv oder Dreieck teilt. Das reduziert die Verarbeitungserfordernisse und macht den Betrieb des Open GL Formats effizienter. In dem bevorzugten Ausführungsbeispiel können die Scheitelpunktpuffer **334** bis zu sieben Scheitelpunkte aufrecht erhalten. Das garantiert einen maximalen Durchsatz für ein Primitiv im schlechtesten Fall, d. h. unabhängige Dreiecke. Die Scheitelwertpuffer **334** arbeiten ebenso bei einer optimalen Geschwindigkeit für Punkte, Linien und Dreiecke und sind im wesentlichen optimal für Vierlingsprimitive.

[0072] Jeder der Scheitelpunktakkumulationspuffer **332** und der Scheitelpunktpuffer **334** sind mit einem Sammelpuffer **324** gekoppelt. Der Sammelpuffer **324** stellt entsprechende Ausgänge an die Ausgangspuffer **304** bereit, wie gezeigt. Die Scheitelwertpuffer **334** sind gekoppelt, um Ausgänge an CF-Busausgangs-FIFOs **144** bereitzustellen. Der Sammelpuffer **324** ist ebenso gekoppelt, um Ausgänge an die CF-Busausgangs-FIFOs **144** bereitzustellen. Der Sammelpuffer **324** wird zum Senden aller nicht-geometrischen Daten an die Gleitkommablöcke **152A**–**152F** verwendet. Der Sammelpuffer **324** kann bis zu 32 32-Bit Worten halten. Es wird bemerkt, dass der Betrieb des Kopierens von Daten in die CF-Busausgangs-FIFOs **144** überlappend mit dem Betrieb des Kopierens neuer Daten in den Sammelpuffer **324** für einen optimalen Durchsatz sein können.

[0073] Wie oben erwähnt, schließt der Befehlsblock **142** eine Mehrzahl von Registern **326** ein, die mit dem Ausgang des Multiplexers **314** gekoppelt sind. Die Register **326** stellen ebenso einen Ausgang an die UPA Ausgangspuffer **304** bereit. Ein Registerblock **326** umfasst 16 Steuer- und Statusregister, die das Format und einen Fluss von Daten steuern, die zu den entsprechenden Gleitkommablöcken **152A**–**152F** gesendet werden.

[0074] Jeder der Scheitelpunktpuffer **334** und der Sammelpuffer **324** stellen einen 48-Bit-Ausgang an die CF-Busausgangs-FIFOs **144** bereit. Die CF-Busausgangs-FIFOs **144** ermöglichen dem Befehlsblock **142** schnell ein Primitiv von den Scheitelpunktpuffern **334** in den Ausgangs-FIFO **144** zu kopieren, während der letzte der vorangegangenen Primitive noch über den CF-Bus übertragen wird. Das ermöglicht dem Graphikbeschleuniger **112**, einen stetigen Fluss von Daten über jeden der Punkt-zu-Punkt-Busse aufrecht zu erhalten. In dem bevorzugten Ausführungsbeispiel haben die CF-Busausgangs-FIFOs **144** ausreichend Raum, um ein komplettes Primitiv zu halten, ebenso wie zusätzliche Speicher zum Glätten des Datenflusses. Die CF-Busausgangs-FIFOs **144** stellen entsprechende 8-Bit Ausgänge an einen Busschnittstellenblock **336** bereit. Die Busschnittstelle **336** ist die letzte Stufe des Befehlsprozessors **142** und ist an den CF-Bus, wie gezeigt, gekoppelt. Zusätzlich stellt die CF/CD-Busschnittstelle **336** "Direct Port" Zugriffe an den CDC-Bus bereit, die multiplex auf dem CF-Bus sind, wie oben erwähnt.

[0075] Der Befehlsblock **142** schließt ebenso eine Round-Robin-Schlichtungslogik **334** ein. Diese Round-Robin-Schlichtungslogik **334** umfasst einen Schaltkreis zum Bestimmen, welcher der entsprechenden Gleitkommaprozessoren **152A**–**152F** das nächste Primitiv empfängt. Wie oben diskutiert, umfasst der Graphikbeschleuniger **112** der vorliegenden Erfindung separate Punkt-zu-Punkt-Busse sowohl in die entsprechenden Gleitkommaprozessoren **152A**–**152F** hinein, als auch aus ihnen heraus. Somit ist die Round-Robin-Schlichtungslogik **334** eingeschlossen, um die Primitive gleich zwischen den Chips zu verteilen und somit einen gleichen Datenfluss über alle der Punkt-zu-Punkt-Busse simultan aufrecht zu erhalten. In dem bevorzugten Ausführungsbeispiel verwendet die Round-robin-Schlichtungslogik **2334** ein "nächster erhältlicher Round Robin"-Schlichtungsschema, das einen Unterbus über-

springt, der gestaut, d. h. voll ist.

[0076] Für Information über eine Ausführungsform aus dem Stand der Technik eines Befehlsprozessors sei bitte auf U.S. Patent 5,408,605 verwiesen, mit dem Titel "Command Preprocessor for a High Performance Three Dimensional Graphics Accelerator", das hiermit durch Bezugnahme in seinem Umfang in die Offenbarung aufgenommen wird.

Fig. 6 – Gleitkommaprozessorblockdiagramm

[0077] Bezugnehmend nun auf **Fig.** 6, ist ein Blockdiagramm gezeigt, das einen der Gleitkommablöcke oder Prozessoren **152** gemäß dem bevorzugten Ausführungsform der vorliegenden Erfindung darstellt. Jeder der entsprechenden Gleitkommaprozessoren **152A**–**152F** ist identisch und somit wird nur einer hier der Einfachheit halber beschrieben. Wie gezeigt, schließt jeder der Gleitkommablöcke **152** drei Hauptfunktionseinheiten oder Kernprozessoren ein. Diese sind F-Kern **352**, L-Kern **354** und S-Kern **356**. Der F-Kernblock **352** ist zum Empfangen von Daten von dem CF-Bus gekoppelt, die von dem Befehlsblock **142** übertragen werden. Der F-Kernblock **352** stellt Ausgangsdaten sowohl an den L-Kernblock **354**, als auch an den S-Kernblock **356** bereit. Der L-Kernblock **354** stellt ebenso Daten an den S-Kernblock **356** bereit. Der S-Kernblock **356** stellt Ausgangsdaten an den FD-Bus bereit.

[0078] Der F-Kernblock **352** führt alle gleitkommaintensiven Operationen durch, einschließlich Geometrietransformation, Clip-Testen, Flächenbestimmung, Perspektiventeilung und Bildschirmraumumwandlung. Der F-Kernblock **352** führt, wenn erforderlich, auch Clipping durch. In dem bevorzugten Ausführungsbeispiel ist der F-Kernblock **352** voll programmierbar unter Verwenden eines 36-Bit Mikroanweisungsworts, das in einem 32k Wort SRAM gespeichert ist.

[0079] Der L-Kernblock **354** führt im Wesentlichen alle Beleuchtungsberechnungen unter Verwenden eines On-Chip RAM-basierten Mikrocodes durch. Beleuchtungsberechnungen sind auf das Farbe-zu-Scheitelpunktformat abgestimmt. Der L-Kernblock **354** beinhaltet ebenso eine effiziente Dreifachwortauslegung für effizientere Beleuchtungsberechnungen. Diese Dreifachwortauslegung wird mit einem 48-Bit Datenwort betrieben, das 16-Bit Festkommawerte umfasst. Somit kann eine Anweisung die gleiche Funktion auf allen drei Farbkomponenten (RGB) ausführen, die alle drei Komponenten eines Normals (N_x, N_y, und N_z) in einem Zyklus sind. Die mathematischen Einheiten umfassen in dem L-Kernblock **354** automatische Klammerwerte auf die erlaubten Bereiche, und erlauben somit keine zusätzlichen Zweige.

[0080] Der S-Kernblock führt Set-Up-Berechnungen für alle Primitive durch. Diese Set-Up-Berechnungen involvieren eine Berechnung der Abstände in mehrfachen Dimensionen von einem Scheitelpunkt zu einem anderen und Berechnen von Flanken entlang dieser Kanten. Für Dreiecke werden die Flanken der Z-Tiefe, der Farbe und der UV (für Textur) ebenso in der Richtung einer Abtastlinie berechnet.

[0081] Wie gezeigt, schließen alle der Gleitkommablöcke **152** eine CF-Busschnittstellenlogik **362** ein, die an den CF-Bus gekoppelt sind. Jeder der Gleitkommablöcke **152** schließt eine FD-Busschnittstellenlogik **366** ein, die mit dem FD-Bus gekoppelt ist. Jeder Gleitkommablock **152** schließt einen Bypass-Bus oder Datenpfad **364** ein, der als der Datenübertragungspfad durch einen entsprechenden Gleitkommablock **152** für den CD-Bus dient. Daten, die über den CD-Bus gesendet werden, d. h. die direkt zu dem FD-Bus gesendet werden, reisen auf dem Datenübertragungsbus **364** und bypassen somit die Gleitkommalogik, die von dem Gleitkommablock **152** umfasst ist. Der Betrieb dieses Bypass-Busses **364** ist klarer in **Fig.** 10 gezeigt, und wird in Verbindung mit **Fig.** 10 diskutiert.

[0082] Im Allgemeinen können Daten, die an den Gleitkommablock **152** bereitgestellt werden, eine von drei Bestimmungen haben, diese sind der F-Kernblock **353**, der L-Kernblock **354** oder direkt hinaus an den FD-Bus, d. h. eine CD-Bus Übertragung. In dem bevorzugten Ausführungsbeispiel umfassen Daten, die für den F-Kernblock **352** bestimmt sind, 32-Bit Worte, einschließlich 32-Bit IEEE Gleitkommazahlen und andere 32-Bit Daten. Daten, die für den L-Kernblock **354** bestimmt sind, umfassen 48-Bit Worte, die drei 16-Bit Festkommazahlen umfassen.

[0083] Wie in **Fig.** 6 gezeigt, liest der Gleitkommablock **152** sechs kombinierte Eingangs- und Ausgangspuffer ein, ebenso wie zwei spezialisierte Puffer, die eine Kommunikation zwischen dem F-Kernblock **352** und dem L-Kernblock **354** bereitstellen.

[0084] Wie gezeigt, schließt ein Gleitkommablock **152** einen Gleiteingangspuffer (FI-Puffer) **372** ein, der Daten von dem CF-Bus empfängt, der durch den Befehlsblock **142** bereitgestellt wurde. Der FI-Puffer **372** ist doppeltgepuffert und hält 32 32-Bit Eingänge in jedem Puffer. Das erste Wort, Wort Null, das in dem FI-Puffer **372** gespeichert ist, umfasst einen op-Code, der den F-Kernblock **352** informiert, welche Mikrocoderoutine für die empfangenen Geometrieprimitive abgeschickt werden. Nur der Header und X, Y und Z Koordinaten werden an diesen Puffer bereitgestellt.

[0085] Der Gleitkommablock **152** schließt auch einen F-Kern an L-Kern Puffer (FL-Puffer) **374** ein. Der FL-Puffer **374** ist doppeltgepuffert und hält 16 16-Bit Eingänge in jedem Puffer. Der F-Kernblock **352** wird betrieben, um drei F-Kernworte in ein L-Kernwort zu schreiben oder zu kombinieren, das an den FL-Puffer **374** bereitgestellt wird. Von der L-Kernperspektive, erscheint jeder Puffer in dem FL-Puffer **374** als fünf 48-Bit Eingänge. Während Beleuchtungsoperationen, werden drei X, Y, Z Koordinaten von dem F-Kernblock **352** durch den FL-Puffer **374** an den L-Kernblock **354** gesendet. Diese drei X, Y, Z Koordi-

naten werden verwendet, um eine Beleuchtungsrichtung zu berechnen. Wenn die Beleuchtungsattribute geschrieben sind, werden jedoch fünf separate Werte von dem F-Kernblock **352** an den L-Kernblock **354** durch den FL-Puffer **374** gesendet, wobei diese fünf Werte Werte für Emissions-, Umgebungs-, Diffusions-, Spiegel und Spiegelexponentvariablen sind.

