



(19)
Bundesrepublik Deutschland
Deutsches Patent- und Markenamt

(10) **DE 698 35 159 T2** 2007.06.14

(12) **Übersetzung der europäischen Patentschrift**

(97) **EP 1 021 759 B1**

(21) Deutsches Aktenzeichen: **698 35 159.2**

(86) PCT-Aktenzeichen: **PCT/US98/12666**

(96) Europäisches Aktenzeichen: **98 930 354.0**

(87) PCT-Veröffentlichungs-Nr.: **WO 1999/019791**

(86) PCT-Anmeldetag: **17.06.1998**

(87) Veröffentlichungstag
der PCT-Anmeldung: **22.04.1999**

(97) Erstveröffentlichung durch das EPA: **26.07.2000**

(97) Veröffentlichungstag
der Patenterteilung beim EPA: **05.07.2006**

(47) Veröffentlichungstag im Patentblatt: **14.06.2007**

(51) Int Cl.⁸: **G06F 9/30** (2006.01)

G06F 9/302 (2006.01)

G06F 7/544 (2006.01)

(30) Unionspriorität:

948679 **10.10.1997** **US**

(73) Patentinhaber:

**Advanced Micro Devices, Inc., Sunnyvale, Calif.,
US**

(74) Vertreter:

**Grünecker, Kinkeldey, Stockmair &
Schwanhäusser, 80538 München**

(84) Benannte Vertragsstaaten:

DE, GB

(72) Erfinder:

**OBERMAN, Stuart, Sunnyvale, CA 94086, US;
JUFFA, Norbert, San Jose, CA 95134, US**

(54) Bezeichnung: **MIKROPROZESSOR MIT Extremwertbefehlen und Vergleichsbefehlen**

Anmerkung: Innerhalb von neun Monaten nach der Bekanntmachung des Hinweises auf die Erteilung des europäischen Patents kann jedermann beim Europäischen Patentamt gegen das erteilte europäische Patent Einspruch einlegen. Der Einspruch ist schriftlich einzureichen und zu begründen. Er gilt erst als eingelegt, wenn die Einspruchsgebühr entrichtet worden ist (Art. 99 (1) Europäisches Patentübereinkommen).

Die Übersetzung ist gemäß Artikel II § 3 Abs. 1 IntPatÜG 1991 vom Patentinhaber eingereicht worden. Sie wurde vom Deutschen Patent- und Markenamt inhaltlich nicht geprüft.

Beschreibung

Gebiet der Erfindung

[0001] Diese Erfindung betrifft Computersysteme und Mikroprozessoren und betrifft insbesondere eine Multimediaausführungseinheit, die in einem Mikroprozessor enthalten ist, um Hochgeschwindigkeitsmultimediaanwendungen zu unterstützen. Die Erfindung betrifft ferner Extremwertfunktionen und Vektorverarbeitung, die in Mikroprozessor gestützten Systemen eingerichtet sind.

Beschreibung des Stands der Technik

[0002] Mikroprozessoren erreichen typischerweise ein besseres Leistungsverhalten, indem Verarbeitungsaufgaben in mehrere Pipeline-Stufen bzw. Schritte unterteilt werden. Auf diese Weise können Mikroprozessoren unabhängig verschiedene Teile mehrerer Befehle während eines einzelnen Taktzyklus ausführen. Im hierin verwendeten Sinne bezeichnet der Begriff „Taktzyklus“ ein Zeitintervall, während welchem die Pipeline-Stufen eines Mikroprozessors ihre vorgesehenen Funktionen ausführen. Am Ende des Taktzyklus werden die sich ergebenden Werte an die nächste Pipeline-Stufe weitergegeben.

[0003] Mikroprozessorgestützte Computersysteme wurden ursprünglich für Geschäftsanwendungen eingesetzt, zu denen u. a. die Textverarbeitung und die Tabellenkalkulation gehört. Computersysteme werden jedoch zunehmend zur Verwendung von Echtzeitanwendungen eingesetzt, zu denen Multimediaanwendungen, etwa die Video- und Audioverarbeitung, die Videoaufnahme und Wiedergabe, Telefonie und Spracherkennung gehören. Da diese Multimediaanwendungen rechenintensiv sind, wurden diverse Verbesserungen in Mikroprozessoren eingerichtet, um das Multimediaverhalten zu verbessern. Beispielsweise wurden einige Mikroprozessoren für Allgemeinzwecke mit Multimediaausführungseinheiten erweitert, die ausgebildet sind, gewisse spezielle Befehle auszuführen, die insbesondere für Multimediaberechnungen zugeschnitten sind. Diese Befehle werden häufig als „vektorierte“ Befehle eingerichtet, in denen Operanden für die Befehle in separate Abschnitte oder Vektoren unterteilt werden, die unabhängig gemäß der Befehlsdefinition verarbeitet werden können. Beispielsweise kann ein vektorisierter Addierbefehl ein Paar aus 32-Bit-Operanden enthalten, wovon jeder in vier 8-Bit-Abschnitte unterteilt ist. Beim Ausführen eines derartigen vektorisierten Addierbefehls werden entsprechende 8-Bit-Abschnitte jedes Operanden unabhängig und gleichzeitig addiert, um vier separate und unabhängige Addierergebnisse zu erhalten. Das Einrichten derartiger vektorisierter Befehle in einem Computersystem fördert die Ausnutzung der Parallelität und führt typischerweise zu einem verbesserten Leistungsverhalten für gewisse Anwendungen. Eine Art einer üblicherweise benutzten Funktion in Multimediaanwendungen ist eine Vergleichsfunktion. Eine Vergleichsfunktion wird typischerweise durch das Ausführen eines Vergleichsbefehls eingerichtet, deren Wert eines Operanden mit einem weiteren Vergleich, um zu bestimmen, ob der Wert des ersten Operanden größer ist, gleich ist oder kleiner ist als der andere. Ein Vergleichsbefehl kann als ein vektorisierter Befehl behandelt werden, wobei entsprechende Abschnitte zugeordneter Operanden unabhängig von anderen Abschnitten des Operanden verglichen werden.

[0004] Ein weiterer Satz aus Funktionen, die häufig in der Multimediaverarbeitung eingesetzt werden, sind Extremwertfunktionen. Im hierin verwendeten Sinne sind „Extremwertfunktionen“ jene Funktionen, die entweder einen minimalen Wert, der aus mehreren Werten ausgewählt wird, oder einen maximalen Wert, der aus mehreren Werten ausgewählt wird, als Ergebnis der Funktion zurückgeben. In typischen Multimediasystemen wird ein minimaler Wert und ein maximaler Wert durch das Ausführen mehrerer sequenziell ausgeführter Befehle erhalten. Beispielsweise kann zunächst ein Vergleichsbefehl ausgeführt werden, um die relativen Größen eines Paares aus Operandenwerten zu bestimmen, und nachfolgend kann ein bedingter Verzweigungsbefehl ausgeführt werden, um zu bestimmen, ob eine Verschiebeoperation bzw. „Move“-Operation ausgeführt werden muss, um den Extremwert in ein Zielregister oder einen anderen Speicherplatz zu übertragen. Diese Abfolgen aus Befehlen sind in Multimediaanwendungen üblich, etwa Abschneidealgorithmen in Graphikerzeugungssystemen. Da Extremwertfunktionen durch das Ausführen mehrerer Befehle eingerichtet sind, wird jedoch ein relativ großer Betrag an Verarbeitungszeit durch derartige Operationen verbraucht.

[0005] EP-A-0 678 808 beschreibt eine Vorrichtung, um zu bestimmen, welcher von zwei Operanden der größte oder der kleinste ist, wobei einer von dem anderen subtrahiert und das Ergebnis verwendet wird, um einen Multiplexer anzusteuern, um eine Auswahl zwischen den beiden Operanden zu treffen.

[0006] WO 96/17292A beschreibt einen Mikroprozessor, der ausgebildet ist, Operanden als gepackte Daten von einer ersten und einer zweiten Stelle zu vergleichen und um ein entsprechendes Ergebnis mit gepackten Daten an eine dritte Stelle zu übermitteln.

[0007] Es besteht daher ein Bedarf, eine Multimediaausführungseinheit in einem Mikroprozessor vorzusehen, die in der Lage ist, einen Extremwert durch das Ausführen eines einzelnen Befehls zu erhalten. Es besteht ferner ein Bedarf, eine Multimediaausführungseinheit bereitzustellen mit einer effizienten Hardwareeinrichtung der Extremwertbefehle.

Überblick über die Erfindung

[0008] Die zuvor dargestellten Probleme werden größtenteils mittels einer Ausführungseinheit gemäß der vorliegenden Erfindung gelöst. In einer Ausführungsform wird eine Ausführungseinheit zum Ausführen eines ersten Befehls bereitgestellt, der ein Opcode- bzw. Operationscodierungsfeld, ein erstes Operandenfeld und ein zweites Operandenfeld aufweist. Die Ausführungseinheit umfasst ein erstes Eingangsregister zum Empfangen eines ersten Operanden, der durch einen Wert des ersten Operandenfeldes spezifiziert ist, und ein zweites Eingangsregister zum Empfangen eines zweiten Operanden, der durch einen Wert des zweiten Operandenfeldes angegeben ist. Die Ausführungseinheit umfasst ferner eine Komparatoreinheit, die angeschlossen ist, einen Wert des Operationscodierungsfeldes für den ersten Befehl zu empfangen. Die Komparatoreinheit bzw. Vergleichseinheit ist angeschlossen, die Werte des ersten und des zweiten Operanden entsprechend von dem Eingangsregister und dem zweiten Eingangsregister zu empfangen. Die Ausführungseinheit umfasst ferner einen Multiplexer, der mehrere Eingangssignale empfängt. Diese Eingangssignale enthalten einen ersten konstanten Wert, einen zweiten konstanten Wert und die Werte des ersten und des zweiten Operanden. Wenn der decodierte Operationscodierungswert, der von dem Komparator empfangen wird, angibt, dass der erste Befehl ein Vergleichsbefehl oder eine Extremwertfunktion ist, überträgt der Komparator ein oder mehrere Steuersignale an den Multiplexer, um einen Ausgang des Multiplexers als das Ergebnis des ersten Befehls auszuwählen. Wenn der erste Befehl einer von mehreren Extremwertbefehlen ist, wählen die einen oder mehreren Steuersignale, die von der Komparatoreinheit übermittelt werden, zwischen dem ersten Operanden und dem zweiten Operanden aus, um das Ergebnis des ersten Befehls zu bestimmen. Wenn der erste Befehl einer von mehreren Vergleichsbefehlen ist, wählen die einen oder die mehreren Steuersignale, die von der Komparatoreinheit übermittelt werden, zwischen dem ersten konstanten Wert und dem zweiten konstanten Wert aus, um das Ergebnis des ersten Befehls zu bestimmen. Im Falle von Vergleichsbefehlen kann der Wert der ersten oder der zweiten Konstanten vorteilhafterweise ausgewählt werden, um eine Maske zur Verwendung in nachfolgenden Befehlen zu bilden. In einer weiteren Ausführungsform ist eine ähnliche Ausführungseinheit vorgesehen, die Vektoroperanden handhabt.

[0009] Die Extremwertfunktionen werden somit in einem einzelnen Befehl eingerichtet. Dies führt vorteilhafterweise zu einer verbesserten Ausführung dieser Befehle, was insbesondere in Multimediaanwendungen wichtig ist. Eine effiziente Hardwareeinrichtung wird ebenso erreicht, da die für die Extremwertoperationen verwendete Schaltung auch für eine Vielzahl von Vergleichsoperationen genutzt wird.

[0010] Allgemein gesagt, betrifft die vorliegende Erfindung einen Mikroprozessor, der ausgebildet ist, einen ersten Befehl auszuführen, wobei eine codierte Darstellung des ersten Befehls ein Operationscodierungsfeld, ein erstes Operandenfeld und ein zweites Operandenfeld aufweist. Der Mikroprozessor umfasst eine Ausführungseinheit, die angeschlossen ist, einen decodierten Wert aus dem Operationscodierungsfeld, einen ersten Operanden, der durch einen Wert des ersten Operandenfeldes und einen zweiten Operanden, der durch einen Wert des zweiten Operandenfeldes angegeben ist, zu empfangen, wobei die Ausführungseinheit ausgebildet ist, eine Extremwertoperation an dem ersten Operanden und dem zweiten Operanden in Reaktion auf das Empfangen des decodierten Wertes des Operationscodierungsfeldes auszuführen. Die Ausführungseinheit ist ferner ausgebildet, einen Ausgabewert der Extremwertoperation als Ergebnis des ersten Befehls zu übermitteln.

[0011] Die vorliegende Erfindung betrifft ferner eine Ausführungseinheit in einem Mikroprozessor, um einen ersten Befehl auszuführen, wobei eine codierte Darstellung des ersten Befehls ein Operationscodierungsfeld, ein erstes Operandenfeld und ein zweites Operandenfeld aufweist. Die Ausführungseinheit umfasst ein erstes Eingangsregister, das angeschlossen ist, einen ersten Operanden, der durch einen Wert des ersten Operandenfeldes spezifiziert ist, zu empfangen, und umfasst ein zweites Eingangsregister, das angeschlossen ist, einen durch einen Wert des zweiten Operandenfeldes spezifizierten zweiten Operanden zu empfangen. Die Ausführungseinheit umfasst ferner eine Komparatoreinheit, die angeschlossen ist, den ersten Operanden aus dem ersten Eingangsregister und den zweiten Operanden aus dem zweiten Eingangsregister zu empfangen. Die Komparatoreinheit ist angeschlossen, einen decodierten Operationscodierungswert des Operationscodierungsfeldes auf einem Bus für den decodierten Operationscodierungsfeld zu empfangen. Ferner umfasst die Ausführungseinheit einen Multiplexer, der angeschlossen ist, mehrere Eingangssignale mit dem ersten Operanden aus dem ersten Eingangsregister, dem zweiten Operanden aus dem zweiten Eingangsregister, einem

ersten konstanten Wert und einem zweiten konstanten Wert zu empfangen. Der Multiplexer ist ausgebildet, eines der mehreren Eingangssignale bzw. Eingänge auszuwählen, der als Ergebnis des ersten Befehls in Reaktion auf das Empfangen eines oder mehrerer Steuersignale aus der Komparatoreinheit übermittelt wird. Die Komparatoreinheit ist ausgebildet, das eine oder die mehreren Steuersignale in Reaktion auf das Empfangen des decodierten Operationscodierungswertes zu erzeugen, und wenn der decodierte Operationscodierungswert angibt, dass der erste Befehl einer von mehreren Extremwertbefehlen ist, sind der eine oder die mehreren Steuerbefehle verwendbar, um den ersten Operanden oder den zweiten Operanden als den Ausgangswert auszuwählen. Wenn der decodierte Operationscodierungswert angibt, dass der erste Befehl einer von mehreren Vergleichsbefehlen ist, sind das eine oder die mehreren Steuersignale verwendbar, um den ersten konstanten Wert oder den zweiten konstanten Wert als en Ausgangswert auszuwählen.

