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

(10) **DE 601 11 324 T2** 2006.05.18

(12) # Übersetzung der europäischen Patentschrift

(97) **EP 1 193 716 B1**
(21) Deutsches Aktenzeichen: **601 11 324.1**
(96) Europäisches Aktenzeichen: **01 307 876.1**
(96) Europäischer Anmeldetag: **17.09.2001**
(97) Erstveröffentlichung durch das EPA: **03.04.2002**
(97) Veröffentlichungstag der Patenterteilung beim EPA: **08.06.2005**
(47) Veröffentlichungstag im Patentblatt: **18.05.2006**

(51) Int Cl.[8]: ***G11C 29/00*** (2006.01)
G11C 7/10 (2006.01)
G01R 31/3193 (2006.01)

(30) Unionspriorität:
665892 **20.09.2000** **US**

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

(73) Patentinhaber:
Agilent Technologies Inc., A Delaware Corp., Palo Alto, Calif., US

(72) Erfinder:
Cook, III, John H., Fort Collins, US; Singh, Preet P., Freemont, US; de la Puente, Edmundo, Cupertino, US

(74) Vertreter:
Schoppe, Zimmermann, Stöckeler & Zinkler, 82049 Pullach

(54) Bezeichnung: **Fehlerdiagnose-RAM eines Speichertesters mit nach Grösse und Geschwindigkeit konfigurierbaren SDRAM Speichereinheiten**

Beschreibung

Hintergrund der Erfindung

[0001] Elektronikvorrichtungen und -fähigkeiten sind im täglichen Leben ganz gewöhnlich geworden. Neben Personalcomputern zu Hause tragen viele Menschen mehr als ein Produktivitätswerkzeug für verschiedene und vielfältige Zwecke bei sich. Viele persönliche Produktivitätselektronikvorrichtungen umfassen irgendeine Form von nichtflüchtigem Speicher. Mobiltelefone verwenden einen nichtflüchtigen Speicher, um benutzerprogrammierte Telefonnummern und Konfigurationen zu speichern und zu halten, wenn die Leistung abgeschaltet wird. PCMCIA-Karten verwenden einen nichtflüchtigen Speicher, um Informationen selbst dann zu speichern und zu halten, wenn die Karte aus ihrem Schlitz in dem Computer entfernt wird. Viele andere gängige Elektronikvorrichtungen nutzen ebenfalls die Langzeitspeicherfähigkeit eines nichtflüchtigen Speichers bei nicht mit Leistung versorgten Anordnungen.

[0002] Hersteller von nichtflüchtigen Speichern, die Elektronikausrüstungshersteller beliefern, benötigen Tester, um die richtige Operation der Speicher, die sie herstellen, auszuführen und zu verifizieren. Aufgrund des Volumens von nichtflüchtigen Speichern, die zu konstant niedrigen Preisen hergestellt und verkauft werden, ist es sehr wichtig, die Zeit zu minimieren, die benötigt wird, um ein einzelnes Teil zu testen. Käufer von nichtflüchtigen Speichern sind darauf angewiesen, dass Speicherhersteller eine hohe Lieferungsausbeute liefern, aufgrund der Kosteneinsparungen, die mit der Praxis eines Einbauens der Speichervorrichtungen in teurere Anordnungen mit geringem oder keinem Testen zusammenhängen. Dementsprechend muss der Speichertestprozess ausreichend effizient sein, um einen großen Prozentsatz von nichtkonformen Teilen und bevorzugt alle nichtkonformen Teile in einem einzigen Testprozess zu identifizieren.

