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

(10) **DE 695 31 270 T2** 2004.04.22

(12) # Übersetzung der europäischen Patentschrift

(97) **EP 0 685 798 B1**
(21) Deutsches Aktenzeichen: **695 31 270.7**
(96) Europäisches Aktenzeichen: **95 303 292.7**
(96) Europäischer Anmeldetag: **17.05.1995**
(97) Erstveröffentlichung durch das EPA: **06.12.1995**
(97) Veröffentlichungstag der Patenterteilung beim EPA: **16.07.2003**
(47) Veröffentlichungstag im Patentblatt: **22.04.2004**

(51) Int Cl.[7]: **G06F 13/26**

(30) Unionspriorität:

251843	**31.05.1994**	**US**
251799	**31.05.1994**	**US**
252281	**31.05.1994**	**US**
251849	**31.05.1994**	**US**
251850	**31.05.1994**	**US**
252086	**31.05.1994**	**US**

(74) Vertreter:
Patentanwälte von Kreisler, Selting, Werner et col., 50667 Köln

(84) Benannte Vertragsstaaten:
AT, BE, DE, DK, ES, FR, GB, GR, IE, IT, LU, NL, PT, SE

(73) Patentinhaber:
Advanced Micro Devices, Inc., Sunnyvale, Calif., US

(72) Erfinder:
Gephardt, Douglas, Austin, Texas 78748, JP;
MacDonald, James, Buda, Texas 78610, US;
Schmidt, Rodney, Austin, Texas 78746, US;
Mahalingaiah, Rupaka, Austin, Texas 78729, US

(54) Bezeichnung: **Unterbrechungssteuerungsgeräte in symmetrischen Mehrprozessorsystemen**

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

[0001] Die Erfindung betrifft Computersysteme und insbesondere Interrupt-Steuerungs-Architekturen und -Schemata zur Verwendung in symmetrischen Mehrfachverarbeitungssystemen.

[0002] Computersysteme, in denen Mehrfachverarbeitungseinheiten verwendet werden, versprechen eine ökonomische Handhabung von Betriebsfunktionen, welche diejenigen derzeitiger Systeme auf Einzelprozessor-Basis übertreffen. In einer Mehrfachverarbeitungs-Umgebung werden, statt die gesamte Verarbeitung für eine Anwendung in einem einzelnen Prozessor zu konzentrieren, die Tasks in Gruppen oder "Stränge" unterteilt, die von separaten Prozessoren gehandhabt werden können. Die gesamte Verarbeitungslast wird dadurch auf mehrere Prozessoren verteilt, und die verteilten Tasks können gleichzeitig parallel ausgeführt werden. Die Betriebssystem-Software verteilt verschiedene Teile des Programm-Codes auf die verschiedenen auszuführenden Stränge und weist typischerweise jedem Strang eine Prioritätsebene zu.

[0003] **Fig.** 1 zeigt ein Blockschaltbild eines sogenannten symmetrischen Mehrfachverarbeitungssystems **10**, das mehrere Verarbeitungseinheiten **12A**–**12C** aufweist. Jede Verarbeitungseinheit **12A**–**12C** enthält einen Prozessor-Kern **14A**–**14C**, einen Cache-Speicher **16A**–**16C** bzw. ein Bus-Interface **18A**–**18C**. Die Verarbeitungseinheiten **12A**–**12C** sind über einen Systembus **22** mit einem Hauptspeicher **20** verbunden. Ferner ist ein Paar von I/O-Einrichtungen 24 und 26 mit dem Systembus **22** verbunden.

[0004] Das Mehrfachverarbeitungssystem **10** gemäß **Fig.** 1 ist symmetrisch in dem Sinn, dass sämtliche Verarbeitungseinheiten **12A**–**12C** gemeinsam den gleichen Speicherraum (d. h. den Hauptspeicher **20**) benutzen und mittels der gleichen Adress-Abbildung auf den Speicherraum zugreifen. Das Mehrfachverarbeitungssystem **10** ist ferner symmetrisch in dem Sinn, dass sämtliche Verarbeitungseinheiten **12A**–**12C** gemeinsam den gleichen Zugriff auf das gleiche I/O-Subsystem benutzen.

[0005] Generell werden eine einzelne Kopie der Betriebsystem-Software sowie eine einzelne Kopie jeder User-Anwendungsdatei in dem Hauptspeicher **20** gespeichert. Jede Verarbeitungseinheit **12A**–**12C** führt ihre Ausführungsvorgänge aus diesen Einzelkopien des Betriebssystems und der User-Anwenderdateien heraus durch. Obwohl die Verarbeitungs-Kerne **14A**–**14C** den Code gleichzeitig ausführen können, ist zu beachten, dass nur eine der Verarbeitungseinheiten **12A**–**12C** zu einem gegebenen Zeitpunkt den Master-Status über den System-Bus übernehmen kann. Somit ist ein (nicht gezeigter) Unterscheidungsmechanismus vorgesehen, um gleichzeitig Bus-Anforderungen zweier oder mehr Verarbeitungseinheiten in Rangfolge zu bringen und einer der beiden Verarbeitungseinheiten basierend auf einem vorbestimmten Unterscheidungs-Algorithmus den Master-Status zuzuerkennen. Es sind verschiedene von Bus-Unterscheidungs-Algorithmen weithin bekannt.

[0006] Der jeweilige Hochgeschwindigkeits-Cache-Speicher **16A**–**16C** jeder Verarbeitungseinheit **12A**–**12C** speichert die Daten, auf die von der jeweiligen Verarbeitungseinheit zuletzt zugegriffen wurde, zusammen mit Adress-Tags, welche die Haupt-Speicheradresse angeben, der die zugewiesenen Daten entsprechen. Da Programme dazu tendieren, wiederholt die gleichen Code-Abschnitte auszuführen und auf die gleichen Datenstrukturen zuzugreifen, sind die meisten Stellen, auf die zugegriffen wird, bereits in dem Cache vorhanden, falls der Cache hinreichend groß ist.

[0007] Die Cache-Mechanismen bieten zwei bedeutende Vorteile. Erstens erfolgt, da die Caches mit Hochgeschwindigkeits-Speicher implementiert sind und auf sie ohne Bus-Unterscheidungs- und Puffer-Verzögerungen zugegriffen werden kann, ein Zugriff auf eine in einem jeweiligen Cache gespeicherte Stelle sehr viel schneller als ein Hauptspeicher-Zugriff. Zweitens wird, da ein Zugriff auf eine in dem jeweiligen Cache gespeicherte Stelle keinen Zugriff auf den System-Bus erfordert, die Bus-Verwendung seitens jedes Prozessors beträchtlich reduziert. Der System-Bus ist somit verfügbar für das Bedienen anderer angeforderter Transaktionen. Typischerweise ist, je höher die "Treffer-Rate", die Gesamt-Systemleistung um so besser. Die Treffer-Rate ist derjenige Prozentanteil von Zugriffen durch einen bestimmten Prozessor-Kern, der auf Stellen erfolgt, die bereits in dem Cache gespeichert sind. Gut konzipierte Systeme mit moderat großen Caches können Treffer-Raten von mehr als neunzig Prozent erreichen.

[0008] Eine wichtige Erwägung im Zusammenhang mit Mehrfachverarbeitungssystemen, bei denen Cache-Speicher verwendet werden, ist die Datenkohärenz. Da durch den Hauptspeicher **20** gespeicherte Mehrfachkopien von Daten (und Instruktionen) gleichzeitig in einem oder mehreren der Cache-Speicher **16A**–**16C** vorhanden sein können, muss ein spezialisierter Mechanismus verwendet werden, um die Integrität der Daten in dem Fall aufrechtzuerhalten, dass eines der Speicher-Subsysteme aktualisiert wird (d. h. mit neuen Daten beschrieben wird). Man stelle sich beispielweise eine Situation vor, in der ein bestimmter Abschnitt von Daten in dem Cache-Speicher **16A** durch den Verarbeitungs-Kern **14A** aktualisiert wird, jedoch nicht in dem entsprechenden Abschnitt des Hauptspeichers aktualisiert wird. Falls der Verarbeitungs-Kern **14B** anschließend auf den gleichen Code-Abschnitt zugreift, muss ein zuverlässiger Mechanismus vorhanden sein, um zu verfolgen, welcher Abschnitt aktuell ist und welcher Abschnitt nicht mehr gültig ist, um zu gewährleisten, das der Verarbeitungs-Kern **14B** auf die korrekten Daten zugreift. Dafür sind verschiedene Techniken mit dem Ziel entwickelt worden, die Cache-Kohärenz effizient aufrechtzuerhalten, einschließlich derjenigen, die auf sogenannten Durchschreib- oder Zurückschreib-Techniken basieren. Verschiedene Cache-Kohärenz-Techniken des Stan-

des der Technik sind in einer Fülle von Veröffentlichungen beschrieben und werden hier nicht weiter erläutert.

[0009] Eine weitere wichtige Erwägung im Zusammenhang mit symmetrischen Mehrfachverarbeitungssystemen ist die Handhabung und Verteilung von Interrupts, die durch verschiedene System-Ressourcen erzeugt werden. Beispielsweise können bei dem System gemäß **Fig.** 1 die I/O-Einrichtungen **24** und **26** auf der Basis des Auftretens (oder Nicht-Auftretens) eines bestimmten Ereignisses jeweils ein betreffendes Interrupt-Signal aktivieren. Wie Fachleuten auf dem Gebiet ersichtlich sein wird, werden Interrupts routinemäßig durch System-Ressourcen wie etwa Tastatur-Vorrichtungen, Drucker und Zeitgeber u. a. erzeugt. Zahlreiche Systeme nehmen auch Software-Interrupts auf, wobei ein Interrupt auf einen Software-Befehl hin aktiviert werden kann. Aufgrund der Anzahl verschiedener Interrupts, die in einem System auftreten können, besteht der Wunsch nach der Entwicklung eines Mechanismus zum effizienten Handhaben und Verteilen der Interrupts dahingehend, dass eine optimale System-Leistungsfähigkeit und Bus-Nutzung erzielt wird.

[0010] Bei einer Technik zum Handhaben von Interrupts wird ein zentralisierter Interrupt-Controller verwendet, der in der Lage ist, mehrere Interrupts zu empfangen und die Interrupts zu prioritisieren und unter den verschiedenen Verarbeitungseinheiten zu verteilen. Ein Problem, das bei einer zentralisierten Interrupt-Steuertechnik auftritt, besteht darin, das die Gesamt-Anzahl von Interrupts, die aufgenommen werden kann, typischerweise durch die Anzahl von Eingangs-Stiften beschränkt ist, die für den zentralisierten Interrupt-Controller vorgesehen sind. Anders ausgedrückt können beispielsweise, falls der zentralisierte Interrupt-Controller insgesamt sechzehn Interrupt-Eingangsstifte aufweist, typischerweise nicht mehr als sechzehn Interrupt-Erzeugungseinrichtungen in dem System aufgenommen werden. Ferner muss bei derartigen Systemen mit jeder Interrupt-Quelle eine speziell vorgesehene Interrupt-Leitung verbunden sein. Derartige speziell vorgesehene Interrupt-Leitungen sind möglicherweise an Fernverkabelungsnetzwerken, die eine oder mehr Peripherieeinrichtungen mit dem Computersystem verbinden, nicht verfügbar. Wiederum kann somit die System-Flexibilität beschränkt sein.

[0011] Ein weiteres Problem, das im Zusammenhang mit der Interrupt-Handhabung in symmetrischen Mehrfachverarbeitungssystemen auftreten kann, ist das Auftreten ungültiger oder "unechter" Interrupts: Wie generell bekannt ist, wird, wenn ein ebenen-getriggertes Interrupt durch eine bezeichnetre Verarbeitungseinheit getriggert wird, ein I/O-Befehl typischerweise der Interrupt-Bedienungs-Routine zugewiesen, die, wenn sie ausgeführt wird, die Interrupt-Quelle dazu veranlasst, das Interrupt-Signal zu deaktivieren. Anschließend wird ein End Of Interrupt-(EOI-) Befehl ausgeführt, um den Interrupt-Controller mitzuteilen, dass die Interrupt-Bedienung abgeschlossen worden ist. Ein unechtes Interrupt kann auftreten, falls eine signifikante Wartezeit zwischen der Zeit, zu der die Verarbeitungseinheit den I/O-Befehl ausführt (um die Interrupt-Quelle dazu zu veranlassen, das Interrupt-Signal zu deaktivieren) und der Zeit eingeführt wird, zu der das Interrupt-Signal tatsächlich deaktiviert wird. Eine derartige Wartezeit kann z. B. eintreten, falls die I/O-Einrichtung an einem Fern-Bus über mehrere Bus-Brücken-Einheiten angeordnet ist. Falls die Interrupt-Quelle das Interrupt-Signal nicht deaktiviert, bevor der zentralisierte Interrupt-Controller auf den End of Interrupt-Befehl reagiert, kann die fortgesetzte Aktivierung des Interrupts von dem Interrupt-Controller detektiert werden, wodurch ein unbeabsichtigtes Re-Initiieren des Interrupts verursacht wird.

[0012] Die Handhabung von Zeitmessungs-Interrupts bildet ein weiteres Problem bei symmetrischen Mehrfachverarbeitungssystemen. Wie generell bekannt ist, gibt es bei einem Mehrfachverarbeitungssystem einige Interrupts, die von sämtlichen Verarbeitungseinheiten auf Broadcast-Weise gehandhabt werden müssen. Beispielsweise wird bei zahlreichen Mehrfachverarbeitungssystemen das Zeitmessungs-Interrupt dazu verwendet, das Ende eines Zeitabschnitts zu signalisieren und somit jede Verarbeitungseinheit dazu zu veranlassen, eine Task-Umschaltung auszuführen. Dies wird typischerweise erzielt, indem entweder gleichzeitig das Zeitmessungs-Interrupt an sämtliche Verarbeitungseinheiten gesendet wird oder in dem das Zeitmessungs-Interrupt an die erste Verarbeitungseinheit in der Kette gesendet wird und über Interprozessor-Interrupts (IPIs) an die anderen weitergeleitet wird. Bei beiden dieser Schemata existieren jedoch Nachteile. Falls sämtliche Verarbeitungseinheiten zur gleichen Zeit unterbrochen werden, versuchen sie, auf den gleichen gemeinsam benutzten Interrupt-Handhabungs-Code zuzugreifen und arretieren gleichzeitig System-Ressourcen, die gemeinsam benutzt werden. Dies kann ein hohes Maß an Konkurrenz verursachen und kann es erforderlich machen, dass einige Verarbeitungseinheiten warten müssen, bis die erforderlichen Ressourcen verfügbar werden. Falls das Zeitmessungs-Interrupt der ersten Verarbeitungseinheit in der Kette zugeführt wird und den anderen unter Verwendung von Interprozessor-Interrupts zugeführt wird, zieht dies ein erhöhtes Maß an Software-Gesamtaufwand und Bus-Nutzung nach sich.

[0013] Zu den weiteren Problemen bei der Interrupt-Handhabung in symmetrischen Mehrfachverarbeitungssystemen zählt die Integration von System-Management-Interrupts (SMI) und die Prioritisierung von Interrupts. Es besteht der Wunsch nach der Schaffung von Mechanismen in symmetrischen Mehrfachverarbeitungssystemen, die sowohl eine effiziente Integration von SMI-Interrupts in dem System als auch eine flexible Interrupt-Prioritisierung ermöglichen.

[0014] Die oben angeführten Probleme werden zum Großteil durch ein symmetrisches Mehrfachverarbeitungssystem gemäß dem unabhängigen Anspruch 1 gelöst.

[0015] Verschiedene detailliertere Implementierungen der Erfindung sind in den abhängigen Ansprüchen definiert.

[0016] Weitere Aufgaben und Vorteile der Erfindung werden, jedoch nur im Sinne von Beispielen, aus der folgenden detaillierten Beschreibung und anhand der beigefügten Zeichnungen ersichtlich:

[0017] **Fig.** 1 zeigt ein Blockschaltbild eines typischen symmetrischen Mehrfachverarbeitungssystems mit mehreren Verarbeitungseinheiten.

[0018] **Fig.** 2 zeigt ein Blockschaltbild eines symmetrischen Mehrfachverarbeitungssystems mit einem zentralisierten Interrupt-Controller-Mechanismus gemäß einer Ausführungsform der vorliegenden Erfindung.

[0019] **Fig.** 3 zeigt ein Blockschaltbild einer zentralen Interrupt-Steuereinheit.

[0020] **Fig.** 4 zeigt ein Blockschaltbild eines I/O-Interrupt-Controllers.

[0021] **Fig.** 5 zeigt ein Blockschaltbild zur Veranschaulichung eines Interrupt-Kanals.

[0022] **Fig.** 6 zeigt ein Schaubild eines Interrupt-Kanal-Steuerregisters und seiner zugeordneten Felder.

[0023] **Fig.** 7 zeigt ein Blockschaltbild zur Veranschaulichung der mit dem symmetrischen Mehrfachverarbeitungssystemen verbundenen Hardware während eines Kaskadierungs-Modus eines Interrupt-Eingangssignal-Prozessors.

[0024] **Fig.** 8 zeigt ein Blockschaltbild zur Veranschaulichung der für einen weiteren Kaskadierungs-Modus des symmetrischen Mehrfachverarbeitungssystems vorgesehenen Hardware.

[0025] **Fig.** 9 zeigt ein Blockschaltbild eines Prozessor-Interrupt-Generators.

[0026] **Fig.** 10 zeigt ein Blockschaltbild zur Veranschaulichung eines Prozessor-Kanals.

[0027] **Fig.** 10A zeigt ein Schaubild zur Veranschaulichung eines CPU-Kanal-Steuerregisters zusammen mit seinen zugeordneten Feldern.

[0028] **Fig.** 10B zeigt ein Schaubild zur Veranschaulichung eines Interprozessor-Interrupt-Registers zusammen mit seinen zugeordneten Feldern.

[0029] **Fig.** 11 zeigt ein Flussdiagramm zur Veranschaulichung der Arbeitsweise der Boot-Verarbeitungseinheit beim System-Rücksetzen.

[0030] **Fig.** 12 zeigt ein Schaubild zur Darstellung der Hardware zur Aktivierung der CPU-Kanal-Register-Einheit jedes CPU-Kanals.

[0031] **Fig.** 13 zeigt ein Blockschaltbild der Veranschaulichung von Schaltungsanordnungen, die in der zentralen Interrupt-Steuereinheit ausgebildet sind.

[0032] **Fig.** 14 zeigt ein Flussdiagramm zur Darstellung der Initialisierungs-Sequenz für jede Slave-Verarbeitungseinheit.

[0033] **Fig.** 15 zeigt ein Blockschaltbild der Veranschaulichung einer der Verarbeitungseinheiten, die mit der zentralen Interrupt-Steuereinheit verbunden sind.

[0034] **Fig.** 16 zeigt ein Blockschaltbild eines Mehrfachverarbeitungssystems, das speziell definierte Interrupt-Zyklen zum Initiieren eines Interrupts aufnimmt.

[0035] **Fig.** 17 zeigt ein Blockschaltbild der Veranschaulichung eines Interrupt-Kanals mit Hardware, die zum Dekodieren eines Interrupt-Zyklus konfiguriert ist.

[0036] **Fig.** 18 zeigt ein Blockschaltbild der Veranschaulichung eines Teils der zentralen Interrupt-Steuereinheit einschließlich eines internen Teils des zentralen Controllers.

[0037] **Fig.** 19 zeigt ein Flussdiagramm zur Veranschaulichung der Zufuhr eines Interrupts während des Broadcast-Modus.

[0038] **Fig.** 20 zeigt ein Blockschaltbild der zentralen Steuereinheit mit einem programmierbaren Wartezeit-Zeitgeber.

[0039] **Fig.** 21 zeigt ein Blockschaltbild eines symmetrischen Mehrfachverarbeitungssystems mit einer Energiehandhabungseinheit, die zum Aktivieren eines System-Handhabungs-Interrupts in der Lage ist.

[0040] **Fig.** 22 zeigt ein Flussdiagramm zur Veranschaulichung der Arbeitsweise des symmetrischen Mehrfachverarbeitungssystems bei Aktivierung eines System-Handhabungs-Interrupts.

[0041] Obwohl die Endung in Form verschiedener Modifikationen und alternativer Ausgestaltungen realisiert werden kann, werden als Beispiel spezielle Ausführungsformen der Erfindung in den Zeichnungen gezeigt und hier detailliert beschrieben. Es wird jedoch darauf hingewiesen, dass die Zeichnungen und die detaillierte Beschreibung nicht die Absicht verfolgen, die Erfindung auf die bestimmte offenbarte Form zu beschränken, sondern dass im Gegenteil die Erfindung sämtliche Modifikationen, Äquivalente und Alternativen umfasst, die in den Bereich der angefügten Ansprüche fallen.

[0042] Die als nächstes zu erläuternde **Fig.** 2 zeigt ein Blockschaltbild eines symmetrischen Mehrfachverarbeitungssystems **200** mit einem zentralisierten Interrupt-Controller-Mechanismus. Das System **200** weist mehrere Verarbeitungseinheiten **202-1** bis **202-m** auf, die über einen CPU-Lokal-Bus **206** mit einem Hauptspeicher **204** verbunden sind. Jede Verarbeitungseinheit **202-1** bis **202-m** enthält einen jeweiligen Verarbeitungs-Kern **204-1** bis **204-m**, einen jeweiligen Cache-Speicher **205-1** bis **205-m** und ein jeweiliges Bus-Interface **206-1** bis **206-m**. Eine Bus-Brücke **208** verbindet den CPU-Lokal-Bus **207** mit einem I/O-Bus **210**. Mit dem I/O-Bus **210** sind mehrere I/O-Peripherieeinrichtungen **212-1** bis **212-n** verbunden. Ferner sind eine zusätzliche I/O-Einrich-

tung **214** und ein Interrupt-Controller **216** mit dem I/O-Bus **210** verbunden. Die I/O-Einrichtungen **212-1** bis **212-n** sind mit einer zentralen Interrupt-Steuereinrichtung **220** verbunden. Auf die I/O-Einrichtungen **212-1** bis **212-n** kann von jeder der Verarbeitungseinheiten **202-1** bis **202-m** über die Bus-Brücke **208** zugegriffen werden.

[0043] Die zentrale Interrupt-Steuereinrichtung **220** ist vorgesehen zum Handhaben von Interrupts, die aus den I/O-Einrichtungen **212-1** bis **212-n** und dem Interrupt-Controller **216** zu empfangen werden, und zum Verteilen der Interrupts unter den. Verarbeitungseinheiten **202-1** bis **202-m**. Die zentrale Interrupt-Steuereinrichtung **220** handhabt ferner Interprozessor-Interrupts und Software-Interrupts, die von den Verarbeitungseinheiten **202-1** bis 202-m erzeugt werden. In ihrer bevorzugten Form ist die zentrale Interrupt-Steuereinrichtung **220**, wie nachstehend erläutert, mit einer Vielfalt programmierbarer Merkmale programmiert, um eine optimale System-Flexibilität aufzunehmen.

[0044] Bei dem I/O-Bus **210** kann es sich um jeden Bus handeln, der dazu geeignet ist, Peripherieeinrichtungen wie CD-ROM-Einheiten, Lokalbereichsnetzwerk-(LAN-) Einheiten und Drücker mit dem Computersystem **200** zu verbinden. Zu den Beispielen für Peripheriebus-Standards zählen ein Bus gemäß ISA (industry standard architecture), ein Bus gemäß EISA (Extended Industry Standard Architecture) und ein Bus gemäß PCI (peripheral component interconnect). Die Bus-Brücke **208** bildet ein Interface zwischen dem I/O-Bus **210** und dem CPU-Lokal-Bus **207**.

[0045] Bei den Verarbeitungs-Kerne **204-1** bis **204-m** handelt es sich um Datenverarbeitungseinheiten, die entsprechend einem vorbestimmten Instruktions-Set arbeiten. Zu den Beispielen für diese Verarbeitungseinheiten zählen Verarbeitungseinheiten vom Modell 80486, Pentium-kompatible Verarbeitungseinheiten und PowerPC- Verarbeitungseinheiten. Es wird jedoch darauf hingewiesen, dass die Verarbeitungseinheiten **202-1** bis **202-m** auch entsprechend anderen Instruktions-Sets arbeiten könnten.

[0046] Die Cache-Speicher **205-1** bis **205-m** sind als Hochgeschwindigkeits-Speichereinrichtungen implementiert. Jeder Cache-Speicher **205-1** bis **205-m** ist mit einem (in der Figur nicht getrennt gezeigten) Cache-Controller verbunden, der die Übertragung von Daten zwischen dem zugehörigen Verarbeitungs-Kern **204-1** bis **204-m**, dem zugehörigen Cache-Speicher **205-1** bis **205-m** und dem CPU-Lokal-Bus **207** organisiert und handhabt. In dere bevorzugten Form arbeitet der Cache-Controller jeder Verarbeitungseinheit gleichzeitig mit dem zugehörigen Verarbeitungs-Kern, damit die maximale gestützte Leistungsfähigkeit erzielt wird.