[0012] Die vorliegende Erfindung betrifft ferner eine Ausführungseinheit in einem Mikroprozessor zum Ausführen eines ersten Befehls, wobei eine codierte Darstellung des ersten Befehls ein Operationscodierungsfeld, ein erstes Operandenfeld und ein zweites Operandenfeld aufweist. Die Ausführungseinheit umfasst ferner ein erstes Eingangsregister, das angeschlossen ist, einen ersten Operanden zu empfangen, der durch einen Wert des ersten Operandenfeldes angegeben ist, wobei der erste Operand einen ersten Vektorwert gefolgt von einem zweiten Vektorwert aufweist; die Ausführungseinheit umfasst ferner ein zweites Eingangsregister, das angeschlossen ist, einen zweiten Operanden zu empfangen, der durch einen Wert des zweiten Operandenfeldes angegeben ist, wobei der zweite Operand einen dritten Vektorwert gefolgt von einem vierten Vektorwert aufweist. Die Ausführungseinheit umfasst ferner eine Komparatoreinheit, die angeschlossen ist, die ersten Operanden aus dem ersten Eingangsregister und den zweiten Operanden aus dem zweiten Eingangsregister zu empfangen. Die Komparatoreinheit ist angeschlossen, einen decodierten Operationscodierungswert des Operationscodierungsfeldes auf einem Bus für decodierte Operationscodierung zu empfangen. Des weiteren umfasst die Ausführungseinheit einen ersten Multiplexer, der angeschlossen ist, mehrere Eingänge bzw. Eingangssignale mit dem ersten Vektorwert aus dem ersten Eingangsregister, dem dritten Vektorwert aus dem zweiten Eingangsregister, einen ersten konstanten Wert und einen zweiten konstanten Wert zu empfangen. Der erste Multiplexer ist ausgebildet, eines der mehreren ersten Eingangssignale auszuwählen, das als ein erster Teil eines Vektoregebnisses des ersten Befehls in Reaktion darauf zu ermitteln ist, dass ein erster Satz aus Steuersignalwerten aus der Komparatoreinheit empfangen wird. Die Komparatoreinheit ist ferner ausgebildet, den ersten Satz aus Steuersignalwerten in Reaktion auf das Empfangen des decodierten Operationscodierungswertes zu erzeugen. Wenn der decodierte Operationscodierungswert angibt, dass der erste Befehl einer von mehreren Extremwertbefehlen ist, ist der erste Satz aus Steuersignalwerten verwendbar, um den ersten Vektorwert oder den dritten Vektorwert als den ersten Teil des Vektoregebnisses auszuwählen. Wenn der decodierte Operationscodierungswert angibt, dass der erste Befehl einer von mehreren Vergleichsbefehlen ist, ist der erste Satz aus Steuersignalwerten verwendbar, um den ersten konstanten Wert oder den zweiten konstanten Wert als den ersten Teil bzw. Bereich des Vektoregebnisses auszuwählen.

[0013] Die vorliegende Erfindung betrifft ferner ein Verfahren zum Ausführen eines ersten Befehls in einer Ausführungseinheit eines Mikroprozessors, wobei eine codierte Darstellung des ersten Befehls ein Operationscodierungsfeld, ein erstes Operandenfeld und ein zweites Operandenfeld aufweist. Das Verfahren umfasst das Übermitteln mehrerer erster Eingangssignale an eine Komparatoreinheit innerhalb der Ausführungseinheit, wobei die mehreren ersten Eingangssignale bzw. Eingänge einen ersten Operanden, der durch einen Wert des ersten Operandenfeldes angegeben ist, einen zweiten Operanden, der durch einen Wert des zweiten Operandenfeldes angegeben ist, und einen decodierten Operationscodierungswert enthalten, der einem codierten Operationscodierungswert des Operationscodierungsfeldes entspricht. Das Verfahren umfasst ferner das Übermitteln mehrerer zweiter Eingangssignale an einen Multiplexer innerhalb der Ausführungseinheit, wobei die mehreren zweiten Eingangssignale den ersten Operanden, den zweiten Operanden, einen ersten konstanten Wert und einen zweiten konstanten Wert enthalten. Des weiteren umfasst das Verfahren das Erzeugen eines Satzes aus Steuersignalwerten von der Komparatoreinheit in Reaktion auf das Empfangen der mehreren ersten Eingangssignale. Das Verfahren umfasst des weiteren das Übermitteln eines der mehreren zweiten Eingangssignale aus dem Multiplexer als Ergebnis des ersten Befehls in Reaktion auf das Empfangen des Satzes aus Steuersignalwerten. Das Ergebnis wird aus dem ersten Operanden und dem zweiten Operanden gemäß dem Satz aus Steuersignalwerten ausgewählt, wenn der decodierte Operationscodierungswert angibt, dass der erste Befehl einem von mehreren Extremwertbefehlen entspricht. Das Ergebnis wird aus dem ersten konstanten Wert und dem zweiten konstanten Wert gemäß dem Satz aus Steuersignalwerten ausgewählt, wenn der decodierte Operationscodierungswert angibt, dass der erste Befehl einen von mehreren Vergleichsbefehlen entspricht.

[0014] Schließlich betrifft die vorliegende Erfindung einen Mikroprozessor, der ausgebildet ist, einen ersten Befehl auszuführen. Der Mikroprozessor umfasst einen Befehls cachespeicher, der ausgebildet ist, eine codier-

te Darstellung des ersten Befehls zu speichern, wobei die codierte Darstellung ein Operationscodierungsfeld, ein erstes Operandenfeld und ein zweites Operandenfeld aufweist. Der Mikroprozessor umfasst ferner eine Decodiereinheit, die angeschlossen ist, die decodierte Darstellung des ersten Befehls aus dem Befehlspeicher zu empfangen, wobei die Decodiereinheit ausgebildet ist, einen decodierten Operationscodierungswert in Reaktion auf das Empfangen eines Wertes des Operationscodierungsfeldes zu erzeugen. Der Mikroprozessor umfasst ferner eine Ausführungseinheit, die mit der Decodiereinheit verbunden ist, wobei die Decodiereinheit ferner ausgebildet ist, zu veranlassen, dass ein erster Operand und ein zweiter Operand an die Ausführungseinheit übermittelt werden. Der erste Operand ist durch einen Wert des ersten Operandenfeldes spezifiziert, während der zweite Operand durch einen Wert des zweiten Operandenfeldes spezifiziert ist.

[0015] Die Ausführungseinheit umfasst ein erstes Eingangsregister, das angeschlossen ist, einen ersten Operanden, der durch einen Wert des ersten Operandenfeldes angegeben ist, zu empfangen; die Ausführungseinheit umfasst ferner ein zweites Eingangsregister, das angeschlossen ist, einen zweiten Operanden zu empfangen, der durch einen Wert des zweiten Operandenfeldes angegeben ist. Die Ausführungseinheit umfasst ferner eine Komparatoreinheit, die angeschlossen ist, den ersten Operanden aus dem ersten Eingangsregister und dem zweiten Operanden aus dem zweiten Eingangsregister zu empfangen. Die Komparatoreinheit ist ferner angeschlossen, einen decodierten Operationscodierungswert des Operationscodierungsfeldes auf einem Bus für die decodierte Operationscodierungen zu empfangen. Die Ausführungseinheit umfasst ferner einen Multiplexer, der angeschlossen ist, mehrere Eingänge bzw. Eingangssignale zu empfangen, die den ersten Operanden aus dem ersten Eingangsregister, den zweiten Operanden aus dem zweiten Eingangsregister, einen ersten konstanten Wert und einen zweiten konstanten Wert aufweisen. Der Multiplexer ist ausgebildet, eines der mehreren Eingangssignale auszuwählen, so dass dieses als Ergebnis des ersten Befehls in Reaktion darauf weitergeleitet wird, dass ein Steuersignalwert aus der Komparatoreinheit empfangen wird. Die Komparatoreinheit ist ausgebildet, den Steuersignalwert in Reaktion auf das Empfangen des decodierten Operationscodierungswertes zu erzeugen. Wenn der decodierte Operationscodierungswert angibt, dass der erste Befehl einer von mehreren Extremwertbefehlen ist, ist der Steuersignalwert verwendbar, um den ersten Operanden oder den zweiten Operanden als den Ausgangswert auszuwählen. Wenn der decodierte Operationscodierungswert angibt, dass der erste Befehl einer von mehreren Vergleichsbefehlen ist, ist das Steuersignal verwendbar, um den ersten konstanten Wert oder den zweiten konstanten Wert als den Ausgabewert auszuwählen.

Kurze Beschreibung der Zeichnungen

[0016] Weitere Aufgaben und Vorteile der Erfindung gehen aus dem Studium der folgenden detaillierten Beschreibung bei Bezugnahme zu den begleitenden Zeichnungen hervor, in denen:

[0017] [Fig. 1](#) eine Blockansicht eines Mikroprozessors ist;

[0018] [Fig. 2a](#) bis [Fig. 2c](#) das Format und die Funktion eines Minimalwertbefehls gemäß einer Ausführungsform der vorliegenden Erfindung zeigen;

[0019] [Fig. 3a](#) bis [Fig. 3c](#) das Format und die Funktion eines Maximalwertbefehls, gemäß einer Ausführungsform der vorliegenden Erfindung zeigen;

[0020] [Fig. 4a](#) bis [Fig. 4c](#) das Format und die Funktion eines Vergleichsbefehls auf Gleichheit gemäß einer Ausführungsform der vorliegenden Erfindung zeigen;

[0021] [Fig. 5a](#) bis [Fig. 5c](#) das Format und die Funktion eines Vergleichsbefehls „größer als“ gemäß einer Ausführungsform der vorliegenden Erfindung zeigen;

[0022] [Fig. 6a](#) bis [Fig. 6c](#) das Format eines Vergleichsbefehls „größer als oder gleich“ gemäß einer Ausführungsform der Erfindung zeigen;

[0023] [Fig. 7a](#) bis [Fig. 7b](#) das Format und die Operanden darstellen, die in denen in den **Fig. 2** bis **6** gezeigten Befehlen gemäß einer Ausführungsform der vorliegenden Erfindung verwendet werden;

[0024] [Fig. 8](#) eine Blockansicht einer Ausführungseinheit ist, die ausgebildet ist, die in den **Fig. 2** bis **6** gezeigten Befehle gemäß einer Ausführungsform der Erfindung auszuführen; und

[0025] [Fig. 9](#) eine Blockansicht einer Vektorausführungseinheit ist, die ausgebildet ist, die in den **Fig. 2** bis **6**

gezeigten Befehle gemäß einer Ausführungsform der vorliegenden Erfindung auszuführen.

Detaillierte Beschreibung der Erfindung

[0026] **Fig. 1** zeigt eine Blockansicht einer Ausführungsform eines Mikroprozessors **10**. Wie gezeigt, umfasst der Mikroprozessor **10** einen Vordecodierungslogikblock **12**, der mit einem Befehlschachespeicher **14** (der einen Befehls-TLB **16** enthält) verbunden ist. Eine Cache-Steuerung **18** ist mit dem Vordecodierungsblock **12** und dem Befehlschachespeicher **14** sowie einer Busschnittstelleneinheit **24** und einem Datencache-Speicher **26** (der einen Daten-TLB **28** enthält) verbunden. Der Mikroprozessor **10** umfasst ferner eine Decodiereinheit **20**, die Befehle aus dem Befehlschachespeicher **14** empfängt, die an eine Ausführungsmaschine **3** bzw. Ausführungseinheit **30** in Übereinstimmung mit einem Eingangssignal, das von einer Verzweigungslogikeinheit **22** empfangen wird, weitergeleitet werden.

[0027] Die Ausführungsmaschine **30** umfasst einen Ablauforganisationspuffer bzw. einen Disponierpuffer **32**, der angeschlossen ist, Eingangssignale von der Decodiereinheit **20** zu empfangen. Der Disponierpuffer **32** ist angeschlossen, decodierte Befehle zu mehreren Ausführungseinheiten **36A** bis G entsprechend den Eingangssignale, die von einer Befehlssteuereinheit **34** empfangen werden, zu übermitteln. Die Ausführungseinheiten **36A** bis G enthalten eine Ladeeinheit **36A**, eine Speicher- bzw. Schreibeinheit **36B**, eine Integer-X-Einheit bzw. Ganzzahl-X-Einheit **36C**, eine Multimediaeinheit **36D**, eine Integer-Y-Einheit bzw. Ganzzahl-Y-Einheit **36E**, eine Fließkommaeinheit **36F** und eine Verzweigungsauflösungseinheit **36G**. Die Ladeeinheit **36A** empfängt eine Eingabe aus dem Datencachespeicher **26**, während die Schreib- bzw. Speichereinheit **36B** mit dem Datencachespeicher **26** über eine Schreibwarteschlange **38** in Verbindung steht. Blöcke, die hierin mit einem Bezugszeichen gefolgt von Buchstaben bezeichnet sind, werden gemeinsam auch durch das Bezugszeichen alleine benannt. Beispielsweise werden die Ausführungseinheiten **36A** bis **36G** auch als Ausführungseinheiten **36** bezeichnet.

[0028] Allgemein gesagt, ist die Multimediaausführungseinheit **36D** in dem Mikroprozessor **10** ausgebildet, eine effiziente Implementierung von Extremwertbefehlen zu ermöglichen. Wie nachfolgend detaillierter beschrieben ist, verwendet die Ausführungseinheit **36D** einen Hardwareaufbau, der Vergleichsoperationen ausführt, um auch Minimalwertbefehle und Maximalwertbefehle auszuführen. In dieser Weise richtet die Ausführungseinheit **36D** vorteilhafterweise diese Extremwertbefehle als spezielle Befehle in einem einzelnen Taktzyklus ein, wobei das Leistungsverhalten in Anwendungen, etwa dreidimensionalen Graphikerzeugungspresen und Audioverarbeitung verbessert wird.

[0029] In einer Ausführungsform ist der Befehlschachespeicher **14** in Sektoren eingeteilt, wobei jeder Sektor zwei 32-Byte-Cache-Zeilen enthält. Die beiden Cache-Zeilen eines Sektors teilen sich eine gemeinsame Markierung, besitzen jedoch unterschiedliche Zustandsbits, die den Status der Zeile angeben. Daher können zwei Arten an Cache-Nichttreffern (und zugeordnete Cache-Fülloperationen) stattfinden: Sektorenersetzung und Cachezeilenersetzung. Im Falle einer Sektorenersetzung begründet sich der Nichttreffer durch einen Unterschied der Markierung in dem Befehlschachespeicher **14**, wobei die erforderliche Cachezeile von einem externen Speicher über die Busschnittstelleneinheit **24** bereitgestellt wird. Die Cachezeile innerhalb des Sektors, die nicht benötigt wird, wird dann als ungültig markiert. Im Falle einer Cache-Zeilenersetzung stimmt die Markierung mit der angeforderten Adresse überein, jedoch ist die Zeile als ungültig markiert. Die erforderliche Cache-Zeile wird von einem externen Speicher bereitgestellt, aber anders als bei der Sektorenersetzung bleibt die Cache-Zeile innerhalb des Sektors, die nicht angefordert wurde, in dem gleichen Zustand. In alternativen Ausführungsformen können andere Organisationsformen für den Befehlschachespeicher **14** eingesetzt werden, oder es können diverse andere Ersetzungsstrategien verwendet werden.