[0003] In dem Maße, in dem nichtflüchtige Speicher größer, dichter und komplexer werden, müssen die Tester in der Lage sein, die gesteigerte Größe und Komplexität zu handhaben, ohne die Zeit beträchtlich zu steigern, die benötigt wird, um dieselben zu testen. Speichertester laufen oft durchgehend, und die Testzeit wird als einer der wichtigsten Faktoren bei den Kosten des Endteils betrachtet. Da Speicher weiterentwickelt und verbessert werden, muss der Tester in der Lage sein, sich ohne Weiteres auf die Veränderungen einzustellen, die an der Vorrichtung vorgenommen wurden. Ein anderer Sachverhalt, der speziell auf das Testen von nichtflüchtigen Speichern zutrifft, besteht darin, dass wiederholte Schreiboperationen in Zellen der Speicher die Gesamtlebensdauerleistung des Teils verschlechtern können. Hersteller von nichtflüchtigen Speichern haben auf viele der Testprobleme mit einem Einbauen von speziellen Testmodi in die Speichervorrichtungen reagiert. Diese Testmodi werden durch den Käufer des Speichers überhaupt nicht benutzt, auf dieselben kann jedoch durch den Hersteller zugegriffen werden, um alle oder wesentliche Teile der Speicher in möglichst wenig Zeit und möglichst effizient zu testen. Einige nichtflüchtige Speicher können auch während des Testprozesses repariert werden. Der Tester sollte deshalb in der Lage sein, Folgendes zu identifizieren: die Notwendigkeit einer Reparatur; einen Ort der Reparatur; den benötigten Reparaturtyp; und derselbe muss dann in der Lage sein, die geeignete Reparatur durchzuführen. Ein derartiger Reparaturprozess erfordert einen Tester, der in der Lage ist, einen spezifischen nichtkonformen Abschnitt des Speichers zu erfassen und zu isolieren. Um die speziellen Testmodi und die Reparaturfunktionen voll auszunutzen, ist es von Vorteil, dass ein Tester in der Lage ist, ein Testprogramm auszuführen, das eine bedingte Verzweigung basierend auf einer erwarteten Antwort von der Vorrichtung unterstützt.

[0004] Aus einer Konzeptperspektive ist der Prozess des Testens von Speichern ein algorithmischer Prozess. Zum Beispiel umfassen typische Tests ein sequentielles Inkrementieren oder Dekrementieren von Speicheradressen, während Nullen und Einsen in die Speicherzellen geschrieben werden. Gewöhnlich wird eine Sammlung von Einsen und Nullen, die während eines Speicherzyklus geschrieben oder gelesen werden, als ein „Vektor" bezeichnet, während der Begriff „Muster" sich auf eine Sequenz von Vektoren bezieht. Herkömmlicherweise umfassen Tests ein Schreiben von Mustern in den Speicherraum, wie z. B. Schachbrettstrukturen, wandernde Einsen und Schmetterlingsmuster. Ein Testentwickler kann ein Programm, um diese Muster zu erzeugen, einfacher und effizienter mit der Hilfe von algorithmischen Konstrukten erzeugen. Es ist auch leichter, ein Testmuster, das algorithmisch kohärent ist, zu bereinigen und logische Verfahren zu verwenden, um Abschnitte des Musters, die nicht erwartungsgemäß wirksam sind, zu isolieren. Ein Testmuster, das algorithmisch unter Verwendung von Anweisungen und Befehlen erzeugt wird, die in Programmierschleifen wiederholt werden, verbraucht in einem Testerspeicher weniger Platz. Dementsprechend ist es erwünscht, in einem Speichertester eine algorithmische Testmustererzeugungsfähigkeit zu haben.

[0005] Eine präzise Signalflankenplatzierung und -erfassung muss ebenfalls bei der Wirksamkeit eines nichtflüchtigen Testers berücksichtigt werden. Um Teile zu erfassen, die allgemein bei einem Medianwert konform

sind, während sie innerhalb der spezifizierten Grenzwerte nicht konform sind, muss ein Tester für nichtflüchtige Speicher in der Lage sein, jede Signalflanke zeitlich relativ zu einer anderen Signalflanke genau zu platzieren. Es ist auch wichtig, in der Lage zu sein, genau zu messen, zu welchem Zeitpunkt eine Signalflanke empfangen wird. Dementsprechend sollte ein Tester für nichtflüchtige Speicher eine ausreichende Flexibilität und Steuerung der Zeitgebung und der Platzierung von Stimuli und Antworten von dem Testobjekt (Speicher) aufweisen.