[0047] Der CPU-Lokal-Bus **207** hat eine vorbestimmte Bit-Breite und ist der primäre Bus des Computersystems. Der Hauptspeicher **204** ist ein physischer Speicher mit vorbestimmter Größe und kann mit einem DRAM (dynamischer RAM-Speicher) implementiert sein. Ein (nicht separat gezeigter) Speicher-Controller ist dem Hauptspeicher **204** zugeordnet und steuert und organisiert die Übertragung von Daten-, Adress- und Steuersignalen, die zwischen dem CPU-Lokal-Bus **207** und dem Hauptspeicher **204** übermittelt werden.

[0048] Der Interrupt-Controller **216** ist zum Sortieren und Handhaben von Interrupt-Signalen vorgesehen, die aus verschiedenen Interrupt-Quellen abgeleitet werden, wie z. B. der I/O-Einrichtung **214**. Bei dem als Beispiel gezeigten Interrupt-Controller kann es sich etwa um einen von Advanced Micro Devices hergestellten programmierbaren Interrupt-Controller der Serie des Modells 8259A handeln. Der programmierbare Interrupt-Controller 8259A ist beschrieben in der Veröffentlichung mit dem Titel "MOS Microprocessors and Peripherals", pp. 3–371 bis 3–388 (Advanced Micro Devices, Inc. 1987).

[0049] Bevor mit einer detaillierten Erläuterung der zentralen Interrupt-Steuereinrichtung **220** fortgefahren wird, ist zunächst anzumerken, dass in der zentralen Interrupt-Steuereinrichtung **220** eine Vielfalt von Konfigurationsregistern ausgebildet ist. Die Programmierung dieser Konfigurationsregister und der Zugriff auf sie können über den I/O-Bus **210** erfolgen. Somit ist der zentrale Interrupt-Controller **220** unabhängig von dem Typ des CPU-Lokal-Busses **207** und kann deshalb unter Verwendung unterschiedlicher Typen von Verarbeitungseinheiten in einer Vielfalt von Systemen verwendet werden. Folglich kann die zentralen Interrupt-Steuereinheit **220** in Verbindung mit verschiedenen Verarbeitungssystemen verwendet werden und ist kompatibel mit diesen.

[0050] Anhand der als nächstes zu erläuternden **Fig.** 3 werden Einzelheiten zu der zentralen Interrupt-Steuereinheit **220** aufgeführt. **Fig.** 3 ist ein Blockschaltbild einer Ausführungsform der zentralen Interrupt-Steuereinheit **220**, die einen zentralen Controller **302** aufweist, der mit einem I/O-Interrupt-Controller **304** und mit einem Prozessor-Interrupt-Controller **306** verbunden ist. Ferner ist eine I/O-Bus-Interface-Einheit **308** gezeigt, die mit dem zentralen Controller **302** verbunden ist.

[0051] Die I/O-Bus-Interface-Einheit **308** bildet ein Interface zwischen dem I/O-Bus **210** und dem zentralen Controller **302**, um das Programmieren der zentralen Steuereinheit **220** zu ermöglichen sowie weitere Funktionen der zentralen Steuereinheit **220** aufzunehmen, wie unten noch detailliert erläutert wird. Es versteht sich, dass die zentrale Interrupt-Steuereinheit **220** alternativ oder zusätzlich eine CPU-Bus-Interface-Einheit enthalten kann, um zwecks Programmierens und anderer Funktionen den CPU-Lokal-Bus **206** mit der zentralen Steuereinheit **220** zu verbinden.

[0052] Wie bereits erläutert ist die zentrale Steuereinheit **220** in der Lage, Interrupts aus zahlreichen verschiedenen I/O-Einrichtungen zu empfangen. Diese Interrupts werden an mehreren Interrupt-Stiften empfangen, die

mit INTR1-INTRn gekennzeichnet sind, und werden dem Interrupt-Controller **304** zugeführt. Die zentrale Interrupt-Steuereinheit **220** ist derart konfiguriert, dass jedes Interrupt INTR1-INTRn individuell programmiert werden kann, um einen spezifischen Typ von Interrupt zu bezeichnen, einen bestimmten Zuführ-Modus zu bezeichnen und seine Prioritätsebene anzugeben. Zusätzlich kann jeder Interrupt-Stift in einem kaskadierten Modus verwendet werden, um die Anzahl von Interrupt-Signalen zu erweitern, die an dem bestimmten Stift empfangen und identifiziert werden können. Dies wird weiter unten erläutert.

[0053] Der zentrale Controller **220** prioritisiert die verschiedenen Interrupt-Signale und führt sie dem Prozessor-Interrupt-Generator **306** zu, der als Reaktion die Interrupt-Signale einer oder mehrerer der Verarbeitungseinheiten **202-1** bis 202-m zuführt, und zwar basierend auf dem Zuführ-Modus für jedes Interrupt und die aktuelle Task-Priorität jeder Verarbeitungseinheit. Der zentrale Controller **302** hält einen Interrupt-Stapel und eine Einrichtungs-Tabelle für das System aufrecht und hält ferner die aktuellen Task-Prioritäten sämtlicher Verarbeitungseinheiten aufrecht. Der zentrale Controller **220** enthält schließlich einen Mechanismus zum Verteilen gewählter Interrupts, die von sämtlichen Prozessoren auf Broadcast-Weise gehandhabt werden müssen. Dieser Verteilungsmechanismus wird nachstehend detaillierter erläutert.

[0054] Wie bereits erwähnt, leitet der Prozessor-Interrupt-Generator **306** die verschiedenen Interrupts einer (oder mehreren) bezeichneten Ziel-Verarbeitungseinheit(en) zu. Bei dieser Ausführungsform ist die zentrale Interrupt-Steuereinheit **220** derart konfiguriert, dass die Interrupts unter maximal 256 Verarbeitungseinheiten verteilt werden. Die Anzahl der in dem System vorgesehenen Verarbeitungseinheiten wird bei der System-Initialisierung programmiert, wie aus der nachstehenden Beschreibung ersichtlich ist.

[0055] **Fig.** 4 zeigt ein Blockschaltbild eines I/O-Interrupt-Controllers **304**. Der I/O-Interrupt-Controller **304** empfängt über Stifte INTR1, INTR2, ... INTRn Interrupts aus I/O-Einrichtungen. Der I/O-Interrupt-Controller **304** weist mehrere Interrupt-Kanäle **402-1** bis **402-n** auf, die mit den betreffenden Stiften INTR1-INTRn verbunden sind. Ein zentrales Controller-Interface **404** ist mit jedem der Interrupt-Kanäle **402-1** bis **402-n** verbunden. Die Interrupt-Kanäle **402-1** bis **402-n** bilden speziell zugeordnete Kanäle, durch die Interrupts, die an den betreffenden zugehörigen Interrupt INTR1-INTRn empfangen werden, verarbeitet werden. Bei einer Ausführungsform weist der I/O-Interrupt-Controller **220** insgesamt sechzehn Interrupt-Eingangs-Stifte auf, von denen jeder mit sechzehn Interrupt-Signalen kaskadiert werden kann, um ein Maximum von **256** Interrupt-Vektoren zu unterstützen.

[0056] In jedem der Interrupt-Kanäle **402-1** bis **402-n** sind mehrere (in **Fig.** 4 nicht gezeigte) Register vorgesehen, um das Verarbeiten jedes eintreffenden interrupts zu steuern. Diese Register werden entweder in dem Speicherraum oder dem I/O-Raum des Systems abgebildet. Weitere Details zu den internen Registern der Interrupt-Kanäle **402-1** bis **402-n** werden nachstehend aufgeführt.

[0057] Jeder Interrupt-Kanal **402-1** bis **402-n** detektiert die Ausgabe eines Interrupt-Signals an seinem zugehörigen Eingangs-Stift und verarbeitet das Interrupt-Signal, um zu prüfen, ob das Interrupt an die Verarbeitungseinheiten übermittelt werden soll. Interrupt-Eigenschaften können individuell programmiert werden, und keinem der Interrupt-Kanäle **402-1** bis **402-n** ist eine implizierte positionale Abhängigkeit zugefügt.

[0058] **Fig.** 5 ist ein Blockschaltbild zur Veranschaulichung jedes der Interrupt-Kanäle **402-1** bis **402-n**. Der Interrupt-Kanal **402** gemäß **Fig.** 5 weist einen Interrupt-Eingangs-Prozessor **502** auf, der mit einer Registereinheit **504** und einer Interrupt-Akzeptanz-Einheit **506** verbunden ist. Der Interrupt-Eingangs-Prozessor **502** verarbeitet das Interrupt-Signal (oder die Interrupt-Signale, falls der Kaskaden-Modus programmiert ist) am INTR-Stift und bestimmt den Transfer-Modus für kaskadierte Interrupts. Falls der Stift als kaskadierter Stift programmiert ist, wird der Index des aktuellen kaskadierten Interrupts in einem Kaskaden-Interrupt-Adress-(CIS-) Register bestimmt und gespeichert. Anzumerken ist, dass, wenn der Interrupt-Kanal **402** in einem Kaskaden-Modus betrieben wird, die Anzahl der Register-Einheiten **504** und die Anzahl der Interrupt-Akzeptanz-Einheiten **506** efektiv fünfzehn Mal dupliziert werden, so dass für jedes mögliche kaskadierte Interrupt-Signal ein separater Interrupt-Unterkanal vorgesehen ist. Diese Unterkanäle sind in **Fig.** 1 durch unterbrochene Linien gekennzeichnet.

[0059] Jedes Interrupt-Signal ist mit einem programmierbaren Steuerregister **504A**, einem Ziel-CPU-Register **504B**, einem Affinitäts-CPU-Register **504C** und einem ID-(Vektor-) Register **504D** verbunden. Basierend auf der Information in dem Steuerregister **504A** verarbeitet die Interrupt-Akzeptanz-Einheit **506** das Signal am INTR-Stift. Falls das Interrupt ein echtes, aktiviertes und akzeptables Signal ist, wird es an das zentrale Controller-Interface **404** (gemäß **Fig.** 4) weitergegeben, um einer oder mehreren der Verarbeitungseinheiten zugeführt zu werden.

[0060] Wie bereits angeführt, kann jeder INTR-Stift in einem Kaskaden-Modus programmiert werden, in dem der Interrupt-Stift ein kaskadiertes Signal empfangen kann, das sechzehn distinkte Interrupts repräsentiert. Wenn ein Stift als zu kaskadierend programmiert ist, werden dem Interrupt-Kanal sechszehn einzelne Sets von Registereinheiten und Interrupt-Akzeptanz-Einheiten zugewiesen, wie in **Fig.** 5 in unterbrochenen Linien gezeigt ist. Die Kaskaden-Modi der zentralen Steuereinheit **220** werden nachstehend detaillierter erläutert.

[0061] Als nächstes werden die verschiedenen Register jedes Interrupt-Kanals (oder Unterkanals) erläutert. Wie bereits erwähnt, ist für jedes mögliche Interrupt-Signal ein separates Set von Registern vorgesehen. Diese

Register sind bezeichnet als das Steuerregister **504A**, das Ziel-CPU-Register **504B**, das Affinitäts-CPU-Register **504C** und das ID-Register **504D**. Diese Register sind für die Software erfassbar und sind entweder in dem speicherabgebildeten oder dem I/O-abgebildeten System-Raum angeordnet. Für jedes Interrupt-Steuersignal ist ein separates Steuerregister **504** vorgesehen, das programmierbar ist. Das Steuerregister **504A** für jedes Interrupt-Signal definiert und diktiert die Funktionalität jedes INTR-Stifts, und jedes kann durch ein 32-Bit-Register realisiert sein. **Fig.** 6 zeigt die Felder, die dem Steuerregister **504A** zugeordnet sind, und Tabelle 1 beschreibt die verschiedenen Felder der Steuerregisters **504A**. Tabelle 2 gibt die Kaskaden-Modus-Kodierung an, Tabelle 3 gibt die Zufuhr-Modus-Kodierung an, und Tabelle 4 gibt die Status-Bit-Kodierung an. Wie aus den Tabellen 1–4. ersichtlich ist, speichert das Steuerregister **504A** verschiedene Information zum Definieren des Typs von Interrupt-Signal, das dem Kanal zugeführt wird, des Modus des Interrupt-Stifts (d. h. normaler oder Kaskaden-Modus), darüber, ob das Interrupt-Signal derzeit maskiert ist, der dem Interrupt zugewiesenen Prioritätsebene, und des Zuführ-Modus, zusammen mit weiteren Parametern, die dem Interrupt zugeordnet sind.

Tabelle 1: IIC-Steuerregister-Felder

Feld	Name	Bits	Erläuterung
TT	Trigger-Typ	1	Interrupt ist flanken-getriggert oder ebenen-getriggert
IP	Interrupt-Polarität	1	Polarität des Interrupts (aktiv high oder low)
EN	Aktivierung	1	Entsprechendes Interrupt wird verwendet
CSD	Kaskade	1	Aktiviert den Stift als kaskadiertes Interrupt
MSK	Maske	1	Interrupt ist maskiert
CM	Kaskaden-Modus	2	Kaskaden-Interrupt-Vektor-Zuführmodus - vgl. Tabelle 2
PL	Prioritätsebene	4	dem Interrupt zugewiesene Prioritätsebene
DM	Zuführ-Modus	3	Modus des Zuführens des Interrupts - vgl. Tabelle 2
AC	Affinitäts-CPU	1	Dieses Interrupt hat eine Affinitäts-CPU
ST	Status	2	aktueller Status des Interrupts - vgl. Tabelle 4
EOI	EOI aktiviert	1	Aktiviert-EOI-Wartezeit-Zeitgeber für ebenen-getriggerte Interrupts
ISA	ISA-Modus	2	System befindet sich im ISA-Modus

Tabelle 2: Kaskaden-Modus-Definition

CM (1:0)	Definition
00	Seriell-Kodierungs-Modus
01	Seriell-16-Bit-Modus
10	8259-Modus
11	TBD

Tabelle 3: Zuführ-Modus-Definition

DM (2:0)	Definition
000	Fixiert: Ausgeben des Interrupts an die CPU/CPUs im Ziel-CPU-Register
001	Niedrigste Priorität: Ausgeben des Interrupts an den mit niedrigster Priorität arbeitenden Prozessor
010	Broadcast: Zuführen des Interrupts an sämtliche CPUs
011	TBD
100	NMI: Zuführen eines ebenen-getriggerten Interrupts an die Ziel-CPUs als NMI
101	Rücksetzen
110	SMI: Zuführen eines ebenen-getriggerten Interrupts an die Ziel-CPUs als SMI
111	TBD

Tabelle 4: Status-Bit-Definition

ST (1:0)	Definition
00	inaktiv
01	wird bedient (ausgegeben und bestätigt)
10	aus dem CIC ausgegeben (jedoch nicht bestätigt)
11	in der Warteschlange des CC eingereiht

[0062] Nochmals gemäß **Fig.** 5 hängt die Definition des Ziel-CPU-Registers **504B** von dem Zuführ-Modus und dem aktuellen Status des zugehörigen Interrupt-Signals ab. Falls das Interrupt nicht bedient wird, hat das Ziel-CPU-Register **504B** die ID der Verarbeitungseinheit oder Gruppe von Verarbeitungseinheiten, auf die das Interrupt abzielt. Falls das Interrupt bedient wird, hat das Ziel-CPU-Register **504B** die ID der Verarbeitungseinheit, die das Interrupt bedient. Falls der Zuführ-Modus Broadcast lautet oder die niedrigste Priorität hat, führt dieses Register keine zugehörige Bedeutung: Das Affinitäts-CPU-Register **504C** hält die ID der Verarbeitungseinheiten **202-1** bis **202-m** (**Fig.** 2), die das Interrupt zuletzt bedient haben. Das ID-Register **504D** enthält das ID (oder den Vektor) des Interrupts.

[0063] Interrupts werden von der jeweiligen Interrupt-Akzeptanzeinheit **506** verarbeitet, bevor sie über das zentrale Controller-Interface **404** an den zentralen Controller **220** weitergegeben werden. Falls das Interrupt aktiviert ist (EN oder ICR) und nicht maskiert ist (MSK von ICR), wird des dem zentralen Controller **302** zusammen mit der Information über den Zuführ-Modus, die Ziel-Verarbeitungseinheit (falls gegeben), die Prioritätsebene und die Interrupt-ID zugeführt.

[0064] Wie oben angeführt, ermöglicht die Architektur, das jeder Interrupt-Stift INTR1-INTRn der zentralen Interrupt-Steuereinheit **220** entweder als direktes Interrupt oder als kaskadiertes Interrupt programmiert wird. Falls das Kaskaden-Bit in dem Steuerregister **504A** gesetzt ist, nimmt der zugehörige Interrupt-Kanal insgesamt fünfzehn zusätzliche Erweiterungs-Interrupts auf. Jedes dieser Erweiterungs-Interrupts ist mit einem speziell zugeordneten Interrupt-Steuerregister (ICR) **504A**, einem Ziel-CRU-Register **504B**, einem Affinitäts-CPU-Register **504C** und einem ID-Register **504D** verbunden, wie in **Fig.** 5 in unterbrochener Linie gezeigt ist. Außer der Tatsache, dass die CSD-, CM- und ISA-Felder der Erweiterungs-Interrupt-Steuerregister undefiniert sind, sind diese Register mit den in den Tabellen 1–4 definierten identisch.

[0065] Das CM-Feld bestimmt das Verfahren, das zum Zugreifen auf eines von sechszehn Interrupts im Kaskaden-Modus verwendet wird. Der Index eines von sechszehn Interrupts wird durch das CIA-Register definiert, das in dem Interrupt-Eingabe-Prozessor angeordnet ist. Der CM-Modus bestimmt den Mechanismus des Berechnens des CIA.

[0066] **Fig.** 7 und 8 zeigen Hardware-Konfigurationen zum Kaskadieren eines bestimmten Stifts. Anzumerken ist, dass die zentrale Interrupt-Steuereinheit **220** die Interrupt-Erweiterung über drei verschiedene Kaskadierungs-Modi unterstützt. Die ersten beiden Modi ermöglichen, das ein einzelner physischer Interrupt-Stift, INTR1-INTRn die Interrupts mehrerer I/O-Einrichtungen gewählten Interrupt-Unterkanälen zuleitet. Der dritte Modus ermöglicht die Integration eines herkömmlichen Interrupt-Controllers vom Typ 8259 in dem Computersystem. Dies ermöglicht die Kompatibilität mit traditioneller PC-Hardware und -Software.

[0067] Zunächst ist gemäß **Fig.** 7 eine Hardware-Konfiguration für den als "seriell kodiert" bezeichneten Modus gezeigt. **Fig.** 7 ist ein Blockschaltbild zur Darstellung interner Teile des Interrupt-Eingangs-Prozessors **502**, die aktiviert werden, wenn der bestimmte Interrupt-Kanal in den seriell kodierten Kaskaden-Modus gesetzt wird. Gemäß **Fig.** 6 enthält der Interrupt-Eingangs-Prozessor **502** eine Steuereinheit **570**, die mit einer Schiebe-Steuerung **572** und einer seriellen Dekodierschaltung **574** verbunden ist. Die Steuereinheit **570** reagiert auf die CSD- und CM-Felder des entsprechenden Steuerregisters **504A**. Wenn das CM-Feld (d. h. das Kaskaden-Modus-Feld) angibt, dass der aktuelle Modus der seriell kodierte Kaskaden-Modus ist, aktiviert die Steuereinheit **570** die Schiebe-Steuerung **572** und die serielle Dekodierschaltung **574** derart, dass seriell übertragene kodierte Daten an dem entsprechenden Interrupt-Stift INTR dekodiert werden, um Aktivität eines bestimmten Interrupt-Signals zu identifizieren. Der Interrupt-Eingabe-Prozessor **502** ist mit einer Fern-Interrupt-Handhabungseinrichtung **580** verbunden gezeigt, die einen Interrupt-Daten-Kodierer enthält, der mit einem Schieberegister **584** und mit einer parallelen Interrupt-Detektionsschaltung **586** verbunden ist.

[0068] Mehrere Interrupt-Signale 0 bis 15 werden der Parallel-Interrupt-Detektionsschaltung **586** zugeführt. Die Fern-Interrupt-Handhabungseinrichtung **580** nimmt die Einrichtungs-Interrupts an und teilt der zentralen Interrupt-Steuereinheit den Status jedes Interrupt-Signals durch eine kodierte serielle Meldung auf der INTR-Leitung des zugehörigen Interrupt-Kanals mit. Die Parallel-Interrupt-Detektionsschaltung **586** überwacht die Interrupt-Signale, die der Fern-Interrupt-Handhabungseinrichtung **580** zugeführt werden. Falls in Übergang in einem der Interrupt-Signale auftritt, veranlasst der Interrupt-Daten-Kodierer **620**, dass eine kodierte serielle Meldung über das Schieberegister **584** an den Interrupt-Eingabe-Prozessor **502** gesendet wird. Die seriellen Daten werden auf einer Seriell-kodiert-Meldungs-Leitung übertragen, die der INTR-Leitung des Interrupt-Kanals zugeführt wird. Das Schieberegister **624** gibt jedes Mal, wenn eine seriell kodierte Meldung übertragen wird, ein Synchronisier-Schiebe-Taktsignal an den Interrupt-Eingabe-Prozessor **502** aus.

[0069] Gemäß einer Ausführungsform weisen die kodierten Meldungen eine kaskaidierte Interrupt-Signal-Nummer und einen Interrupt-Zustand in einer 6-Bit-Form auf. Ein mögliches Kodierungsschema ist wie folgt ausgelegt:

Bits 5:4	Interrupt-Zustand	00	Interrupt im Übergang auf low
		01	Interrupt im Übergang auf high
		10	reserviert
		11	reserviert

[0070] Bits 3:0 Interrupt-Anzahl für kaskadierte Interrupt-Eingangssignale mit den Nummern 0 bis 15

[0071] Als Beispiel sei eine Situation angenommen, in der das Interrupt-Signal 3 von low auf high übergeht. Dieser Übergang wird von der Parallel-Interrupt-Detektionsschaltung **586** detektiert. Der Interrupt-Daten-Kodierer **580** erzeigt als Reaktion einen kodierten Wert, der den Typ der Transaktion, die erfolgt ist, und das bestimmte Interrupt-Signal angibt, das de Transaktion getätigt hat. Falls beispielsweise das oben angeführte Kodierungsschema verwendet wird, repräsentiert ein kodierter Wert "010111" einen Übergang auf high ("01") in dem Interrupt-Signal 7 ("0111"). Der kodierte Wert wird dann dem Schieberegister **584** zugeführt, das eine serielle Übertragung initiiert, die seitens der Schiebe-Steuereinheit **572** empfangen wird. Die serielle Dekodiereinheit **574** dekodiert dann die empfangene Meldung entsprechend dem Kodierungsschema des Interrupt-Daten-Kodierers **582** und gibt die Meldung an die Interrupt-Akzeptanzeinheit **506** des bezeichneten Interrupt-Unterkanals aus. Die Interrupt-Akzeptanzeinheit **406** für das entsprechende Interrupt-Signal gibt das Interrupt dann an den zentralen Controller **302** weiter, falls das Interrupt aktviert ist (EN des Steuerregisters) und nicht maskiert ist (MSK des Steuerregisters). Ähnlich wie entsprechend der vorherigen Beschreibung wird, wenn eine Interrupt-Akzeptanzeinheit **506** ein Interrupt an den zentralen Controller **302** weitergibt, das Interrupt zusammen mit der Information über den Zuführ-Modus, die Ziel-CPU (falls vorhanden), die Prioritätsebene und die Interrupt-ID für das Interrupt-Signal weitergegeben.

[0072] Gemäß der seriell kodierten Kaskaden-Konfiguration gemäß **Fig.** 6 ist der serielle Kanal nur aktiv, wenn Aktivität auf einem oder mehreren der Interrupt-Signal-Leitungen auftritt, und bildet somit eine für niedrigen Energieverbrauch ausgelegte und elektrisch sehr erweiterungsfähige Technik. Ferner sind in dem Kodie-

rungsschema reservierte Werte vorgesehen, um zusätzliche Typen von Meldungen aufzunehmen.