[0030] Der Mikroprozessor **10** führt das Vorabholen nur in einer Ausführungsform lediglich im Falle der Sektorenersetzungen aus. Während des Sektorenersetzens wird die erforderliche Cache-Zeile gefüllt. Wenn diese erforderliche Cachezeile sich in der ersten Hälfte des Sektors befindet, wird die andere Cachezeile in dem Sektor vorabgeholt. Wenn diese erforderliche Cachezeile in der zweiten Hälfte des Sektors liegt, wird keine Vorabholung ausgeführt. Zu beachten ist, dass andere Vorabholverfahren in unterschiedlichen Ausführungsformen des Mikroprozessors **10** eingesetzt werden können.

[0031] Wenn Cache-Zeilen aus Befehlsdaten von dem externen Speicher mittels der Busschnittstelleneinheit **24** ausgelesen werden, werden diese Daten dem Vordecodierungslogikblock **12** zugeführt. In einer Ausführungsform sind die von dem Mikroprozessor **10** verarbeiteten und in dem Cachespeicher **14** gespeicherten Daten in ihrer Länge variabel (beispielsweise der x86-Befehlssatz). Da insbesondere das Decodieren von Befehlen mit variabler Länge besonders komplex ist, ist die Vordecodierlogik **12** ausgebildet, zusätzliche Informati-

onen bereitzustellen, die in dem Befehls-Cachespeicher **14** zur Unterstützung der Decodierung zu speichern sind. In einer Ausführungsform erzeugt die Vordecodierlogik **12** Vordecodierbits für jedes Byte in dem Befehls-Cachespeicher **14**, die die Anzahl an Bytes bis zum Beginn des nächsten Befehls mit variabler Länge kennzeichnen. Diese Vordecodierbits werden der Decodiereinheit **20** zugeführt, wenn Befehlsbytes von dem Cachespeicher **14** angefordert werden.

[0032] Der Befehls-Cachespeicher **14** ist als ein 32-K-Byte-Zwei-Wege-teilassoziativer Cachespeicher mit Rückschreibung in einer Ausführungsform des Mikroprozessors **10** eingerichtet. Die Cache-Zeilengröße beträgt in dieser Ausführungsform 32-Bytes. Der Cachespeicher **14** enthält ferner einen TLD mit 64 Einträgen, der zum Übersetzen von linearen Adressen in physikalische Adressen verwendet wird. In anderen Ausführungsformen sind viele andere Variationen eines Befehls-Cachespeichers **14** möglich.

[0033] Von der Cachesteuerung **18** werden Befehlsabholadressen dem Befehls-Cachespeicher **14** zugeführt. In einer Ausführungsform können bis zu 16 Bytes pro Taktzyklus aus dem Cache-Speicher **14** abgerufen werden. Die abgerufene Information wird in einem Befehlspuffer abgelegt, der die Decodiereinheit **20** speist. In einer Ausführungsform des Mikroprozessors **10** kann das Abholen zusammen mit einem einzelnen Ausführungsstrom mit sieben anhängigen durchgeführten Verzweigungen vorstatten gehen.

[0034] In einer Ausführungsform ist die Befehlsabhollogik innerhalb der Cachesteuerung **18** ausgebildet, beliebige **16** zusammenhängende Befehlsbytes innerhalb einer 32-Byte-Grenze des Cachespeichers **14** abzurufen. Es gibt keine weiteren Beeinträchtigungen, wenn die 16 Bytes eine Cache-Zeilengrenze überschreiben. Befehle werden in den Befehlspuffer geladen, wenn die aktuellen Befehle von der Decodiereinheit **20** aufgenommen werden. In anderen Ausführungsformen sind andere Konfigurationen der Cachesteuerung **18** möglich.

[0035] Die Decodierlogik **20** ist ausgebildet, mehrere Befehle pro Prozessortaktzyklus zu decodieren. In einer Ausführungsform erhält die Decodiereinheit **20** Befehls- und Vordecodierbytes aus dem Befehlspuffer (in x86-Format), erkennt tatsächliche Befehls Grenzen und erzeugt entsprechende „Risc-Operationen“. Risc-Operationen sind interne Befehle mit festgelegtem Format, wovon die meisten von den Mikroprozessor **10** in einem einzelnen Taktzyklus ausführbar sind. Risc-Operationen werden kombiniert, um in einer Ausführungsform des Mikroprozessors **10** jede Funktion des x86-Befehlssatzes zu bilden.

[0036] Der Mikroprozessor **10** verwendet eine Kombination aus Decodieren, um die x86-Befehle in Risc-Operationen umzuwandeln. Die Hardware umfasst drei Mengen an Decodieren: zwei parallele Kurzdecodierer, einen Langdecodierer und einen vektorisierenden Decodierer. Die parallelen Kurzdecodierer übersetzen die meisten typischerweise verwendeten x86-Befehle (Übertragen (Move), Schiebeoperationen, Verzweigungen, etc.) in 0, 1 oder 2 Risc-Operationen. Die Kurzdecodierer operieren lediglich auf x86-Befehlen, die bis zu 7 Bytes lang sind. Des Weiteren sind sie ausgebildet, bis zu zwei x86-Befehle pro Taktzyklus zu decodieren. Die häufig verwendeten x86-Befehle, die größer als 7 Bytes sind, sowie die meisten relativ häufig verwendeten Befehle, die bis zu 7 Bytes lang sind, werden von dem Langdecodierer verarbeitet.

[0037] Der Langdecodierer in der Decodiereinheit **20** führt lediglich eine Decodierung pro Taktzyklus aus und erzeugt bis zu 4 Risc-Operationen. Alle anderen Übersetzungen (komplexe Befehle, Interrupts, etc.) werden aus einer Kombination des Vektordecodierers und Risc-Operationssequenzen, die von einem chipinternen ROM abgeholt werden, verarbeitet. Für komplexe Operationen liefert die Vektordecodierlogik den ersten Satz aus Risc-Operationen und eine Anfangsadresse zu einer Sequenz weiterer Risc-Operationen. Die Risc-Operationen, die aus dem chipinternen ROM abgeholt werden, sind von der gleichen Art, wie sie von dem Hardwaredecodieren erzeugt werden.

[0038] In einer Ausführungsform erzeugt die Decodiereinheit **20** eine Gruppe aus vier Risc-Operationen bei jedem Taktzyklus. Für Taktzyklen, in denen 4 Risc-Operationen nicht erzeugt werden können, ordnet die Decodiereinheit **20** Risc-NOP-Operationen in dem verbleibenden Zeitfenstern beim Gruppieren an. Diese Einteilungen von Risc-Operationen (und möglicherweise von NOP-Operationen) werden dann einem Disponierpuffer **32** zugeführt.

[0039] Es ist zu beachten, dass in einer weiteren Ausführungsform ein Befehlsformat, das nicht dem x86 entspricht, in dem Befehls-Cache-Speicher **14** gespeichert und nachfolgend von der Decodiereinheit **20** decodiert werden kann.

[0040] Die Befehlssteuerlogik **34** enthält die notwendige Logikschaltung, um eine Ausführung „außerhalb der

Reihenfolge" von Befehlen zu verwalten, die in dem Disponierpuffer **32** gespeichert sind. Die Befehlssteuerlogik **34** verwaltet ferner die Datenweiterleitung, die Registerumbenennung, die gleichzeitig Ausgabe und den Abschluss von Risc-Operationen und die spekulative Ausführung. In einer Ausführungsform hält der Disponierpuffer **32** bis zu 24 Risc-Operationen gleichzeitig, was sich auf maximal 12 x86-Befehle beläuft. Wenn möglich kann die Befehlssteuerlogik **34** gleichzeitig eine Risc-Operation (aus dem Puffer **32**) zu einer beliebigen verfügbaren Ausführungseinheit **36** ausgeben. Insgesamt kann die Steuerlogik **34** zu 6 Risc-Operationen pro Taktzyklus in einer Ausführungsform ausgeben und bis zu 4 Risc-Operationen abschließen.

[0041] In einer Ausführungsform enthält der Mikroprozessor **10** sieben Ausführungseinheiten (**36A** bis G). Die Schreibereinheit **36A** und die Ladeeinheit **36B** sind in einer zweistufigen Pipeline-Ausführung vorgesehen. Die Schreibereinheit **36A** führt Schreiboperationen für den Datenspeicher und die Register durch, die nach einem Taktzyklus für eine Ladeoperation verfügbar sind. Die Ladeeinheit **36** führt das Auslesen des Speichers durch. Die Daten aus diesen Leseoperationen sind nach zwei Taktzyklen verfügbar. Die Lade- und Speichereinheit sind auch in anderen Ausführungsformen mit unterschiedlichen Wartezeiten möglich.

[0042] Die Ausführungseinheit **36C** (Integer-x-Einheit) ist eine Festkomma-Ausführungseinheit, die ausgebildet ist, auf allen Alu-Operationen, sowie Multiplikationen, Divisionen (sowohl vorzeichenbehaftet oder als auch ohne Vorzeichen), Schiebeoperationen und Rotationsoperationen zu operieren. Im Gegensatz dazu ist die Ausführungseinheit **36E** (Integer-Y-Einheit) eine Festkomma-Ausführungseinheit, die ausgebildet ist, auf den Basiswort- und Doppelwort-Alu-Operationen (ADD, AND, CMP, etc.) zu operieren.

[0043] Die Ausführungseinheit **36D** (Multimediaeinheit) ist ausgebildet, das Funktionsverhalten von Software zu beschleunigen, die unter Anwendung von Multimediabefehlen erstellt wurde. Anwendungen, die vorteilhafteweise Multimediabefehle ausnutzen können, sind Graphik, Video- und Audiokomprimierung und Dekomprimierung, Spracherkennung und Telefonanwendungen. Die Ausführungseinheit **36D** ist in einer Ausführungsform ausgebildet, Multimediabefehle in einem einzelnen Taktzyklus auszuführen. Viele dieser Befehle sind so gestaltet, dass sie die gleiche Operation an mehreren Datensätzen gleichzeitig ausführen (Vektorverarbeitung). In einer Ausführungsform verwendet die Multimediaeinheit **36D** Register, die auf den Stapel der Fließkommaeinheit **36F** abgebildet werden.

[0044] Die Ausführungseinheit **36F** enthält eine IEEE 754-kompatible Fließkommaeinheit, die so gestaltet ist, die Geschwindigkeit von Software zu beschleunigen, die dem x86-Befehlssatz verwendet. Fließkomma-Software wird typischerweise zur Manipulierung von Zahlen verwendet, die entweder sehr groß oder sehr klein sind, ein hohes Maß an Genauigkeit erfordern, oder sich aus komplexen mathematischen Operationen, etwa transzendenten Zahlen, ergeben. Fließkomma-Einheiten umfassen eine Addiereinheit, eine Multipliziereinheit und eine Dividier/Quadratwurzeleinheit. In einer Ausführungsform sind diese Einheiten mit geringer Verarbeitungszeit ausgebildet, Fließkomma-Befehle mit nur bis zu 2 Taktzyklen auszuführen.

[0045] Die Ausführungseinheit **36G** (die Verzweigungsauflösungseinheit) ist von der Verzweigungsvorhersagelogik **22** getrennt, dahingehend, dass diese bedingte Verzweigungen, etwa JCC und LOOP auflöst, nachdem die Verzweigungsbedingung bewertet wurde. Die Verzweigungsauflösungseinheit **36G** erlaubt eine effiziente spekulative Ausführung, wodurch der Mikroprozessor **10** in der Lage ist, Befehle hinter unbedingten Verzweigungen auszuführen, bevor bekannt ist, ob die Verzweigungsvorhersage korrekt war. Wie zuvor beschrieben ist, ist der Mikroprozessor **10** ausgebildet, in einer Ausführungsform bis zu 7 anhängige Verzweigungen zu handhaben.

[0046] Die Verzweigungsvorhersagelogik **22**, die mit der Decodiereinheit **20** verbunden ist, ist ausgebildet, die Genauigkeit zu erhöhen, mit der bedingte Verzweigungen in dem Mikroprozessor **10** vorhergesagt werden. 10 bis 20% der Befehle in typischen Anwendungen beinhalten bedingte Verzweigungen. Die Verzweigungsvorhersagelogik **22** ist ausgebildet, diese Art an Programmverhalten und dessen negativen Auswirkungen auf die Befehlsausführung, etwa die Unterbrechung auf Grund verzögerter Befehlsabholung, zu handhaben. In einer Ausführungsform umfasst die Verzweigungsvorhersagelogik **22** eine Verzweigungsgeschichtetabelle mit 8192-Einträgen, einen Verzweigungszielcachespeicher mit 16×16 Byte-Einträgen und einen Rücksprungadressenstapel mit 16 Einträgen.

[0047] Die Verzweigungsvorhersagelogik **22** bietet einen adaptiven Zweiebenengeschichtsalgorithmus unter Anwendung der Verzweigungsgeschichtetabelle. Diese Tabelle enthält Information über ausgeführte Verzweigungen, sagt einzelne Verzweigungen voraus und sagt das Verhalten von Gruppen von Verzweigungen voraus. In einer Ausführungsform enthält die Verzweigungsgeschichtetabelle keine vorausgesagten Zieladressen, um Speicherplatz zu sparen. Diese Adressen werden stattdessen während des Decodierschritts neben-

her berechnet.

[0048] Um eine Beeinträchtigung im Hinblick auf Taktzyklen bei der Cachespeicherabholung zu vermeiden, wenn eine Verzweigung als vorhergesagt genommen wird, liefert ein Verzweigungszielcachespeicher innerhalb der Verzweigungslogik **22** die ersten 16 Bytes an dieser Adresse direkt an den Befehlspeicher (wenn ein Treffer in dem Verzweigungszielcachespeicher auftritt). In einer Ausführungsform erreicht diese Verzweigungsvorhersage Logik Verzweigungsvorhersageraten von über 95%.

[0049] Die Verzweigungslogik **22** umfasst ferner eine spezielle Schaltung, die so gestaltet ist, um CALL- und RET-Befehle zu optimieren. Diese Schaltung ermöglicht es, dass die Adresse eines auf den CALL-Befehl im Speicher folgenden Befehls auf einen Rücksprungadressenstapel geschoben wird. Wenn der Mikroprozessor **10** einen RET-Befehl antrifft, holt die Verzweigungslogik **22** diese Adresse aus dem Rücksprungstapel ab und beginnt mit der Abholung.

[0050] Wie der Befehlspeicher **14** ist auch der Datencachespeicher **26** als ein Zweigeiteilassoziativer 32-K-Byte-Speicher organisiert. In einer Ausführungsform enthält der Daten-TLB **28** 128 Einträge, die zum Übersetzen von linearen Adressen in physikalische Adressen verwendet werden. Wie der Befehlspeicher **14** ist auch der Datencachespeicher **26** in Sektoren eingeteilt. Der Datencachespeicher **26** besitzt ein MESI (modifizier-exklusivgemeinsam-ungültig) Protokoll, um den Cache-Zeilenzustand zu überwachen, obwohl auch andere Variationen möglich sind.