[0086] Der Gleitkommablock **152** schließt einen L-Kerneingangspuffer (LI-Puffer) **376** ein, der Daten empfängt, die über den CF-Bus gesendet wurden, die von dem Befehlsblock **142** bereitgestellt wurden und diese Daten an den L-Kernblock **354** bereitstellen. Der LI-Puffer **376** umfasst fünf Puffer, von denen jeder sieben 48-Bit Eingänge hält. Diese sieben 48-Bit Eingänge umfassen drei Scheitelpunktnormale, drei Scheitelpunktfarben und ein Wort mit drei Alpha-Werten. Der FI-Puffer **372** und der LI-Puffer **376** umfassen kollektiv den Gleitkommablockeingangspuffer **155** (**Fig.** 4).

[0087] Der Gleitkommablock **152** schließt auch einen FLL-Puffer **378** ein, der zwischen dem F-Kernblock **352** und dem L-Kernblock **354** verbunden ist. Der FLL-Pufer **378** ist ein FIFO, der für ein Übermitteln von Beleuchtungs- und Abschwächungsfaktoren von dem F-Kernblock **352** an den L-Kernblock **354** verwendet wird. Diese Abschwächungsfaktoren umfassen drei X, Y, Z Positionswerte, drei Abschwächungswerte und ein Abschwächungsverschiebungswort, das drei gepackte Werte enthält. Ein FLF-Puffer **380** wird auch zwischen dem F-Kernblock **352** und dem L-Kernblock **354** bereitgestellt. Der FLF-Puffer ist ein bidirektionaler Puffer, der für ein Kommunizieren von Daten zwischen dem F-Kernblock **352** und dem L-Kernblock **354** unter einer F-Kernsteuerung verwendet wird.

[0088] Ein L-Kern an S-Kern Puffer (LS-Puffer) **386** ist zwischen den L-Kernblock **354** und den S-Kernblock **356** gekoppelt. Der LS-Puffer **386** ist ein Doppelpuffer, bei dem jeder Puffer vier 48-Bit Worte hält.

[0089] Der Gleitkommablock **152** schließt auch einen F-Kern an S-Kern Puffer (FS-Puffer) **384** ein, der für ein Übertragen von Daten von dem F-Kernblock **352** an den S-Kernblock **356** verwendet wird. Der FS-Puffer umfasst fünf Puffer, wobei jeder 32 32-Bit Werte hält. Diese fünf Puffer sind angepasst an die Pipeline-Stufen des L-Kernblocks **354** ausgelegt, diese sind die zwei FL-Puffer, die zwei LS-Puffer, plus ein Primitiv, das in dem L-Kernblock **354** gespeichert werden kann. Daten, die von dem F-Kernblock **352** durch diesen Puffer an den S-Kernblock **356** übertragen werden, schließen einen Absendecode ein, der anzeigt, welche Mikrocodeprozedur in dem S-Kernblock **356** laufen soll.

[0090] Schließlich schließt der Gleitkommablock **152** einen S-Kernausgangspuffer (SO-Puffer) **158** ein, der zwischen den S-Kernblock **356** und die FD-Busschnittstelle **366** gekoppelt ist. Der SO-Puffer **158** sammelt Daten, die über den FD-Bus an den entsprechenden Zeichnungsprozessoren **172A–172B** gesendet werden. Der SO-Puffer **158** wird doppeltgepuffert und hält 32 32-Bit Worte in jedem Puffer. Der SO-Puffer **158** hält bis zu zwei Primitiven, die Festkommadaten der Reihenfolge umfassen, wie sie in den entsprechenden Zeichnungsprozessoren **172A–172B** benötigt werden. Die SO-Puffer **158** schließen ein separates Statusregister ein, das anzeigt, wie viele Worte gültig sind, so dass die minimale Anzahl von Zyklen verwendet werden, um die Daten über dem Bus zu übertragen. Der SO-Puffer **158** umfasst den Gleitkommablockausgangspuffer **158**.

[0091] Zur Information über eine andere Ausführungsform des Gleitkommablock **152** sei bitte auf U.S. Patent 5,517,611 verwiesen, mit dem Titel "Floating Point Processor for a high Performance Three Dimensional Graphics Accelerator", das hiermit durch Bezugnahme in seinem Umfang in die Offenbarung aufgenommen wird.

Fig. 7 – Zeichnungsprozessorblockdiagramm

[0092] Bezugnehmend nun auf **Fig.** 7 ist ein Blockdiagramm gezeigt, das einen der entsprechenden Zeichnungsprozessoren **172** darstellt. Alle der entsprechenden Zeichnungsprozessoren **172A** und **172B** sind identisch, und somit wird nur einer dieser der Einfachheit halber hier beschrieben. Der Zeichnungsprozessor **172** verwaltet das Sequenzieren der 3DRAM Chips. Jeder Zeichnungsprozessor **172** umfasst eine 3DRAM Aufstellungslogik für sowohl interne Pixel-Caches, als auch Videoausgangsauffrischung. Diese Ressourcen werden gesteuert durch Aufreihen wiedergegebener Pixel, bevor diese den 3DRAM erreichen und Vorfühlen der Pixel-Adressen in dieser Reihe, um 3DRAM Cache-Verluste vorherzusagen.

[0093] Wie gezeigt schließt jeder Zeichnungsprozessor **172** einen FD-Schnittstellenblock **402** als Schnittstelle für einen FD-Bus ein. Der FD-Busschnittstellenboock **402** ist an eine CDC-Busschnittstellenlogik **412** gekoppelt. Die CDC-Busschnittstellenlogik **412** ist an einen Scratch-Puffer **414** und eine Direkt-Port-Einheit **416** gekoppelt. Die Direkt-Port-Einheit **416** empfängt Eingänge von einer Rahmenpufferschnittstellenlogik **436** und stellt einen Ausgang an eine Pixel-Daten-Mux-Logik **432** bereit. Die CDC-Busschnittstellenlogik **412** ist ebenso gekoppelt, um Ausgangsdaten an einen DC-Bus bereitzustellen. Die FD-Busschnittstelle **402** stellt Ausgänge an die Primitiv-Akkumulationspuffer **404** bereit.

[0094] Wie oben bemerkt, umfasst der FD-Bus sechs unabhängige Busse, die nur auf einer Pro-Wortbasis synchronisiert sind. Die FD-Busschnittstelle **402** dient zwei Funktionen. Erstens wandelt die FD-Busschnittstelle **402** jeden Satz der drei 11-Bit Datenstücke, die über den FD-Bus übertragen werden, zurück in ein 32-Bit Wort plus ein Steuerungs-Bit. Zweitens richtet die FD-Busschnittstelle **402** empfangene Daten von dem FD-Bus entweder an die Primitivakkumulationspuffer **404** oder an die CD-Busschnittstellenlogik **412**.

[0095] Die CDC-Busschnittstellenlogik **412** arbeitet mit 32-Bit Datenworten. Wie oben beschrieben umfasst der CDC-Bus Teile von anderen Bussen, einschließlich des CF-Bus und des FD-Bus und wird verwendet, um dem Befehlsblock **142** zu erlauben, Pixel in die 3DRAM Chips **192** und **194** zu übertragen. Der DC-Bus erlaubt das Lesen von Registern von dem Zeichnungsprozessor **172**, ebenso wie das Lesen von Pixeln von einen 3DRAM. Daten, die auf dem CD-Bus an einen der Zeichnungsprozessoren **172** bereitgestellt werden, erfordern einen Header als ein erstes Wort. Daten, die zurück an den DC-Bus bereitgestellt werden, haben keinen Header, da der Befehlsblock **142** immer weis, was erforderlich ist.

[0096] Der Zeichnungsprozessor **172** schließt auch ein Zählstands-Board **418** ein, das die Verfolgung einer Primitivreihenfolge hält, wie durch den Befehlsprozessor **142** spezifiziert. Wie gezeigt, empfängt die Zählstands-Board-Logik einen F Num-Eingang und stellt einen Ausgang an die Primitivakkumulationspuffer **404** bereit. Der Befehlsblock **142** stellt einen 3-Bit Code an den Zeichnungsprozessor **172** immer dann bereit, wenn ein (unicast) Primitiv in einen der CF-Busausgangs-FIFOs kopiert wird. Der Code spezifiziert, welcher der sechs Gleitkommablockprozessoren **152A**–**152F** das Primitiv empfängt. Der Code schließt auch ein Bit ein, das anzeigt, ob das Primitiv geordnet oder ungeordnet ist. Für alle geordneten Primitive ist erforderlich, dass sie in der Reihenfolge herauskommen, in der sie eingegeben wurden. Ungeordnete Primitive können aus den Primitivakkumulationspuffern **404** immer entnommen werden, wenn sie verfügbar sind. Einige Primitive, wie Text oder Marker geben mehrfache Primitive für jeden Primitiveingang aus, und diese Primitive sind vorzugsweise wegen der Effizienz in ungeordneter Weise platziert. Jedoch müssen alle Attribute, die an den Zeichnungsprozessor **172** gesendet werden, relativ zu den Primitiven, die sie modifizieren können, geordnet bleiben. Zusätzlich gibt es Fälle, mit Linien und Dreiecken, bei denen eine strikte Ordnung auch eingehalten werden muss. Die Zählstands-Board-Logik **418** hält die Verfolgung von wenigsten 64 Primitiven fest. Die Zählstands-Board-Logik **418** stellt ein Signal zurück an den Befehlsblock **142** bereit, wenn die Zählstands-Board-Logik **418** nahezu voll ist, um ein Überfließen des Zählstands-Board-Puffers **418** zu verhindern.

[0097] Wie oben erwähnt empfangen die Primitivakkumulationspuffer **404** Ausgänge von der FD-Busschnittstelle **402** und von der Zählstands-Board-Logik **418**. Die Primitivakkumulationspuffer **404** stellen einen Ausgang an eine Kantenläuferlogik **422** bereit, die wiederum einen Ausgang an eine Spannenfülllogik **424** bereitstellt. Die Spannenfülllogik **424** stellt einen Ausgang an einen Texturpixelprozessor **426** bereit. Die Spannenfülllogik **424** stellt auch einen Ausgang an die direkte Porteinheit **416** bereit. Die Primitivakkumulationspuffer **404** stellen ebenso einen Ausgang an eine Texturausdehnerlogik **428** bereit. Die Texturausdehnerlogik **428** ist an einen Texturspeicher **430** gekoppelt. Der Texturspeicher **430** stellt Daten an den Texturpixelprozessor **426** bereit. Der Texturspeicher **430** stellt auch Daten an die direkte Porteinheit **416** bereit. Der Texturpixelprozessor **426** und die direkte Porteinheit **416** stellen jeweils Daten an den Pixeldatenmultiplexer **432** bereit. Der Pixeldatenmultiplexer **432** stellt seinen Ausgang an einen Pixelprozessor **434** bereit. Der Pixelprozessor **434** stellt seinen Ausgang an die Rahmenpufferschnittstelle **436** bereit und stellt auch seinen Ausgang an die direkte Porteinheit **416** bereit.

[0098] Die Primitivakkumulationspuffer **404** werden verwendet, um Primitivdaten zu akkumulieren, bis ein komplettes Primitiv empfangen worden ist. Somit bilden die Daten schließlich komplette Primitive, da Daten von den sechs Gleitkommaprozessoren **152A**–**152F** gesammelt werden. Die Primitivakkumulationspuffer **404** schließen genügend Raum ein, um ein komplettes Primitiv zu halten plus einem ausreichenden Speicher, um einen Teil eines zweiten Primitivs zu halten, um den Pipeline-Fluss geglättet zu halten. Die sechs Primitivakkumulationspuffer **404** sind gefüllt, da Daten von jedem der sechs Gleitkommaprozessoren **152A**–**152F** kommen. Sobald das Primitiv vollkommen empfangen worden ist, kommt im Allgemeinen das nächste hinter diesem. Somit schließen die Primitivakkumulationspuffer **404** ausreichende Extrapufferung ein, um die vollständigen Primitive aus dem Primitivakkumulationspuffer **404** an die Kantenläuferlogik **422** zu übertragen, bevor die Daten von den Daten, die von dem nächsten Primitiv kommen, voll werden. In dem bevorzugten Ausführungsbeispiel sind die Primitivakkumulationspuffer **404** einige Wörter länger als das längste Primitiv (Dreieck) das bearbeitet wird. Die Primitivakkumulationspuffer **404** stellen einen 64-Bit Ausgang an die Kantenläuferlogik **422** bereit. Die Primitive werden von den Primitivakkumulationspuffern **404** eins nach dem anderen entfernt, basierend auf dem Inhalt der Zählstands-Board-Logik **418**.