[0073] **Fig.** 8 ist ein Blockschaltbild einer alternativen Kaskadierungs-Konfiguration. Schaltungsbereiche, die denjenigen gemäß **Fig.** 7 entsprechen, sind mit identischen Bezugszeichen versehen. Bei dieser Konfiguration wird, statt dass ein Wert kodiert wird, der die Aktivität eines bestimmten Interrupt-Signals angibt, der Zustand der Parallel-Interrupt-Detektionsschaltung **586** über das Schieberegister **584** kontinuierlich und direkt dem Interrupt-Eingangs-Prozessor **502** mitgeteilt. Als solches erzeugt das Schieberegister **584** kontinuierlich ein serielles Signal, das den Zustand der Parallel-Interrupt-Detektionsschaltung **586** angibt, und die Schiebe-Steuereinheit **572** konvertiert die Signalübertragung in parallele Daten. Die parallelen Daten werden dann von dem Meldungs-Dekodierer **590** dekodiert, welcher den entsprechenden Interrupt-Akzeptanz-Einheit **506** eine detektierte Interrupt-Signal-Übertragung zusammen mit der zugehörigen Steuer- und Vektor-Information in der entsprechenden Registereinheit **504** zuführt.

[0074] In dieser Konfiguration taktet der Interrupt-Eingangs-Prozessor **502** kontinuierlich das Fern-Interrupt-Schieberegister **584**. Er muss dann verfolgen, zu welchem Interrupt die aktuellen Daten gehören, und diese dem entsprechenden Kanal zuführen. Die Daten sind einfach "interrupt high" und "interrupt low". Der zentrale Controller (oder die Interrupt-Akzeptanz-Einheit **506**) muss dann bestimmen, ob die Daten eine Veränderung im Interrupt-Zustand repräsentieren und somit, welche Maßnahme gegebenenfalls getroffen werden sollte.

[0075] Als nächstes werden weitere Aspekte der zentralen Interrupt-Steuereinheit **220** gemäß **Fig.** 2 erläutert. In **Fig.** 9 ist ein Blockschaltbild gezeigt, das einen Prozessor-Interrupt-Generator **306** zeigt. Wie bereits angeführt, empfängt der Prozessor-Interrupt-Generator **306** Interrupt-Information aus dem zentralen Controller **302** und erzeugt mit INT1-INTm bezeichnete Interrupt-Signale, die den Verarbeitungseinheiten zugeführt werden. Wie in der Figur gezeigt, weist der Prozessor-Interrupt-Generator **306** ein zentrales Controller-Interface **602**, ein Interprozessor-Interrupt-(IPI-) und Software-Interrupt-Register-Set **604** und ein Set von CPU-Kanälen **601-1** bis **606-m** auf. Jede Verarbeitungseinheit in dem System empfängt ein Interrupt von einem zugeordneten CPU-Kanal **601-1** bis **606-m** des Prozessor-Interrupt-Generators **306**. Die CPU-Kanäle **601-1** bis **606-m** empfangen Interrupts von dem zentralen Controller **302** (**Fig.** 3) durch das Controller-Interface **602** und geben diese an die entsprechende(n) Verarbeitungseinheiten) aus.

[0076] **Fig.** 10 ist in Blockschaltbild zur Darstellung jedes der Prozessor-Kanäle **601-1** bis **606-m**. Der CPU-Kanal **6006** gemäß **Fig.** 10 enthält eine CPU-Kanal-Registereinheit **650** und eine Interrupt-Warteschlange **652**, die mit einer Interrupt-Ausgangs-Steuereinheit **654** verbunden ist. Die Interrupt-Ausgangs-Steuereinheit **654** gibt anhängige Interrupts an die entsprechende Verarbeitungseinheit aus.

[0077] Die CPU-Kanal-Registereinheit **650** enthält ein Aktuell-Task-Prioritätsregister **650A**, ein Aktuell-Interrupt-ID-Register **650B**, ein Prozessor-ID-Register **650C** und ein Steuerregister **650D**. Als nächstes werden die Funktionen- und Bit-Definitionen jedes dieser Register beschrieben.

[0078] Jeder Verarbeitungseinheit in dem System ist ein speziell vorgesehenes Steuerregister **650D** zugeordnet, um, wie an der zentralen Interrupt-Steuereinheit **220** ersichtlich, die Funktionalität zu diktieren. Dies sind programmierbare 32-Bit-Register, die entweder im I/O- oder im Speicher-Raum des Systems abgebildet. sind. **Fig.** 10A zeigt ein CPU-Kanal-Steuer-(CIG-) Register **650D** zusammen mit seinen zugehörigen Feldern, und die Tabellen 5 bis 7 beschreiben jedes der Felder in dem Steuerregister.

Tabelle 5: CIG-Steuerregister-Felder

Feld	Name	Bits	Erläuterung
EN	Aktivierung	1	Dieser Kanal wird gerade benutzt
IEN	Interrupt-Aktivierung	1	Die mit diesem Kanal verbundene CPU akzeptiert die Interrupts
ITM	Interrupt-Übertragungs-Modus	2	Mechanismen zum Ausgeben der Interrupts an die CPU - vgl. Tabelle 6
RMI	Real-Modus-Interrupt	1	Dieses Interrupt wird im "Real-Modus" zugeführt
IML	Interrupt-Masken-Ebene	4	Maskieren sämtlicher Interrupts auf oder unter dieser Prioritätsebene
IST	Interrupt-Status	2	Reflektiert den Status des Interrupts auf diesem Kanal - vgl. Tabelle 7

Tabelle 6: Interrupt-Übertragungsmechanismus-Definition

ITM(1:0)	Definition
00	Zuführen der Interrupts auf separaten Stiften
01	Zuführen der Interrupts auf dem CPU-Bus
10	Verwenden des speziell vorgesehenen Interrupt-Zuführ-Busses (z.B. 4 Bit breit) zum Zuführen der Interrupts
11	TBD

Tabelle 7: Interrupt-Status-Definition

IST(1:0)	Definition
00	Kein Interrupt wird bedient
01	Interrupt wird von der CPU bedient
10	Interrupt an die CPU ausgegeben, noch nicht bestätigt
11	TBD

[0079] Das Prozessor ID-Register **650C** enthält die ID der Verarbeitungseinheiten **202-1** bis **202-m**, die dem speziellen Kanal zugeordnet sind. Das aktuelle Interrupt-ID-Register **650B** ist vorgesehen zum Speichern der ID (Vektor) des Interrupts, das von der mit dem Kanal verbundenen Verarbeitungseinheit bedient wird. Das Aktuell-Interrupt-Register **650B** ist nur gültig, wenn das Status-Feld des Steuerregisters anzeigt, das ein Interrupt bedient wird. Das Aktuell-Task-Prioritäts-Register **650A** reflektiert die Priorität des Tasks, das von der dem Kanal zugewiesenen Verarbeitungseinheit ausgeführt wird.

[0080] Wie nochmals aus **Fig.** 6 ersichtlich ist, stellt das Interprozessor-Interrupt- und Software-Interrupt-Register-Set **604** ein Set von Registern bereit, auf das an den gleichen Stellen von sämtlichen CPU-Kanälen zugegriffen werden kann. Der Platz in diesem Register bietet jedem CPU-Kanal einzigartige Register-Ansichten, indem die Prozessor-ID als Index verwendet wird. Somit werden, wenn zwei Prozessoren Lese-/Schreib-Zyklen für diese Register erzeugen, die an der gleichen logischen Stelle abgebildet sind, die Prozessoren tatsächlich auf zwei separate physische Register zugreifen. Die Verarbeitungseinheiten schreiben diese Register zum Initiieren von Interprozessor-Interrupts oder zum Planen von Software-Interrupts.

[0081] **Fig.** 10B zeigt das Interprozessor-Interrupt-(IPI-) Register-Format und seine Felder. Sämtliche IPI-Register unterliegen dem Zugriff durch die Software entweder an einer I/O-Stelle oder einer Speicherstelle des Systems. Die ID der Verarbeitungseinheit wird als ein Index verwendet, um zu bestimmen, auf welches Register zugegriffen wird. Die Tabellen 8 bis 11 enthalten Beschreibungen der verschiedenen Felder in jedem IPI-Register.

Tabelle 8: CIG-IPI-Register-Felder

Feld	Name	Bits	Erläuterung
TT	Trigger-Typ	1	Interrupt wird flanken-getriggert oder ebenen-getriggert
IP	Interrupt-Polarität	1	Polarität des Interrupts (aktiv high oder low)
EN	Aktivierung	1	Entsprechendes Interrupt wird gerade verwendet
ST	Status	2	Aktueller Status des IPI - vgl. Tabelle 9
DTC	Ziel-Code	2	Ziel-Code des IPI - vgl. Tabelle 10
PL	Prioritätsebene	4	dem Interrupt zugeordnete Prioritätsebene
DM	Zuführ-Modus	3	Modus zum Zuführen des Interrupts - vgl. Tabelle 11
DID	Ziel-ID	8	Ziel-Prozessor-ID, für den das IPI planmäßig bestimmt ist
SID	Ausgangs-ID	8	ID des Ausgangs-Prozessors

Tabelle 9: IPI-Register-Status-Feld-Definition

ITM(1:0)	Definition
00	inaktiv
01	wird bedient (ausgeführt und bestätigt)
10	ausgegeben aus dem CIC (jedoch nicht bestätigt)
11	in der Warteschlange des CC aufgereiht

Tabelle 10: IPI-Register-Ziel-Code-Feld-Definition

DTC(1:0)	Definition
00	Ziel-ID-Feld (DID) des IPIU-Registers
01	selbst
10	Broadcast
11	sämtliche Prozessoren außer demselben

Tabelle 11: IPI-Register-Zuführ-Modus-Feld-Definition

DTC(2:0)	Definition
000	Festgelegt: Zuführen des Interrupts zu der/den CPU(s) in dem Ziel-CPU-Register
001	Unterste Priorität: Zuführen des Interrupts zu dem mit niedrigster Priorität arbeitenden Prozessor
010	TBD
011	TBD
100	NMI: Zuführen eines ebenen-getriggerten Interrupts zu sämtlichen CPUs als NMI
101	Rücksetzen:
110	TBD
111	TBD

[0082] Eine Verarbeitungseinheit führt einen Schreibvorgang zu seinem IPI-Register durch, wenn sie ein Interprozessor-Interrupt geplant hat. Falls eine Verarbeitungseinheit mehrere Interprozessor-Interrupts planen kann, muss sie das ST-(Status-) Feld des IPI-Registers überwachen. Falls dieses Feld inaktiv ist, dann kann die Verarbeitungseinheit ein Interprozessor-Interrupt in das System einführen. Falls eine Verarbeitungseinheit ein Interprozessor-Interrupt einführt, ohne den Status des IPI-Registers zu prüfen, und falls das ST-Feld nicht inaktiv ist, dann sind die Ziele des aktuellen Interprozessor-Interrupts und jedes zuvor geplanten Interprozessor-Interrupts unbestimmt. Anzumerken ist, dass ferner für jeden Prozessor-Kanal ein Software-Interrupt-Register vorgesehen sein kann, das ein Format hat, welches mit dem für das Interprozessor-Interrupt-Register spezifizierten identisch ist. Bei Software-Interrupts jedoch wird ein angefordertes Interrupt nur der interrupt-anfordernden Verarbeitungseinheit zugeführt.

[0083] Nochmals anhand von **Fig.** 2 werden als nächstes Einzelheiten zum Hochfahren des Computersystems **200** sowie zur Initialisierung der verschiedenen Konfigurationsregister in der zentralen Interrupt-Steuereinheit **220** erläutert. Während der System-Konfiguration wird eine der Verarbeitungseinheiten **202-1** bis **202-m** als die "Boot"-Verarbeitungseinheit bezeichnet.

[0084] In der folgenden Erläuterung wird angenommen, dass die Verarbeitungseinheit **202-1** als Boot-Verarbeitungseinheit bezeichnet worden ist. **Fig.** 11 ist ein Flussdiagramm zur Veranschaulichung der Arbeitsweise des Boot-Verarbeitungseinheit beim System-Rücksetzen. Während eines Schritts **852** beginnt die Verarbeitungseinheit **202-1** einen Energieeinschalt-Selbsttestvorgang und einen Initialisierungsvorgang. Anzumerken ist, dass anfangs die Verarbeitungseinheiten **202-1** bis **202-m** von der zentralen Steuereinheit **220** im Rücksetzzustand gehalten werden. Während des Schritts **854** initialisiert die Verarbeitungseinheit **202-1** die Registereinheit **504** (d. h. das Steuerregister **504A** und das ID-(Vektor-) Register **504D**) jedes Interrupt-Kanals. Wie bereits erwähnt, ist die Registereinheit **504** jedes Interrupt-Kanals in dem I/O- oder Speicher-Raum des Computersystems abgebildet. Jedes Register jedes Interrupt-Kanals ist mit einer vorbestimmten und einzigartigen Adresse bezeichnet. Die Initialisierungsdaten, die der Registereinheit **504** jedes Interrupt-Kanals zugeführt werden, werden typischerweise in dem BIOS-Code des Hauptspeichers **204** gespeichert. Als solcher ist der BIOS-Code zum Initialisieren der Register-Einheit **504** jedes Interrupt-Kanals abhängig von der bestimmten System-Konfiguration (d. h. von Anzahl und Typ der interrupt-erzeugenden Ressourcen) und muss von der Systemprogrammiereinrichtung geliefert werden.

[0085] Der CPU-Kanal **606-1** der zentralen Interrupt-Steuereinheit **220**, der die Verbindung zur Verarbeitungseinheit **202-1** herstellt, muss ebenfalls initialisiert werden. Anzumerken ist jedoch, dass das Aktuell-Task-Prioritäts-Register **650A**, das. Aktuell-Interrupt-ID-Register **650B**, das Prozessor-ID-Register **650C** und das Steuerregister **650D** für einen bestimmten CPU-Kanal **606-1** an den gleichen System-Adress-Stellen (entweder I/O- oder Speicher-Raum) wie die entsprechenden Register für die anderen CPU-Kanäle angeordnet und abgebildet sind. Dies bedeutet, dass die Adresse des Aktuell-Task-Prioritäts-Registers **650A** für jeden CPU-Kanal **606-1** bis **606-m** identisch ist. Gleichermaßen ist die Adresse des Aktuell-Interrupt-ID-Registers **650B** für jeden CPU-Kanal identisch, was auch für die Adress-Werte für das Prozessor-ID-Register **650C** und das Steuerregister **650D** jedes CPU-Kanals gilt. Somit ist jeder Verarbeitungseinheit **202-1** bis **202-m** ein Verarbeitungseinheit-ID-Wert zugeordnet, der in einen bezeichneten Befehl zum Initialisieren oder Aktualisieren der CPU-Kanal-Registereinheit **650** jedes CPU-Kanals **606-1** bis **606-m** eingebettet ist. Dies wird im Folgenden detaillierter erläutert.

[0086] **Fig.** 12 ist ein Schaubild mit einer detaillieren Darstellung der Hardware, die ermöglicht, die CPU-Kanal-Registereinheit **650** jedes CPU-Kanals **606-1** bis 606-m während der normalen Ausführung zu initialisieren

und zu aktualisieren. **Fig.** 12 zeigt ein ID-Register **902-1**, das mit der Verarbeitungseinheit **202-1** verbunden ist. Identische ID-Register **902-2** bis **902-m** sind ebenfalls mit den Verarbeitungseinheiten **202-1** bis **202-m** verbunden. Jedes ID-Register **902-2** bis **902-m** enthält einen Wert, der die bestimmte Verarbeitungseinheit eindeutig identifiziert. Der ID-Wert jeder Verarbeitungseinheit kann ein hartverdrahteter Wert sein oder kann während der System-Konfiguration zugeführt werden. Falls beispielsweise fünfzehn Verarbeitungseinheiten in dem System angeschlossen sind, können sich die ID-Werte in den ID-Registern **902-1** bis **902-16** jeweils von 0 bis 15 bewegen. Auf jedes ID-Register **902-1** bis **902-m** kann durch einen Software-Befehl über zugehörige Steuer-Dekodieren **904-1** bis **904-m** jeder Verarbeitungseinheit zugegriffen werden. Die ID-Register können in jedem Speicher- oder I/O-Raum abgebildet werden. Anzumerken ist jedoch, dass jede Verarbeitungseinheit auf ihr entsprechendes ID-Register **902** über den gleichen Adress-Wert zugreift. Beispielsweise kann das ID-Register **902** jeder Verarbeitungseinheit **202-1** bis **202-m** an einer Speicherstelle 2000:H abgebildet werden. Somit wird, falls ein bezeichneter Verarbeitungs-Kern **204-1** bis **204-m** einen Lese-Zyklus an die Speicherstelle 2000:H durchführt, der in dem entsprechenden ID-Register **902** für diese Verarbeitungseinheit befindliche Wert an den Verarbeitungs-Kern ausgegeben. In diesen Situationen liest jeder Verarbeitungs-Kern einen einzigartigen Wert.

[0087] **Fig.** 13 zeigt die in der zentralen Interrupt-Steuereinheit **220** ausgebildete Schaltungsanordnung, die das Schreiben (oder Lesen) von Daten aus der jeweiligen CPU-Kanal-Registereinheit **650** jedes CPU-Kanals **606-1** bis **606-m** ermöglicht. Bei dieser Darstellung wird angenommen, dass insgesamt sechzehn Verarbeitungseinheiten in dem System angeschlossen sein können; es versteht sich jedoch, das die Schaltung alternativ zur Aufnahme von z. B. **256** einzigartigen Verarbeitungseinheiten konfiguriert sein kann. **Fig.** 13 zeigt die CPU-Kanal-Registereinheiten **650-1** bis **650-16** für die separaten sechzehn CPU-Kanäle. Wie bereits erwähnt, enthält jede CPU-Kanal-Registereinheit **650-1** bis **650-16** ein Aktuell-Task-Prioritätsregister **650A**, ein Aktuell-Interrupt-ID-Register **650B**, ein Prozessor-ID-Register **650C** und ein Steuerregister **650D**. Jedes dieser Register ist zum Empfangen (oder Ausgeben) von Daten aus den Datenleitungen des CPU-Lokal-Busses **207** geschaltet. Ferner ist eine 4-zu-16-Dekodierschaltung **920** an ihren Eingängen mit gewählten Datenleitungen des CPU-Lokal-Busses **207** verbunden. Die Ausgänge der 4-zu-16-Dekodierschaltung **920** sind mit jeweiligen Selekt-Leitungen der CPU-Kanal-Registereinheiten **650-1** bis **650-16** verbunden. Anzumerken ist, dass die CPU-Kanal-Registereinheit **650-1** bis **650-16** aus der Dekodierschaltung **920**. ein separates Selekt-Signal erhält. Ferner ist ein Adress-Dekodierer **922** an seinem Eingang mit den Adress-Leitungen des CPU-Lokal-Busses **207** verbunden. An einem Ausgang des Adress-Dekodierers sind vier Latch-Aktivierungs-Leitungen vorgesehen. Mit den Latch-Aktivierungs-Eingängen jedes Aktuell-Task-Prioritätsregisters **650A** der CPU-Kanal-Registereinheiten **650-1** bis **650-16** ist eine Adress-Dekodierer-Aktivierungs-Leitung, und in ähnlicher Weise sind mit jedem Aktuell-Interrupt-ID-Register **650B**, jedem Prozessor-ID-Register **650C** und jedem Steuerregister **650D** Adress-Dekodierer-Aktivierungs-Leitungen verbunden.

[0088] Entsprechend der Hardware-Implementierung gemäß **Fig.** 12 und 13 kann, wenn die CPU-Kanal-Registereinheit **650** eines bezeichneten CPU-Kanals **606-1** bis **606-m** initialisiert oder aktualisiert werden muss, die Betriessystem-Programmiereinrichtung den Code derart ausführen, dass der Prozessor-ID-Wert in einem bezeichneten ID-Register **902** als Index eingebettet ist, um angefügte Daten den korrekten CPU-Kanal-Registereinheiten **650** zuzuführen. Es sei beispielsweise eine Situation angenommen, in der das ID-Register **902** jeder Verarbeitungseinheit **202-1** bis **202-m** an einer Speicherstelle 2000:H abgebildet wird und bei der das Steuerregister **650D** jedes CPU-Kanals **606-1** bis **606-m** an einer I/O-Adresse 3000:H abgebildet wird. Falls das Betriebssystem die Konfigurations-Information in dem Steuerregister **650D** für eine bestimmte Verarbeitungseinheit aktualisieren muss, kann die Programmiereinrichtung zuerst die bezeichnete Verarbeitungseinheit dazu veranlassen, an einen Speicher-Lese-Zyklus an der Speicherstelle 2000:H auszuführen, um den Wert in dem ID-Register der bestimmten Verarbeitungseinheit zu lesen. Die Programmiereinrichtung kann dann einen Befehl zum Anfügen des ID-Werts an die Konfigurations-Daten veranlassen, die in dem zugeordneten Steuerregister **650D** gespeichert werden sollen. Anschließend wird ein I/O-Schreib-Befehl an die Adress-Stelle 3000:H ausgeführt, um die kombinierte Information (d. h. die Konfigurations-Daten zusammen mit dem Prozessor-ID-Wert) zu schreiben. Dieser I/O-Zyklus wird durch den Adress-Dekodierer **922** dekodiert, der als Reaktion veranlasst, dass die Steuerregister **650D** jeder CPU-Kanal-Registereinheit **650-1** bis **650-16** aktiviert werden. Der Prozessor-ID-Wert, der an die Konfigurations-Daten angefügt ist, wird dann von dem 4-zu-16-Bit-Dekodierer **920** dekodiert, der ein Selekt-Signal an eine gewählte der CPU-Kanal-Registereinheiten **650-1** bis **650-16** ausgibt. Dies bewirkt, dass die Konfigurations-Daten nur in den gewählten und aktivierten Registern gespeichert werden. Die Konfigurations-Daten werden dadurch an den bezeichneten CPU-Kanal **606-1** bis **606-m** ausgegeben, ohne dass separate, speziell vorgesehene Adress-Stellen für die Konfigurations-Register jedes CPU-Kanals erforderlich sind. Anzumerken ist, dass Zyklen zum Aktualisieren der Register jedes CPU-Kanals sowie Lese-Zyklen gleichzeitig durchgeführt werden. Ein Beispiel eines Codes, der die erforderliche Prozessor-ID-Lese-Operation durchführt, sowie eines Codes zum Anfügen der ID an die Konfigurations-Daten und zum Schreiben der Konfigurations-Daten an einen bezeichneten CPU-Kanal ist wie folgt ausgelegt:

OS Code to Init a given CPU's control reg

```
procedure init_channel

    id: = get mem(2000) ; - read ID register
    command: = id << 28 or config data
    put io (3000), command
    end
```

[0089] Wie nochmals anhand von **Fig.** 11 ersichtlich ist, muss, nachdem die Boot-Verarbeitungseinheit die I/O-Kanäle in der zentralen Interrupt-Steuereinheit **220** initialisiert hat (die Konfigurations-Register jedes I/O-Kanals werden an speziell vorgesehenen Stellen separat von den Konfigurations-Registern der anderen I/O-Kanäle abgebildet), die Boot-Verarbeitungseinheit **202-1** den CPU-Kanal **606-1** initialisieren. Dies wird erreicht durch Verwendung des oben im Zusammenhang mit **Fig.** 12 und 13 beschriebenen Verfahrens. Somit liest während des Schritts **856** die Verarbeitungseinheit **202-1** ihr entsprechendes ID-Register **902**. Während des Schritts **858** fügt die Verarbeitungseinheit **202** ihren ID-Register-Wert an die gewünschten Konfigurations-Daten an, die in einem bezeichneten Register der CPU-Kanal-Registereinheit **650** gespeichert werden muss. Die Verarbeitungseinheiten **202-1** führt dann einen Zyklus zum Schreiben der kombinierten Daten in das gewählte Register der CPU-Kanal-Registereinheit **650** durch. Anzumerken ist, dass während dieses Zyklus der 4-zu-16-Dekodierer **920** gemäß **Fig.** 13 verwendet wird, um die Registereinheit **650-1** des CPU-Kanals **906-1** zu wählen. Ähnliche Operationen können initialisiert werden, um weitere Initialisierungs-Daten in andere Register der CPU-Kanal-Registereinheit **650** des CPU-Kanals **606-1** zu schreiben. Für eine Implementierung werden das Aktuell-Task-Prioritätsregister **650A** und das Steuerregister **650D** durch die Verarbeitungseinheit **202-1** während der Initialisierungs-Sequenz mit Initialisierungs-Daten beschrieben. Nachdem die Verarbeitungseinheit **202-1** ihren CPU-Kanal **606-1** initialisiert hat, gibt die Verarbeitungseinheit **202-1** an die zentrale Interrupt-Steuereinheit **220** einen Befehl aus, der die zentralen Interrupt-Steuereinheit **220** dazu veranlasst, die übrigen Verarbeitungseinheiten **202-1** bis **202-m** aus dem Rücksetzzustand freizugeben (Schritt 862). Anschließend (Schritt **864**) wartet die Boot-Verarbeitungseinheit **202-1** darauf, dass die Slave-Initialisierungs-Sequenz gemäß **Fig.** 14 abgeschlossen wird.