[0051] [Fig. 2a](#) zeigt das Format eines Fließkommaminalwertbefehls („PFMIN“) **100** gemäß einer Ausführungsform der Erfindung. Wie dargestellt, enthält der PFMIN-Befehl **100** einen Operationscodierungswert **101** und zwei Operanden, d. h. ein erstes Operandenfeld **102a** und ein zweites Operandenfeld **102b**. Der von dem ersten Operandenfeld **102a** spezifizierte Wert ist so gezeigt, dass dieser „mmreg1“ ist, was in einer Ausführungsform eines der Register auf dem Stapel der Fließkommaausführungseinheit **36F** ist. Der von dem zweiten Operandenfeld **102** angegebene Wert ist als ein weiteres der Fließkommastapelregister oder als eine Speicherstelle gezeigt. In einer weiteren Ausführungsform spezifiziert das zweite Operandenfeld **102b** einen unmittelbaren Wert.

[0052] In einer Ausführungsform spezifiziert der Befehl **100** (und andere Befehle, die nachfolgend mit Bezug zu den [Fig. 3a](#), [Fig. 4a](#), [Fig. 5a](#) und [Fig. 6a](#) beschrieben sind), Operanden (etwa die durch die Operandenfelder **102** angegebenen Werte), die mehr als einen einzigen unabhängigen Wert innerhalb eines vorgegebenen Register aufweisen, das als ein Operand angegeben ist. D. h. Register, etwa mmreg1, das in [Fig. 2a](#) angegeben ist, sind Vektorregister.

[0053] Das Format eines derartigen Registers **600** ist in [Fig. 7a](#) gezeigt. Das Register **600** enthält zwei separate Vektorgrößen, d. h. einen ersten Vektorwert **602a** und einen zweiten Vektorwert **602b**. In einer Ausführungsform sind alle Fließkommaregister in der Ausführungseinheit **36F**, auf die von dem Befehl **100** und von anderen Befehlen, wie sie hierin beschrieben sind, zugegriffen werden kann, in ähnlicher Weise organisiert. Die Vektorwerte **602** enthalten jeweils in einer Ausführungsform eine 32-Bit-Fließkommazahl mit einfacher Genauigkeit. In anderen Ausführungsformen werden die Vektorwerte **602** in einer anderen numerischen Darstellung, etwa einem Fest-Kommaformat gespeichert.

[0054] Das Format der Werte mit einfacher Genauigkeit, die in den Vektorwerten **603** gespeichert sind, ist in [Fig. 7b](#) gezeigt. Wie gezeigt, enthält das Format **610**, das dem IEEE-Fließkommaformat entspricht, ein Vorzeichenbit **612** (S), einen Exponentenwert **614** (E) und einen signifikanten Wert **616** (F). Der Wert einer Zahl V, die in dem Format **610** dargestellt ist, kann somit dargestellt werden als:

$$V = (-1)^S \cdot 2^{E-\text{bias}} \cdot (1.F).$$

[0055] In anderen Ausführungsformen sind andere Fließkommaformate für die Vektorwerte **602** möglich.

[0056] [Fig. 2b](#) ist eine Darstellung eine Pseudocodierung zur Veranschaulichung der Funktion für den PFMIN-Befehl **100** gezeigt. Wie gezeigt, wird beim Ausführen des PFMIN-Befehls **100** ein Vergleich eines ersten Vektorteils (etwa dem Wert **602**) des Wertes, der von dem ersten Operandenfeld **102A** angegeben ist, mit einem ersten Vektorbereich des zweiten Operanden **102B** ausgeführt. Gleichzeitig wird ein Vergleich eines zweiten Vektorbereichs (etwa Wert **602B**) des Wertes, der von dem ersten Operandenfeld **102A** angegeben ist, mit einem zweiten Vektorbereich des Wertes, der von dem zweiten Operandenfeld **102B** angegeben ist, ausgeführt.

[0057] Wenn der erste Vektorbereich des von dem ersten Operandenfeld **102A** angegebenen Wertes als kleiner als der erste Vektorbereich des Wertes, der von dem zweiten Operandenfeld **102B** angegeben ist, erkannt wird, wird der Wert des ersten Vektorbereichs des Wertes, der von dem ersten Operandenfeld **102A** spezifiziert ist, als ein erster Bereich bzw. Teil eines Ergebnisses des Befehls **100** weitergegeben. Ansonsten wird der Wert des ersten Vektorteils eines Wertes, der von einem zweiten Operandenfeld **102B** angegeben ist, als der erste Vektorteil des Ergebnisses des Befehls **100** weitergegeben. Wenn in ähnlicher Weise der zweite Vektorteil des Ergebnisses, der von dem ersten Operandenfeld **102A** spezifiziert wird, als kleiner als der zweite Vektorteil des Wertes, der von dem zweiten Operandenfeld **102B** angegeben ist, erkannt wird, wird der Wert des zweiten Vektorteils des Wertes, der von dem ersten Operandenfeld **102A** spezifiziert ist, als ein zweiter Teil eines Ergebnisses des Befehls **100** weitergegeben. Ansonsten wird der Wert des zweiten Vektorteils des Wertes, der von dem zweiten Operandenfeld **102B** spezifiziert ist, als der zweite Vektorteil des Ergebnisses des Befehls **100** weitergegeben. Dieses Sequenz aus Operationen bewirkt das Ausführen der Minimalwertfunktion. [Fig. 2c](#) zeigt ist eine Tabelle, die die Ausgabe des Befehls **100** bei diversen Eingaben zeigt, wobei Fälle eingeschlossen sind, in denen die Operanden **102** Null sind oder in einem nicht unterstützten Format vorliegen.

[0058] Das Ergebnis (der erste und der zweite Vektorteil) des Befehls **100** wird nachfolgend in das Register mmreg1 innerhalb der Fließkommaausführungseinheit **36F** geschrieben. In einer weiteren Ausführungsform des Befehls **100** kann der Ergebniswert in mmreg2, einer Speicherstelle oder einem dritten Register abgelegt werden, das durch einen weiteren Operanden spezifiziert ist. Zu beachten ist, dass in anderen Ausführungsformen des Befehls **100** die Operanden nicht vektorisiert sind und damit lediglich einen einzelnen Wert enthalten. Des weiteren ist zu beachten, dass in noch weiteren Ausführungsformen der Operanden **102** diese Werte zusätzlich Vektorwerte über die beiden in [Fig. 7a](#) gezeigten Vektorwerte hinausgehend aufweisen können.

[0059] [Fig. 3a](#) zeigt das Format eines Fließkommaximalwertbefehls („PFMAX“) **200** gemäß einer Ausführungsform der Erfindung. Das Format des PFMAX-Befehls **200** ist ähnlich zu dem Format, wie es zuvor für den PFMIN-Befehl **100** beschrieben ist. Wie gezeigt, enthält der PFMAX-Befehl **200** einen Operationscodierungswert **201** und zwei Operanden, d.h. ein erstes Operandenfeld **202a** und ein zweites Operandenfeld **202b**. Der Wert, der von dem ersten Operandenfeld **202a** spezifiziert ist, ist als „mmreg1“ gezeigt, das in einer Ausführungsform eines der Register auf dem Stapel der Fließkommaausführungseinheit **36F** ist. Der von dem zweiten Operandenfeld **202B** spezifizierte Wert ist als ein weiteres der Fließkommastapelregister oder als eine Speicherstelle gezeigt. In einer weiteren Ausführungsform spezifiziert das zweite Operandenfeld **202b** einen unmittelbaren Wert.

[0060] [Fig. 3b](#) zeigt eine Pseudocodierung, die die Operation des PFMAX-Befehls **200** darstellt. Wie gezeigt, wird beim Ausführen diese PFMAX-Befehls **200** ein Vergleich eines ersten Vektorteils (etwa der Wert **602A**) des von dem ersten Operandenfeld **202A** spezifizierten Wertes mit einem ersten Vektorteil des Wertes des zweiten Operanden **202B** ausgeführt. Gleichzeitig wird ein Vergleich eines zweiten Vektorteils (etwa der Wert **602B**) des von dem ersten Operandenfeld **202A** spezifizierten Wertes mit einem zweiten Vektorteil des von dem zweiten Operandenfeld **202B** spezifizierten Wertes ausgeführt.

[0061] Wenn der erste Vektorteil des von dem ersten Operandenfeld **202A** spezifizierten Wertes als größer als der erste Vektorteil des von dem zweiten Operandenfeld **202B** spezifizierten Wertes erkannt wird, wird der Wert des ersten Vektorteils des von dem ersten Operandenfeld **202A** angegebenen Wertes als ein erster Teil eines Ergebnisses des Befehls **200** weitergegeben. Ansonsten wird der Wert des ersten Vektorteils des Wertes, der von dem zweiten Operandenfeld **202** spezifiziert wird, als der erste Vektorteil des Ergebnisses des Befehls **200** ausgegeben. Wenn in ähnlicher Weise der zweite Vektorteil des von dem ersten Operandenfeld **202** spezifizierten Wertes als größer als der zweite Vektorteil des Wertes, der von dem zweiten Operandenfeld **202B** spezifiziert wird, erkannt wird, wird der zweite Wert des zweiten Vektorteils des von dem ersten Operandenfeld **202A** spezifizierten Wertes als ein zweiter Teil eines Ergebnisses des Befehls **200** weitergegeben. Ansonsten wird der Wert des zweiten Vektorteils des von dem zweiten Operandenfeld **202B** angegebenen Wertes als der zweite Vektorteil des Ergebnisses des Befehls **200** verwendet. Die Abfolge an Operationen bewirkt das Ausführen einer Maximalwertfunktion. [Fig. 3c](#) ist eine Tabelle, die die Ausgabe des Befehls **200** bei diversen Eingaben zeigt, wozu Fälle gehören, in denen die Operanden **202** Null sind oder in nicht unterstützten Formaten vorliegen.

[0062] Das Ergebnis (sowohl der erste als auch der zweite Vektorteil) des Befehls **200** wird nachfolgend in das Register mmreg1 innerhalb der Fließkommaausführungseinheit **36F** geschrieben. In einer weiteren Ausführungsform des Befehls **200** kann der Ergebniswert in mmreg2, einer Speicherstelle oder einem dritten Register abgelegt werden, das durch einen zusätzlichen Operanden spezifiziert ist. Zu beachten ist, dass in anderen Ausführungsformen des Befehls **200** die Operanden nicht vektorisiert sind, und damit lediglich einen ein-

zelen Wert enthalten. Des weiteren ist zu beachten, dass in noch weiteren Ausführungsformen von Operanden **202** diese Werte weitere Vektorwerte enthalten können, die über die beiden in [Fig. 7a](#) gezeigten Vektorwerte hinausgehen.

[0063] In [Fig. 4a](#) ist das Format eines Fließkommagleichheitsvergleichsbefehls („PFCMPEQ“) **300** gemäß einer Ausführungsform der Erfindung gezeigt. Das Format des PFCMPEQ-Befehls **300** ist ähnlich zu jenem Format, das für die Befehle **100** und **200** beschrieben ist. Wie gezeigt, enthält der PFCMPEQ-Befehl **300** einen Operationscodierungswert **301** und zwei Operanden, d. h. ein erstes Operandenfeld **302A** und ein zweites Operandenfeld **302B**. Der von dem ersten Operandenfeld **302** spezifizierte Wert ist als „mmreg1“ gezeigt, das in einer Ausführungsform eines der Register auf dem Stapel der Fließkommaausführungseinheit **36F** ist. Der von dem zweiten Operandenfeld **302B** angegebene Wert ist als ein weiteres Register der Fließkommastapelregister oder als eine Speicherstelle gezeigt. In einer weiteren Ausführungsform gibt das zweite Operandenfeld **302B** einen unmittelbaren Wert an.

[0064] In [Fig. 4b](#) ist eine Pseudocodierung gezeigt, die die Operation des PFCMPEQ-Befehls **300** zeigt. Wie gezeigt, wird beim Ausführen des PFCMPEQ-Befehls **300** ein Vergleich eines ersten Vektorteils (etwa des Wertes **602A**) des von dem ersten Operandenfeld **302** spezifizierten Wertes mit einem ersten Vektorteil des Wertes des zweiten Operandenfeldes **302B** ausgeführt. Gleichzeitig wird auch ein Vergleich des zweiten Vektorteils (etwa des Wertes **602B**) des von dem ersten Operandenfeld **302A** spezifizierten Wertes mit einem zweiten Vektorteil des von dem zweiten Operandenfeld **302B** spezifizierten Wertes ausgeführt.

[0065] Wenn der erste Vektorteil des von dem ersten Operandenfeld **302A** spezifizierten Wertes als gleich zu dem ersten Vektorteil des von dem zweiten Operandenfeld **302B** spezifizierten Wertes erkannt wird, wird der Wert des ersten Vektorteils des von dem ersten Operandenfeld **302A** angegebenen Wertes als erster Teil eines Ergebnisses des Befehls **300** weitergegeben. Ansonsten wird der Wert des ersten Vektorteils des von dem zweiten Operandenfeld **302B** spezifizierten Wertes als der erste Vektorteil des Ergebnisses des Befehls **300** ausgegeben. Wenn in ähnlicher Weise der zweite Vektorteil des von dem ersten Operandenfeldes **302A** spezifizierten Wertes als gleich zu dem zweiten Vektorteil des von dem zweiten Operandenfeld **302B** spezifizierten Wertes erkannt wird, wird der Wert des zweiten Vektorteils des von dem ersten Operandenfeld **302A** angegebenen Wertes als ein zweiter Teil eines Ergebnisses des Befehls **300** weitergeleitet. Ansonsten wird der Wert des zweiten Vektorteils des von dem zweiten Operandenfeldes **302B** angegebenen Wertes als der zweite Vektorteil des Ergebnisses des Befehls **300** ausgegeben. Diese Sequenz aus Operationen bewirkt das Ausführen einer Vergleichsfunktion für die Gleichheit. [Fig. 4c](#) ist eine Tabelle, die die Ausgabe des Befehls **300** mit diversen Eingaben zeigt, wobei Fälle enthalten sind, in denen die Operanden **302** sind oder in einem nicht unterstützten Format vorliegen.

[0066] Das Ergebnis (sowohl der erste als auch der zweite Vektorteil) des Befehls **300** wird nachfolgend in das Register mmreg1 in der Fließkommaausführungseinheit **36F** geschrieben. In einer weiteren Ausführungsform des Befehls **300** kann der Ergebniswert in mmreg2, einer Speicherstelle oder einem dritten Register, das durch einen weiteren Operanden spezifiziert ist, abgelegt werden. Zu beachten ist, dass in anderen Ausführungsformen des Befehls **300** die Operanden nicht vektorisiert sind und damit nur einen einzelnen Wert enthalten. Des weiteren ist zu beachten, dass in noch weiteren Ausführungsformen der Operanden **302** diese Werte weitere Vektorwerte und damit mehr Vektorwerte als die beiden Vektorwerte aufweisen können, die in [Fig. 7a](#) gezeigt sind.