[0099] Die Kantenläuferlogik **422** teilt Primitive in Teile, die leicht durch die Spannenfülleinheit **424** gehandhabt werden können. Für Dreiecke läuft die Kantenläuferlogik **422** entlang der zwei laufenden Kanten und erzeugt ein Paar vertikaler Spannen, die an den nächsten Pixelabtastpunkt angepasst sind, die dann an die Spannenfülleinheit **424** gesendet werden. Die Kantenläufereinheit **422** führt auch eine ähnliche Anpassung für Linien durch, sendet eine Linienbeschreibung an die Spannenfeldeinheit **424**, die sehr ähnlich zu einer Dreieckspanne ist. Die Kantenläuferlogik **422** umfasst zwei 16 × 24 Vervielfacher, die zum Durchführen dieser Anpassungen verwendet werden. Die Kantenläuferlogik **422** schließt weiterhin mehrere Addierer ein, die Zählungen verfolgen, die zum Durchführen anderer Berechnungen verwendet werden. Andere Primitive als Dreiecke und Linien werden aufgespalten, abhängig von der effizientesten Verwendung von Ressourcen. Sowohl gezackte

als auch anti-aliaste Punkte werden direkt durch die Logik mit einem Minimum an Anpassungen gesendet, wie Hinzufügen von 0,5 zu zackigen Punkten. Große Punkte werden durch die Kantenläuferlogik **422** als individuelle Pixel bereitgestellt. Die Kantenläuferlogik **422** wandelt Polygone und Rechtecke in horizontale Spannen um. Die Kantenläuferlogik **422** modifiziert in keiner Weise Bresenham-Linien, bevor sie an eine Spannenfülleinheit **424** geschickt werden.

[0100] Die Spannenfülleinheit **424** führt eine Interpolation von Werten über beliebig orientierte Spannen aus, gewöhnlich für Dreiecke und Linien, und führt auch Filtergewichttafelverweise für anti-aliaste Linien durch. Für optimierte Primitive, einschließlich Dreieckspannenpaare, Rechtecke und Polygonspannen, und anti-aliaste Linien und Punkte werden zwei Pixel pro Zyklus erzeugt. Alle anderen Primitive erzeugen ein Pixel pro Zyklus. Die letzte Stufe der Spannenfülleinheit **424** führt auch ein Schwanken aus, das 12-Bit Farben in 8-Bit Werte umwandelt, durch Verwenden eines 4 × 4 Bildschirmraumschwankungsmusters. Die Spannenfülllogik **424** stellt einen Ausgang an den Texturpixelprozessor **426** bereit.

[0101] Der Texturpixelprozessor **426** führt Texturberechnungen durch und steuert den Verweis von Texeln in dem Texturspeicher **430**. Der Texturpixelprozessor **426** stellt eine an dem Pixel zu verschmelzende Farbe durch den Pixelprozessor **424** her. Der Texturpixelprozessor **426** lässt Daten auf einen Pixeldatenmultiplexer **432** für alle anderen Primitive außer für texturierte Dreiecke durch.

[0102] Wie oben erwähnt, stellen die Primitivakkumulationspuffer **404** einen Ausgang an den Texturausdehner **428** bereit. Der Texturausdehner **428** arbeitet, um empfangene Texturen zum Speichern in dem Texturspeicher **430** auszudehnen. Der Texturspeicher **430** wird somit direkt von den Primitivakkumulationspuffern **404** geladen und ist mit dem Texturpixelprozessor für Texel-Verweise verbunden. Der Texturspeicher **430** ist ausgelegt, um genug Daten für eine Texturkarte eines 16 × 16 Texel-Bereichs zu halten, einschließlich aller der kleineren Mipmaps. Der Texturspeicher **430** ist vorzugsweise doppeltgepuffert, so dass ein Puffer geladen werden kann, während der laufende Puffer in Verwendung ist. Es wird bemerkt, dass der 16 × 16 Texel-Bereich tatsächlich als ein 17 × 17 Feld gespeichert wird, um die Interpolation für einen korrekten Betrieb zu ermöglichen.

[0103] Wie oben erwähnt, empfängt der Pixeldatenmultiplexer **432** Eingangsdaten von dem Texturpixelprozessor **426** und der direkten Porteinheit **416**. Die Pixeldaten-Mux-Logik **432** schlichtet zwischen Pixeln, die von der Spannenfülleinheit **424** kommen, und denen die von dem CD-Bus kommen. Pixel von dem CD-Bus erhalten immer Priorität. Der Pixeldatenmultiplexer **432** stellt seinen Ausgang an den Pixelprozessor **434** bereit.

[0104] Der Pixelprozessor **434** führt ein Mischen, Antialiasing, Tiefen-Cueing und Einstellen für logische Operationen in den 3DRAM 192 und 194 bereit. Der Pixelprozessor **434** umfasst auch eine Logik, die betreibbar ist, um ein Pixelschreiben vor Operationen wie Linienmusterung, Schablonenmusterung, V-Port-Clipping und so fort zu bewahren. Der Pixelprozessor **434** stellt einen Ausgang an die Rahmenpufferschnittstelle **436** bereit.

[0105] Die Rahmenpufferschnittstelle **436** umfasst eine Logik, die notwendig ist, um Pixel von den 3DRAM Speichern **192** und **194** zu lesen und zu schreiben. Die Rahmenpufferschnittstelle **436** verwaltet die Pegel **1** (L1) und Pegel **2** (L2) Caches in den 3DRAM Chips. Das wird durch Vorausschauen auf die zu schreibenden Pixel und Paging in dem benötigten Cache durchgeführt, während andere Pixelzugriffe auftreten. Die Rahmenpufferschnittstelle 436 wiederum ist mit jedem der 3DRAM Speicher **192** und **194** gekoppelt, wie gezeigt.

Fig. 8 – CF-Busdiagramm

[0106] Bezugnehmend nun auf **Fig.** 8, ist ein Blockdiagramm gezeigt, das den CF-Bus ebenso darstellt, wie die relevanten Puffer innerhalb des Befehlsblocks **142** und entsprechende Gleitkommaprozessoren **152A**–**152F**. Wie oben beschrieben ist der Befehlsprozessor **142** an die entsprechenden Gleitkommablöcke **152A**–**152F** gekoppelt. Wie in **Fig.** 8 gezeigt, werden die Daten in sechs separate CF-Busausgangs-FIFOs **144A**–**144F** separiert, da Daten die Scheitelpunktpuffer **344** in dem Befehlsblock (**Fig.** 5) verlassen. Die CF-Busausgangs-FIFOs **144A**–**144F** werden kollektiv als FIFOs **144** in **Fig.** 5 bezeichnet. Jeder CF-Busausgangs-FIFO **144A**–**144F** ist mit einem entsprechenden Gleitkommablock **152** verbunden, und jeder CF-Busausgangs-FIFO **144A**–**144F** wird unabhängig betrieben, während Daten an den Gleitkommablock **152** gesendet werden, mit dem er verbunden ist. Alle Datenübertragungen auf dem CF-Bus sind 48-Bit Worte plus ein 6-Bit Code. Jedes Wort wird als sechs 8-Bit Stücke übermittelt, signifikante Bits zuerst, und der Code wird als sechs 1-Bit Stücke übermittelt.

[0107] Die 48-Bit Worte werden unter den sechs separaten Pfaden synchronisiert. Das erste 8-Bit Stück eines 48-Bit Worts wird auf dem gleichen Zyklus für alle sechs Pfade übertragen. Wenn einer der Pfade keine Daten fertig hat, wenn eine 48-Bit Übertragung beginnt, muss er warten, bis zu dem nächsten 48-Bit Wortübertragungszyklus. Es besteht jedoch keine Synchronisierung relativ zu dem Beginn von Primitiven. Die Worte eines Primitivs können wann immer sie für ein Übertragen verfügbar sind, übertragen werden.

[0108] Da die Datenstücke durch den entsprechenden Gleitkommaprozessor **152** empfangen werden, werden sie in ein 48-Bit Wort wieder zusammengefügt. Der 6-Bit Code ist ebenso zusammengefügt und informiert den Gleitkommaprozessor **152**, was mit den Daten zu tun ist. Gleitkommadaten, wie für

Durchlassdaten, werden von den niedrigeren 32 Bits gezogen und in dem FI-Puffer **372** zum Verarbeiten durch den F-Kern **352** gespeichert. Normale, die als drei 16-Bit Zahlen in ein 48-Bit Wort gepackt sind, werden in dem LI-Puffer **376** zum Verarbeiten durch einen L-Kern **354** gespeichert. Kombinierte Farben und Scheitelpunkte werden mit 16 Bits entpackt, die zu dem LI-Puffer **376** gehen, und 32 Bits entpackt, die zu dem FI-Puffer **372** gehen.

CD-Bus leiht CF-Busdatenleitungen

[0109] Wie in **Fig.** 8 gezeigt, schließt der CF-Bus Extraleitungen ein, die als der CD-Bus bezeichnet werden. Logischerweise ist der CD-Bus unabhängig von dem CF-Bus. Der CD-Bus teilt oder "leiht" jedoch die Datenleitungen von dem CF-Bus und verwendet die Gleitkommaprozessoren **152** als Pufferchips. Wie gezeigt, stellen drei der CF-Busausgangs-FIFOs **144A**–**144C** Daten an entsprechende Multiplexer **502A**–**502C** bereit. Diese Multiplexer empfangen auch 8-Bit Daten, die den CD-Bus umfassen. Ein 3-Bit Teil des CD-Bus wird auch an der letzten Ausgangsstufe des Befehlsblocks **142** bereitgestellt.

[0110] Wenn ein 32-Bit Wort von dem Befehlsblock **142** an den Zeichnungsprozessor **172** übertragen werden muss, wird ein Zyklus von dem CF-Bus "geliehen". Die Übertragung von den CF-Busausgangs-FIFOs **144** wird für einen Zyklus angehalten und die CD-Bus-Daten werden auf den Bus gerichtet. Um an den 11-Bit Datenpfad von den Gleitkommaprozessoren **152** an die Zeichnungsprozessoren **172** anzupassen, werden drei weitere Leitungen zu jedem der ersten drei Befehl-zum-Gleiten(CF)-Datenpfade hinzugefügt. Das stellt 33 Bits zum Übertragen des 32-Bit Worts bereit unter Verwenden von drei der sechs Gleitkommaprozessoren **152**.

[0111] Die Daten, die über den CD-Bus übertragen werden, werden nach der letzten Stufe eines Befehlsprozessorausgangs eingefügt und zurück aus dem Datenstrom in den Gleitkommaprozessor **152** gezogen, vor jeder Verarbeitungsstufe. Die einzige Unterbrechung der CF-Busdatenübertragungen ist der eine geliehene Zyklus, um die Daten durchzuübertragen. In dem bevorzugten Ausführungsbeispiel weisen alle sechs Gleitkommaprozessoren **152** diesen einen Zyklus "hiccup" auf, auch wenn drei von ihnen keine speziellen Daten annehmen. Mehr Details über die CF-Busübertragungen an den Gleitkommaprozessorausgängen sind unten enthalten.