[0090] **Fig.** 14 ist ein Flussdiagramm, das die Initialisierungs-Sequenz jeder der Slave-Verarbeitungseinheiten **202-2** bis **202-m** zeigt. Wenn die Master-Verarbeitungseinheit **202-1** die zentrale Interrupt-Steuereinheit **220** dazu veranlasst, die übrigen Verarbeitungseinheiten aus dem Rücksetzzustand freizugeben, liest jede Verarbeitungseinheit **202-1** bis **202-m** das betreffende ihr zugeordnete ID-Register **902-2** bis **902-m** während des Schritts **870**, fügt den ID-Wert mit dem im Steuerregister **650D** zu speichernden Konfigurations-Daten hinzu (Schritt **872**) und schreibt die kombinierten Daten in die CPU-Kanal-Registereinheit **650** (Schritt **874**). Die Dekodiereinheit **920** gemäß **Fig.** 13 ist während dieser Zyklen dahingehend aktiv, dass sie die korrekte CPU-Kanal-Registereinheit **650-1** bis **650-16** entsprechend der während jedes bestimmten Zyklus identifizierten Prozessor-ID wählt. Ähnliche Operationen werden initialisiert, um das Task-Prioritätsregister **650A** jedes CPU-Kanals zu initialisieren (Schritt **876**). Anzumerken ist wiederum, dass, da ein eindeutiger ID-Wert, der in jedem der ID-Register **902-2** bis **902-m** enthalten ist, den in jedes Register der CPU-Kanal-Registereinheit **650** geschriebenen Daten hinzugefügt wird, jede Verarbeitungseinheit **202-1** bis 202-m ihre eigene CPU-Kanal-Konfiguration durchführt.

[0091] Es wurde bereits angeführt, wie nochmals gemäß **Fig.** 2 und 4 ersichtlich, das jeder bezeichnete Interrupt-Kanal **402-1** bis **402-n** in einem Modus programmierbar ist, der als "8259"-Modus bezeichnet wird. Dies macht es möglich, dass ein programmierbarer Interrupt-Controller wie der Interrupt-Controller **216** mit der zentralen Interrupt-Steuereinheit **220** verbunden wird, wenn ein bestimmter Interrupt-Kanal im 8259-Modus programmiert wird (wie durch das CM-Feld des zugeordneten Steuerregisters **504A** angezeigt). Währen der 8259-Betriebsart wird das Interrupt-Signal aus dem 8259-Interrupt-Controller entsprechend seiner programmierten Priorität durch die zentrale Interrupt-Steuereinheit **220** geschickt, und das Bestätigungs-Signal von der Empfangs-Verarbeitungseinheit **202-1** bis **202-m** wird zurück durch die zentralen Interrupt-Steuereinheit **220** zu dem Interrupt-Controller **216** geschickt. Dies ist in **Fig.** 15 dargestellt die eine der Verarbeitungseinheiten **202** zeigt, welche mit der zentralen Interrupt-Steuereinheit **220** und mit einem Puffer **219** verbunden ist. Wenn ein ISA-interrupt an einem Interrupt-Eingang des 8259-Interrupt-Controllers **216** empfangen wird, werden die Interrupts durch die zentrale Interrupt-Steuereinheit **220** hindurchgeleitet und entsprechend der programmier-

ten Prioritätsebene und weiteren zentralen Steuerungs-Leit-Aspekten an eine bezeichnete Verarbeitungseinheit **202** übermittelt. Wenn die bezeichnete Verarbeitungseinheit **202** den Zyklus bestätigt, wird das Interrupt-Bestätigungs-Signal INTA durch die zentrale Interrupt-Steuereinheit **220** hindurchgeschickt und wird an der Interrupt-Bestätigungs-Leitung des 8259-Interrupt-Controllers **216** empfangen. Der Interrupt-Controller **216** steuert als Reaktion den Interrupt-Vektor auf einem X-Bus **210** (oder einem anderen Bus), und der Interrupt-Vektor wird über einen Puffer **219** an die Verarbeitungseinheit **202** übermittelt. Anzumerken ist, dass der Puffer **219** in einer Bus-Brücke **208** eingebettet sein kann. Somit reagiert bei dem 8259-Kaskaden-Modus die zentrale Interrupt-Steuereinheit **220** nicht direkt auf den Interrupt-Bestätigungs-Zyklus der Empfangs-Verarbeitungseinheit **202** und erlaubt stattdessen, dass die Vektor-Information aus dem 8259-Interrupt-Controller 216 zugeführt wird. Anzumerken ist ferner, dass die wie oben beschrieben vorgesehene Aufnahme des 8259-Kaskadierungs-Modus vorteilhafterweise die Verwendung integrierter Interrupt-Quellen erlaubt, wie z. B. der IC-Schaltung vom Modell 82C206, die einen System-Zeitgeber **834** und einen Echtzeit-Taktgeber **835** enthält.

[0092] Nochmals gemäß **Fig.** 2 kann das Mehrfachverarbeitungssystem **200** ferner derart konfiguriert sein, dass es die Übertragung von Interrupt-Information über verschiedene Interfaces unter Verwendung eines speziell definierten Zyklus ermöglicht, der über einen oder mehrere der im System enthaltenen Busse übermittelt wird. Dies ist am besten im Zusammenhang mit **Fig.** 16 verständlich. **Fig.** 16 ist ein Blockschaltbild des in **Fig.** 2 generell gezeigten Mehrfachverarbeitungssystems mit einer zusätzlichen I/O-Einrichtung **280**, die mit einer zweiten I/O-Brücke **282** verbunden ist. Der zweite I/O-Bus **282** ist über eine Bus-Brücke **284** mit dem I/O-Bus 210 verbunden. Die Bus-Brücke **284** kann beispielsweise als Andockstation zum Anschließen eines tragbaren Computers vorgesehen sein, wie er durch die I/O-Einrichtung **280** des Mehrfachverarbeitungssystems repräsentiert ist. Bei dem System gemäß **Fig.** 16 kann die I/O-Einrichtung **280** ein Interrupt-Signal an die Bus-Brücke **284** ausgeben. Aufgrund der Kosten und der möglichen Nichtverfügbarkeit speziell vorgesehener Interrupt-Stifte, welche die Bus-Brücke **284** mit der zentralen Interrupt-Steuereinheit **220** verbinden, kann die Bus-Brücke **284** jedoch möglicherweise nicht so konfiguriert sein, dass sie auf einer speziell zugeordneten Leitung ein Interrupt-Signal ausgibt, das von der zentralen Interrupt-Steuereinheit **220** empfangen wird. Stattdessen kann auf die Ausgabe eines Interrupts von der I/O-Einrichtung **280** hin die Bus-Brücke **284** einen spezialisierten Zyklus oder einen Speicher- oder I/O-Zyklus für eine speziell zugeordnete Speicherstelle durchführen, auf den ein bestimmter Interrupt-Kanal der zentralen Interrupt-Steuereinheit **220** reagiert. **Fig.** 17 zeigt einen Interrupt-Kanal einschließlich der Hardware, die konfiguriert ist zum Dekodieren eines Interrupt-Zyklus, wie er von der Bus-Brücke **284** ausgeführt wird, und zum Ausgeben eines entsprechenden Interrupt-Signals in einem entsprechenden Interrupt-Kanal. Gemäß **Fig.** 17 reagiert eine Steuereinheit **290** auf ein CM-Feld des Steuerregisters **504A** und aktiviert entsprechend den Interrupt-Zyklus-Dekodierer **292**, falls der Interrupt-Kanal-Modus als ein I/O-Bus-Modus bezeichnet ist. Wenn die Steuereinheit **290** den Interrupt-Zyklus-Dekodierer **292** aktiviert, wird der von der Bus-Brücke **284** erzeugte spezialisierte Interrupt-Zyklus von dem Interrupt-Zyklus-Dekodierer **292** detektiert, der entsprechend ein Interrupt-Signal an dem INTR-Eingang des Interrupt-Eingabe-Prozessors **502** aktiviert. Anzumerken ist, dass in Situationen, in denen der I/O-Bus **210** ein PCI-Standard-Konfigurations-Bus ist, der spezielle Interrupt-Zyklus durch vor-spezifizierte Kodierung der Zyklus-Definitions-Bits des PCI-Busses definiert sein kann. Alternativ kann der spezielle Interrupt-Zyklus als ein Zyklus für eine vorbestimmte Adresse im Speicher- oder I/O-Raum des Systems definiert sein.

[0093] Als nächstes wird die Prioritisierung von Interrupts durch den zentralen Controller **302** erläutert. **Fig.** 18 ist ein Blockschaltbild, das einen Teil der zentralen Interrupt-Steuereinheit **220** einschließlich des I/O-Interrupt-Controllers **304**, des zentralen Controllers **302** und des Prozessor-Interrupt-Controllers **306** zeigt. Statt automatisch ein bestimmtes Interrupt je nach seinem Interrupt-Vektor mit einer festen Priorität zu versehen, erlaubt die zentrale Interrupt-Steuereinheit **220** jedem Interrupt, einen separaten programmierbaren Interrupt-Vektor und eine separate Priorität zu haben. Die Priorität wird in dem PL-Feld des zugeordneten Steuerregisters **504A** für den Interrupt-Kanal gespeichert. Wie bereits erwähnt, wird bei der System-Initialisierung der Vektor für jeden Interrupt-Kanal gesetzt. Zudem wird auch die Prioritätsebene für den Interrupt-Kanal gesetzt. Nachdem ein bestimmtes Interrupt-Request durch den I/O-Interrupt-Controller **304** akzeptiert worden ist, werden der Interrupt-Vektor und die Prioritäts-Daten durch einen Interrupt-Planer **305** des zentralen Controllers **302** verarbeitet, der entsprechend den Interrupt-Vektor und die Prioritäts-Daten für jedes Interrupt in einer anhängigen Interrupt-Warteschlange **652** (**Fig.** 10) eines bezeichneten CPU-Kanals des Prozessor-Interrupt-Generators **306** liefert. Die Interrupts werden den verschiedenen Interrupt-Warteschlangen der CPU-Kanäle auf prioritisierte Weise zugeführt, und zwar basierend auf der vom Interrupt-Steuerregister angegebenen Prioritätsebene sowie basierend auf den aktuellen Task-Prioritäten der verfügbaren Verarbeitungseinheiten. **Fig.** 18 zeigt eine Dekodiereinheit **309** in der I/O-Bus-Interface-Einheit **308**, die ein separates Programmieren der Vektor-Information und der Prioritäts-Information für jedes Interrupt über bezeichnete I/O- oder Speicher-Zyklen auf dem I/O-Bus **210** ermöglicht. Bei einer einfachen Konfiguration werden die anhängigen Interrupt-Requests den Interrupt-Warteschlangen der CPU-Kanäle in ihrer prioritisierten Reihenfolge zugeführt.

[0094] Es wurde, wiederum gemäß **Fig.** 2, bereits festgestellt, dass möglicherweise gewählte Interrupts wie z. B. ein Zeitgeber-Ablauf-Interrupt auf Boadcast-Weise an jede Verarbeitungseinheit **202-1** bis **202-m** ausge-

geben werden müssen. Falls der Zuführ-Modus für ein bestimmtes Interrupt-Signal. als ein Broadcast-Modus bezeichnet wird (d. h. das DM-Feld des Steuerregisters **504A** für das Interrupt), arbeitet der zentrale Controller **220** entsprechend der Auto-Kettenbildungs-Technik, wie in dem Flussdiagramm gemäß **Fig.** 19 dargestellt. Gemäß **Fig.** 19 wird, falls das Interrupt-Anforderungs-Signal für ein als Broadcast bezeichnetes Interrupt entsprechend der Bestimmung gemäß Schritt **470** aktiviert wird, wird während des Schritts **472** das Interrupt an die Verarbeitungseinheiten **202-1** bis **202-m** ausgegeben, welche die niedrigste aktuelle Task-Prioritätsebene aufweist (wie durch das Aktuell-Task-Prioritätsregister **650A** für diesen CPU-Kanal angezeigt). Während des Schritts **474** bedient die bezeichnete Verarbeitungseinheit das Interrupt und gibt einen End Of Interrupt-(EOI-) Befehl an den zentralen Controller **302** zurück. Falls sämtliche Verarbeitungseinheiten **202-1** bis **202-m** das Interrupt noch nicht empfangen haben (Schritt **476**), wird das Interrupt an die nächste Verarbeitungseinheit **202-1** bis **202-m** ausgegeben, die dieses Interrupt noch nicht empfangen hat und die den niedrigsten aktuellen Task-Prioritätswert hat (unter denjenigen übrigen Verarbeitungseinheiten, die das Interrupt noch nicht empfangen haben). Diese nächste Verarbeitungseinheit bedient dann das Interrupt und gibt einen End 0f Interrupt-Befehl an den zentralen Controller **302** zurück. Dieser Vorgang wird wiederholt, bis jede Verarbeitungseinheit **202-1** bis **202-m** das Interrupt bedient hat. Wenn sämtliche Verarbeitungseinheiten das Interrupt empfangen und bedient haben, wird das Busy-Bit für das Interrupt (d. h. der Inaktiv-Status des ST-Felds des Steuerregisters **504A** für den Interrupt-Kanal) gelöscht (Schritt **478**). Es wird darauf hingewiesen, das ein ähnlicher Auto-Kettenbildungsvorgang ausgeführt werden kann, falls statt sämtlicher Verarbeitungseinheiten ein bezeichnetes Set von Verarbeitungseinheiten ein bestimmtes Interrupt empfangen muss. Gemäß der oben beschriebenen Auto-Kettenbildungs-Technik, bei der Interrupts wie z. B. Zeitgeber-Ablauf-Interrupts an zwei oder mehr der Verarbeitungseinheiten ausgegeben werden, gibt der zentrale Controller auf intelligente Weise das Interrupt selektiv an die Verarbeitungseinheit aus, für welche die niedrigste aktuelle Prioritätsebene angezeigt ist. Das Interrupt wird nicht an die nachfolgenden Verarbeitungseinheiten ausgegeben, bis die vorherige Verarbeitungseinheit die von ihr durchgeführte Bedienung des Interrupts beendet hat. Folglich wird die Bus-Konkurrenz minimiert, und die System-Leistung wird maximiert, indem die Verarbeitungseinheiten mit den niedrigsten aktuellen Prioritäts-Werten unterbrochen werden, bevor Verarbeitungseinheiten mit relativ hohen Task-Prioritäten unterbrochen werden.

[0095] Die zentrale Interrupt-Steuereinheit **220** ist ferner konfiguriert zum Verhindern des Auftretens unechter Interrupts. Wie bereits erwähnt, wird, wenn ein ebenen-getriggertes Interrupt von einer bezeichneten Verarbeitungseinheit bedient wird, typischerweise ein I/O-Befehl derjenigen Interrupt-Bedienungs-Routine zugeordnet, die, wenn sie ausgeführt wird, veranlasst, dass die Interrupt-Quelle das Interrupt-Signal deaktiviert. Anschließend wird ein End Of Interrupt-(EIO-) Befehl ausgeführt, um der zentralen Interrupt-Steuereinheit **220** mitzuteilen, dass die Interrupt-Bedienung abgeschlossen ist. Ein unechtes Interrupt kann auftreten, falls eine signifikante Wartezeit zwischen der Zeit, zu der die Verarbeitungseinheit den I/O-Befehl ausführt (um die Interrupt-Quelle zum Deaktivieren des Interrupt-Signals zu veranlassen) und der Zeit eingeführt wird, zu der das Interrupt-Signal deaktiviert wird. Eine derartige Wartezeit kann z. B. auftreten, falls die I/O-Einrichtung an einem entfernten Bus über mehrere Bus-Interface-Einheiten hinweg angeordnet ist. Falls die Interrupt-Quelle das Interrupt-Signal nicht deaktiviert, bevor der zentralisierte Interrupt-Controller auf die End Of Interrupt-Befehl reagiert, kann die fortgesetzte Aktivierung des Interrupts von der zentralen Interrupt-Steuereinheit **220** detektiert werden, was somit zur Folge hat, dass das Interrupt unbeabsichtigt re-initiiert wird.

[0096] Gemäß **Fig.** 20 ist die zentrale Steuereinheit **220** vorteilhafterweise mit einem programmierbaren Wartezeit-Zeitgeber **595** verbunden, der durch eine Dekodiereinheit **596** mit dem I/O-Bus **210** verbunden ist. Der programmierbare Wartezeit-Zeitgeber **595** kann von einem System-User dahingehend programmiert werden, dass er eine programmierbare Zeitverzögerung zwischen den Zeitpunkt, zu dem der zentrale Controller **302** einen End Of Interrupt-Befehl empfängt, und den Zeitpunkt setzt, zu dem der zentrale Controller **302** das Status-(ST-) Feld des Steuerregisters **504A** des bestimmten Interrupt-Kanals rücksetzt. Es ist ersichtlich, dass der I/O-Interrupt-Controller **304** derart konfiguriert ist, dass ein bezeichnetes Interrupt-Signal nicht überwacht wird, falls der Status anzeigt, dass das Interrupt bedient wird, oder aus der zentralen Interface-Steuereinheit **220** an eine bestimmte Verarbeitungseinheit ausgegeben worden ist, oder in dem zentralen Controller **302** in die Warteschlange eingereiht worden ist. Nachdem der zentrale Controller **302** den Status für ein bestimmtes Interrupt-Signal in den inaktiven Zustand rückgesetzt hat, fährt der Interrupt-I/O-Controller **304** damit fort, das bestimmte Interrupt-Signal auf nachfolgende Aktivierungen zu überwachen. Anzumerken ist, dass, da der programmierbare Zeitmesser **595** das Rücksetzen des ST-Felds des Steuerregisters für den bestimmten Interrupt-Kanal verzögert, unechte Interrupts verhindert werden können. Anzumerken ist ferner, dass der programmiere Zeitmesser **595** selektiv für jedes Interrupt über das EOI-Feld des Kanal-Steuerregisters **504A** dieses Interrupts aktiviert werden kann. Ferner ist anzumerken, dass der Dekodieren **596** als integraler Bestandteil der I/O-Bus-Interface-Einheit 308 ausgebildet sein kann.

[0097] Gemäß **Fig.** 21 und 22 schließlich wird als nächstes die Integration von System-Handhabungs-Interrupts in das symmetrische Mehrfachverarbeitungssystem gemäß **Fig.** 2 erläutert. Wie generell bekannt ist, werden System-Handhabungs-Interrupts allgemeinhin z. B. bei der Implementierung der Systemenergiehand-

habung verwendet. Bei einem symmetrischen Mehrfachverarbeitungssystem jedoch können SMI-Interrupts typischerweise nicht in der gleichen Weise gehandhabt werden, in der normale Interrupts und NMIs (nicht maskierbare Interrupts) gehandhabt werden. Deshalb ist die zentrale Interrupt-Steuereinheit **220** konfiguriert zum optimalen Handhaben von System-Handhabungs-Interrupts, die aus einer System-Handhabungs-Quelle wie z. B. der Energie-Handhabungseinheit **990** empfangen werden können. Der zentrale Controller **302** der zentralen Interrupt-Steuereinheit **220** handhabt System-Handhabungs-Interrupts, indem eine der Verarbeitungseinheiten **202-1** bis **202-m** als Master-SMM-(System-Handhabungs-Modus-)Handhabungseinheit bezeichnet wird. Die Master-SMM-Verarbeitungseinheit kann durch System-Konfiguration bezeichnet werden. Sämtliche SMM-Interrupt-Anforderungen (d. h. das SMI-Signal aus der Energiehandhabungseinheit **990**) werden der Master-SMM-Verarbeitungseinheit zugeleitet. Die übrigen Verarbeitungseinheiten **202-1** bis **202-m** werden als Slave-Verarbeitungseinheiten bezeichnet. **Fig.** 21 ist ein Flussdiagramm, dass die Arbeitsweise der Master SMM-Verarbeitungseinheit und der Slave-SMM-Verarbeitungseinheit zeigt.

[0098] Wenn sich eine Verarbeitungseinheit in dem System-Handhabungs-Modus befindet und die anderen Verarbeitungseinheiten einen normalen Code ausführen, kann dies System-Probleme verursachen, z. B. dass eine aktive Verarbeitungseinheit auf eine Peripherieeinrichtung zugreift, die gerade von der SMM-Master-Verarbeitungseinheit abgeschaltet worden ist. Wenn die SMM-Master-Verarbeitungseinheit feststellt, dass ihre Aktionen eine Auswirkung auf die anderen Prozessoren in dem System habe, gibt die Master-Verbeitungseinheit einen Befehl aus, der den zentralen Controller **302** dazu veranlasst, ein SMI an sämtliche anderen CPUs in dem System auszugeben. Es ist ein Feld des Steuerregisters **650D** (oder ein separates Register der CPU-Registereinheit **650**) für jeden CPU-Kanal **606** vorgesehen, das pro Verarbeitungseinheit einen SMM-Code-/Flag-Wert für jede Slave-SMM-Verarbeitungseinheit hält. Anfangs zeigt dieses Flag "HOLD" für jede Slave-SMM-Verarbeitungseinheit an. Beim Eintrag von SMM und der Feststellung des HOLD-Codes in diesem Register pausiert jede Slave-SMM-Verarbeitungseinheit und fragt ihr entsprechendes SMM-Code-/Flag-Feld ab, bis sich dieses verändert. Wenn die Master-SMM-Verarbeitungseinheit die erforderlichen Aktionen für die System-Handhabung speichert (z. B. das Herunterfahren einer Peripherieeinrichtung), ändert sie den Status der anderen Verarbeitungseinheit-SMM-Code-Flags mit einem speziellen Befehl (d. h. es ist zu beachten, dass normalerweise eine Verarbeitungseinheit nicht andere Verarbeitungseinheit-Kanal-Register beeinflussen kann), so dass sie mit der SMM-Code-Ausführung fortfahren können. Anzumerken ist, dass der zentrale Controller **302** derart konfiguriert ist, dass er der Master-Verarbeitungseinheit ermöglicht, die SMM-Code-Flags der übrigen Verarbeitungseinheit-CPU-Kanal-Register während des System-Handhabungs-Modus zu schreiben. Mögliche Flag-Werte könnten "RETURN", "CONFIG CHANGE", "SHUTDOWN" etc. sein. Die Sklave-SMM-Verarbeitungseinheiten können dann die bestmöglichen vorzunehmenden Aktionen bestimmen, um korrekt auf den neuen Code-Flag-Wert zu reagieren. Anzumerken ist, dass die Master-SMM-Verarbeitungseinheit und die Slave-SMM-Verarbeitungseinheiten gemeinsam den gleichen SMM-Code-Raum benutzen können. Somit muss der SMM-Code den Master-/Slave-Status durch Software-Steuerung identifizieren.

[0099] Wie aus **Fig.** 20 und 21 zusammen ersichtlich ist, aktiviert, wenn die Energie-Handhabungseinheit **990** das System-Handhabungs-Interrupt ausgibt, der zentrale Controller **302** (**Fig.** 3) der zentralen Interrupt-Steuereinheit **220** das SMI-Eingangssignal der Verarbeitungseinheit **202-1**, die der bezeichnete SMM-Master für dieses System ist. Dies veranlasst die Verarbeitungseinheit **202-1** zum Beginnen der Ausführung der SMM-Bedienungs-Routine in dem Schritt **332**. Während des Schritts **333** bestimmt die Verarbeitungseinheit **202-1** die erforderliche Aktion (entsprechend dem bestimmten SMM-Codes, der von der System-Programmiereinrichtung geliefert wird) und stellt während des Schritts **334** fest, ob die erforderlichen Aktionen eine Auswirkung auf die übrigen Verarbeitungseinheiten in dem System haben können. Falls die angeforderte Aktion keine Auswirkung auf die übrigen Verarbeitungseinheiten hat, wird während des Schritts **335** die SMI-Bedienungs-Routine abgeschlossen, und die Verarbeitungseinheit **202-1** kehrt anschließend zum normalen Betrieb zurück.

[0100] Falls hingegen die angeforderte Aktion während der System-Handhabungs-Bedienungs-Routine eine Auswirkung hinsichtlich des Betriebs der übrigen Verarbeitungseinheiten haben kann, veranlasst die Verarbeitungseinheit **202-1** das Code-/Flag-Register in jedem der übrigen CPU-Kanäle dazu, anzuzeigen, dass die übrigen Verarbeitungseinheiten anhalten sollen (Schritt **336**). Anzumerken ist, dass in der CPU-Kanal-Registereinheiten **650** jedes CPU-Kanals **606** ein Code-/Flag-Register (oder -Feld) für jede Slave-Verarbeitungseinheit eingebettet ist. Anschließend veranlasst die Verarbeitungseinheit **202-1** im Schritt **337** den zentralen Controller **302** dazu, an jede der Slave-Verarbeitungseinheiten **202-1** bis **202-m** ein SMI auszugeben. Die Verarbeitungseinheit **202-1** schließt anschließend ihre verlangten System-Handhabungs-Modus-Operationen ab (Schritt **339**) und veranlasst die Code-/Flag-Register der Slave-Verarbeitungseinheiten **202-1** bis **202-m** dazu, anzuzeigen, dass eine Veränderung in der System-Konfiguration eingetreten ist (Schritt **339**). Die System-Handhabungs-Interrupt-Routine für die Verarbeitungseinheit **202-1** wird dann abgeschlossen, und die Verarbeitungseinheit **202-1** kehrt zum normalen Betrieb zurück.