[0067] In [Fig. 5A](#) ist das Format eines Fließkommavergleichsbefehls für „größer als“ (PFCMPGT“) **400** gemäß einer Ausführungsform der Erfindung gezeigt. Das Format des PFCMPGT-Befehls **400** ist ähnlich zu jenen, die zuvor für die Befehle **100**, **200** und **300** beschrieben sind. Wie gezeigt, enthält der PFCMPGT-Befehl **400** einen Operationscodierungswert **401** und zwei Operanden, d. h. ein erstes Operandenfeld **402A** und ein zweites Operandenfeld **402B**. Der Wert, der von dem ersten Operandenfeld **402A** spezifiziert ist, ist als „mmreg1“ gezeigt, das in einer Ausführungsform eines der Register auf dem Stapel der Fließkommaausführungseinheit **36F** ist. Der von dem zweiten Operandenfeld **402B** spezifizierte Wert ist als ein weiteres der Fließkommastapelregister oder als eine Speicherstelle gezeigt. In einer weiteren Ausführungsform spezifiziert das zweite Operandenfeld **402B** einen unmittelbaren Wert.

[0068] In [Fig. 5b](#) ist eine Pseudocodierung gezeigt, die das Verhalten des PFCMPGT-Befehls **400** darstellt. Wie gezeigt, wird beim Ausführen des PFCMPGT-Befehls **400** ein Vergleich eines ersten Vektorbereichs (etwa eines Wertes **602A**) des von dem ersten Operandenfeld **402A** spezifizierten Wertes mit einem ersten Vektorteil des Wertes des zweiten Operandenfeldes **402B** ausgeführt. Gleichzeitig wird ein Vergleich eines zweiten Vektorteils (etwa des Wertes **602B**) des von dem ersten Operandenfeld **402A** spezifizierten Wertes mit einem

zweiten Vektorteil des von dem zweiten Operandenfeld **402B** spezifizierten Wertes ausgeführt.

[0069] Wenn der erste Vektorteil des von dem ersten Operandenfeld **402** spezifizierten Wertes als größer als der erste Vektorteil des von dem zweiten Operandenfeld **402D** spezifizierten Wertes erkannt wird, wird der Wert des ersten Vektorteils des von dem ersten Operandenfeld **402A** spezifizierten Wertes als ein erster Teil eines Ergebnisses des Befehls **400** verwendet. Ansonsten wird der Wert des ersten Vektorteils des von dem zweiten Operandenfeld **402B** spezifizierten Wertes als der erste Vektorteil des Ergebnisses des Befehls **400** verwendet. Wenn in ähnlicher Weise der zweite Vektorteil des von dem ersten Operandenfeld **402** spezifizierten Wertes als größer als der zweite Vektorteil des von dem zweiten Operandenfeld **402b** spezifizierten Wertes erkannt wird, wird der Wert des zweiten Vektorteils des von dem ersten Operandenfeld **402A** spezifizierten Wertes als ein zweiter Teil eines Ergebnisses des Befehls **400** verwendet. Ansonsten wird der Wert des zweiten Vektorteils des von dem zweiten Operandenfeld **402B** spezifizierten Wertes als der zweite Vektorteil des Ergebnisses des Befehls **400** weitergegeben. Diese Sequenz aus Operationen bewirkt das Ausführen einer Vergleichsfunktion mit „größer als“. [Fig. 5c](#) ist eine Tabelle, die die Ausgabe des Befehls **400** für vorgegebene Eingaben zeigt, wobei Fälle eingeschlossen sind, in denen die Operanden **402** Null sind oder in nicht unterstützten Formaten vorliegen.

[0070] Das Ergebnis (sowohl der erste als auch der zweite Vektorteil) des Befehls **400** wird nachfolgend in das Register mmreg1 innerhalb der Fließkommaausführungseinheit **36F** geschrieben. In einer weiteren Ausführungsform des Befehls **400** wird der Ergebniswert in mmreg2, einer Speicherstelle oder einem dritten Register, das durch einen weiteren Operanden spezifiziert wird, geschrieben. Zu beachten ist, dass in anderen Ausführungsformen des Befehls **400** die Operanden nicht vektorisiert sind und damit nur einen einzelnen Wert enthalten. Des weiteren ist zu beachten, dass in noch weiteren Ausführungsformen der Operanden **402** diese Werte mehr Vektorwerte als die in [Fig. 7a](#) gezeigten zwei Vektorwerte enthalten können.

[0071] [Fig. 6a](#) zeigt einen Fließkommavergleichsbefehl mit „größer als oder gleich“ („PFCMPGE“) **500** gemäß einer Ausführungsform der Erfindung. Das Format des PFCMPGE-Befehls **500** ist ähnlich zu jenen, wie sie zuvor für die Befehle **100**, **200**, **300** und **400** beschrieben sind. Wie gezeigt, enthält der PFCMPGE-Befehl **500** einen Operationscodierungswert **501** und zwei Operandenfelder, d. h. ein erstes Operandenfeld **502A** und ein zweites Operandenfeld **502B**. Der von dem ersten Operandenfeld **502A** spezifizierte Wert ist als „mmreg1“ gezeigt, das in einer Ausführungsform eines der Register auf dem Stapel der Fließkommaausführungseinheit **36F** ist. Der von dem zweiten Operandenfeld **502B** spezifizierte Wert ist als ein weiteres der Fließkommastapelregister oder als eine Speicherstelle gezeigt. In einer weiteren Ausführungsform spezifiziert das zweite Operandenfeld **502B** einen unmittelbaren Wert.

[0072] In [Fig. 6b](#) ist eine Pseudocodierung gezeigt, die den Ablauf des PFCMPGE-Befehls **500** darstellt. Wie gezeigt, wird beim Ausführen des PFCMPGE-Befehls **500** ein Vergleich eines ersten Vektorteils (etwa des Wertes **602A**) des von dem ersten Operandenfeld **502A** spezifizierten Wertes mit einem ersten Vektorteil des Wertes des zweiten Operandenfeldes **502B** ausgeführt. Gleichzeitig wird auch ein Vergleich eines zweiten Vektorteils (etwa des Wertes **602B**) des von dem ersten Operandenfeld **502A** spezifizierten Wertes mit einem zweiten Vektorteil des von dem zweiten Operandenfeld **502B** spezifizierten Wertes ausgeführt.

[0073] Wenn der erste Vektorteil des von dem ersten Operandenfeld **502** spezifizierten Wertes als größer oder gleich zu dem ersten Vektorteil des von dem zweiten Operandenfeld **502B** spezifizierten Wertes erkannt wird, wird der Wert des ersten Vektorteils des von dem ersten Operandenfeld **502A** spezifizierten Wertes als ein erster Teil eines Ergebnisses des Befehls **500** verwendet. Ansonsten wird der Wert des ersten Vektorteils des von dem zweiten Operandenfeld **502B** spezifizierten Wertes als der erste Vektorteil des Ergebnisses des Befehls **500** verwendet. Wenn in ähnlicher Weise der zweite Vektorteil des von dem ersten Operandenfeld **502** spezifizierten Wertes als größer als der zweite Vektorteil des von dem zweiten Operandenfeld **502B** spezifizierten Wertes erkannt wird, wird der Wert des zweiten Vektorteils des von dem ersten Operandenfeld **502A** spezifizierten Wertes als ein zweiter Teil eines Ergebnisses des Befehls **500** verwendet. Ansonsten wird der Wert des zweiten Vektorteils des von dem zweiten Operandenfeld **502B** spezifizierten Wertes als der zweite Vektorteil des Ergebnisses des Befehls **500** verwendet. Diese Sequenz aus Operationen bewirkt das Ausführen einer Vergleichsfunktion „größer als oder gleich“. [Fig. 6c](#) ist eine Tabelle, die die Ausgabe des Befehls **500** bei verschiedenen Eingängen zeigt, wobei Fälle eingeschlossen sind, in denen die Operanden **502** Null sind oder in einem nicht unterstützten Format vorliegen.

[0074] Das Ergebnis (sowohl der erste als auch der zweite Vektorteil) des Befehls **500** wird nachfolgend in das Register mmreg1 in der Fließkommaausführungseinheit **36F** geschrieben. In einer weiteren Ausführungsform des Befehls **500** wird der Ergebniswert in mmreg2, einer Speicherstelle oder einem dritten Register, das

durch einen weiteren Operanden spezifiziert ist, abgelegt. Zu beachten ist, dass in weiteren Ausführungsformen des Befehls **500** die Operanden nicht vektorisiert sind und damit nur einen einzelnen Wert enthalten. Des Weiteren ist zu beachten, dass in noch weiteren Ausführungsformen der Operanden **502** diese Werte weitere Vektorwerte zusätzlich zu den zwei in [Fig. 7a](#) gezeigten Vektorwerten aufweisen können.

[0075] Zu beachten ist, dass weitere Extremwert- und Vergleichsbefehle in anderen Ausführungsformen eingerichtet sein können.

[0076] [Fig. 8](#) ist eine Blockansicht einer Multimediaausführungseinheit **36D** gemäß einer Ausführungsform der Erfindung. Wie gezeigt, enthält die Ausführungseinheit **36D** Eingangsregister **702A**, **702B**. Das Register **702A** ist mit der Fließkommaeinheit **36F** über einen ersten Operandenbus **710A** verbunden. In ähnlicher Weise ist das Register **702B** mit der Fließkommaeinheit **36F** mittels eines zweiten Operandenbusses **701B** verbunden. Der zweite Operandenbus **710B** ist ebenso mit dem Eingangsregister **702B** verbunden, um Operanden über die Decodiereinheit **20** und den Speicher in einer Ausführungsform zu empfangen. Die Ausführungseinheit **36D** umfasst ferner eine Komparatoreinheit **730**, die angeschlossen ist, um decodierte Operationscodierungswerte über den Bus **720** für decodierte Operationscodierungen von der Decodiereinheit **20** zu empfangen. Die Komparatoreinheit ist ferner angeschlossen, um Eingangssignale (die als „A“ und „B“ bezeichnet sind) von den Registerausgabebussen **704A** bis B, die entsprechend mit den Ausgängen der Register **702A** und **702B** verbunden sind, zu empfangen. Die Komparatoreinheit **730** besitzt einen Auswahlbus **732A** als Ausgang.

[0077] Der Auswahlbus **732A** ist mit einem Ausgabe- bzw. Ausgangsmultiplexer **740A** verbunden, um eine der Eingaben für den Multiplexer **740A** als Ausgang auf einem Ergebnisbus **742** auszuwählen. Die Eingaben an den Multiplexer **740A** enthalten einen ersten konstanten Wert **734A**, einen zweiten konstanten Wert **734B** und Registerausgangsbusse **704A** und **704B**. Die Ausgabe des Multiplexers **740A** wird in ein Ausgangsregister **750** und nachfolgend zu einem Ergebnisziel, etwa einem Fließkommaregister innerhalb der Ausführungseinheit **36F** weitergeleitet.

[0078] Decodierte Versionen von Operationscodierungen, die von dem Mikroprozessor **10** erkannt werden, werden von der Decodiereinheit **20** zu der Komparatoreinheit **730** auf den Bus für decodierte Operationscodierungen **720** transportiert. Wenn der Wert auf dem Bus **720** eine Operationscodierung ist, die einer Extremwertfunktion oder eine Vergleichsfunktion entspricht (d. h. Operationscodierungen etwa Operationscodierung **101**, Operationscodierung **201**, etc.), ist die Komparatoreinheit ausgebildet, Signale auf dem Auswahlbus **732A** während des aktuellen Taktzyklus zu setzen. Diese Signale werden verwendet, um die Ausgabe des Multiplexers **740** auszuwählen, wie dies nachfolgend beschrieben ist. Wenn der Wert auf den Bus **720** nicht einer Extremwertfunktion in einer Vergleichsfunktion entspricht, ist die Komparatoreinheit während des aktuellen Taktzyklus inaktiv.

[0079] Gleichzeitig mit dem Zuführen der decodierten Operationscodierungswerte zu der Ausführungseinheit **36D** werden der erste und der zweite Operand zu der Einheit **36D** über die Busse **710A** bzw. **710B** zugeführt.

[0080] Wie gezeigt wird der erste Operand einem Eingangsregister **702A** auf dem Bus **701A** zugeführt. Der erste Operand wird entsprechend dem Wert des ersten Operandenfeldes ausgewählt, wie dies zuvor gezeigt ist. Beispielsweise kann der Wert des ersten Operandenfeldes, wie es in Befehlschachespeicher **14** abgelegt ist, eine spezielle Registerstelle spezifizieren. Eine nachfolgende Logikschaltung in der Decodiereinheit **20** und die Befehlssteuerlogik **34** sind verantwortlich, den in der speziellen Registerstelle gespeicherten Wert auf dem Bus **710A** weiterzuleiten. Der zweite Operand des Befehls wird in ähnlicher Weise auf dem Bus **710B** transportiert und in dem Register **702B** gespeichert. In der in [Fig. 8](#) gezeigten Ausführungsform sind die in den Registern **702** gespeicherten Operanden nicht vektorisiert; d. h., diese enthalten lediglich einen einzelnen unabhängigen Wert. Eine Ausführungseinheit, die vektorisierte Operanden verarbeitet, ist nachfolgend mit Bezug zu [Fig. 9](#) beschrieben.

[0081] In dem gleichen Taktzyklus, in dem der decodierte Operationscodierungswert der Komparatoreinheit **730** auf dem Bus für decodierte Operationscodierungen **720** zugeführt wird, werden auch der erste und der zweite Operand als Vergleichseingangswerte der Einheit **730** auf den Bussen **704A** bis B zugeführt. Die Komparatoreinheit führt einen Vergleich der zwei Operanden aus und übermittelt entsprechend dem decodierten Operationscodierungswert auf dem Bus **720** Werte auf dem Bus **732A**, die einen der Eingänge für den Multiplexer **740A** auswählen. Die Eingänge des Multiplexers **740A** enthalten den ersten konstanten Wert **734A**, den zweiten konstanten Wert **734B** und die Werte des ersten und des zweiten Operanden auf den Bussen **704A**–B. Diese Eingänge sind als „11“ bis „14“ in [Fig. 8](#) bezeichnet.

[0082] Wenn der decodierte Operationscodierungswert eine Vergleichsoperation angibt, werden auf Grund der Werte auf dem Auswahlbus **732A** die Konstanten **734** entsprechend dem Ergebnis der Vergleichsoperation ausgewählt. Diese Konstanten können ein beliebiger Wert sein, der geeignet ist, so dass dieser in ein Register als das Ergebnis eines Vergleichs geschrieben wird. Wenn andererseits der decodierte Operationscodierungswert eine Extremwertfunktion bezeichnet (Minimalwert oder Maximalwert), werden durch die Wert auf dem Auswahlbus **732A** die Operandenwerte auf den Bussen **704** gemäß den Ergebnissen des Vergleichs ausgewählt. Die speziellen Eingangswerte, die von dem Multiplexer **740A** ausgewählt werden, sind in Tabelle 1 für die diversen Vergleichsergebnisse und die zuvor beschriebenen Befehle gezeigt.