[0006] Speichertester erzeugen bekannterweise Sendevektoren, die an das DUT (Testobjekt) angelegt werden (Stimulus), und Empfangsvektoren, die zurück erwartet werden (Antwort). Die algorithmische Logik, die diese Vektoren erzeugt, kann dies im Allgemeinen, ohne sich damit zu beschäftigen, wie ein bestimmtes Bit in einem Vektor zu oder von einer bestimmten Signalanschlussfläche in dem DUT gelangt. Auf dieser Ebene ist es fast so, als ob es sicher wäre, dass benachbarte Bits in dem Vektor als physisch benachbarte Signale an dem DUT enden. Wenn das Leben doch so einfach wäre!

[0007] In der Realität tendiert die Entsprechung zwischen Bits in einem Vektor auf der „Konzeptebene" und den tatsächlichen Signalen in dem DUT dazu, ziemlich willkürlich zu sein. Falls nichts unternommen wird, um dies zu verhindern, kann es nötig sein, einen oder mehr Sondendrähte zu kreuzen, wenn sie von einer Peripherie herunterkommen, um einen Kontakt mit dem DUT herzustellen. Ein derartiges Kreuzen ist sehr unerwünscht, und herkömmlicherweise wird ein Abbildungsmechanismus in den Weg des Sendevektors eingegliedert, um die Bitpositionen in dem Sendevektor neu anzuordnen, bevor dieselben an das DUT angelegt werden, so dass die Aufgabe des Herstellens eines physischen Kontakts nicht mit Kreuzungen belastet ist. Empfangsvektoren werden dementsprechend an einen Rückwärtsabbildungsmechanismus angelegt, bevor dieselben betrachtet werden. Auf diese Weise können die algorithmische Vektorerzeugung und Vergleichsmechanismen diesen gesamten Sachverhalt ignorieren. Als ein weiteres Beispiel dafür, was derartige Abbildungsvorrichtungen und Rückwärtsabbildungsvorrichtungen leisten können, sei der Fall betrachtet, wenn eine unterschiedliche Instanz des gleichen Typs von DUT auf dem gleichen Wafer angeordnet ist, jedoch mit einer Drehung oder irgendeiner gespiegelten Symmetrie, um Platzverschwendung auf dem Wafer zu vermeiden. Diese Praktiken haben auch eine Wirkung auf die Entsprechung zwischen Vektorbitposition und physischem Signalort, die jedoch durch die geeigneten Abbildungen und Rückwärtsabbildungen verborgen werden kann. Es sei darauf hingewiesen, dass die Abbildungen und Rückwärtsabbildungen, die für diese Situationen benötigt werden, sobald sie einmal für ein bestimmtes DUT identifiziert worden sind, statisch sind und sich im Laufe des Testens für dieses bestimmte DUT nicht zu ändern brauchen.