[0101] Wenn der zentrale Controller **302** die SMI-Signale and die Slave-Verarbeitungseinheiten **202-1** bis **202-m** ausgibt, beginnt jede der Verarbeitungseinheiten **202-1** bis **202-m** während des Schritts 340 mit der Ausführung der SMI-Bedienungs-Routine. Während der Ausführung der SMI-Bedienungs-Routine (Schritt

341) identifizieren sich sämtliche der Verarbeitungseinheiten **202-1** bis **202-m** als Slaves, und anschließend lesen sie ihre Code-/Flag-Register unter Verwendung der betreffende CPU-ID-Register **902-2** bis **902-m** auf die Weise, die zuvor für das Zugreifen auf die Konfigurations-Register **650** jedes CPU-Kanals beschrieben wurde. Falls das Code-/Flag-Register anzeigt, dass die jeweilige Verarbeitungseinheit angehalten werden sollte, bleibt die Verarbeitungseinheit im Effekt inaktiv, bis das Code-Register einen anderen Status anzeigt. Beispielsweise kann das Code-Register der Verarbeitungseinheiten **202-1** bis **202-m** anzeigen, dass die jeweiligen Verarbeitungseinheiten zu ihrer normalen Ausführung zurückkehren sollen, wobei keine weitere Aktion verlangt wird (Schritt **343**). Gleichermaßen kann das Code-/Flag-Register die jeweilige Verarbeitungseinheit anweisen, den Betrieb abzuschließen (Schritt **344**). Anzumerken ist, dass die Master-Verarbeitungseinheit **202-1** oder der zentrale Controller **302** möglicherweise vorher das Code-/Flag-Register der bestimmten Verarbeitungseinheit zum Anzeigen von "return" oder "shutdown" eingestellt haben. Falls das Code/Flag-Register anzeigt, dass sich ein Konfigurations-Parameter in dem System verändert hat (Schritt **345**), wird die I/O-Erlaubnis-Abbildung für die jeweiligen Verarbeitungseinheiten rekonfiguriert, um die System-Veränderung zu reflektieren. Anzumerken ist, dass die Rekonfiguration der I/O-Erlaubnis-Abbildung für jede Verarbeitungseinheit auf herkömmliche Weise durchgeführt wird.

[0102] Mit dem symmetrischen Mehrfachverarbeitungssystem gemäß der obigen Beschreibung wird eine effiziente Handhabung von System-Interrupts erzielt, während eine breite Kompatibilität aufrechterhalten wird. Die Interrupt-Handhabung wird mittels einer zentralisierten Interrupt-Steuereinheit erzielt. Die Interrupt-Steuereinheit erlaubt vorteilhafterweise eine Erweiterung jedes Interrupt-Stifts, indem die Interrupt-Steuereinheit in einen Kaskaden-Modus gesetzt wird. Ferner reagiert die zentrale Steuereinheit auf spezialisierte Interrupt-Zyklen, die den I/O-Einrichtungen und/oder Bus-Brücken-Einrichtungen ermöglichen, eine Initiierung eines Interrupts zu veranlassen, ohne dass eine speziell zugeordnete Interrupt-Leitung erforderlich ist. Die zentrale Interrupt-Steuereinheit ermöglicht ferner, dass jedes Interrupt unabhängig von seiner zugehörigen Vektor-ID priorisiert wird, und verhindert das Auftreten unechter Interrupts, indem ein programmierbarer Wartezeit-Zeitgeber vorgesehen ist, der die zentrale Interrupt-Steuereinheit dazu veranlasst, ihre Reaktion auf End Of Interrupt-(EOI-) Instruktionen zu verzögern. Ferner ist durch die zentralen Interrupt-Steuereinheit eine Auto-Kettenbildungs-Technik dahingehend implementiert, dass Interrupts basierend auf ihren aktuellen Task-Prioritäts-Werten sequentiell an verschiedene Verarbeitungseinheiten ausgegeben werden. Ferner handhabt die zentrale Interrupt-Steuereinheit System-Handhabungs-Interrupts (SMIs) aus Quellen wie z. B. Energie-Handhabungseinheiten und gewährleistet einen korrekten System-Betrieb selbst dann, wenn die angeforderte System-Handhabungs-Funktion Operationen beeinflusst, die von anderen Verarbeitungseinheiten ausgeführt werden.

[0103] Fachleuten auf dem Gebiet werden in voller Kenntnis der obigen Beschreibung zahlreiche Variationen und Modifikationen ersichtlich sein. Die folgenden Ansprüche sind dahingehend zu interpretieren, dass sie sämtliche derartigen Variationen und Modifikationen umfassen.

Patentansprüche

1. Symmetrisches Mehrfachverarbeitungssystem mit:
mehreren Verarbeitungseinheiten (**202**);
einem mit jeder der Verarbeitungseinheiten verbundenen CPU-Lokalbus (**207**);
einem ersten I/O-Bus (**210**);
einer zwischen dem CPU-Lokalbus und dem I/O-Bus angeordneten ersten Bus-Brücke (**208**) zur Bildung eines Interface zwischen dem CPU-Lokalbus und dem I/O-Bus;
mehreren I/O-Einrichtungen (**212**), die mit dem I/O-Bus verbunden sind; und
einer mit den mehreren I/O-Einrichtungen verbundenen programmierbaren zentralen Interrupt-Steuereinheit (**220**) zum Empfangen von Interrupt-Signalen von den mehreren I/O-Einrichtungen und zum selektiven Ausgeben entsprechender Interrupt-Signale an die Verarbeitungseinheiten, wobei die zentrale Interrupt-Steuereinheit aufweist:
mehrere mit den mehreren I/O-Einrichtungen verbundene Interrupt-Kanäle (**402**), von denen jeder das Vorhandensein eines Interrupts einer entsprechenden I/O-Einrichtung detektiert, wobei jeder Interrupt-Kanal ferner eine programmierbare Speichereinheit (**504A**) zum Speichern eines Vektors und eines dem Interrupt der entsprechenden I/O-Einrichtung zugeordneten Prioritätswerts aufweist;
mehrere mit den mehreren Verarbeitungseinheiten verbundene CPU-Kanäle (**606**), von denen jeder Interrupt-Informatioh zum Steuern der Ausgabe eines entsprechenden Prozessor-Interrupts empfängt, wobei die Interrupt-Information den Vektor und den Prioritätswert aufweist, der jedem mehrerer gewählter Interrupts der I/O-Einrichtungen, die ausgegeben worden sind, entspricht;
wobei das symmetrische Mehrfachverarbeitungssystem dadurch gekennzeichnet ist, dass es ferner aufweist:
einen mit den mehreren Interrupt-Kanälen und den mehreren CPU-Kanälen verbundenen zentralen Controller (**302**), der die Interrupt-Information leitet, die jedem mehrerer Interrupts entspricht, welche von einem der meh-

reren Interrupt-Kanäle zu einem bezeichneten CPU-Kanal ausgegeben worden ist.

2. Symmetrisches Mehrfachverarbeitungssystem nach Anspruch 1, bei dem die zentrale Interrupt-Steuereinheit auf einem einzigen IC-Chip ausgebildet ist, und bei dem jeder der mehreren CPU-Kanäle (**606**) ein Task-Prioritäten-Register (**650A**) zum Speichern eines Task-Prioritäten-Werts aufweist, wobei das Task-Prioritäten-Register eines gegebenen CPU-Kanals auf die gleiche Adresse abgebildet wird wie die Task-Prioritäten-Register der anderen CPU-Kanäle.

3. Symmetrisches Mehrfachverarbeitungssystem nach Anspruch 1 oder 2, bei dem jede der mehreren Verarbeitungseinheiten (**202**) ein ID-Register (**902**) zum Speichern eines Prozessor-ID-Werts zur eindeutigen Identifizierung der Verarbeitungseinheit aufweist, wobei der in dem Prozessor-ID-Register einer gegebenen Verarbeitungseinheit gespeicherte Prozessor-ID-Wert zum Identifizieren eines bestimmten der Task-Prioritäten-Register verwendet wird, auf das während eines gegebenen Zyklus zugegriffen werden soll.

4. Symmetrisches Mehrfachverarbeitungssystem nach Anspruch 1, ferner mit einem Interrupt-Controller (**216**), der zwischen einer I/O-Einrichtung (**214**) und dem I/O-Bus (**210**) angeordnet ist, wobei die zentrale Interrupt-Steuereinheit (**220**) ferner ein von dem Interrupt-Controller erzeugtes Interrupt-Signal empfängt.

5. Symmetrisches Mehrfachverarbeitungssystem nach Anspruch 1, ferner mit:
einem zweiten I/O-Bus (**282**), der mit einer I/O-Einrichtung (**280**) verbunden ist; und
einer zwischen dem ersten I/O-Bus und dem zweiten I/O-Bus angeordneten zweiten Bus-Brücke (**284**), die ein Interface zwischen dem ersten und dem zweiten I/O-Bus bilden kann und ferner auf ein Interrupt-Signal von der I/O-Einrichtung hin einen Interrupt-Zyklus erzeugen kann.

6. Symmetrisches Mehrfachverarbeitungssystem nach Anspruch 4, bei dem die zentrale Interrupt-Steuereinheit (**220**) einen Interrupt-Zyklus-Dekodierer (**292**) aufweist, der einen von der zweiten Bus-Brücke erzeugten Interrupt-Zyklus dekodieren kann.

7. Symmetrisches Mehrfachverarbeitungssystem nach Anspruch 1, bei dem bei Ausgabe eines bestimmten Prozessor-Interrupt-Signals ein dem bestimmten Prozessor-Interrupt-Signal entsprechender Vektor an eine Verarbeitungseinheit ausgegeben wird, die das bestimmte Prozessor-Interrupt-Signal empfängt.

8. Symmetrisches Mehrfachverarbeitungssystem nach Anspruch 7, bei dem der dem Prozessor-Interrupt-Signal entsprechende Vektor angibt, welche der mehreren I/O-Einrichtungen ein entsprechendes Interrupt-Signal ausgegeben hat, das die Ausgabe des Prozessor-Interrupt-Signals veranlasst hat.

Es folgen 21 Blatt Zeichnungen

Anhängende Zeichnungen

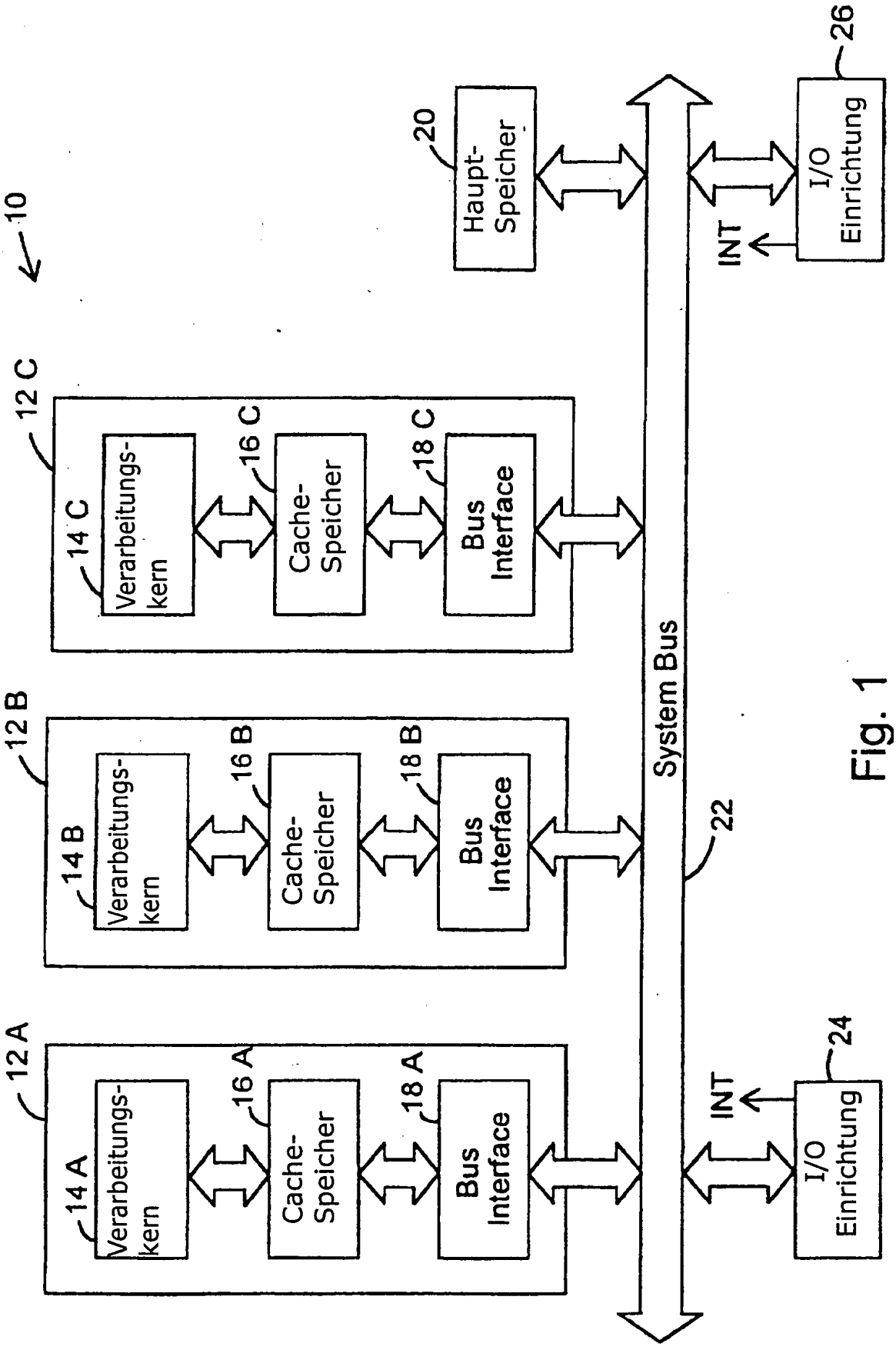


Fig. 1
(Stand der Technik)

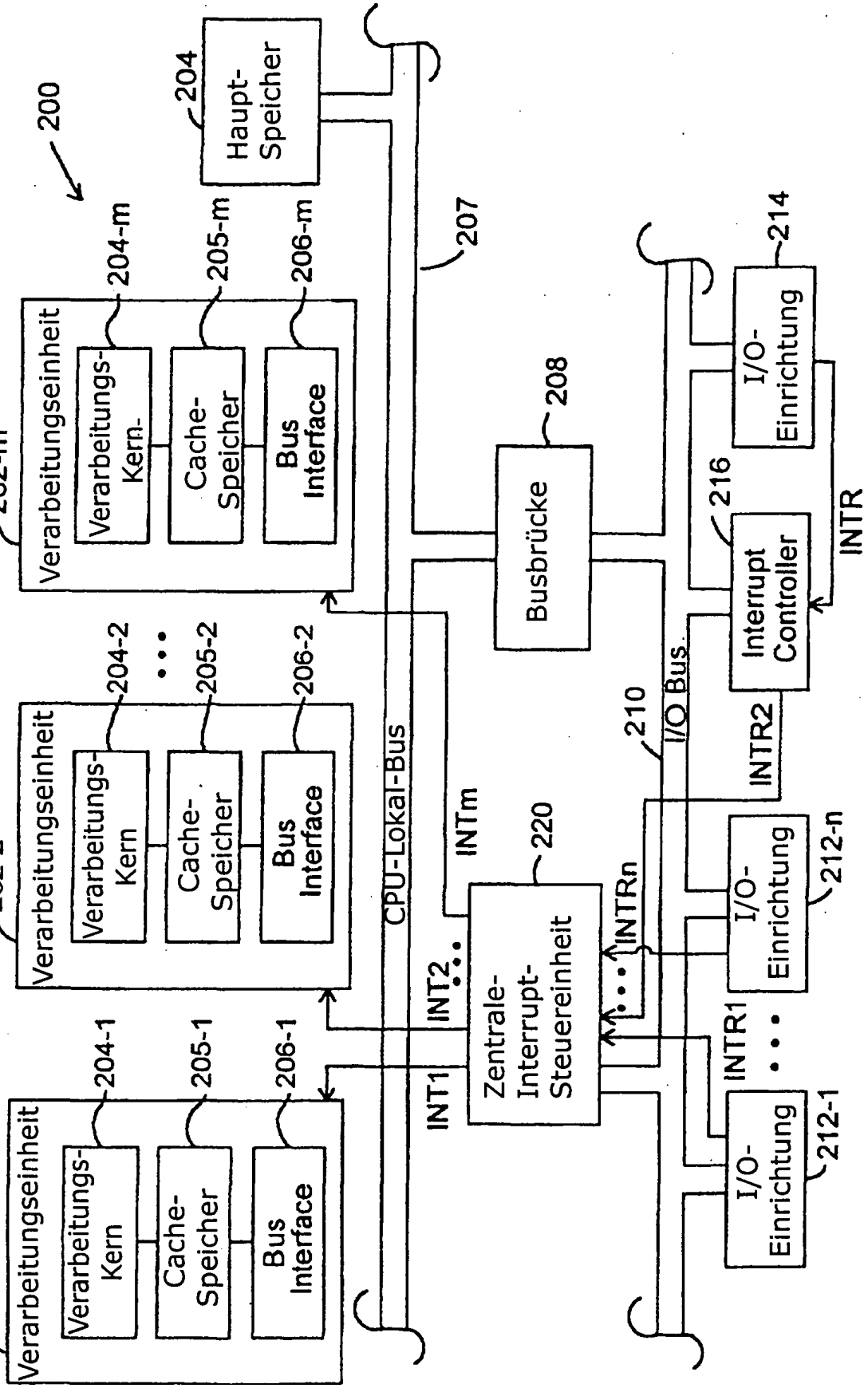

200
Haupt-Speicher
204
207
Verarbeitungseinheit
202-m
Verarbeitungs-Kern
204-m
Cache-Speicher
205-m
Bus Interface
206-m
Verarbeitungseinheit
202-2
Verarbeitungs-Kern
204-2
Cache-Speicher
205-2
Bus Interface
206-2
Verarbeitungseinheit
202-1
Verarbeitungs-Kern
204-1
Cache-Speicher
205-1
Bus Interface
206-1
CPU-Lokal-Bus
Busbrücke
208
I/O-Einrichtung
214
Interrupt Controller
216
INTR
INTR2
I/O Bus
210
INTm
INT2
INT1
Zentrale-Interrupt-Steuereinheit
220
INTRn
INTR1
I/O-Einrichtung
212-n
I/O-Einrichtung
212-1

Fig. 2

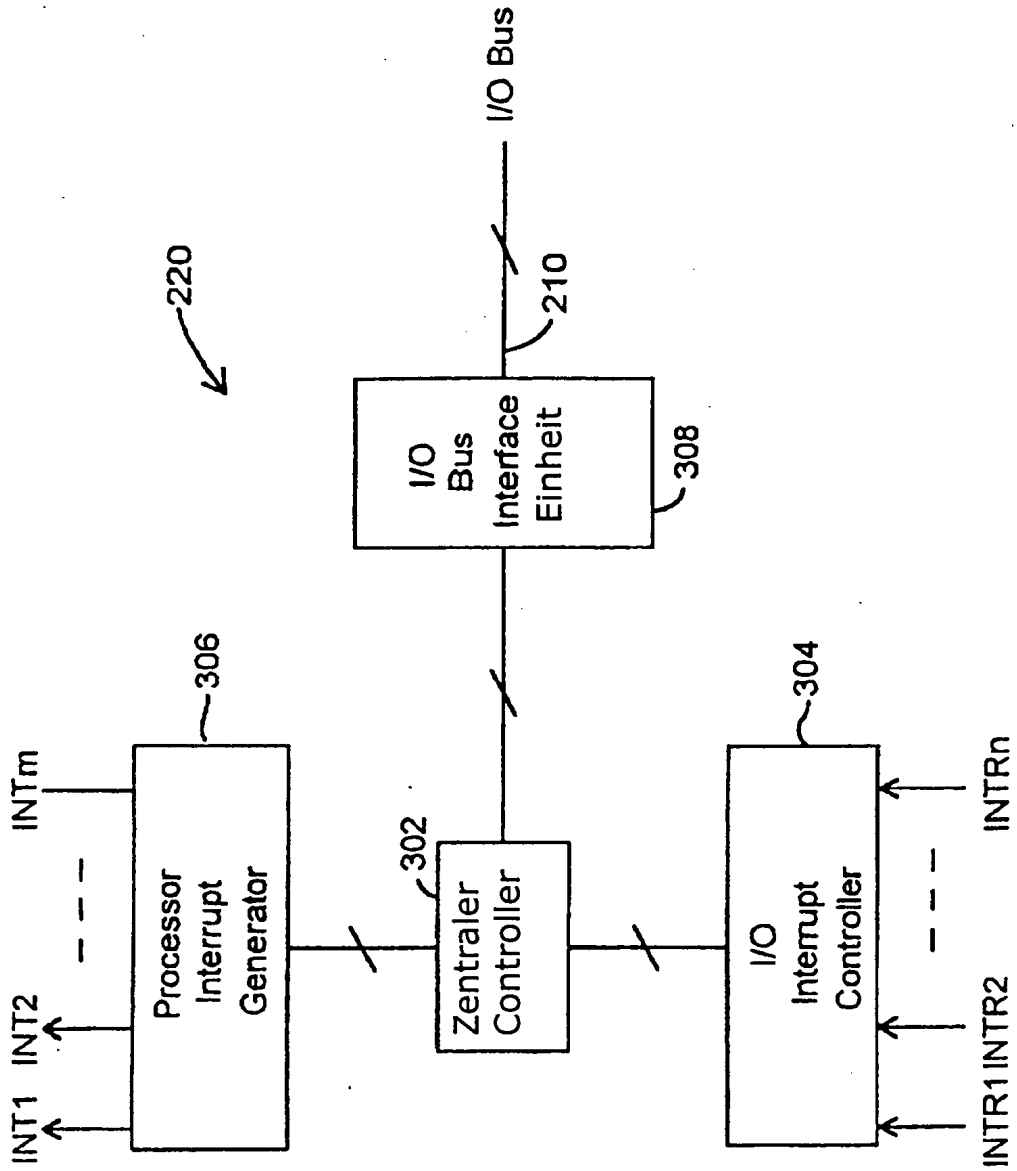

220
I/O Bus
210
I/O Bus Interface Einheit
308
306
304
302
INTm
INT2
INT1
Processor Interrupt Generator
Zentraler Controller
I/O Interrupt Controller
INTRn
INTR2
INTR1