	Vergleichen	Ergebnisse	
Operationscodierung	a>b	a=b	a<b
PFCMIN	14	14	13
PFCMAX	13	14	14
PFCMPEQ	12	11	12
PFCMPGT	11	12	12
PFCMPGE	11	11	12

Tabelle 1

[0083] Der Ausgang des Multiplexers **740A** wird auf dem Ergebnisbus **742** zu einem Ausgangsregister **750** als das Ergebnis des Befehls übermittelt, der dem über den Bussen **720** empfangenen decodierten Operationscodierungswert entspricht. Dieses Ergebnis kann zu einer Vielzahl von Zielen weitergeleitet werden. Wie zuvor mit Bezug zu den Befehlen **100**, **200**, **300**, **400** und **500** beschrieben ist, wird das Ergebnis des Befehls in die Registerstelle in der Fließkommaeinheit **36F** geschrieben, die den ersten Operanden enthält. In anderen Ausführungsformen wird das Ergebnis in ein anderes Register geschrieben, das durch einen zusätzlichen Operandenwert spezifiziert ist.

[0084] In [Fig. 9](#) ist eine Multimediaausführungseinheit **37D** gezeigt. Die Ausführungseinheit **37D** ist im Aufbau ähnlich zu der Ausführungseinheit **36D**; daher sind die Logikblöcke und Busse in der Ausführungseinheit **37D**, die ähnlich sind zu jenen in [Fig. 8](#), der Einfachheit halber identisch bezeichnet. Anders als die Ausführungseinheit **36D** ist jedoch die Ausführungseinheit **37D** ausgebildet, effizient Vektoroperanden zu verarbeiten. Die Ausführungseinheit **37D** ist mit der Ausführungseinheit **36D** im Rahmen des Mikroprozessors **10**, wie er in [Fig. 1](#) gezeigt ist, austauschbar.

[0085] Wie dargestellt, enthält die Ausführungseinheit **37D** Eingangsregister **701A**, **701B**. Das Register **701A** ist mit der Fließkommaeinheit **36F** mittels eines ersten Operandenbusses **710A** gekoppelt. In ähnlicher Weise ist das Register **702B** mit der Fließkommaeinheit **36F** mittels eines zweiten Operandenbusses **710B** verbunden. Der zweite Operandenbus **710B** ist ferner mit dem Eingangsregister **701B** verbunden, um in einer Ausführungsform Operanden über die Decodiereinheit **20** und den Speicher zu empfangen. Anders als die Eingangsregister **702**, die in [Fig. 8](#) gezeigt sind, sind die Eingangsregister **701** vektorisiert. Wie gezeigt, enthält das Register **701A** einen ersten Vektorteil **703A** und einen zweiten Vektorteil **703B**. In gleicher Weise enthält das Register **701B** einen ersten Vektorbereich **703C** und einen zweiten Vektorteil **703D**. Die Ausführungseinheit **37D** enthält ferner eine Komparatoreinheit **731**, die ähnlich zu der Komparatoreinheit **730** ist, die in [Fig. 8](#) gezeigt ist, derart, dass die Komparatoreinheit **731** ebenso einen decodierten Operationscodierungswert auf dem Bus **720** empfängt. Die Komparatoreinheit **731** ist jedoch ausgebildet, zwei Vergleichsoperationen gleichzeitig auszuführen und die beiden Sätze aus Auswahlbussignalen gleichzeitig weiterzugeben. Die Einheit **731** empfängt einen ersten Satz aus Signalen (als Eingangssignale „a₁“ und „b₁“ bezeichnet), die den ersten Vektorteilen **703A** und **703C** entsprechen, sowie einen zweiten Satz aus Signalen zu empfangen (die als Eingangssignale „a₂“ und „b₂“ bezeichnet sind), die den zweiten Vektorteilen **703B** und **703D** entsprechen. In Reaktion auf den Vergleich, der für die Eingangssignale a₁ und b₁ ausgeführt wird, stellt die Komparatoreinheit **731** den Auswahlbus **732A** als Ausgang für den Multiplexer **740A** bereit. In ähnlicher Weise stellt der Komparator **731** Werte auf dem Auswahlbus **732B** für den Multiplexer **740B** in Reaktion auf das Ausführen eines Vergleichs für die Eingangssignale, die als a₂ und b₂ bezeichnete sind, bereit.

[0086] Der Auswahlbus **732A** ist mit dem Ausgangsmultiplexer **740** verbunden, um einen der Eingänge für

den Multiplexer **740A** als Ausgabe auf dem Ergebnisbus **743A** auszuwählen. Die Eingänge des Multiplexers **740A** enthalten einen ersten konstanten Wert **734A**, einen zweiten konstanten Wert **743B** und erste Vektorteile **703A** und **703C**. Der Ausgang des Multiplexers **740A** wird auf einem Ergebnisbus **743A** ausgegeben, so dass dieser in einem ersten Vektorteil **752A** eines Ausgangsregisters **751** gespeichert wird.

[0087] In ähnlicher Weise ist der Auswahlbus **732B** in einem Ausgangsmultiplexer **740B** verbunden, um einen Eingang für den Multiplexer **740B** als Ausgang auszuwählen. Eingänge zu dem Multiplexer **740B** enthalten konstante Werte **734** und zweite Vektorteile **703B** und **703D**. Die ausgewählte Ausgabe des Multiplexers **740B** wird auf einem Ergebnisbus **743B** weitergeleitet, um in einem zweiten Vektorbereich **752B** des Ausgangsregisters **751** gespeichert zu werden.

[0088] Der erste und der zweite Vektorteil des Registers **751** werden nachfolgend zu einem Ergebnisziel, etwa einem Fließkommaregister innerhalb der Ausführungseinheit **36F** weitergeleitet.

[0089] Die Extremwertbefehle **100** und **200** sind somit effizient in dem Mikroprozessor **10** eingerichtet. Durch Modifizieren der Komparator- und Multiplexerhardware, die für Vergleichsoperationen verwendet wird, können die Minimalwert- und Maximalwertbefehle **100** und **200** dem Befehlssatz des Mikroprozessors **10** mit nur minimalem Aufwand hinzugefügt werden. Das Leistungsverhalten des Mikroprozessors **10** in Anwendungen, die häufig diese Extremwertfunktionen verwenden, wird damit vorteilhafterweise verbessert.

[0090] Für den Fachmann werden zahlreiche Variationen und Modifizierungen offensichtlich, sobald er im Besitze der vorliegenden Offenbarung ist. Es ist beabsichtigt, dass die folgenden Patentansprüche so interpretiert werden, dass alle derartigen Variationen und Modifizierungen mit eingeschlossen sind.

Patentansprüche

1. Mikroprozessor (**10**) mit:

einem ersten Eingangsregister (**702A**), das angeschlossen ist, einen ersten Operanden zu empfangen, der einem ersten Befehl entspricht;
 einem zweiten Eingangsregister (**702B**), das angeschlossen ist, einen zweiten Operanden zu empfangen, der dem ersten Befehl entspricht;
 einer Komparatoreinheit (**730**), die angeschlossen ist, den ersten und den zweiten Operanden von dem ersten und dem zweiten Eingangsregister zu empfangen, und
 einem Multiplexer (**740A**), der angeschlossen ist, mehrere Eingangswerte (I1–I4), die den ersten Operanden und den zweiten Operanden mit einschließen, zu empfangen, wobei der Multiplexer ausgebildet ist, einen der mehreren Eingangswerte auszuwählen, so dass dieser als Ergebnis des ersten Befehls in Reaktion auf das Empfangen eines oder mehrerer Steuersignale von der Komparatoreinheit weitergeleitet wird;
dadurch gekennzeichnet, dass die Komparatoreinheit ferner ausgebildet ist, einen decodierten Operationscodierungswert entsprechend dem ersten Befehl zu empfangen;
 dass die mehreren Eingangswerte einen ersten konstanten Wert (**734A**) und einen zweiten konstanten Wert (**734B**) umfassen, und dass
 die Komparatoreinheit ausgebildet ist, das eine oder die mehreren Steuersignale in Reaktion auf das Empfangen des decodierten Operationscodierungswerts zu erzeugen, und wobei, wenn der decodierte Operationscodierungswert angibt, dass der erste Befehl einer von mehreren Extremwertbefehlen ist, das eine oder die mehreren Steuersignale verwendbar sind, um den ersten Operanden oder den zweiten Operanden als den Ausgabewert auszuwählen, und wobei, wenn der decodierte Operationscodierungswert angibt, dass der erste Befehl einer von mehreren Vergleichsbefehlen ist, das eine oder die mehreren Steuersignale verwendbar sind, um den ersten konstanten Wert oder den zweiten konstanten Wert als den Ausgabewert auszuwählen.

2. Mikroprozessor nach Anspruch 1, wobei die mehreren Extremwertbefehle einen Minimalwertbefehl und einen Maximalwertbefehl enthalten.

3. Mikroprozessor nach Anspruch 2, wobei der erste Befehl der Minimalwertbefehl ist, und wobei die Komparatoreinheit ausgebildet ist, in Reaktion auf das Empfangen eines decodierten Operationscodierungswertes, der den Minimalwertbefehl kennzeichnet, zu bestimmen, ob der erste Operand oder der zweite Operand einen kleineren Wert aufweist, und wobei die Komparatoreinheit ferner ausgebildet ist, einen ersten Satz aus Steuersignalwerten, die den kleineren Wert spezifizieren, zu dem Multiplexer zu übermitteln, und wobei der Multiplexer ausgebildet ist, den kleineren Wert als das Ergebnis des ersten Befehls in Reaktion auf das Empfangen des ersten Satzes aus Steuersignalwerten weiterzuleiten.

4. Mikroprozessor nach Anspruch 2, wobei der erste Befehl der Maximalwertbefehl ist, und wobei die Komparatoreinheit ausgebildet ist, in Reaktion auf das Empfangen eines decodierten Operationscodierungswertes, der den Maximalwertbefehl angibt, zu bestimmen, ob der erste Operand oder der zweite Operand einen größeren Wert aufweist, und wobei die Komparatoreinheit ferner ausgebildet ist, einen zweiten Satz aus Steuersignalwerten, die den größeren Wert spezifizieren, zu dem Multiplexer zu übermitteln, und wobei der Multiplexer ausgebildet ist, den größeren Wert als das Ergebnis des ersten Befehls in Reaktion auf das Empfangen des zweiten Satzes aus Steuersignalwerten weiterzuleiten.

5. Mikroprozessor nach Anspruch 2, wobei die mehreren Vergleichsbefehle einen „größer oder gleich“-Befehl, einen „größer“-Befehl und einen „gleich“-Befehl enthalten.

6. Mikroprozessor nach Anspruch 5, wobei der erste Befehl der „größer oder gleich“-Befehl ist, und wobei die Komparatoreinheit ausgebildet ist, zu bestimmen, ob der erste Operand größer oder gleich dem zweiten Operand ist in Reaktion auf das Empfangen eines decodierten Operationscodierungswertes, der den „größer oder gleich“-Befehl kennzeichnet, und wobei die Komparatoreinheit ferner ausgebildet ist, einen dritten Satz aus Steuersignalwerten, die den ersten konstanten Wert oder den zweiten konstanten Wert spezifizieren, zu dem Multiplexer zu übermitteln.

7. Mikroprozessor nach Anspruch 6, wobei der erste Operand größer oder gleich dem zweiten Operand ist, und wobei der dritte Satz aus Steuersignalwerten den ersten konstanten Wert spezifiziert, und wobei der erste konstante Wert als das Ergebnis des ersten Befehls in Reaktion auf das Empfangen des dritten Satzes aus Steuersignalwerten übermittelt wird.

8. Mikroprozessor nach Anspruch 6, wobei der erste Operand kleiner als der zweite Operand ist, und wobei der dritte Satz aus Steuersignalwerten den zweiten konstanten Wert spezifiziert, und wobei der zweite konstante Wert als das Ergebnis des ersten Befehls in Reaktion auf das Empfangen des dritten Satzes aus Steuersignalwerten übermittelt wird.

9. Mikroprozessor nach Anspruch 5, wobei der erste Befehl der „größer“-Befehl ist, und wobei die Komparatoreinheit ausgebildet ist, zu bestimmen, ob der erste Operand größer als der zweite Operand ist in Reaktion auf das Empfangen eines decodierten Operationscodierungswertes, der den „größer“-Befehl kennzeichnet, und wobei die Komparatoreinheit ferner ausgebildet ist, einen vierten Satz aus Steuersignalwerten, die den ersten konstanten Wert oder den zweiten konstanten Wert spezifizieren, zu dem Multiplexer zu übermitteln.

10. Mikroprozessor nach Anspruch 9, wobei der erste Operand größer als der zweite Operand ist, und wobei der vierte Satz aus Steuersignalwerten den ersten konstanten Wert spezifiziert, und wobei der erste konstante Wert als das Ergebnis des ersten Befehls in Reaktion auf das Empfangen des vierten Satzes aus Steuersignalwerten übermittelt wird.

11. Mikroprozessor nach Anspruch 9, wobei der erste Operand kleiner als der zweite Operand ist, und wobei der vierte Satz aus Steuersignalwerten den zweiten konstanten Wert spezifiziert, und wobei der zweite konstante Wert als das Ergebnis des ersten Befehls in Reaktion auf das Empfangen des vierten Satzes aus Steuersignalwerten übermittelt wird.

12. Mikroprozessor nach Anspruch 5, wobei der erste Befehl der „gleich“-Befehl ist, und wobei die Komparatoreinheit ausgebildet ist, zu bestimmen, ob der erste Operand gleich dem zweiten Operand ist in Reaktion auf das Empfangen eines decodierten Operationscodierungswertes, der den „gleich“-Befehl kennzeichnet, und wobei die Komparatoreinheit ferner ausgebildet ist, einen fünften Satz aus Steuersignalwerten, die den ersten konstanten Wert oder den zweiten konstanten Wert spezifizieren, zu dem Multiplexer zu übermitteln.

13. Mikroprozessor nach Anspruch 12, wobei, wenn der erste Operand gleich dem zweiten Operanden ist, der fünfte Satz aus Steuersignalwerten den ersten konstanten Wert spezifiziert, wodurch der Multiplexer veranlasst wird, den ersten konstanten Wert als das Ergebnis des ersten Befehls weiterzuleiten.

14. Mikroprozessor nach Anspruch 7, 10 oder 13, wobei der erste konstante Wert ein oder mehrere Bits umfasst, wobei alle Bits gesetzt oder alle Bits zurückgesetzt sind, und wobei der erste konstante Wert als eine Maske von einem Befehl verwendet werden kann, der nachfolgend zu dem ersten Befehl ausgeführt wird.

15. Mikroprozessor nach Anspruch 12, wobei, wenn der erste Operand nicht gleich dem zweiten Operanden ist, der fünfte Satz aus Steuersignalwerten den zweiten konstanten Wert spezifiziert, wodurch der Multip-

lexer veranlasst wird, den zweiten konstanten Wert als das Ergebnis des ersten Befehls zu übermitteln.