Fig. 9 – FD-Bus

[0112] **Fig.** 9 stellt den FD-Bus dar, der der Bus von den Gleitkommaprozessoren **152** zu den Zeichnungsprozessoren **172** ist. **Fig.** 9 ist ein Blockdiagramm des FD-Bus, das die relevanten Puffer innerhalb eines entsprechenden Gleitkommaprozessors **152** und eines Zeichnungsprozessors **172** zeigt. Es wird bemerkt, dass physikalisch separate Leitungen von jedem der Gleitkommaprozessoren **152** zu jedem der zwei Zeichnungsprozessoren **172** vorhanden sind, wie in **Fig.** 3 und 4 gezeigt, auch wenn **Fig.** 9 nur die Leitungen zu einem der Zeichnungsprozessoren **172** zeigt. Logischerweise sind die Leitungen die gleichen, die zu beiden Zeichnungsprozessoren **172** gehen, da auf ihnen immer die gleichen Daten sind.

[0113] Wie Daten durch die Setup-Einheit (S-Kern) hergestellt werden, werden sie in den SO-Puffer **158** geschrieben. Jedes Wort in diesem Puffer hat 32 Bits. Jedes Wort wird aus dem SO-Puffer **158** in drei 11-Bit Stücken entnommen, signifikante Bits zuerst, und über den FB-Bus mit 11 Bits auf einmal gesendet. Die Datenworte werden dann wieder zurück in 32-Bit Worte in dem Zeichnungsprozessor **172** zusammengefügt. Das 33ste Bit wird auf "1" für das letzte Wort des Primitivs gesetzt. Das eliminiert die Notwendigkeit für jede Wortzahl, die über den Bus gesendet wurde.

[0114] Wie gezeigt, stellt jeder SO-Puffer **158** seinen Ausgang an einen Multiplexer **522** bereit. Der Multiplexer **552** empfängt auch einen 11-Bit Eingang von dem CD-Bus. Wie mit dem CF-Bus, verleiht der FD-Bus ebenso einige von seinen Datenleitungen für den CD-Bus. Logischerweise ist der CD-Bus unabhängig von dem FD-Bus, aber der CD-Bus kann einen Zyklus zu jeder Zeit leihen, um ein 32-Bit Datenwort zu übertragen. Wenn eine CD-Busübertragung stattfindet, wird der FD-Bus für einen Zyklus angehalten und die CD-Busdaten auf den Bus gerichtet. Die 32-Bit Datenübertragung verwendet drei Sätze von 11 Datenleitungen von den Gleitkommaprozessoren **152A**–**152C**. Die Datenleitungen von den Gleitkommaprozessoren **152D**–**152F** werden während dieser Übertragung ignoriert. Wenn die Daten bei den Zeichnungsprozessoren **172** eingehen, werden sie unmittelbar zurück an den internen CD-Bus gerichtet, anstatt in den Primitivakkumulationspuffer **404** zu gehen, wie alle anderen Daten.

Fig. 10 – CDC-Bus

[0115] **Fig.** 10 stellt den CDC-Bus dar, der oben diskutiert wurde. Logischerweise kann der CDC-Bus so wie ein 32-Bit breiter bidirektionaler Datenbus zwischen dem Befehlsprozessor **142** und dem Zeichnungsprozessor **172** sein. Tatsächlich umfasst der CDC-Bus zwei unidirektionale Busse: der CD-Bus, der von dem Befehlsprozessor **142** zu jedem der Zeichnungsprozessoren **172A** und **172B** geht, und der DC-Bus, der von jedem der Zeichnungsprozessoren **172A** und **172B** zu dem Befehlsprozessor **142** geht.

[0116] Der CDC-Bus ist der "direkte Port"-Pfad von dem Befehlsprozessor **142** in den Rahmenpuffer, d. h. die 3DRAM Speicher **192** und **194**. Der CDC-Bus wird zum Schreiben von Pixeln in den Rahmenpuffer verwendet. Der CDC-Bus wird ebenso zum Zurücklesen von Registern und Pixeln, ebenso wie zum Zu-

rücklesen der Inhalte der Gleitkommablöcke SRAM verwendet. Wie unten diskutiert, leiht der CD-Bus einige Leitungen von dem CF-Bus und dem FD-Bus und verwendet die Gleitkommaprozessoren **152A**–**152F** als einen zweistufigen Puffer. Zyklen werden von diesen zwei Bussen ein Wort nach dem anderen auf Anforderung geliehen.

[0117] Wie in **Fig.** 10 gezeigt, wird der CD-Bus über den CF-Bus getragen und wird an den Eingangspuffer **362** der drei entsprechenden Gleitkommablockchips **152A**–**152C** bereitgestellt. Wenn die Datenübertragung eine CF-Busübertragung ist, werden die Daten an die Gleitlogik bereitgestellt, wie gezeigt. Wenn jedoch die Datenübertragung eine CD-Busübertragung ist, werden die Daten von dem entsprechenden FIFO oder einer Busschnittstelle direkt an die entsprechenden Multiplexer **532A**–**532C** in die entsprechenden Gleitkommaprozessoren **152A**–**152C** bereitgestellt. Der Ausgang von jedem der Multiplexer **532A**–**532C** wird durch entsprechende Ausgangspuffer **366** an den FD-Bus, und dann an die entsprechenden Zeichnungsprozessoren **172A** und **172B** bereitgestellt.

[0118] Daten, die entlang des CD-Bus oder Bypassbus übertragen werden, unterbrechen den normalen CF-Busübertragungszyklus und werden aus den entsprechenden Gleitkommablöcken **152** so schnell wie möglich zurückgesendet. Die Übertragungswartezeit durch die Gleitkommablöcke **152** ist zwei Zyklen über diesen Bypassbus. Der Bypassbusdatenpfad **364** ist 11 Bit breit. Wie oben beschrieben, werden drei der entsprechenden Gleitkommaprozessoren, vorzugsweise die Prozessoren **152A**, **152B** und **152C** kollektiv zum Übertragen eines 32-Bit Worts verwendet. Wie ebenfalls oben bemerkt, wird das 33ste Bit dieser drei 11-Bit-Busse verwendet, um ein Ende der Übertragungsbedingung anzuzeigen. Wie gezeigt, empfängt der Bypassbus **364** Daten von der CF-Busschnittstelle **362** und ist gekoppelt, um die Daten an die FD-Busschnittstelle **366** bereitzustellen. Somit verwendet der CD-Bus einen Teil des CF-Bus, eine Teil des FD-Bus und einen internen Datenpfad zu drei der Gleitkommablöcke **152A**–**152C**.

[0119] In der Mehrzahl der Fälle stellt der Befehlsblock **142** Daten an jeden der Zeichnungsblöcke **172A** und **172B** bereit, die durch die Gleitkommalogik in den Gleitkommablöcke **152A**–**152F**, wie oben beschrieben, bereitgestellt werden. In einigen Fällen wünscht jedoch der Befehlsblock **142** Daten direkt an die Zeichnungsblöcke **172A** und **172B** schnell bereitzustellen, ohne einen Durchlass durch die Gleitkommalogik zu erfordern. In diesem Fall verwendet der Befehlsblock **142** den CD-Bus. Der CD-Bus wird primär verwendet, um dem Befehlsblock **142** zu ermöglichen, Daten direkt an den Rahmenpuffer bereitzustellen, bypassend die Gleitkommalogik in den Gleitkommaprozessoren **152**. Wie oben beschrieben, wird ein wesentlicher Teil des CF-Busses "on chip" in drei der Gleitkommablöcke **152A**–**152C** bereitgestellt. Das reduziert den erforderlichen Board-Raum.

[0120] In einem Ausführungsbeispiel kann jeder der entsprechenden Gleitkommablöcke **152** während der Zeit, in der der CD-Bus oder Bypasskanal **364** verwendet wird, um Daten direkt von dem Befehlsblock **142** an die Zeichnungsblöcke **172A** und **172B** zu übertragen, andere Daten während dieser Zeit verarbeiten. Das erlaubt somit, dass zusammentreffende Operationen auftreten, die eine größere Systemeffizienz bereitstellen.

[0121] Wie auch in **Fig.** 10 gezeigt, schließt jeder der Zeichnungsprozessoren **172A** und **172B** einen direkten Datenpfad ein, bezeichnet als den DC-Bus **173**, der an den Befehlsblock **142** gekoppelt ist. Der DC-Bus ist der Datenpfad zurück von jedem der Zeichnungsprozessoren **172A** und **172B** an dem Befehlsprozessor **142**. Der DC-Bus umfasst zwei 16-Bit unidirektionale Punkt-zu-Punkt-Busse. Daten, die über den DC-Bus gesendet wurden, umfassen immer Paare von 16-Bit Worten, die in 32-Bit Worten in dem Befehlsblock **142** gesammelt werden. Wenn Pixel zurückgelesen werden, sind Daten von den zwei Zeichnungsprozessoren **172** unterschiedlich. Der Befehlsprozessor **142** sortiert diese Daten zurück in der Reihefolge, die durch die Host-CPU **102** benötigt wird. Wenn ein einzelnes Pixel von den Zeichnungsprozessoren **172A** und **172B** gelesen wird, sendet nur einer der Zeichnungsprozessoren **172** die Daten zurück und die Hälfte des gesamten 32-Bit breiten Datenpfads bleibt untätig.

[0122] Der DC-Bus stellt einen Rückkehrpfad für Pixel von jedem der Zeichnungsblöcke **172A** und **172B** zurück an den Befehlsblock **142** dar. Somit stellen die Zeichnungsblöcke **172A** und **172B** diese Pixel dann auf dem DC-Bus an den Befehlsblock **142** bereit, wenn der Befehlsblock **142** anfragt, Pixel in den Zeichnungsblöcken **172A** und **172B** zu lesen. Wie gezeigt, schließt der Befehlsblock **142** Puffer ein, die die Daten von dem DC-Bus empfangen. Der DC-Bus ermöglicht dem Befehlsblock **142**, Pixel von einem entsprechenden Rahmenpuffer zu lesen. Der DC-Bus ermöglicht den Zeichnungsblöcken **172A** und **172B** einen Status zurück an den Befehlsblock **142** bereitzustellen, wie während Kontextschaltern.

[0123] Der DC-Bus wird primär verwendet, um dem Befehlsblock **142** zu ermöglichen, Pixel zurück aus den entsprechenden 3DRAM Speichern **192** und **194** zu lesen. Zum Beispiel, wenn ein Fenster von Pixeldaten in den Speichern **192** und/oder 194 gespeichert ist, und dieses Fenster teilweise oder vollständig durch ein anderes Fenster eingeschlossen ist, wünscht die CPU **102**, die eingeschlossenen Daten von einem Speicher zu lesen, so dass diese Daten später wieder angewendet werden können, wenn dieses Fenster nicht länger eingeschlossen ist. In diesem Fall stellt die CPU **102** eine Anfrage bereit, um die Pixeldaten zu dem Befehlsblock **142** zu lesen, und in Antwort auf eine Anfrage von dem Befehlsblock **142** liest jeder der Zeichnungsblöcke **172A** und **172B** die Pixeldaten von den Speichern **192** und **194** und stellt diese Daten zurück bereit an den

DC-Busumkehrpfad an den Befehlsblock **142**. Der Befehlsblock **142** stellt dann wiederum die Daten zurück bereit an die CPU 102 zum Speichern.