Fig. 3

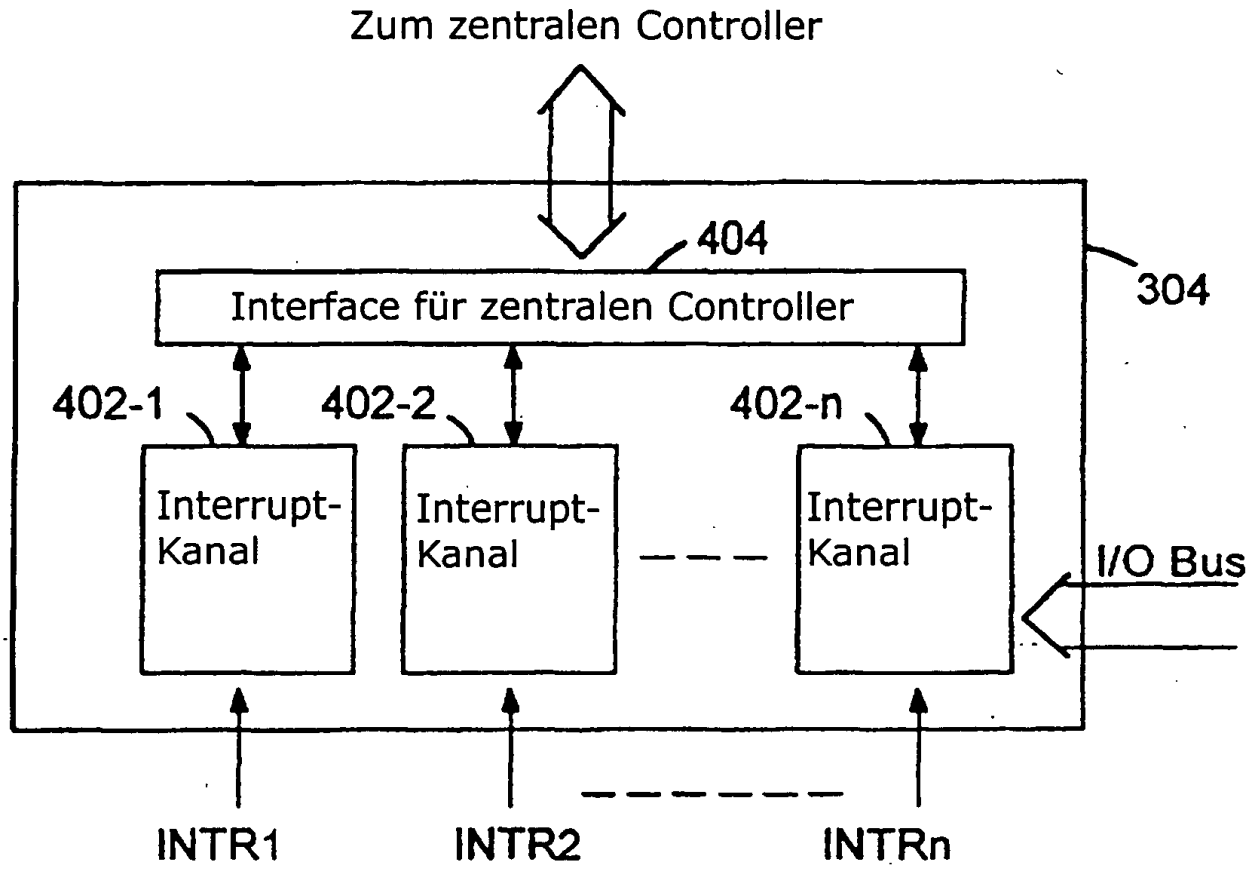

Zum zentralen Controller
404
304
Interface für zentralen Controller
402-1
402-2
402-n
Interrupt-Kanal
Interrupt-Kanal
Interrupt-Kanal
I/O Bus
INTR1
INTR2
INTRn

Fig. 4

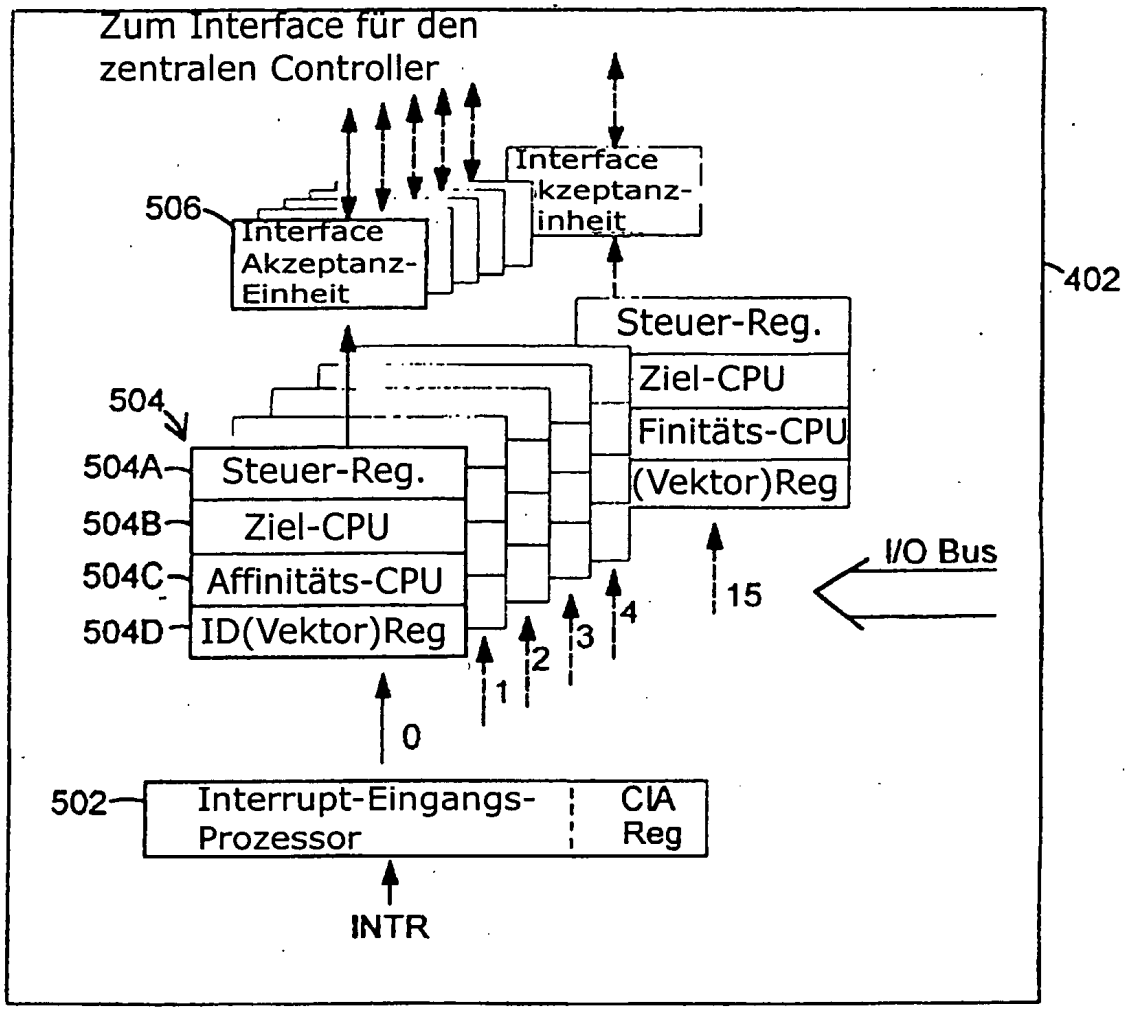
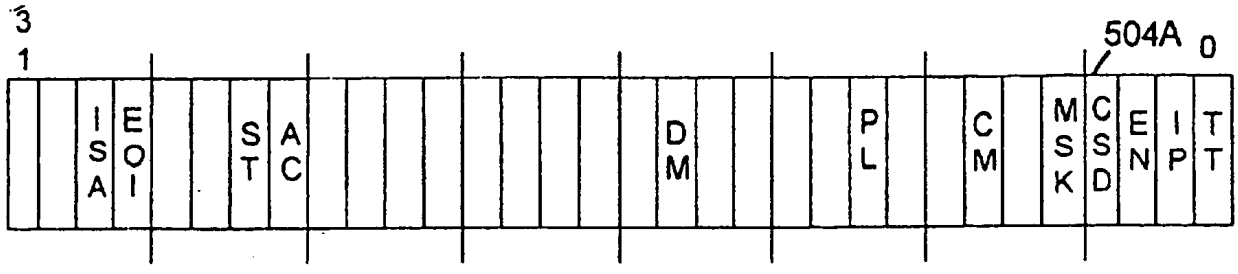


Fig. 5



Interrupt-Kanal-Steuerregister

Fig. 6

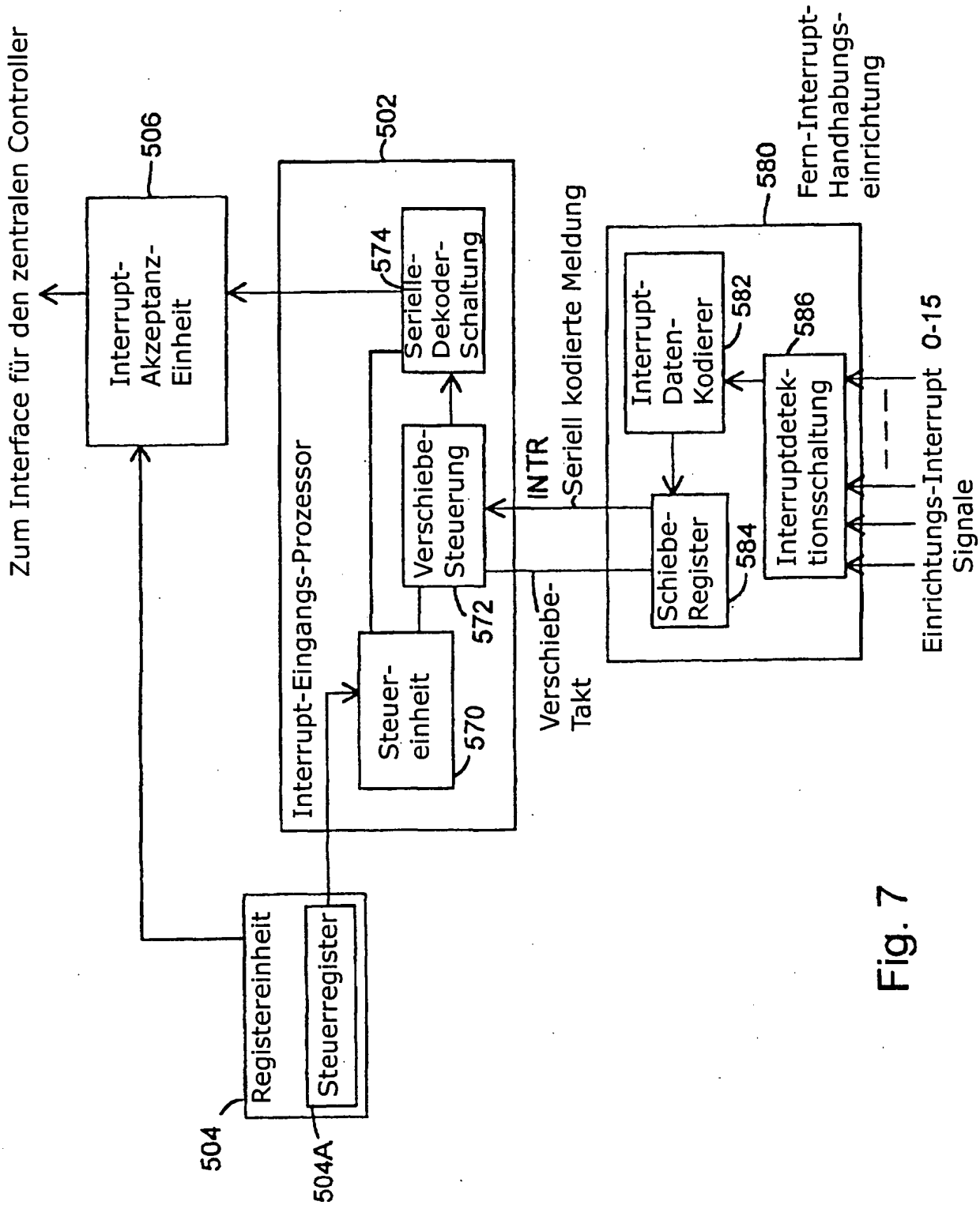

Zum Interface für den zentralen Controller
506
Interrupt-Akzeptanz-Einheit
502
Interrupt-Eingangs-Prozessor
574
Serielle-Dekoder-Schaltung
Verschiebe-Steuerung
572
Steuer-einheit
570
INTR
Seriell kodierte Meldung
Verschiebe-Takt
580
Fern-Interrupt-Handhabungs-einrichtung
Interrupt-Daten-Kodierer
582
Schiebe-Register
584
Interruptdetek-tionsschaltung
586
Einrichtungs-Interrupt 0-15 Signale
Registereinheit
Steuerregister
504
504A

Fig. 7

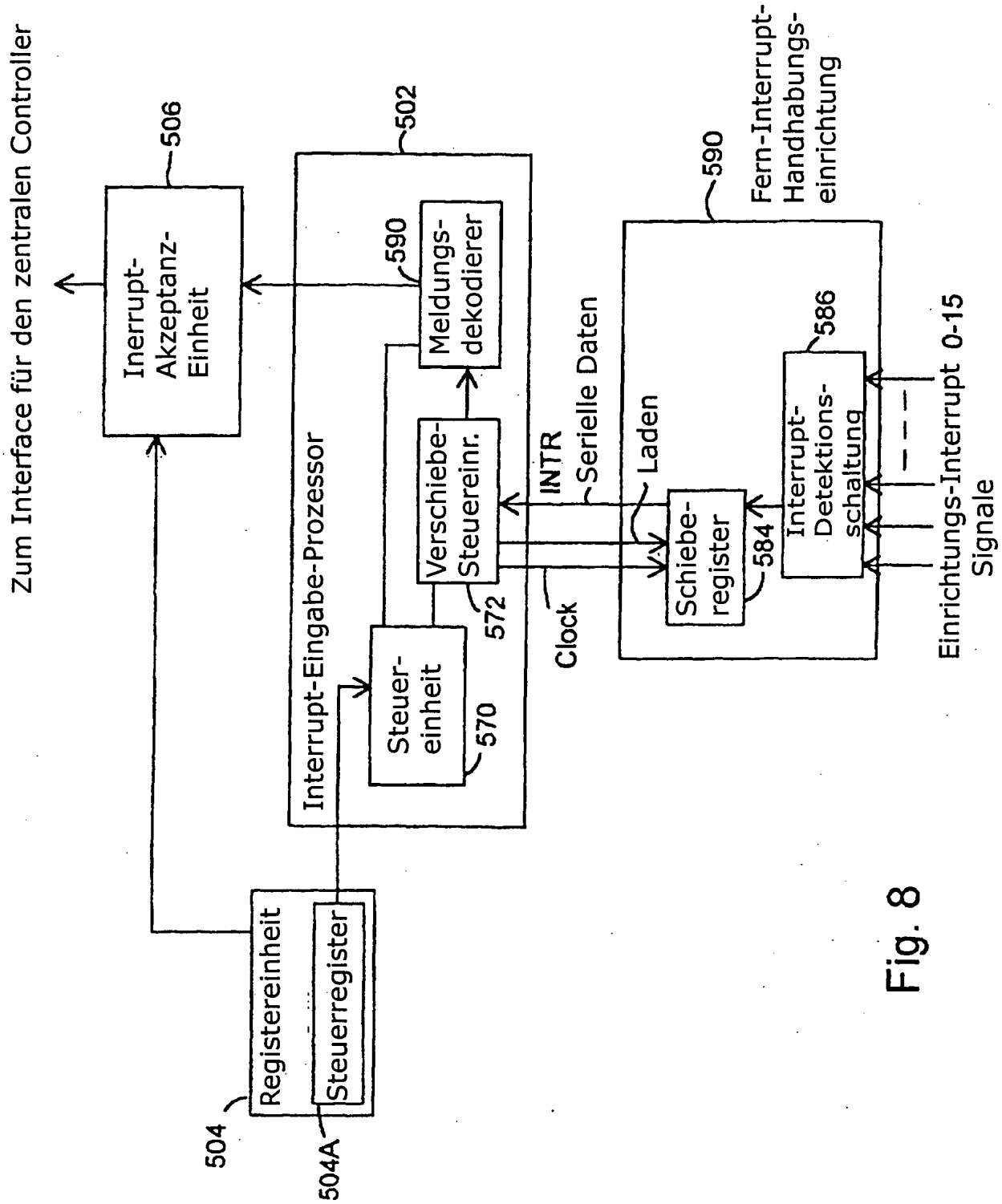

Zum Interface für den zentralen Controller
Inerrupt-Akzeptanz-Einheit
506
502
590
Meldungs-dekodierer
Interrupt-Eingabe-Prozessor
Verschiebe-Steuereinr.
572
Steuer-einheit
570
INTR
Serielle Daten
Laden
Clock
590
Fern-Interrupt-Handhabungs-einrichtung
Schiebe-register
584
Interrupt-Detektions-schaltung
586
Einrichtungs-Interrupt 0-15 Signale
Registereinheit
504
Steuerregister
504A

Fig. 8

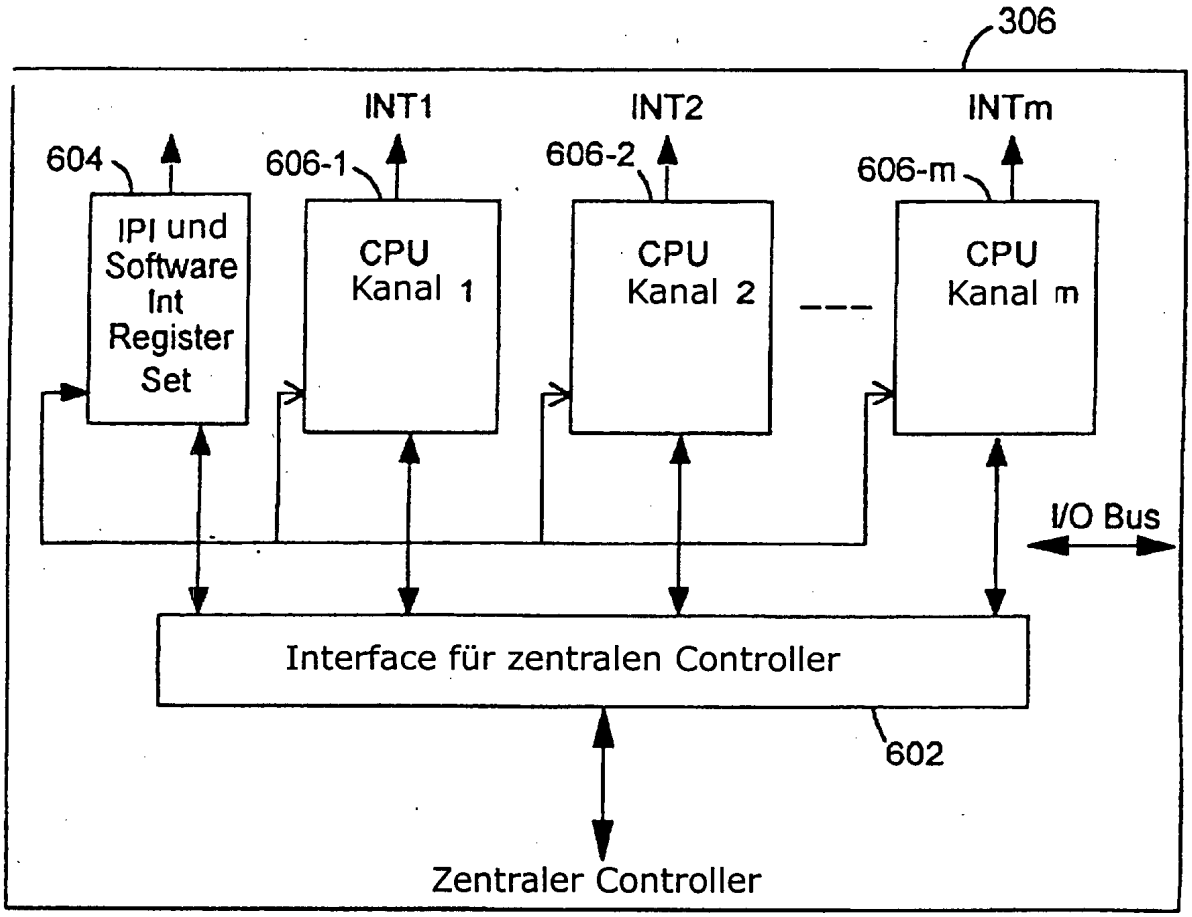

306
INT1
INT2
INTm
604
606-1
606-2
606-m
IPI und Software Int Register Set
CPU Kanal 1
CPU Kanal 2
CPU Kanal m
I/O Bus
Interface für zentralen Controller
602
Zentraler Controller

Fig. 9

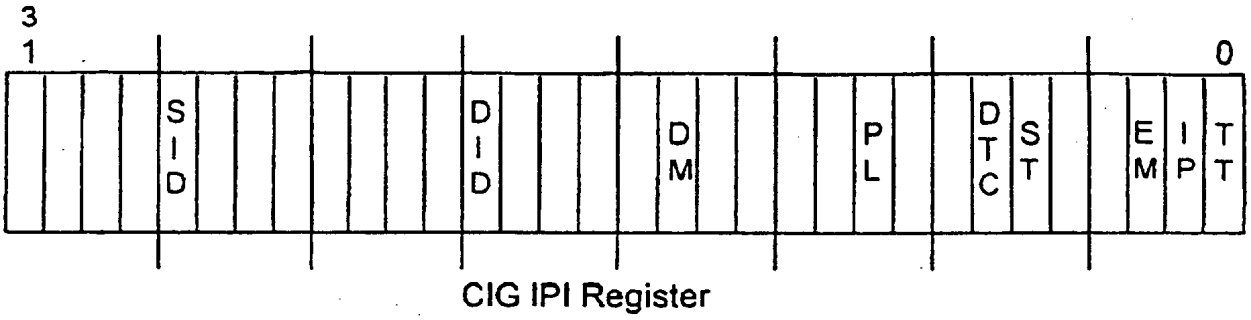

3
1
0
SID
DID
DM
PL
DTC
ST
EM
IP
TT
CIG IPI Register

Fig. 10B

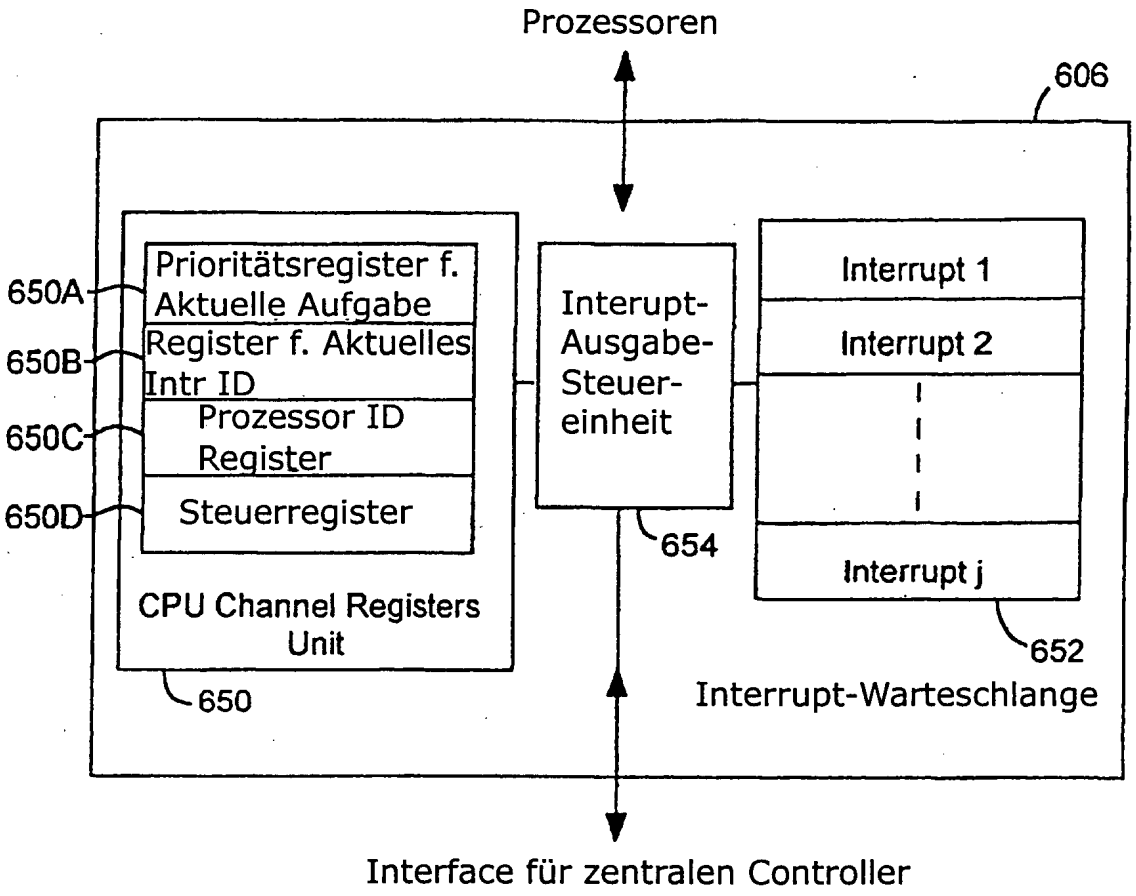
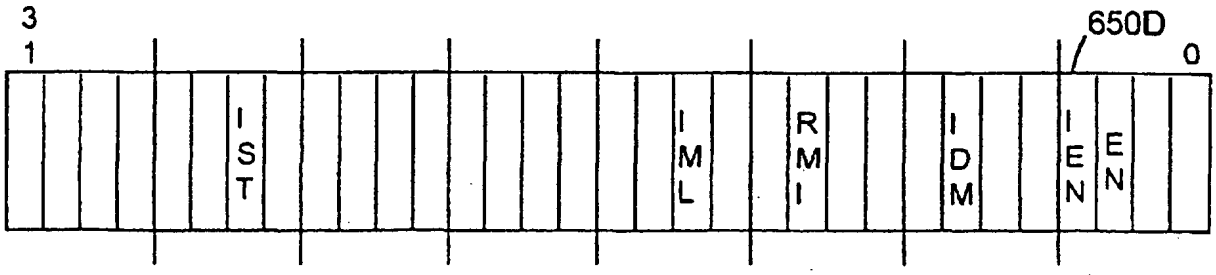