16. Mikroprozessor nach Anspruch 8, 11 oder 15, wobei der zweite konstante Wert mehrere Bits enthält, wobei keines der mehreren Bits gesetzt ist, und wobei der zweite konstante Wert als eine Maske von einem Befehl verwendet werden kann, der nachfolgend zu dem ersten Befehl ausgeführt wird.

17. Mikroprozessor nach Anspruch 1, wobei dem ersten und dem zweiten Operanden entsprechende Werte von einer oder mehreren der folgenden Quellen ausgelesen werden: einem unmittelbaren Wertefeld, das in dem ersten Befehl enthalten ist, aus einem Speicher, aus einem Register.

18. Mikroprozessor nach einem der Ansprüche 1 bis 13, 15 oder 17, wobei der erste Operand und der zweite Operand Fließkommazahlen sind.

19. Mikroprozessor mit:
 einem ersten Eingangsregister (**703A**, B), das angeschlossen ist, einen ersten Operanden zu empfangen, der einem ersten Befehl entspricht;
 einem zweiten Eingangsregister (**703C**, D), das angeschlossen ist, einen zweiten Operanden zu empfangen, der dem ersten Befehl entspricht;
 einer Komparatoreinheit (**731**), die angeschlossen ist, den ersten und den zweiten Operanden von dem ersten und dem zweiten Eingangsregister zu empfangen; und
 dadurch gekennzeichnet, dass der erste Operand einen ersten Vektorwert gefolgt von einem zweiten Vektorwert aufweist;
 dass der zweite Operand einen dritten Vektorwert gefolgt von einem vierten Vektorwert aufweist;
 dass die Komparatoreinheit ferner ausgebildet, einen decodierten Operationscodierungswert zu empfangen, der dem ersten Befehl entspricht; und
 dass ein erster Multiplexer (**740A**) angeschlossen ist, mehrere erste Eingangswerte einschließlich des ersten und des dritten Vektorwertes, eines ersten konstanten Wertes und eines zweiten konstanten Wertes zu empfangen, und wobei der erste Multiplexer ausgebildet ist, einen der mehreren ersten Eingangswerte auszuwählen, der als ein erster Bereich eines Vektorendergebnisses des ersten Befehls zu übermitteln ist, in Reaktion auf das Empfangen eines ersten Satzes aus Steuersignalwerten von der Komparatoreinheit;
 und dass die Komparatoreinheit ausgebildet ist, den ersten Satz aus Steuersignalwerten in Reaktion auf das Empfangen des decodierten Operationscodierungswertes zu erzeugen, und wobei, wenn der decodierte Operationscodierungswert angibt, dass der erste Befehl einer von mehreren Extremwertbefehlen ist, der erste Satz aus Steuersignalwerten verwendbar ist, um den ersten Vektorwert oder den dritten Vektorwert als den ersten Bereich des Vektorendergebnisses auszuwählen, und wobei, wenn der decodierte Operationscodierungswert angibt, dass der erste Befehl einer von mehreren Vergleichsbefehlen ist, der erste Satz aus Steuersignalwerten verwendbar ist, um den ersten konstanten Wert oder den zweiten konstanten Wert als den ersten Bereich des Vektorendergebnisses auszuwählen.

20. Mikroprozessor nach Anspruch 19, der ferner einen zweiten Multiplexer (**740B**) aufweist, der angeschlossen ist, mehrere zweite Eingangswerte einschließlich des zweiten und des vierten Vektorwertes, des ersten konstanten Wertes und des zweiten konstanten Wertes zu empfangen, und wobei der zweite Multiplexer ausgebildet ist, einen der mehreren zweiten Eingangswerte, der als ein zweiter Vektorbereich des Ergebnisses des ersten Befehls zu übermitteln ist, in Reaktion auf das Empfangen eines zweiten Satzes aus Steuersignalwerten von der Komparatoreinheit auszuwählen.

21. Mikroprozessor nach Anspruch 20, wobei die Komparatoreinheit ausgebildet ist, den zweiten Satz aus Steuersignalwerten in Reaktion auf das Empfangen des decodierten Operationscodierungswertes zu erzeugen, und wobei, wenn der decodierte Operationscodierungswert angibt, dass der erste Befehl einer der mehreren Extremwertbefehle ist, der zweite Satz aus Steuersignalwerten von dem zweiten Multiplexer verwendbar ist, um den zweiten Vektorwert oder den vierten Vektorwert als den zweiten Bereich des Vektorendergebnisses auszuwählen, und wobei, wenn der decodierte Operationscodierungswert angibt, dass der erste Befehl einer von mehreren Vergleichsbefehlen ist, der erste Satz aus Steuersignalwerten durch den zweiten Multiplexer verwendbar ist, um den ersten konstanten Wert oder den zweiten konstanten Wert als den zweiten Bereich des Vektorendergebnisses auszuwählen.

22. Verfahren zum Ausführen eines Befehls in einem Mikroprozessor, wobei das Verfahren umfasst.
 Vergleichen mehrerer erster Eingangswerte (**710A**, **710B**) auf der Grundlage eines decodierten Operationscodierungswertes eines Befehls, wobei die mehreren ersten Eingangswerte einen ersten und einen zweiten Operanden von dem Befehl enthalten;

Zusammenfügen mehrerer zweiter Eingangswerte (I1 bis I4), die den ersten Operanden, den zweiten Operanden, einen ersten konstanten Wert und einen zweiten konstanten Wert enthalten; und

Erzeugen eines ersten Befehlsergebnisses durch:

Auswählen aus den mehreren zweiten Eingangswerten des ersten Operanden oder des zweiten Operanden auf der Grundlage der Ergebnisse des Vergleichens, wenn der Befehl einer von mehreren Extremwertbefehlen ist, und

Auswählen aus den mehreren zweiten Eingangswerten des ersten konstanten Wertes oder des zweiten konstanten Wertes auf der Grundlage der Ergebnisse des Vergleichens, wenn der Befehl einer von mehreren Vergleichsbefehlen ist.

23. Verfahren nach Anspruch 22, wobei der erste Operand einen ersten Vektorwert gefolgt von einem zweiten Vektorwert aufweist, und wobei der zweite Operand einen dritten Vektorwert gefolgt von einem vierten Vektorwert aufweist.

24. Verfahren nach Anspruch 23, wobei die mehreren zweiten Eingangswerte den ersten und den dritten Vektorwert enthalten, und wobei das erste Befehlsergebnis ein erstes Vektorergebnis ist und wobei das Verfahren ferner umfasst:

Erzeugen eines zweiten Vektorergebnisses durch:

Auswählen des zweiten Vektorwertes oder des vierten Vektorwertes auf der Grundlage der Ergebnisse des Vergleichens, wenn der Befehl einer von mehreren Extremwertbefehlen ist, und

Auswählen des ersten konstanten Wertes oder des zweiten konstanten Wertes auf der Grundlage der Ergebnisse des Vergleichens, wenn der Befehl einer von mehreren Vergleichsbefehlen ist.

25. Mikroprozessor nach einem der Ansprüche 19 bis 21 oder Verfahren nach einem der Ansprüche 22 bis 24, wobei die Extremwertbefehle Minimalwertbefehle und Maximalwertbefehle enthalten.

26. Mikroprozessor nach einem der Ansprüche 19 bis 21 oder Verfahren nach einem der Ansprüche 22 bis 24, wobei die Vergleichsbefehle „größer oder „gleich“-Befehle, „größer“-Befehle und „gleich“-Befehle enthalten.

Es folgen 9 Blatt Zeichnungen

Anhängende Zeichnungen

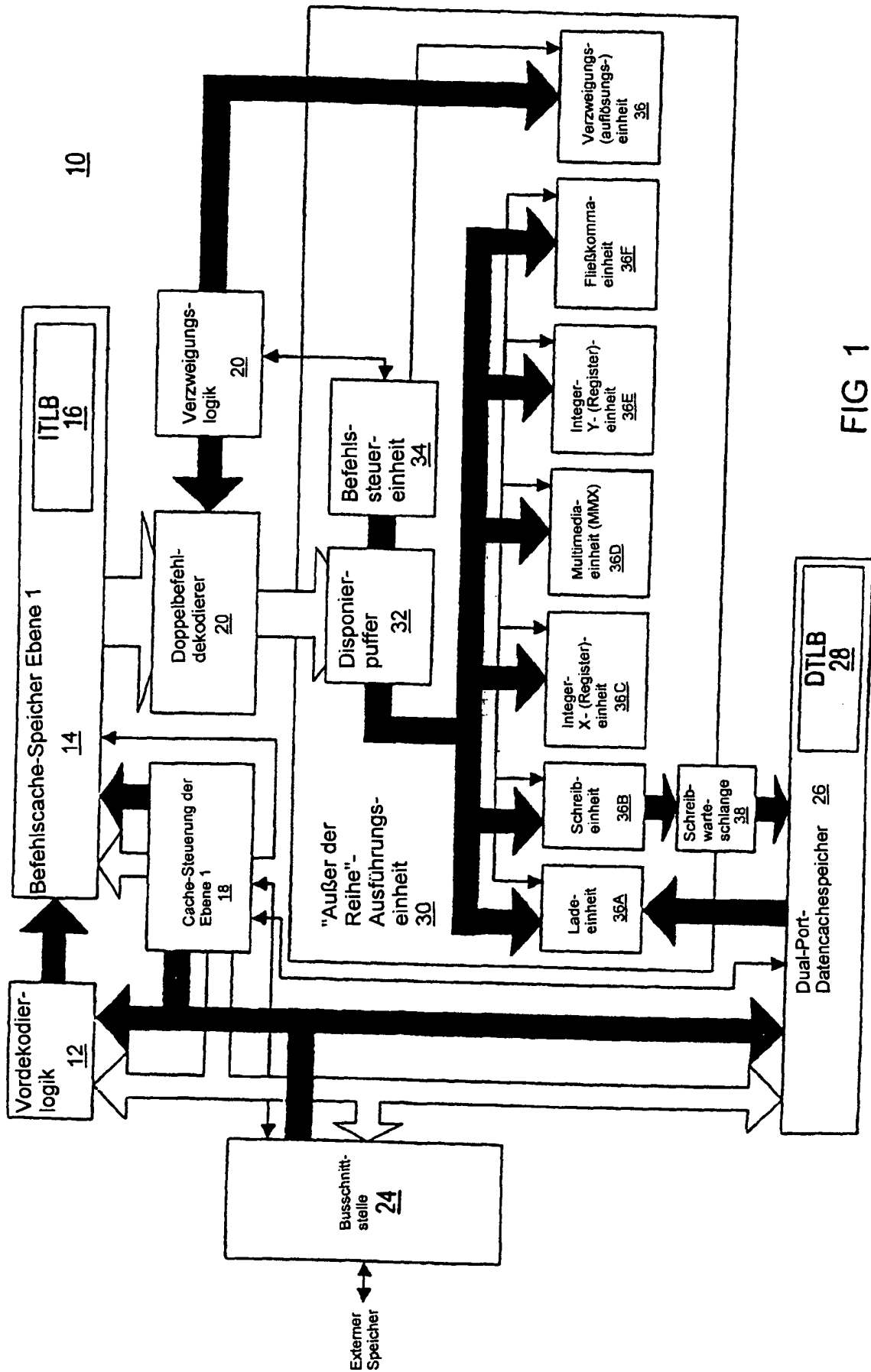


FIG 1

PFMIN		100		
mnemonic	opcode/imm8		Beschreibung	
PFMIN mmmreg1, mmreg2/mem64	0fh 0fh / 94h		gepacktes Fließkommaminimum	
102A	102B	101		

FIG. 2A

```

IF (mmreg1[31:0]<mmreg2/mem64[31:0])
  THEN mmreg1[31:0]=mmreg1[31:0]
ELSE mmreg1[31:0]=mmreg2/mem64[31:0]
IF (mmreg1[63:32]<mmreg2/mem64[63:32])
  THEN mmreg1[63:32]=mmreg1[63:32]
ELSE mmreg1[63:32]=mmreg2/mem64[63:32]
    
```

FIG. 2B

PFMIN	Quelle 2			
Quelle 1 & Ziel		0	normal	nicht unterstützt
	0	+0	Quelle 2, +0*	nicht definiert
	normal	Quelle 1, +0*	Quelle 1/Quelle 2 ***	nicht definiert
	nicht unterstützt	nicht definiert	nicht definiert	nicht definiert
Bemerkungen: * Das Ergebnis ist Quelle 2, wenn Quelle 2 negativ ist, ansonsten ist das Ergebnis positiv Null ** Das Ergebnis ist Quelle 1, wenn Quelle 1 negativ ist, ansonsten ist das Ergebnis positiv Null *** Das Ergebnis ist Quelle 1, wenn Quelle 1 negativ und Quelle 2 positiv ist. Das Ergebnis ist Quelle 1, wenn beide negativ sind und Quelle 1 betragsmäßig größer ist als Quelle 2. Das Ergebnis ist Quelle 1, wenn beide positiv sind und Quelle 1 betragsmäßig kleiner ist als Quelle 2. In allen anderen Fällen ist das Ergebnis Quelle 2				

FIG. 2C

PFMAX

200

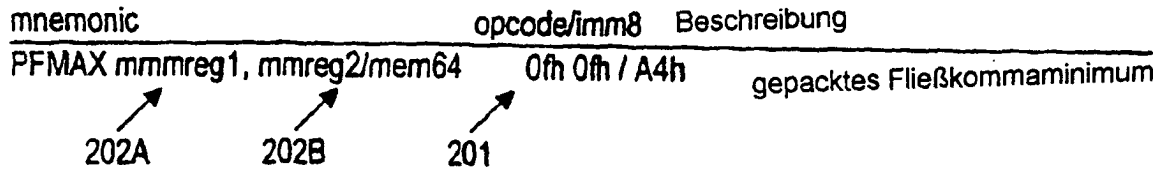


FIG. 3A

```

IF (mmreg1[31:0]>mmreg2/mem64[31:0])
  THEN mmreg1[31:0]=mmreg1[31:0]
  ELSE mmreg1[31:0]=mmreg2/mem64[31:0]
IF (mmreg1[63:32]>mmreg2/mem64[63:32])
  THEN mmreg1[63:32]=mmreg1[63:32]
  ELSE mmreg1[63:32]=mmreg2/mem64[63:32]
    
```

FIG. 3B

PFMAX	Quelle 2			
Quelle 1 & Ziel		0	Normal	nicht unterstützt
	0	+0	Quelle 2, +0*	nicht definiert
	nicht unterstützt	Quelle 1, +0*	Quelle 1/ Quelle 2 ***	nicht definiert
	nicht unterstützt	nicht definiert	nicht definiert	nicht definiert

Bemerkungen:

- * Das Ergebnis ist Quelle 2, wenn Quelle 2 negativ ist, ansonsten ist das Ergebnis positiv Null
- ** Das Ergebnis ist Quelle 1, wenn Quelle 1 negativ ist, ansonsten ist das Ergebnis positiv Null
- *** Das Ergebnis ist Quelle 1, wenn Quelle 1 positiv und Quelle 2 negativ ist. Das Ergebnis ist Quelle 1, wenn beide positiv sind und Quelle 1 betragsmäßig größer ist als Quelle 2. Das Ergebnis ist Quelle 1, wenn beidene negativ sind und Quelle 1 betragsmäßig kleiner ist als Quelle 2. In allen anderen Fällen ist das Ergebnis Quelle 2

FIG. 3C

PFCMPEQ		300		
mnemonic		opcode/imm8	Beschreibung	
PFCMPEQ	mmmreg1, mmreg2/mem64	0fh 0fh / B0h	Fließkommavergleich "gleich"	
	<div style="display: flex; justify-content: space-around;"> 302A ↗ 302B ↗ 301 ↗ </div>			

FIG. 4A

```

IF (mmreg1[31:0]=mmreg2/mem64[31:0])
  THEN mmreg1[31:0]=FFFF_FFFFh
ELSE mmreg1[31:0]=0000_0000h
IF (mmreg1[63:32]=mmreg2/mem64[63:32])
  THEN mmreg1[63:32]=FFFF_FFFFh
ELSE mmreg1[63:32]=0000_0000h
    
```

FIG. 4B

PFCMPEQ	Quelle 2			
Quelle 1 & Ziel		0	Normal	nicht unterstützt
	0	FFFF_FFFFh*	0000_0000h	0000_0000h
	Normal	0000_0000h	0000_0000h, FFFF_FFFFh**	0000_0000h
	nicht unterstützt	0000_0000h	0000_0000h	nicht definiert
Bemerkungen: * Positiv Null ist gleich negativ Null. Das Ergebnis ist FFFF_FFFFh, wenn Quelle 1 und Quelle 2 identische Vorzeichen, Exponenten und Mantissen: Ansonsten 0000_0000h.				