Befehlsblockoperation

[0124] Der Befehlsblock **142** steuert die Reihenfolge von Übertragungen in den entsprechenden Gleitkommablöcken **152A**– **152F**, wie oben beschrieben. Der Befehlsblock **142** wird ebenso betrieben, um alle der Operationen innerhalb des Graphikbeschleunigersystems zu steuern. Jeder der Gleitkommablöcke **152A**–**152F** ist erforderlich zum Anfragen und Empfangen einer Erlaubnis von dem Befehlsblock **142** vor einer entsprechenden Übertragung an die Zeichnungsblöcke **172A** und **172B**. Obwohl nicht in den Figuren gezeigt, schließt jeder der Ausgangs-FIFO-Puffer **152A**–**152F** in den entsprechenden Gleitkommablöcken **152A**–**152F** Steuerleitungen ein, die zurückgekoppelt auf den Befehlsblock **142** sind. Diese Steuerleitungen werden durch die entsprechenden Ausgangs-FIFO-Puffer **152A**–**152F** verwendet, um um eine Erlaubnis des Befehlsblocks **142** für eine Übertragung an die entsprechenden Zeichnungsblöcke **172A** und **172B** anzufragen. Jeder der Eingangs-FIFO-Puffer **152A**–**152F** in den entsprechenden Gleitkommablöcken **152A**–**152F** verwenden ebenso ihre entsprechenden Steuerleitungen auf den entsprechenden 12-Bit Kanälen **154A**–**154F**, um Statusinformation an den Befehlsblock **142** bereitzustellen, einschließlich eines Signals, das einschließt, dass der Puffer voll ist und/oder Daten erfordert, etc.
[0125] Wenn der entsprechende FIFO-Puffer **158A**–**158F** um eine Erlaubnis von dem Befehlsblock **142** anfragt und diese empfängt, dann übermittelt der entsprechende Ausgangs-FIFO-Puffer **158** Primitive an jeden der Zeichnungsblöcke **172A** und **172B**. Der Befehlsblock **142** schließt vorzugsweise Zähler für jede der Eingangsschlangen **155A**–**F** und jede der Ausgangsschlangen **158A**–**F** ein, und wird betrieben, um diese entsprechenden Zähler zu erhöhen, wie Daten von den entsprechenden Puffern empfangen bzw. übertragen werden. Der Befehlsblock **142** stellt ebenso Steuerleitungen an jeden der Zeichnungsblöcke **172A** und **172B** bereit, um eine Reihenfolge zum Ausführen für jedes ihrer empfangenen Primitive anzuzeigen.

Patentansprüche

1. Ein Befehlsprozessor (**142**) für einen Grafikbeschleuniger (**112**) umfassend:
einen oder mehr Eingangspuffer (**302**), die für ein Koppeln an einen Bus (**104**) angepasst sind, zum Empfangen von Geometrie-Eingangsdaten, wobei die Geometrie-Eingangsdaten komprimierte Geometrie-Eingangsdaten und nicht komprimierte Geometrie-Eingangsdaten einschließen;
einen ersten Datenpfad, der mit dem einen oder den mehreren Eingangspuffern zum Übertragen der nicht komprimierten Geometrie-Daten gekoppelt ist;
einen zweiten Datenpfad, der mit dem einen oder den mehreren Eingangspuffern zum Empfangen der komprimierten Geometrie-Daten gekoppelt ist, wobei der zweite Datenpfad eine Geometrie-Dekomprimierungseinheit (**316**) einschließt, die an einen Ausgang der Eingangspuffer (**302**) zum Empfangen der komprimierten Geometrie-Eingangsdaten gekoppelt ist, wobei die Dekomprimierungseinheit (**316**) zum Dekomprimieren der komprimierten Geometrie-Eingangsdaten zum Erstellen dekomprimierter Geometrie-Eingangsdaten betrieben wird;
einen oder mehrere Ausgangspuffer (**304**), die zum Empfangen von Daten von dem ersten Datenpfad und dem zweiten Datenpfad gekoppelt sind, wobei die Ausgangspuffer Ausgangsprimitivdaten bereitstellen.

2. Befehlsprozessor gemäß Anspruch 1, weiter umfassend:
einen Multiplexer (**314**), der einen ersten Eingang einschließt, der an den ersten Datenpfad gekoppelt ist, der die nicht-komprimierten Geometrie-Eingangsdaten empfängt, und einen zweiten Eingang, der an den zweiten Datenpfad gekoppelt ist, zum Empfangen einer Ausgabe von der Geometrie-Dekomprimierungseinheit (**316**), wobei der erste Eingang des Multiplexers (**314**) nicht komprimierte Geometrie-Daten von dem einen oder den mehreren Eingangspuffern empfängt, wobei der zweite Eingang des Multiplexers dekomprimierte Geometrie-Eingangsdaten von der Geometrie-Dekompressionseinheit (**316**) empfängt, wobei der Multiplexer einen Ausgang einschließt, der ausgewählt entweder die nicht komprimierten Geometrie-Daten oder die dekomprimierten Geometrie-Eingangsdaten bereitstellt.

3. Befehlsprozessor gemäß Anspruch 2, weiter umfassend:
einen Formatumwandler (**322**), der zum Empfangen eines Ausgangs des Multiplexers (**314**) gekoppelt ist, wobei der Formatumwandler zum Umwandeln empfangener Daten in einen oder mehrere Formattypen betrieben wird, wobei der Formatumwandler zum Bereitstellen eines Ausgangs an einen oder mehrere Ausgangspuffer (**304**) gekoppelt ist.

4. Befehlsprozessor gemäß Anspruch 2, wobei der eine oder die mehreren Eingangspuffer (**302**) zum Empfangen nicht-geometrischer Daten angepasst ist;
wobei der Befehlsprozessor (**142**) weiterhin umfasst:
einen Sammelpuffer (**324**), der an einen Ausgang des Multiplexers (**314**) zum Empfangen der nicht-geometrischen Daten gekoppelt ist, wobei der Sammelpuffer die nicht-geometrischen Daten an den Ausgangspuffer bereitstellt.

5. Befehlsprozessor gemäß Anspruch 4, weiter-

hin umfassend:
einen Scheitelpunkt-Akkumulationspuffer (**332**), der an einen Ausgang des Formatumwandlers (**322**) gekoppelt ist, der eine Mehrzahl von Geometrie-Eingangsdaten speichert, die für einen oder mehrere Scheitelpunkte erforderlich sind.

6. Befehlsprozessor gemäß Anspruch 5, weiterhin umfassend:
eine Mehrzahl von Scheitelpunktpuffern (**334**), die zum Empfangen eines Ausgangs von dem Scheitelpunkt-Akkumulationspuffer (**332**) gekoppelt sind, wobei die Scheitelpunktpuffer zum Konstruieren geometrischer Primitive verwendet werden, wobei die Scheitelpunktpuffer eine Ausgabe geometrischer Primitivdaten an den Ausgabepuffer (**304**) bereitstellen.

7. Einen 3-D-Grafikbeschleuniger (**112**) zum Durchführen dreidimensionaler Grafikbeschleunigungsfunktionen, umfassend:
einen Rahmenpufferspeicher (**192**, **194**);
den Befehlsprozessor gemäß Anspruch 2 zum Empfangen von Hochpegel-Zeichnungsbefehlen zum Zeichnen dreidimensionaler Objekte;
eine Mehrzahl von Gleitkommablöcken (**155A**–**155F**), die an den Befehlsprozessor zum Durchführen von Fließpunktoperationen gekoppelt sind, wobei die Mehrzahl von Fließpunktblöcken die Hochpegel-Befehle von dem Befehlsprozessor empfangen und geometrische Fließpunktoperationen in Antwort auf die Hochpegel-Befehle durchführen, wobei die Mehrzahl von Fließpunktblöcken jeweils geometrische Primitivdaten produzieren;
einen oder mehrere Zeichnungsblöcke (**172A**, **172B**), die an die Mehrzahl von Fließpunktblöcken (**155A**–**155F**) gekoppelt sind, die geometrische Primitivdaten von der Mehrzahl von Fließpunktblöcken empfangen, wobei der eine oder die mehreren Zeichnungsblöcke an den Rahmenpufferspeicher gekoppelt sind, zum Übertragen dreidimensionaler Objektpixeldaten in den Rahmenspeicherpuffer (**192**, **194**); und
einen Digital/Analog-Umwandler (**196**), der an den Rahmenpufferspeicher zum Empfangen von Pixeldaten von dem Rahmenpufferspeicher gekoppelt ist und einen analogen Ausgang an einen Videomonitor liefert.

8. 3-D-Grafikbeschleuniger gemäß Anspruch 7, wobei der Befehlsprozessor weiterhin umfasst:
einen Formatumwandler (**322**), der zum Empfangen der Ausgabe des Multiplexers (**314**) gekoppelt ist, wobei der Formatumwandler (**322**) zum Umwandeln empfangener Daten in einen oder mehrere Formattypen betrieben wird; wobei der Formatumwandler zum Bereitstellen eines Ausgangs an einen oder mehrere Ausgangspuffer (**304**) gekoppelt ist.

9. 3-D-Grafikbeschleuniger gemäß Anspruch 8, wobei der eine oder die mehreren Eingangspuffer (**302**) in dem Befehlsprozessor zum Empfangen nicht-geometrischer Daten angepasst sind;
wobei der Befehlsprozessor (**142**) weiterhin umfasst:
einen Sammelpuffer (**324**), der an den Ausgang des Multiplexers (**314**) zum Empfangen der nicht-geometrischen Daten gekoppelt sind, wobei der Sammelpuffer (**324**) die nicht-geometrischen Daten an die Ausgangspuffer (**304**) liefert.

10. Ein Verfahren zum Verarbeiten geometrischer Daten in einem Grafikbeschleuniger (**112**), umfassend:
Empfangen von Geometrie-Eingangsdaten, wobei die Geometrie-Eingangsdaten komprimierte Geometrie-Eingangsdaten und nicht komprimierte Geometrie-Eingangsdaten einschließen;
Übertragen der komprimierten Geometrie-Daten auf einen ersten Datenpfad an eine Geometrie-Dekomprimierungseinheit (**316**);
die Dekomprimierungseinheit die komprimierten Geometrie-Eingangsdaten zum Erstellen dekomprimierter Geometrie-Eingangsdaten dekomprimiert;
Übertragen der nicht-komprimierten Geometrie-Daten auf einen zweiten Datenpfad, wobei der zweite Datenpfad eine Geometrie-Dekomprimierungsseinheit nicht einschließt; und
ausgewähltes Bereitstellen entweder der nicht-komprimierten Geometrie-Daten oder der dekomprimierten Geometrie-Daten auf einen dritten Datenpfad.