Fig. 10



CIG-Steuerregister

Fig. 10A

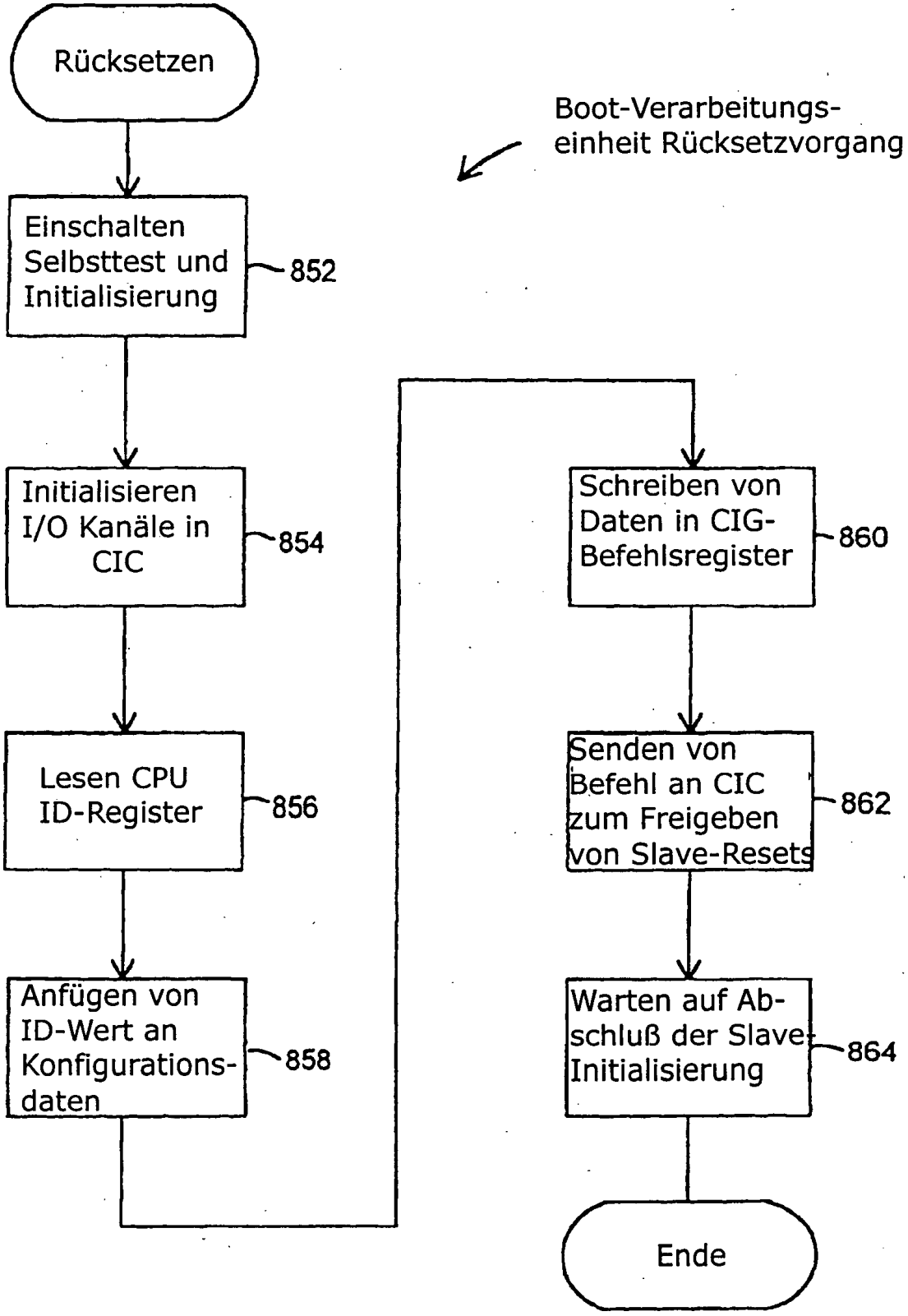

Rücksetzen
Boot-Verarbeitungs-einheit Rücksetzvorgang
Einschalten Selbsttest und Initialisierung
852
Initialisieren I/O Kanäle in CIC
854
Lesen CPU ID-Register
856
Anfügen von ID-Wert an Konfigurations-daten
858
Schreiben von Daten in CIG-Befehlsregister
860
Senden von Befehl an CIC zum Freigeben von Slave-Resets
862
Warten auf Ab-schluß der Slave-Initialisierung
864
Ende

Fig. 11

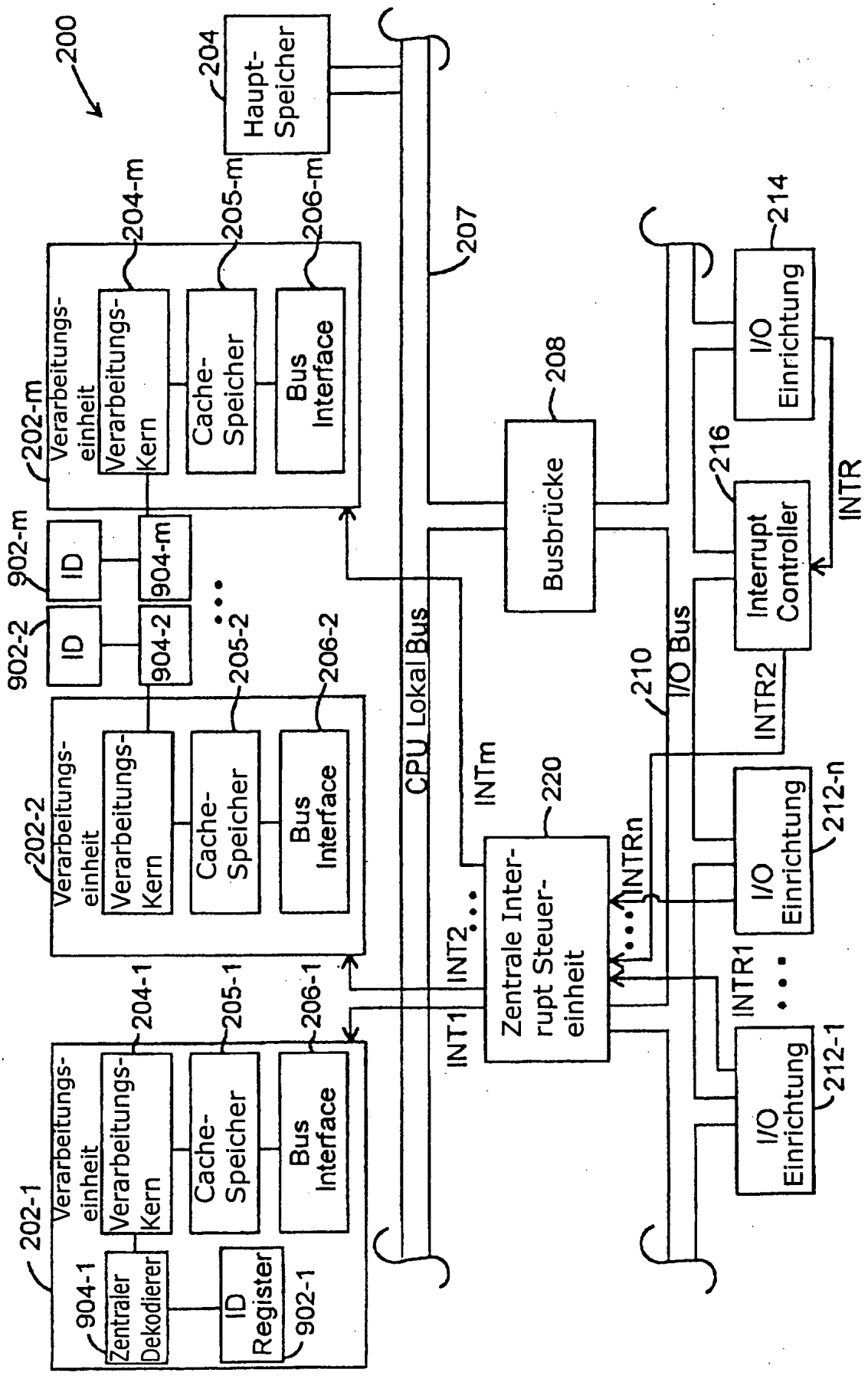

200
Haupt-Speicher
204
204-m
205-m
206-m
202-m
Verarbeitungs-einheit
Verarbeitungs-Kern
Cache-Speicher
Bus Interface
902-m
ID
904-m
902-2
904-2
205-2
206-2
202-2
204-1
205-1
206-1
202-1
904-1
Zentraler Dekodierer
ID Register
902-1
207
CPU Lokal Bus
208
Busbrücke
210
I/O Bus
214
I/O Einrichtung
216
Interrupt Controller
INTR
INTR2
INTR1
INTRn
INTm
INT2
INT1
220
Zentrale Interrupt Steuereinheit
212-n
212-1

Fig. 12

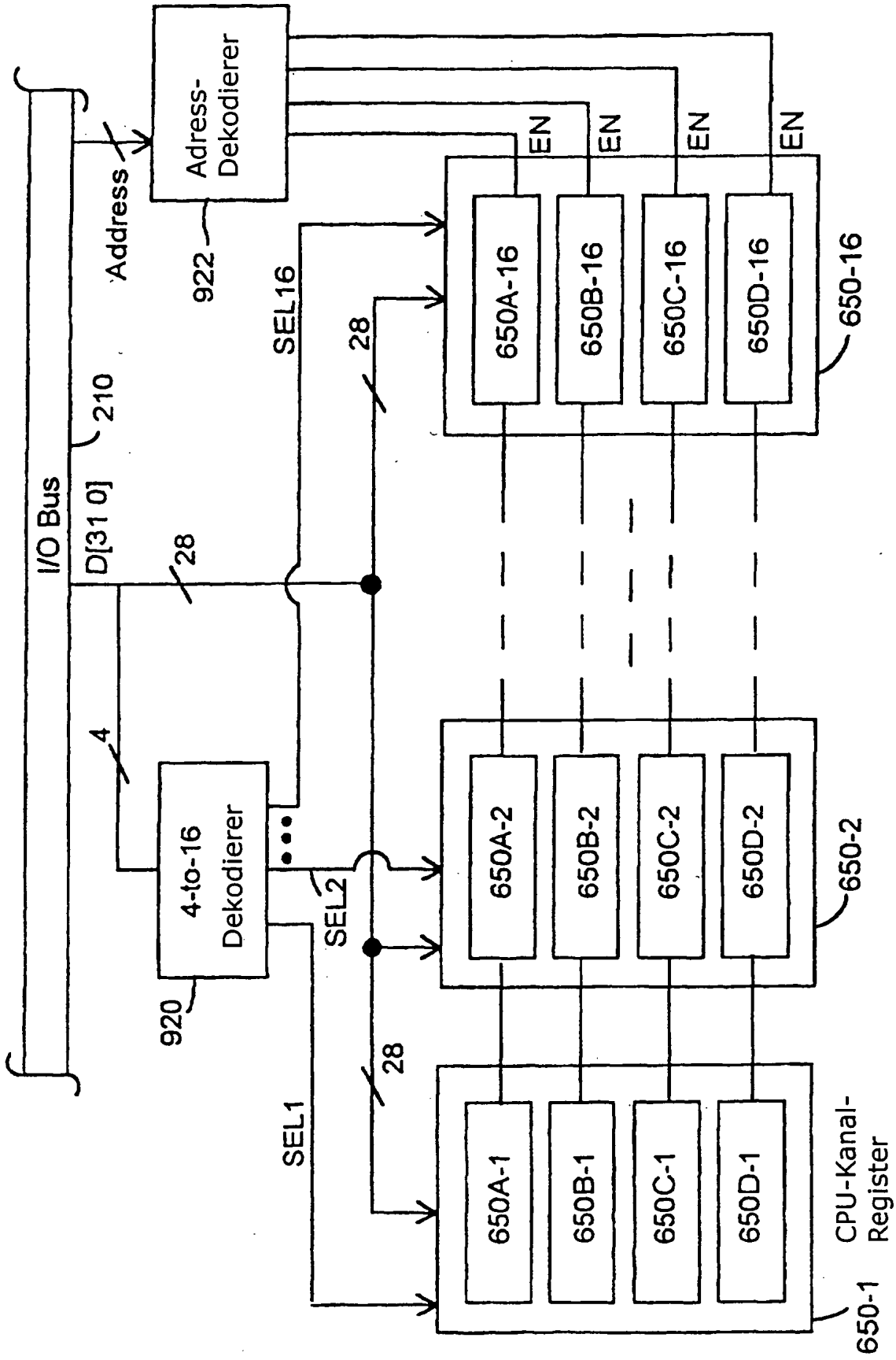

I/O Bus
D[31 0]
210
Address
Adress-
Dekodierer
922
SEL16
28
EN
650A-16
650B-16
650C-16
650D-16
650-16
4
4-to-16
Dekodierer
920
SEL2
650A-2
650B-2
650C-2
650D-2
650-2
SEL1
650A-1
650B-1
650C-1
650D-1
CPU-Kanal-
Register
650-1

Fig. 13

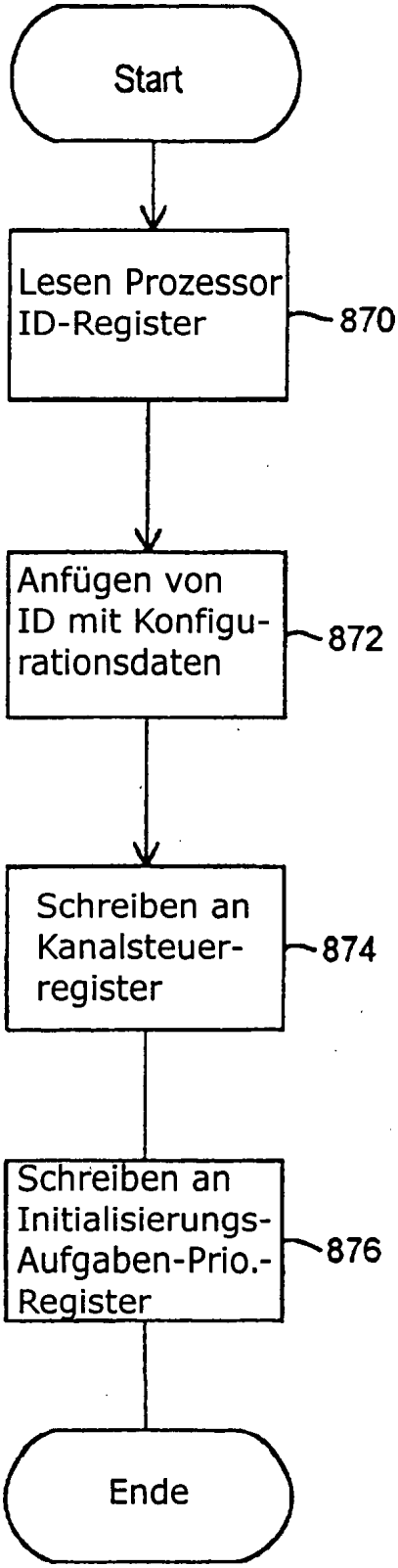

Start
Lesen Prozessor ID-Register
870
Anfügen von ID mit Konfigu-rationsdaten
872
Schreiben an Kanalsteuer-register
874
Schreiben an Initialisierungs-Aufgaben-Prio.-Register
876
Ende

Fig. 14

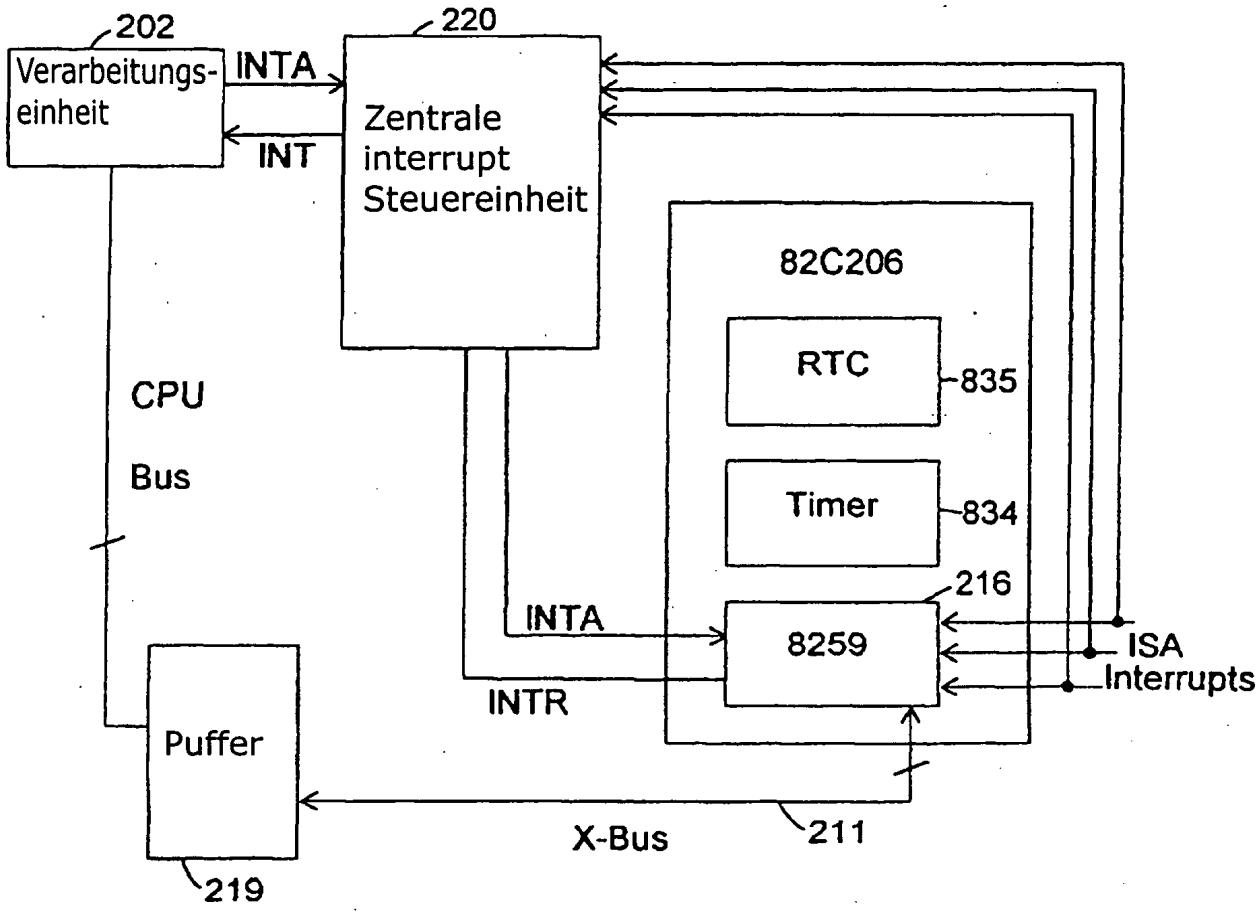

202
Verarbeitungs-
einheit
INTA
INT
220
Zentrale
interrupt
Steuereinheit
82C206
RTC
835
Timer
834
216
8259
ISA
Interrupts
CPU
Bus
INTA
INTR
Puffer
X-Bus
211
219

Fig. 15

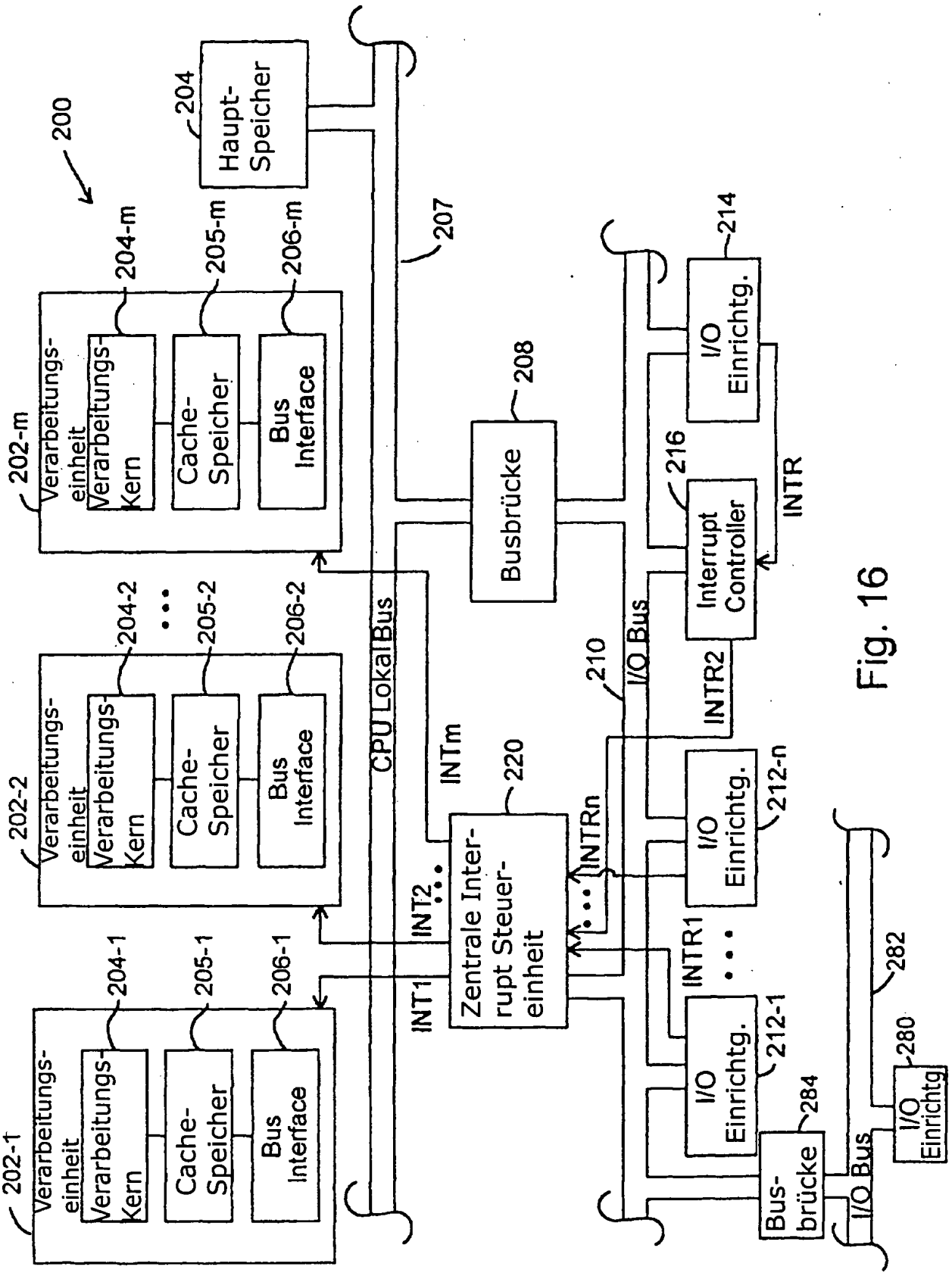

200
204
Haupt-Speicher
207
CPU Lokal Bus
202-m
Verarbeitungs-einheit
Verarbeitungs-Kern
204-m
Cache-Speicher
205-m
Bus Interface
206-m
202-2
Verarbeitungs-einheit
Verarbeitungs-Kern
204-2
Cache-Speicher
205-2
Bus Interface
206-2
202-1
Verarbeitungs-einheit
Verarbeitungs-Kern
204-1
Cache-Speicher
205-1
Bus Interface
206-1
Busbrücke
208
210
I/O Bus
214
I/O Einrichtg.
216
Interrupt Controller
INTR
INTR2
INTm
INT2
INT1
Zentrale Interrupt Steuereinheit
220
INTRn
INTR1
212-n
I/O Einrichtg.
212-1
I/O Einrichtg.
Bus-brücke
284
282
I/O Bus
280
I/O Einrichtg