FIG. 4C

PFCMPGT		400		
mnemonic	opcode/imm8		Beschreibung	
PFCMPGT	mmmreg1, mmmreg2/mem64	0fh 0fh / A0h	gepackter Fließkomma- vergleich "größer"	
	↖ 402A	↖ 402B	↖ 401	

FIG. 5A

```

IF (mmreg1[31:0]>mmreg2/mem64[31:0])
  THEN mmreg1[31:0]=FFFF_FFFFh
ELSE mmreg1[31:0]=0000_0000h
IF (mmreg1[63:32]>mmreg2/mem64[63:32])
  THEN mmreg1[63:32]=FFFF_FFFFh
ELSE mmreg1[63:32]=0000_0000h
    
```

FIG. 5B

PFCMPGT	Quelle 2			
		0	Normal	nicht unterstützt
Quelle 1 & Ziel	0	0000_0000h	0000_0000h, FFFF_FFFFh*	nicht definiert
	Normal	0000_0000h, FFFF_FFFFh**	0000_0000h FFFF_FFFFh***	nicht definiert
	nicht unterstützt	nicht definiert	nicht definiert	nicht definiert
Bemerkungen: * Das Ergebnis ist FFFF_FFFFh, wenn die Quelle 2 negativ ist, ansonsten ist das Ergebnis 0000_0000h ** Das Ergebnis ist FFFF_FFFFh, wenn Quelle 1 positiv ist, ansonsten ist das Ergebnis 0000_0000h. *** Das Ergebnis ist FFFF_FFFFh, wenn Quelle 1 positiv und Quelle 2 negativ ist, oder wenn beide negativ sind und Quelle 1 betragsmäßig kleiner ist als Quelle 2, oder wenn Quelle 1 und Quelle 2 positiv sind und Quelle 1 betragsmäßig größer ist als Quelle 2. In allen anderen Fällen ist das Ergebnis 0000_0000h.				

FIG. 5C

PFCMPGE		500		
mnemonic		opcode/imm8		Beschreibung
PCMPGE	mmreg1, mmreg2/mem64	0Fh 0Fh / 90h		gepackter Fließkommavergleich "größer oder gleich"
	502A	502B	501	

FIG. 6A

```

IF (mmreg1[31:0]>=mmreg2/mem64[31:0])
  THEN mmreg1[31:0]=FFFF_FFFFh
ELSE mmreg1[31:0]=0000_0000h
IF (mmreg1[63:32]>=mmreg2/mem64[63:32])
  THEN mmreg1[63:32]=FFFF_FFFFh
ELSE mmreg1[63:32]=0000_0000h
    
```

FIG. 6B

PFCMPGE	Quelle 2			
Quelle 1 & Ziel		0	normal	nicht unterstützt
	0	FFFF_FFFFh*	0000_0000h, FFFF_FFFFh**	nicht definiert
	normal	0000_0000h, FFFF_FFFFh***	0000_0000h, FFFF_FFFFh****	nicht definiert
	nicht unterstützt	nicht definiert	nicht definiert	nicht definiert
Bemerkungen: * Positiv Null ist gleich negativ Null. ** Das Ergebnis ist FFFF_FFFFh, wenn Quelle 2 negativ ist, ansonsten ist das Ergebnis 0000_0000h. *** Das Ergebnis ist FFFF_FFFFh, wenn Quelle 1 positiv ist, ansonsten ist das Ergebnis 0000_0000h. **** Das Ergebnis ist FFFF_FFFFh, wenn Quelle 1 positiv und Quelle 2 negativ ist, oder wenn beide negativ sind und wenn Quelle1 betragsmäßig kleiner oder gleich als Quelle 2 ist, oder wenn Quelle 1 und Quelle 2 beide positiv sind und wenn Quelle 1 betragsmäßig größer oder gleich als Quelle 2 ist. Ansonsten ist das Ergebnis 0000_0000h.				

FIG. 6C

600

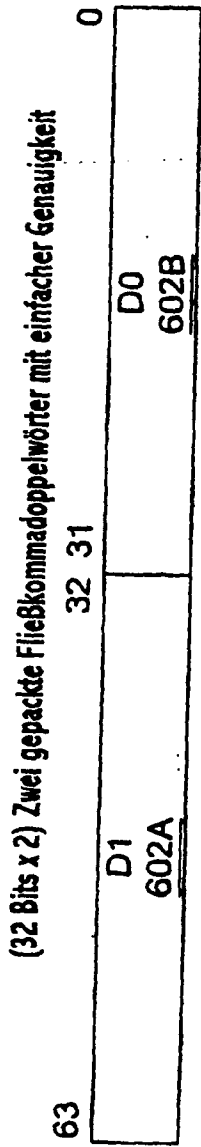


FIG. 7A

610

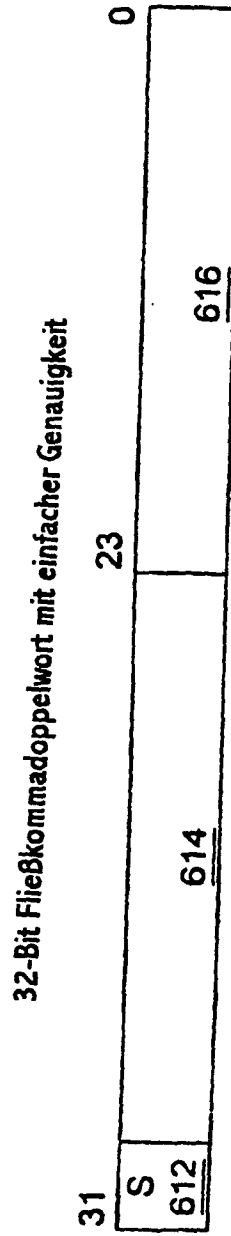


FIG. 7B

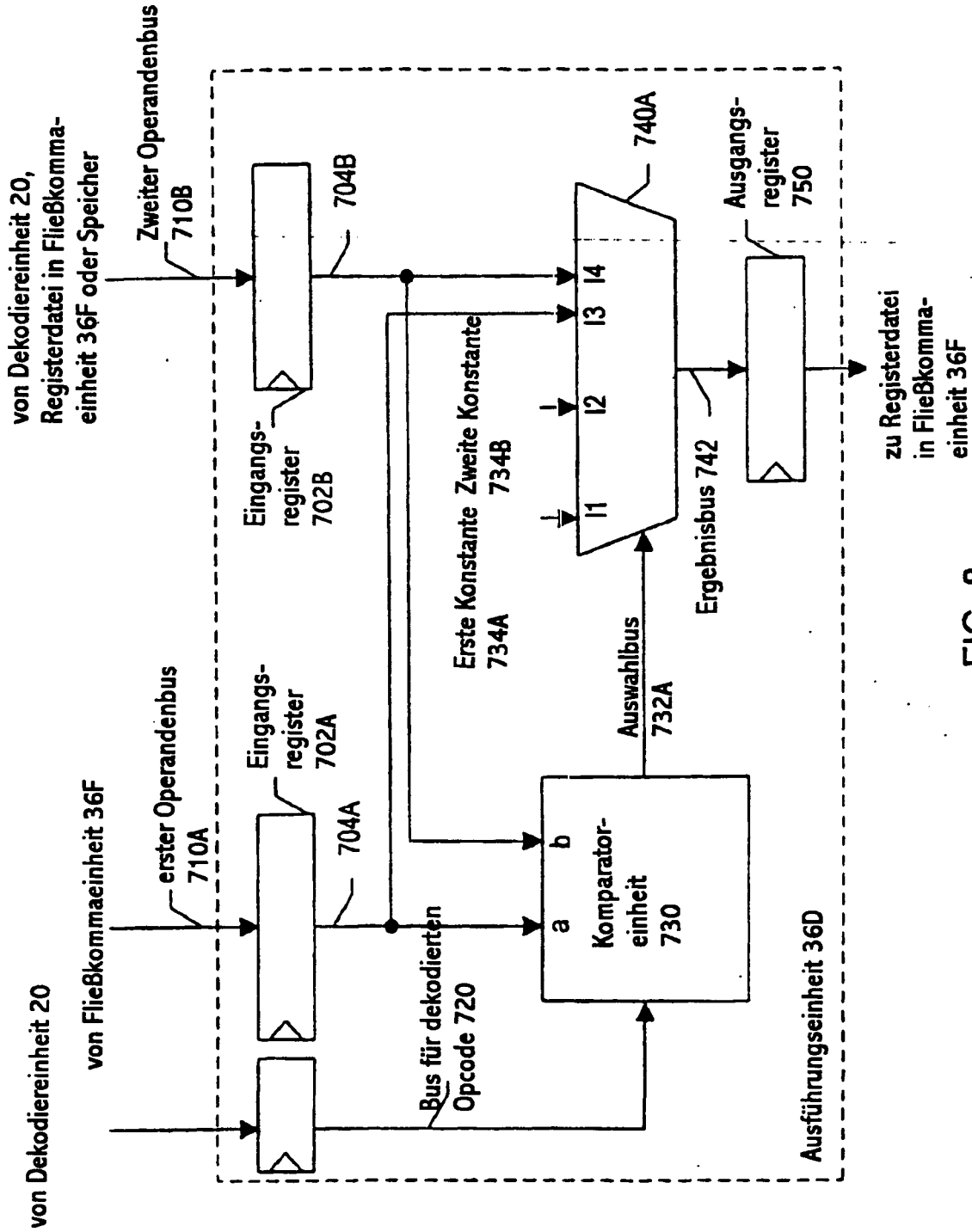


FIG. 8

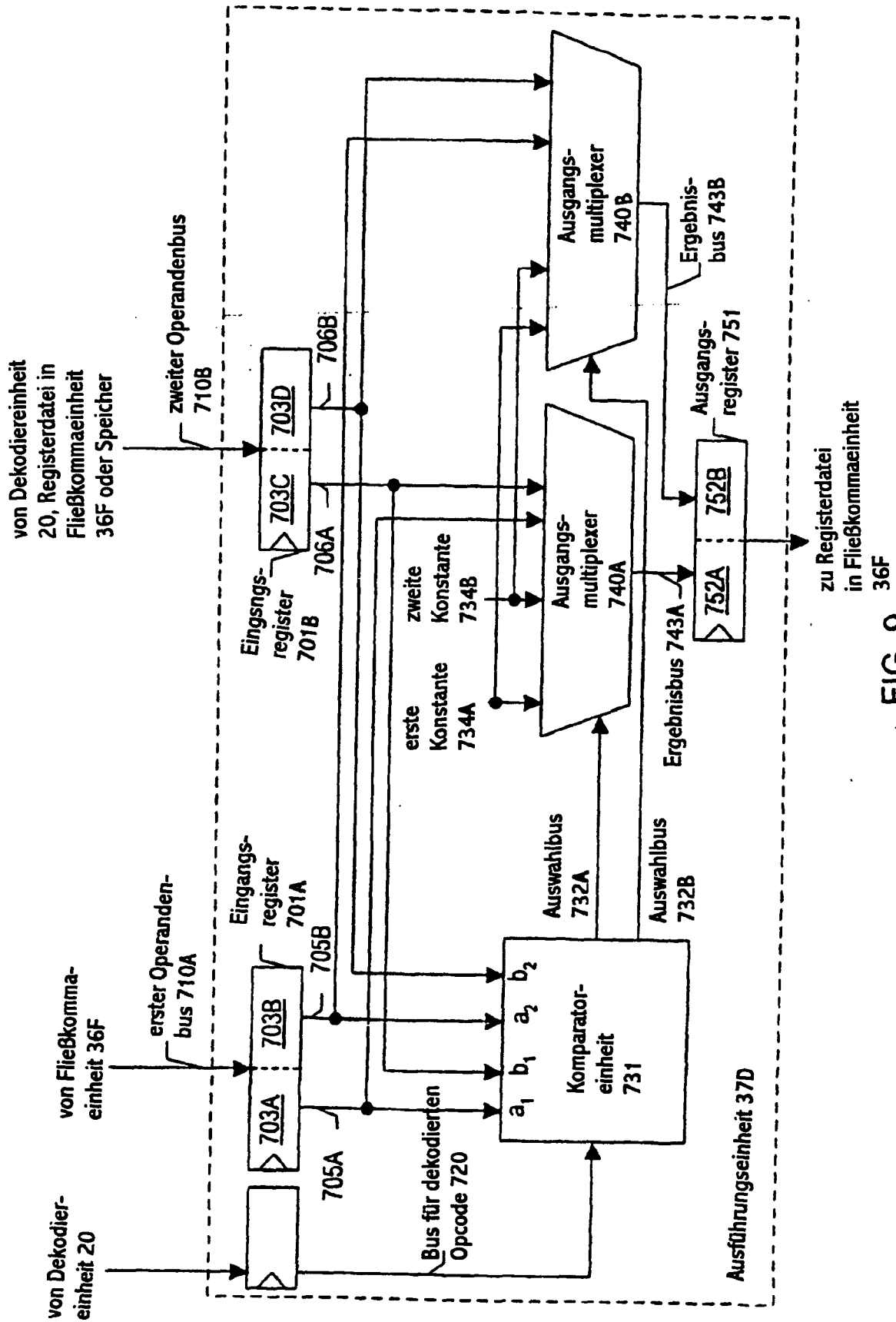


FIG. 9