Es folgen 10 Blatt Zeichnungen

Anhängende Zeichnungen

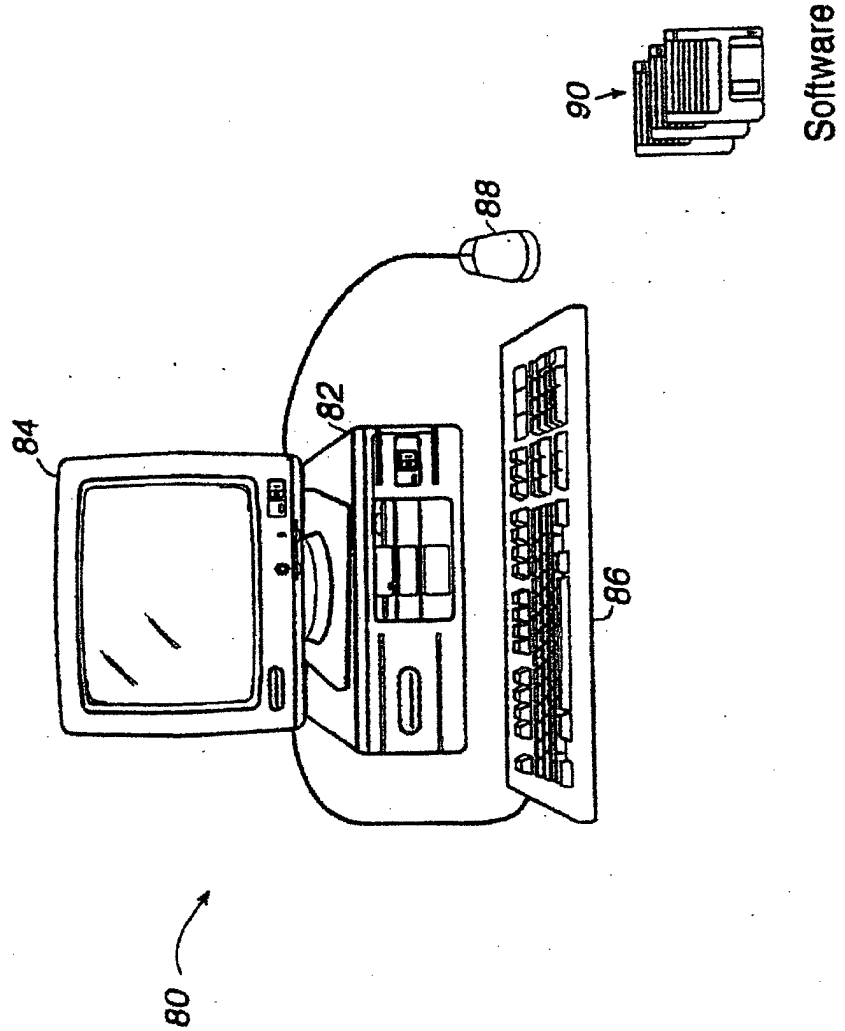


Fig. 1

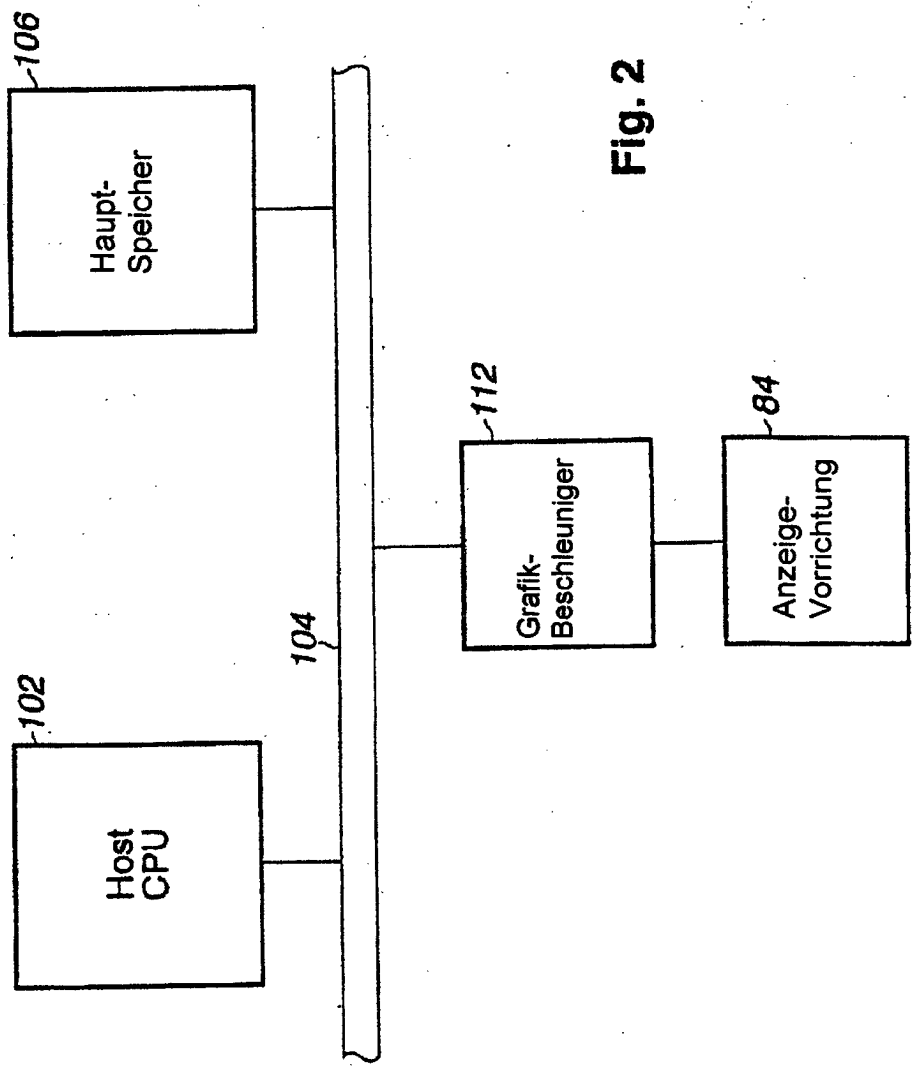

106
Haupt-
Speicher
112
Grafik-
Beschleuniger
84
Anzeige-
Vorrichtung
104
102
Host
CPU

Fig. 2

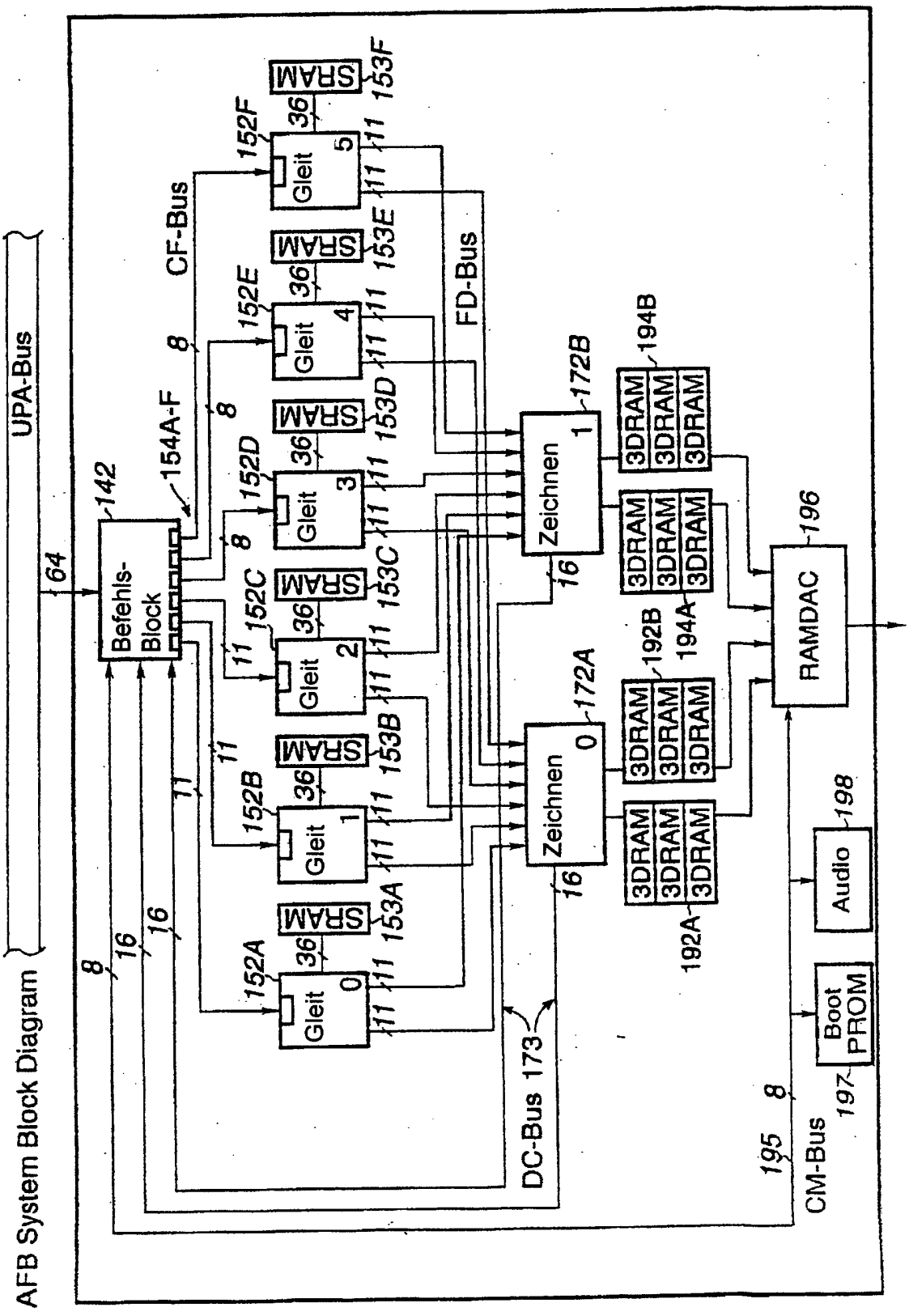

AFB System Block Diagram
UPA-Bus
64
Befehls-Block
142
154A-F
8
CF-Bus
FD-Bus
DC-Bus 173
CM-Bus
195
197
Boot PROM
Audio
198
RAMDAC
196
Zeichnen 0
172A
Zeichnen 1
172B
16
Gleit 0
Gleit 1
Gleit 2
Gleit 3
Gleit 4
Gleit 5
152A
152B
152C
152D
152E
152F
SRAM
153A
153B
153C
153D
153E
153F
36
11
3DRAM
192A
192B
194A
194B

Fig. 3

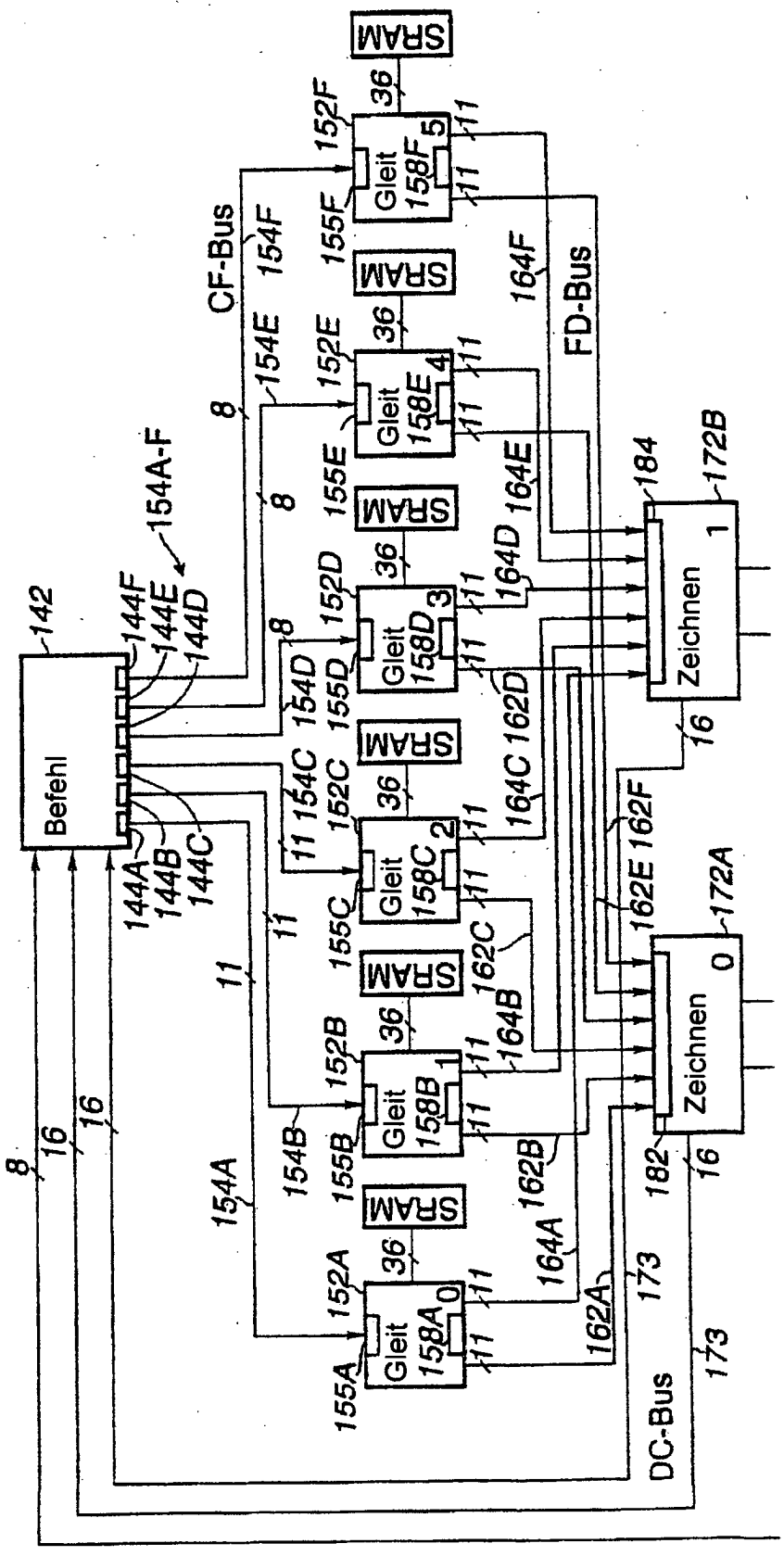
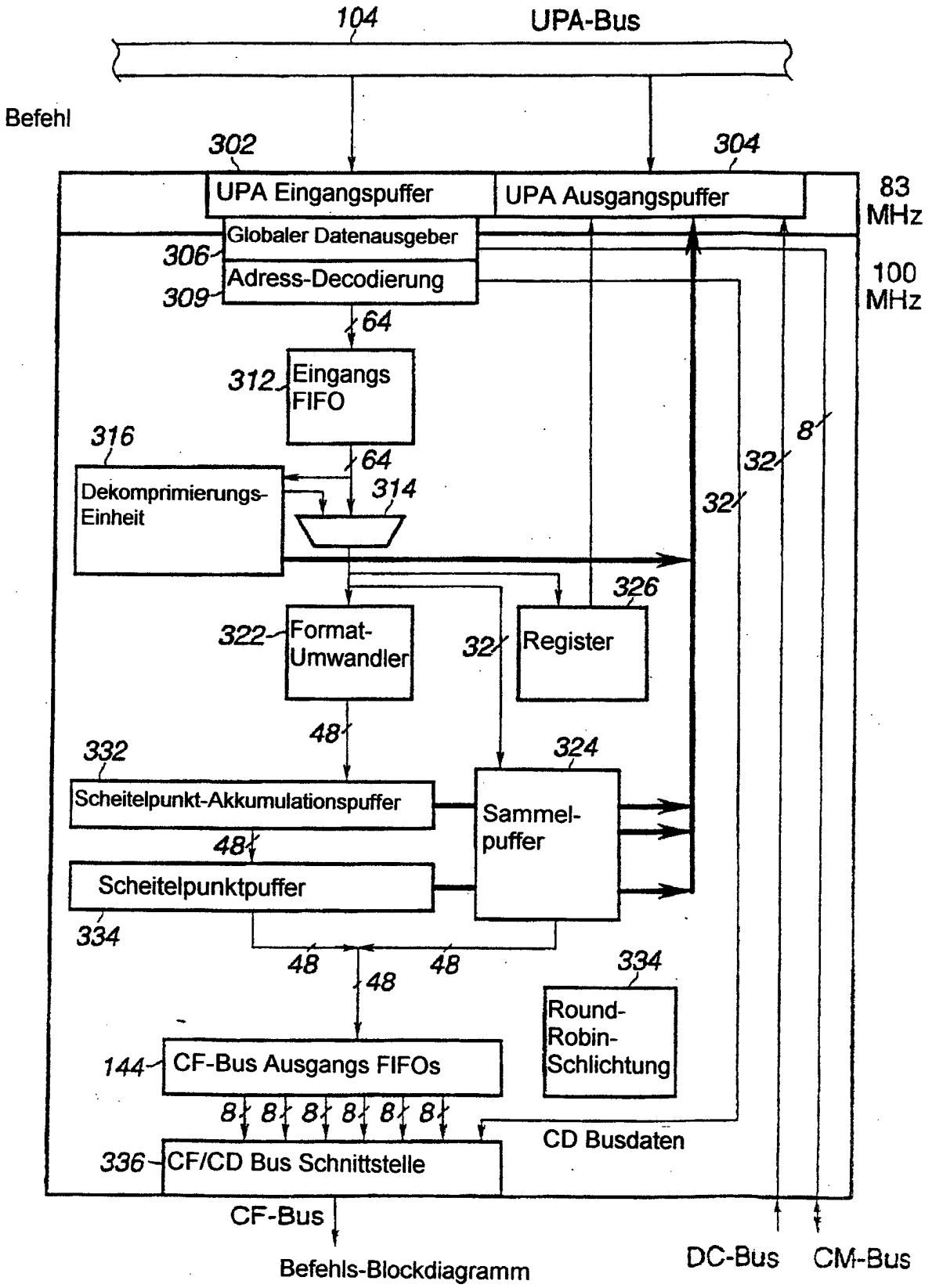