Fig. 16

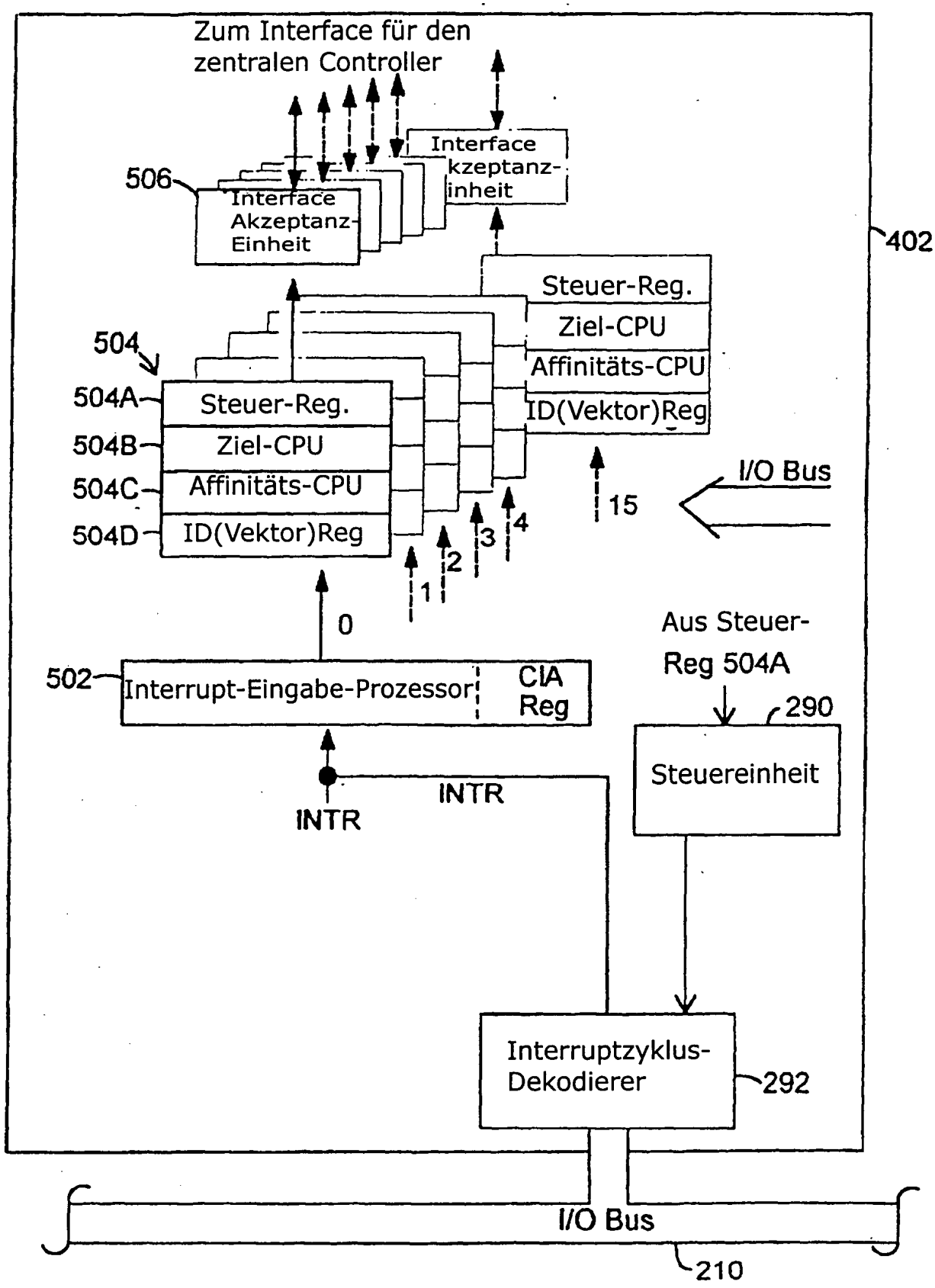

Zum Interface für den zentralen Controller
Interface kzeptanz-inheit
506
Interface Akzeptanz-Einheit
402
Steuer-Reg.
Ziel-CPU
Affinitäts-CPU
ID(Vektor)Reg
504
504A
504B
504C
504D
Steuer-Reg.
Ziel-CPU
Affinitäts-CPU
ID(Vektor)Reg
I/O Bus
15
4
3
2
1
0
Aus Steuer-Reg 504A
290
502
Interrupt-Eingabe-Prozessor
CIA Reg
Steuereinheit
INTR
INTR
Interruptzyklus-Dekodierer
292
I/O Bus
210

Fig. 17

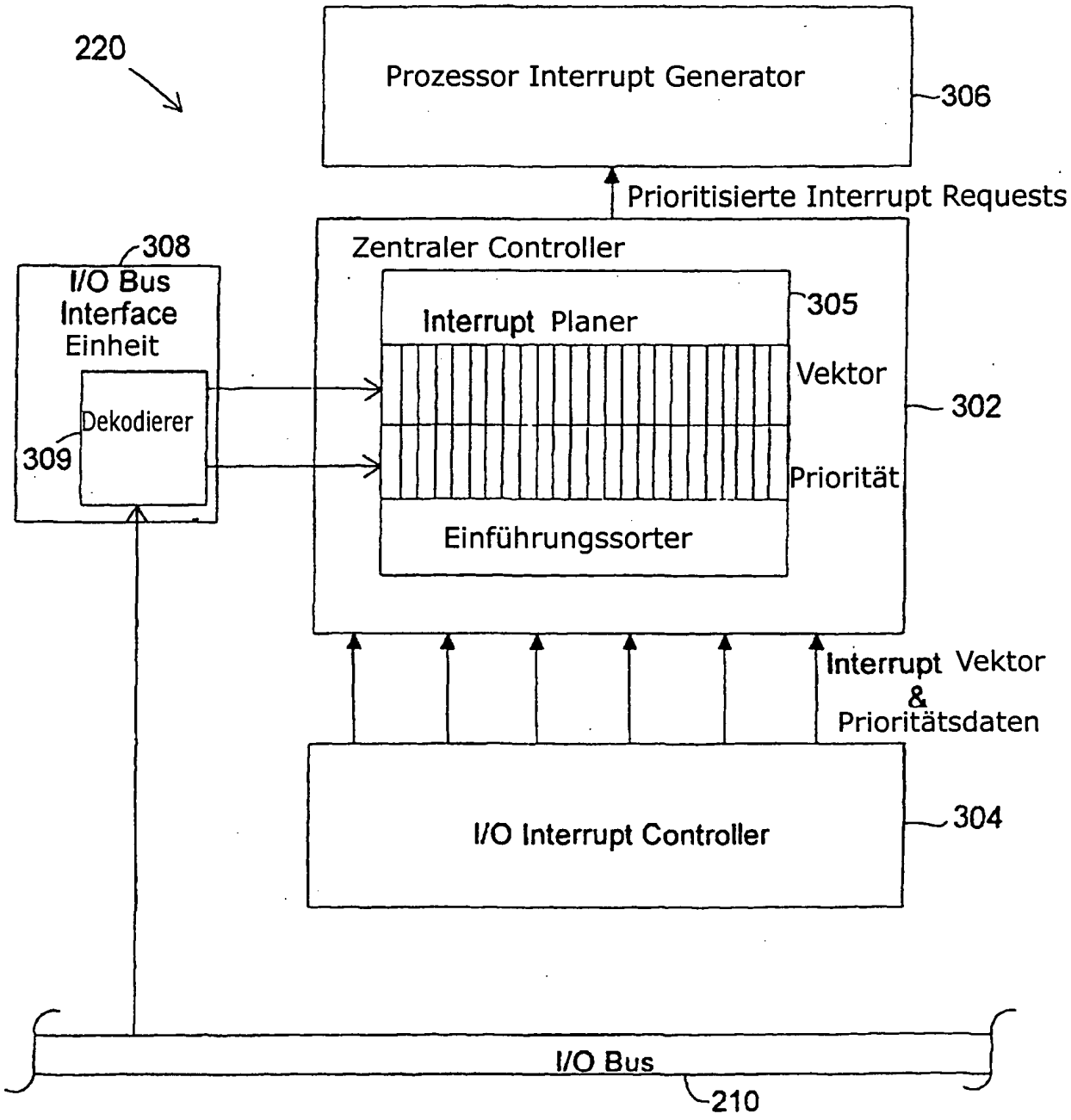

220
Prozessor Interrupt Generator
306
Prioritisierte Interrupt Requests
308
I/O Bus
Interface
Einheit
Dekodierer
309
Zentraler Controller
Interrupt Planer
305
Vektor
302
Priorität
Einführungssorter
Interrupt Vektor
&
Prioritätsdaten
I/O Interrupt Controller
304
I/O Bus
210

Fig. 18

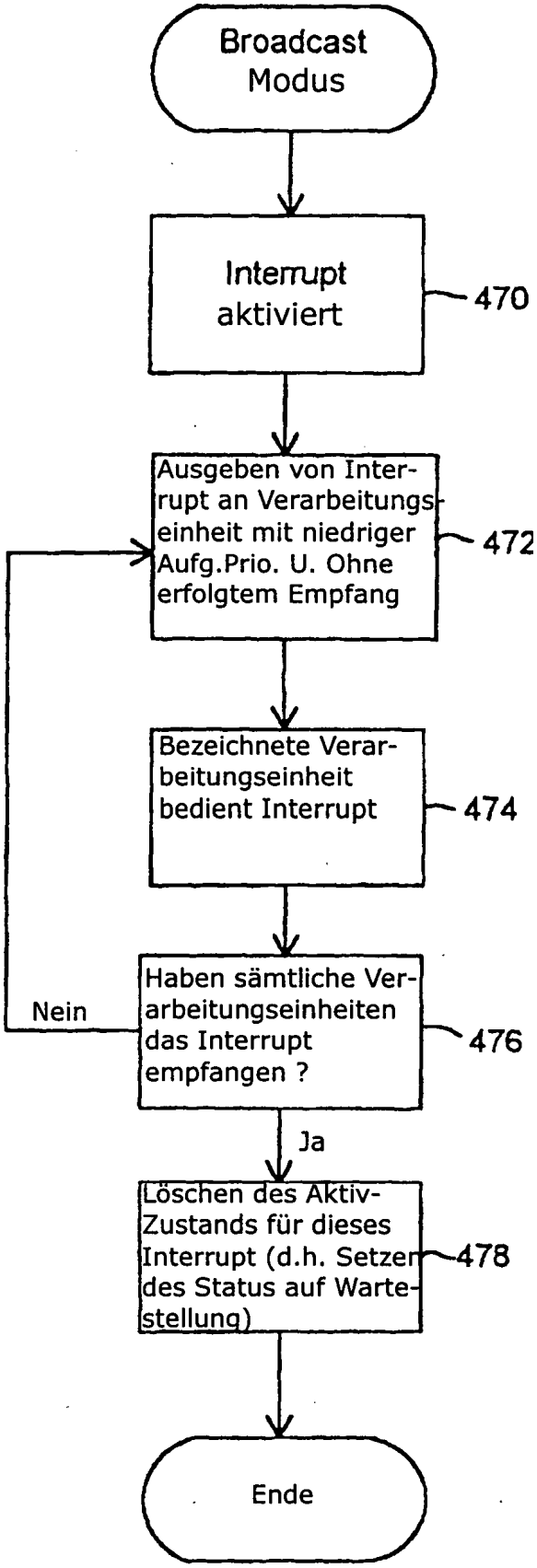

Broadcast Modus
Interrupt aktiviert
470
Ausgeben von Interrupt an Verarbeitungseinheit mit niedriger Aufg.Prio. U. Ohne erfolgtem Empfang
472
Bezeichnete Verarbeitungseinheit bedient Interrupt
474
Haben sämtliche Verarbeitungseinheiten das Interrupt empfangen ?
476
Nein
Ja
Löschen des Aktiv-Zustands für dieses Interrupt (d.h. Setzen des Status auf Wartestellung)
478
Ende

Fig. 19

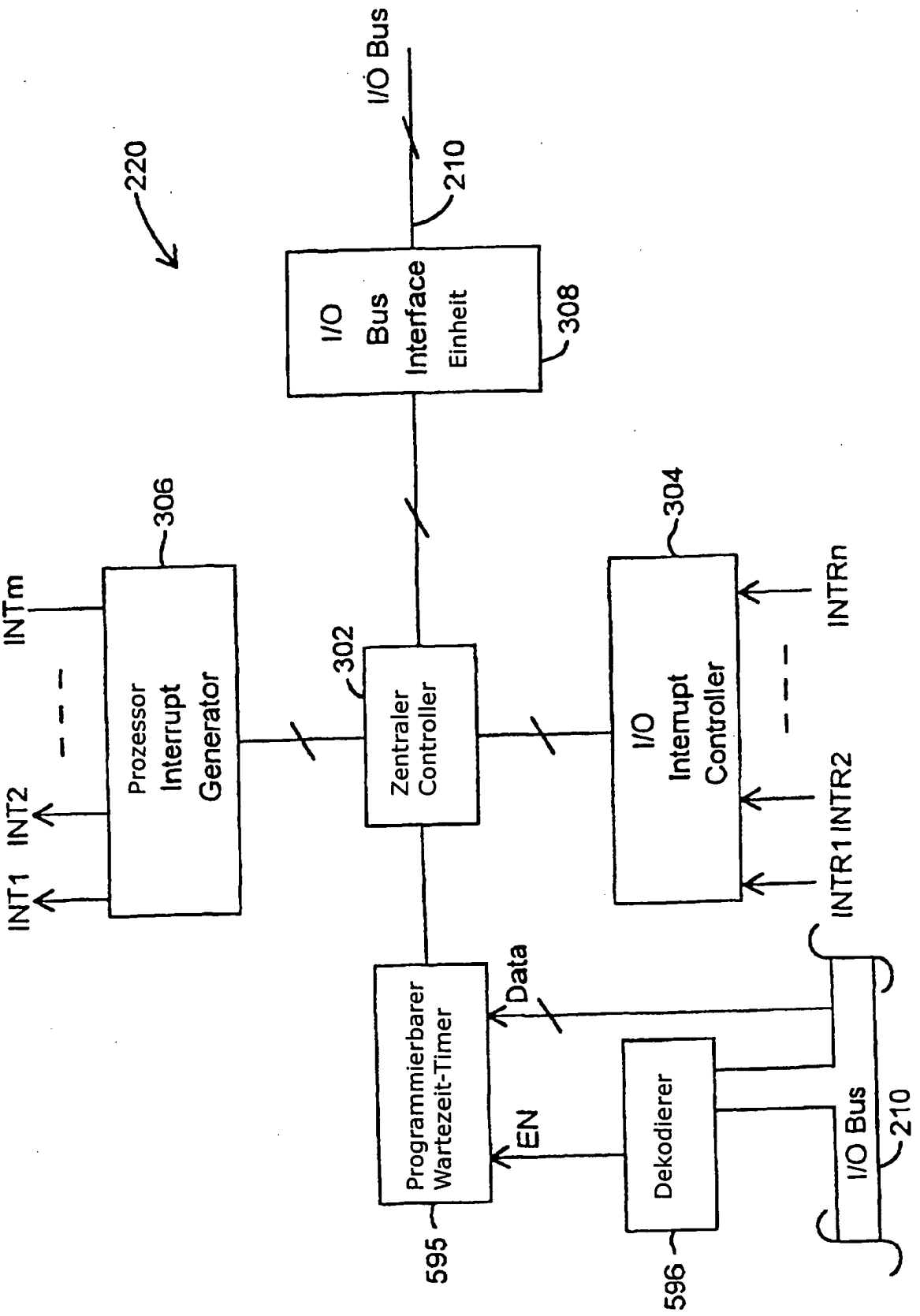

220
I/O Bus
210
I/O Bus Interface Einheit
308
306
304
INTm
INT2
INT1
Prozessor Interrupt Generator
302
Zentraler Controller
I/O Intemupt Controller
INTRn
INTR1 INTR2
Data
EN
Programmierbarer Wartezeit-Timer
Dekodierer
I/O Bus
210
595
596

Fig. 20

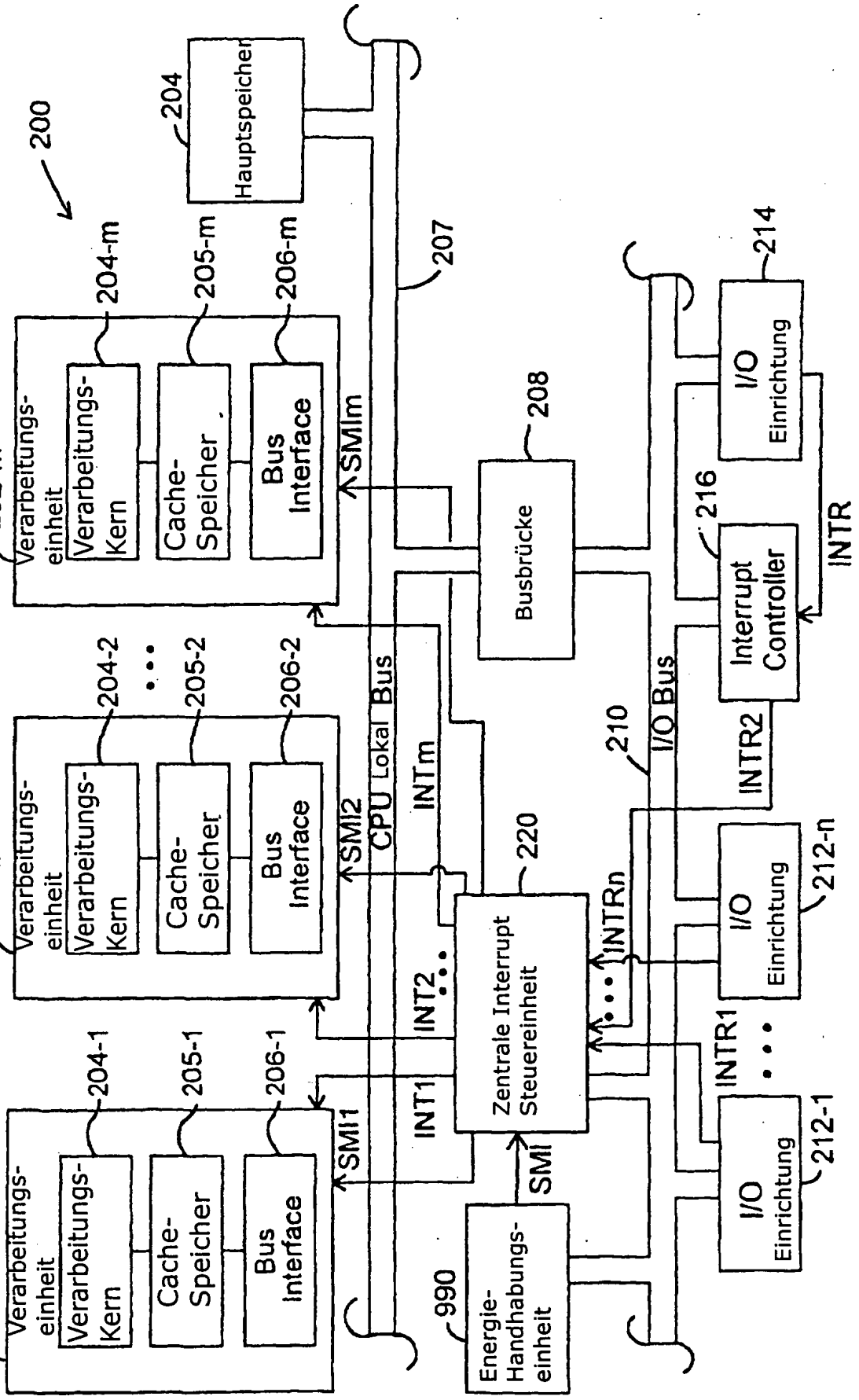

200
Hauptspeicher
204
207
202-m
Verarbeitungs-einheit
Verarbeitungs-Kern
204-m
Cache-Speicher
205-m
Bus Interface
206-m
SMIm
202-2
Verarbeitungs-einheit
Verarbeitungs-Kern
204-2
Cache-Speicher
205-2
Bus Interface
206-2
SMI2
202-1
Verarbeitungs-einheit
Verarbeitungs-Kern
204-1
Cache-Speicher
205-1
Bus Interface
206-1
SMI1
CPU Lokal Bus
INTm
INT2
INT1
Busbrücke
208
Zentrale Interrupt Steuereinheit
220
INTRn
SMI
Energie-Handhabungs-einheit
990
210
I/O Bus
I/O Einrichtung
214
I/O Einrichtung
212-n
I/O Einrichtung
212-1
INTR1
INTR2
Interrupt Controller
216
INTR

Fig. 21

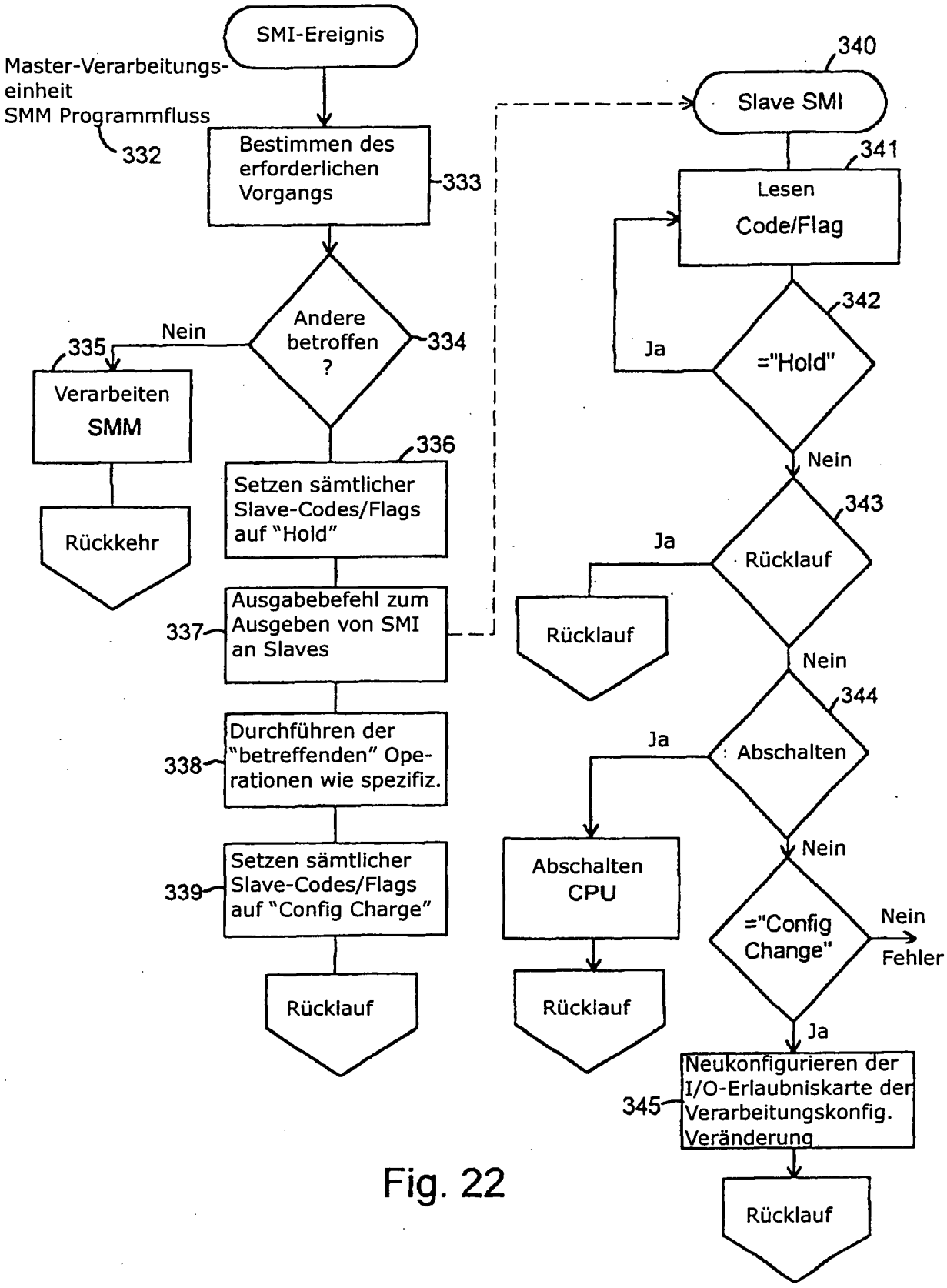

SMI-Ereignis
Master-Verarbeitungs-einheit SMM Programmfluss
332
Bestimmen des erforderlichen Vorgangs
333
Andere betroffen ?
334
Nein
335
Verarbeiten SMM
Rückkehr
336
Setzen sämtlicher Slave-Codes/Flags auf "Hold"
337
Ausgabebefehl zum Ausgeben von SMI an Slaves
338
Durchführen der "betreffenden" Operationen wie spezifiz.
339
Setzen sämtlicher Slave-Codes/Flags auf "Config Charge"
Rücklauf
340
Slave SMI
341
Lesen Code/Flag
342
="Hold"
Ja
Nein
343
Rücklauf
Ja
Rücklauf
Nein
344
Abschalten
Ja
Abschalten CPU
Rücklauf
Nein
="Config Change"
Nein
Fehler
Ja
345
Neukonfigurieren der I/O-Erlaubniskarte der Verarbeitungskonfig. Veränderung
Rücklauf

Fig. 22