Befehl
142
144A
144B
144C
144D
144E
144F
154A-F
CF-Bus
DC-Bus
FD-Bus
SRAM
Gleit
Zeichnen
158A
158B
158C
158D
158E
158F
172A
172B
184
182
173

Fig. 4

Fig. 5


104
UPA-Bus
Befehl
302
304
UPA Eingangspuffer
UPA Ausgangspuffer
83 MHz
Globaler Datenausgeber
306
Adress-Decodierung
309
100 MHz
64
312
Eingangs FIFO
316
64
314
8
32
Dekomprimierungs-Einheit
32
326
322
Format-Umwandler
32
Register
48
332
324
Scheitelpunkt-Akkumulationspuffer
Sammel-puffer
48
Scheitelpunktpuffer
334
48
48
48
334
Round-Robin-Schlichtung
144
CF-Bus Ausgangs FIFOs
8 8 8 8 8 8
CD Busdaten
336
CF/CD Bus Schnittstelle
CF-Bus
Befehls-Blockdiagramm
DC-Bus
CM-Bus

Gleitkommablock

Gleit-Flussdiagramm

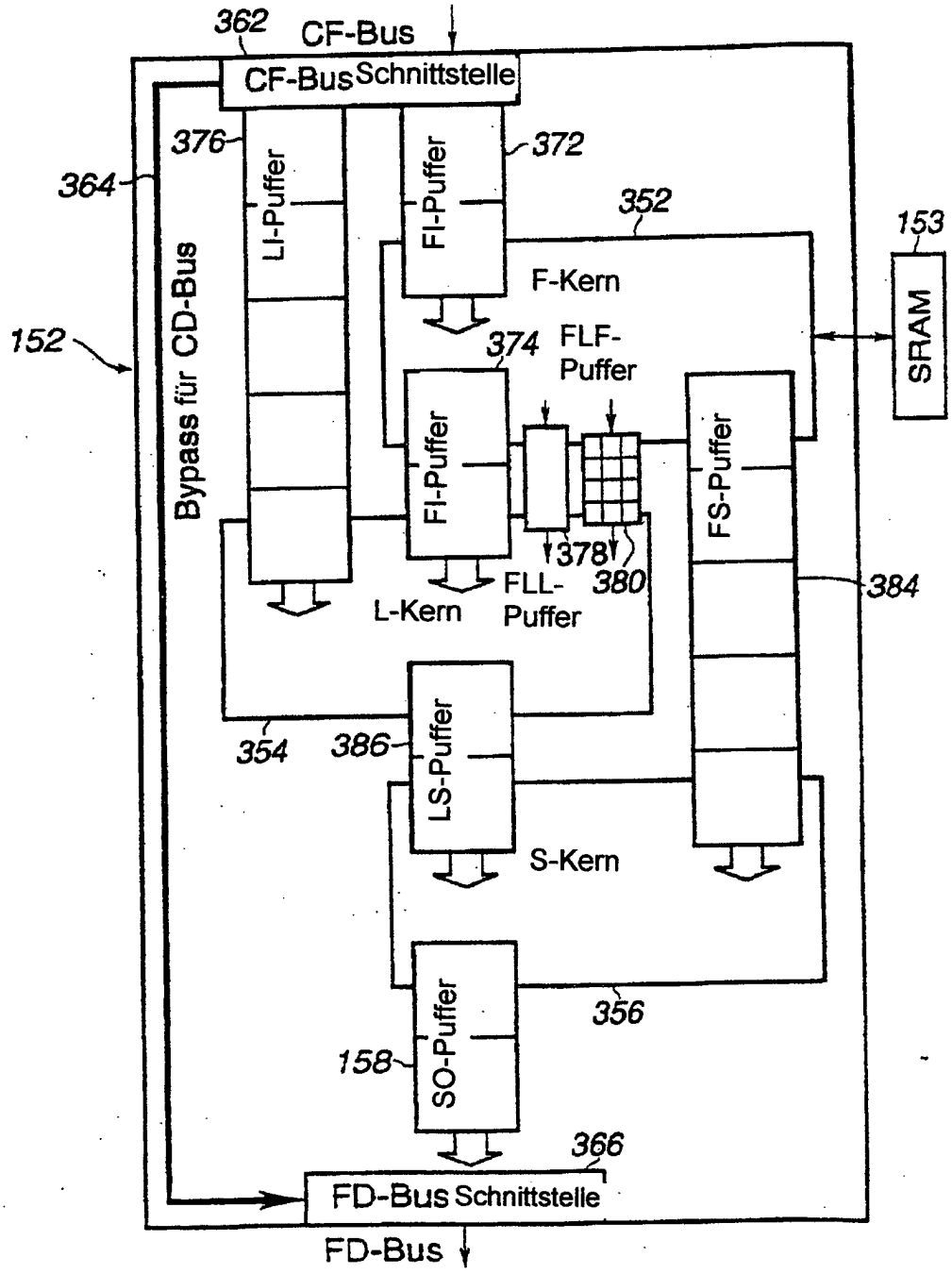
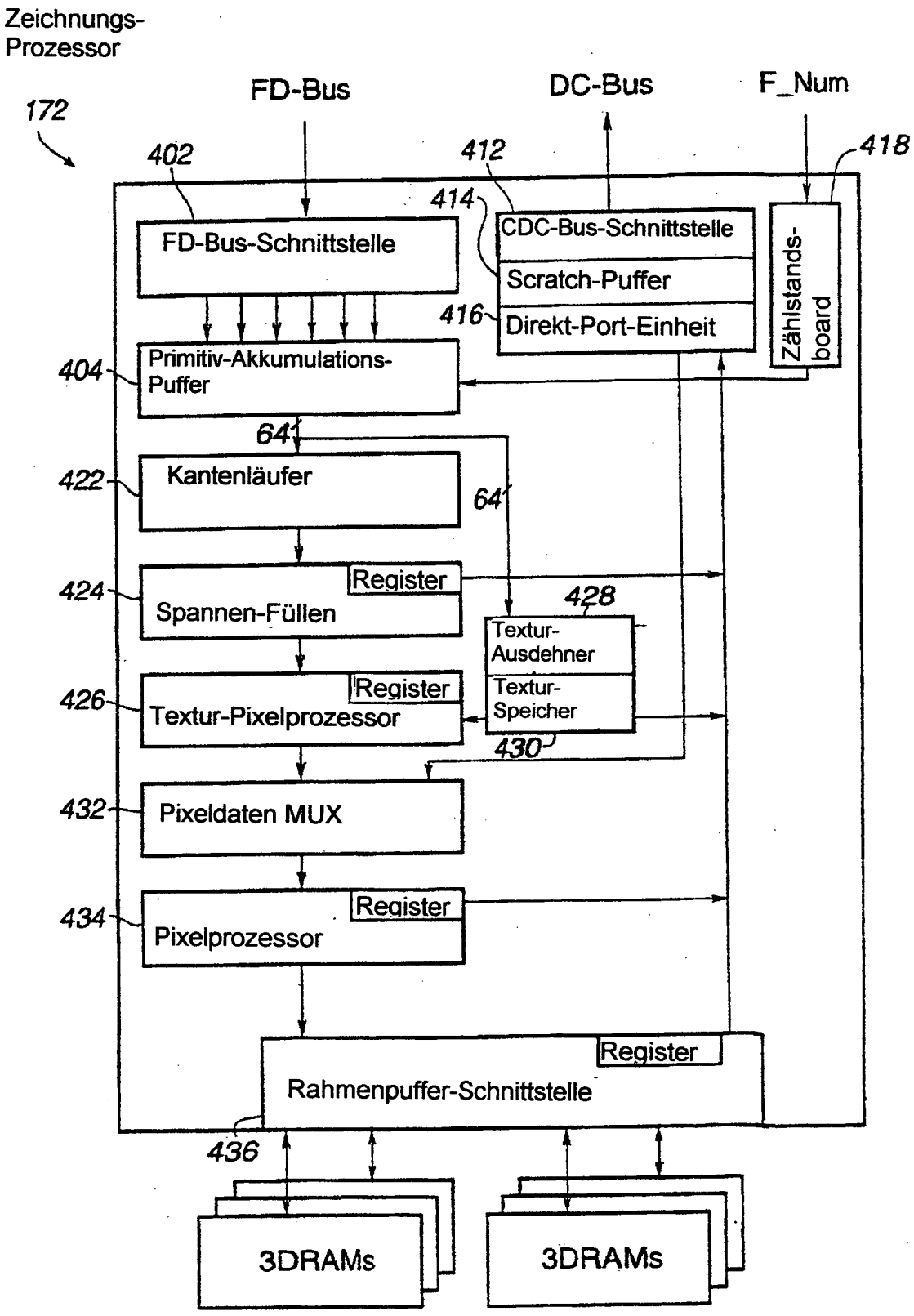
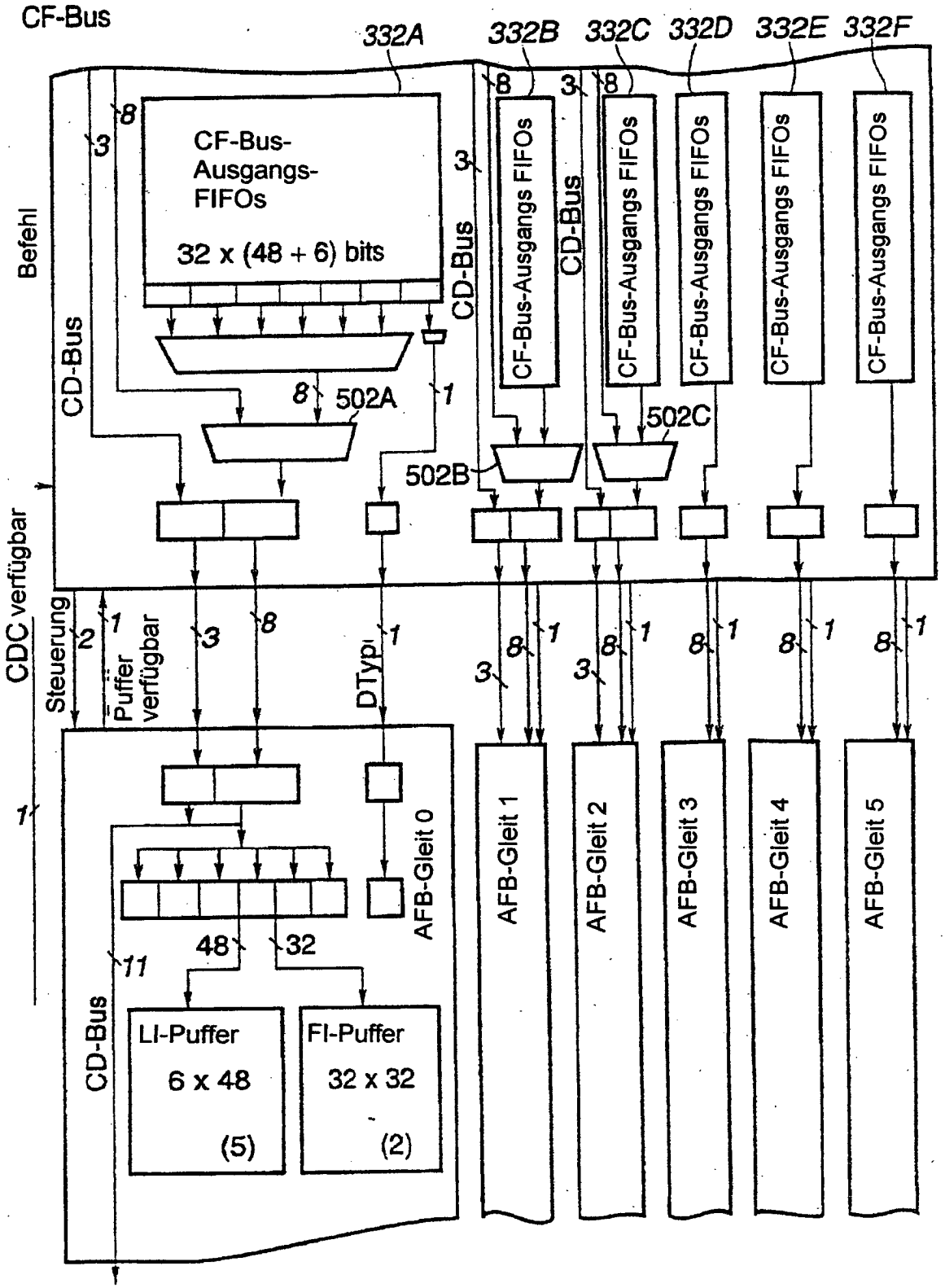


Fig. 6



Zeichnungs Bockdiagramm

Fig. 7



Bus-Blockdiagramm

Fig. 8

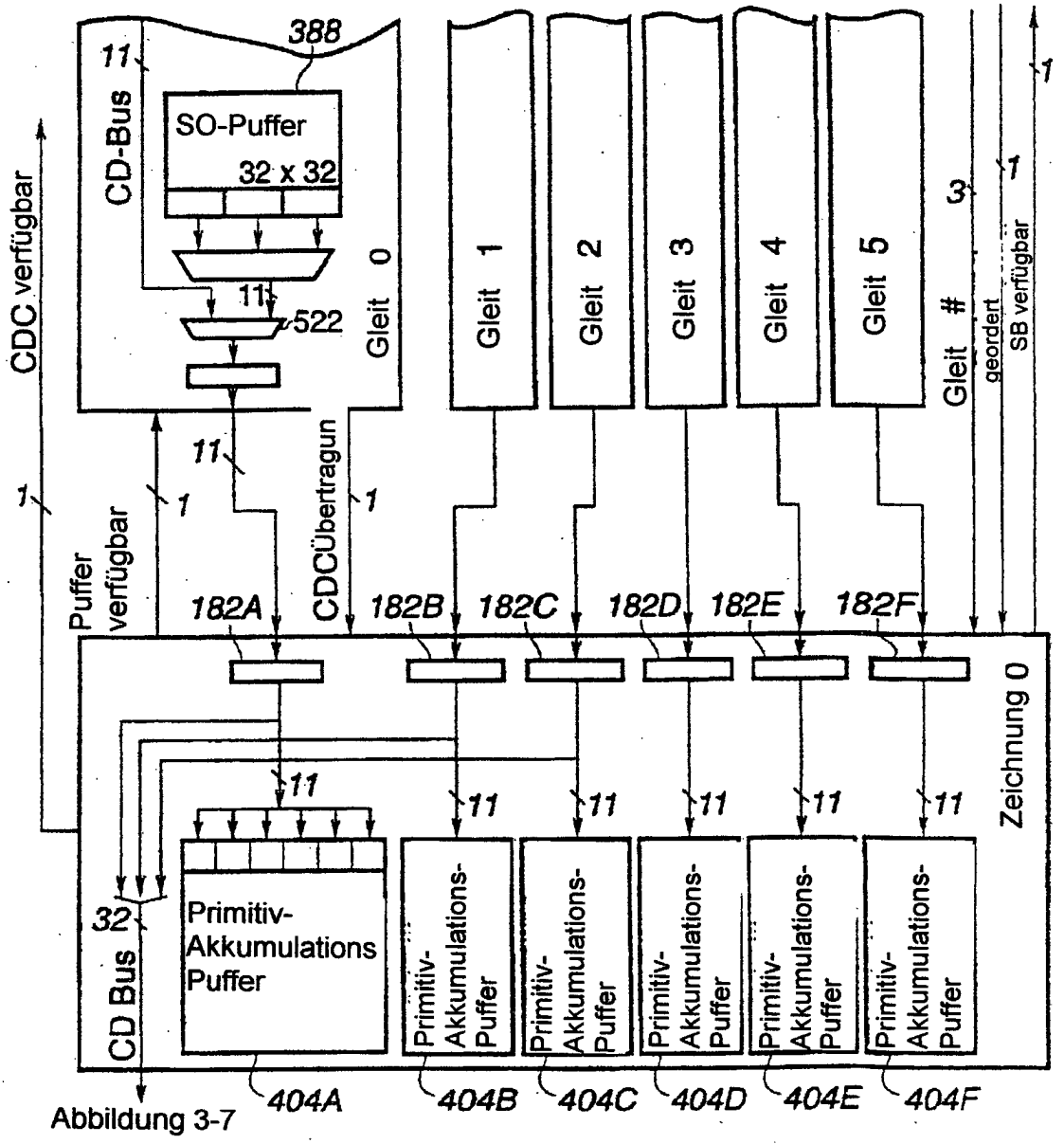


Abbildung 3-7

FD-Bus-Blockdiagramm

Fig. 9

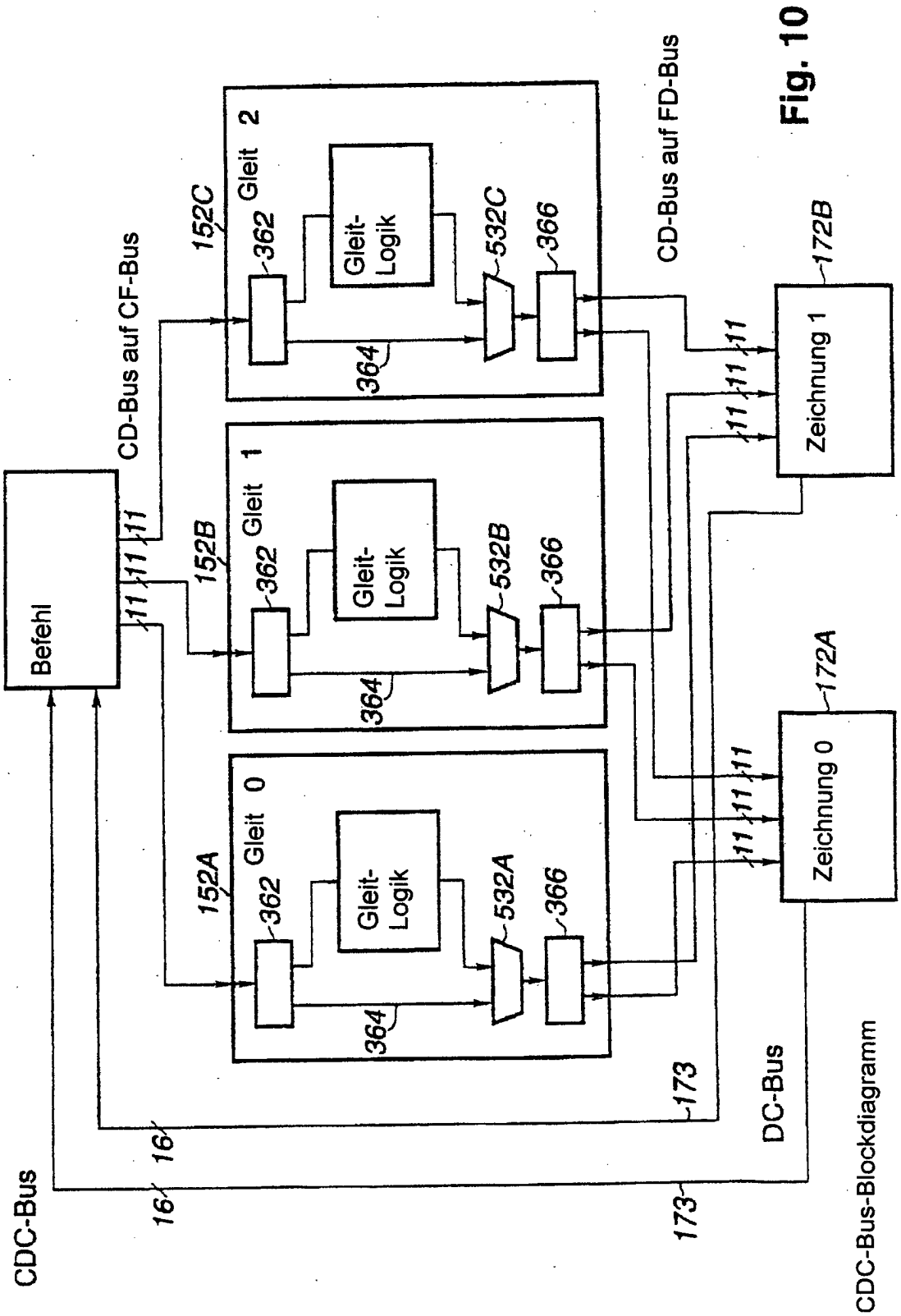


Fig. 10

CDC-Bus-Blockdiagramm