[0008] Es wurde im Vorhergehenden erwähnt, dass das DUT eventuell eine Reparatur zulässt. Dies ist oft selbst für nicht vereinzelte Speicherchips der Fall, die noch ein Teil eines Wafers sind. Wie dies tatsächlich auf der Schaltungsebene erreicht wird, ist für Hersteller derartiger Vorrichtungen klar, so dass es ausreichend ist, einfach darauf hinzuweisen, dass in diese Vorrichtungen eine bestimmte Anzahl von auswählbar zerstörbaren Elementen eingegliedert ist, deren Zerstörung ein Durchschalten ermöglicht, das wiederum die interne Logik einer zugeordneten Schaltung ändert. Diese Fähigkeit wird verwendet, um interne Signale zu Ersatzschaltungen zu leiten, die fehlerhafte ersetzen. Diese Fähigkeit kann wirtschaftlich nicht lohnend sein, außer die Reparatur kann in geringerer Zeit und mit weniger Aufwand vorgenommen werden, als erforderlich wäre, um ein neues Teil herzustellen; ansonsten wäre es kosteneffizienter, das schlechte Teil einfach in die Schrotttonne zu werfen. Insbesondere ist es unerwünscht, einen menschlichen Techniker in die Prozesse des Verstehens der bestimmten Fehler in einem Strom von schlechten Teilen und der Zuständigkeit zum Entscheiden, wie dieselben zu reparieren sind, einzubeziehen. Stattdessen kann ein algorithmischer Mechanismus (Programm) in dem Speichertester entwickelt werden, um den Fehler zu analysieren und seine Reparatur zu versuchen. Das reparierte Teil kann auf der Stelle erneut getestet und sein Schicksal entschieden werden.

[0009] Ein derartiger Operationsmodus umfasst bestimmte Implikationen für den Entwurf des Speichertesters. Das Testen muss mit der Geschwindigkeit durchgeführt werden, die für geeignet angesehen wird, wobei es sich häufig um die höchsten Geschwindigkeiten handelt, mit denen das Teil wirksam sein soll. Eine Echtzeiterfassung von Fehlern kann verwendet werden, um Flags zu setzen und Testalgorithmen zu ändern, um das Verständnis des Fehlers zu verbessern. Das heißt, Tests, die durchgeführt werden, um eine richtige Operation zu verifizieren, sind eventuell nicht diejenigen, die am besten dazu geeignet sind, zu entdecken, warum das Teil überhaupt versagt. Schließlich muss der Speichertester in der Lage sein, eine Spur (d. h. eine verwendbare Aufzeichnung) von Testdaten für eine automatische Analyse zu erzeugen (die entweder sofort oder bei Abschluss eines größeren Testprozesses durchgeführt wird), die bestimmt, ob eine Reparatur versucht werden soll, und falls dies der Fall ist, welche Maßnahmen zu ergreifen sind, um die Reparatur durchzuführen.

[0010] Normalerweise wird der Reparaturversuch aufgeschoben, bis zumindest ein vorläufiges Testen den Umfang oder die Anzahl von wahrscheinlichen Fehlern aufzeigt. Die Anzahl von verfügbaren Ersatzschaltun-

gen ist begrenzt (z. B. auf ein halbes Dutzend, wie es durch eine chancengestützte Kosten-Nutzen-Analyse bestimmt ist), und es hat keinen Sinn, zu versuchen, ein Teil zu reparieren, bei dem gezeigt werden kann, dass dasselbe mehr Hilfe benötigt als verfügbar ist. Falls das Testen des DUT mit einer hohen Geschwindigkeit und ohne unnötige Pausen durchgeführt werden soll, ist es offensichtlich, dass der Speicher des Testers, der verwendet wird, um die Spur zu erzeugen, die die Fehler beschreibt, mit den gleichen hohen Geschwindigkeiten wirksam sein muss, die verwendet werden, um das DUT zu testen. Bei dem Speichertester, der hier beschrieben werden soll, wird dieser Speicher der ECR genannt (Fehlererfassungs-RAM).

[0011] In Betrieb wird ein ECR normalerweise durch die gleiche Adresse adressiert, die an das DUT angelegt wird, und weist eine Datenwortbreite in Bits auf, die zumindest gleich derjenigen des DUT ist. Die Wortbreite ist entlang Potenzen von Zwei (8, 16, 32) einstellbar, wobei eine derartige Einstellbarkeit von einer entsprechenden inversen Veränderung der Adressierbarkeit begleitet ist, so dass die Wortbreite mal die Anzahl von adressierbaren Orten gleich irgendeiner Konstante ist.

[0012] Wenn ein Testkanal für das DUT (ein Bit in einem Ausgangswort oder irgendein anderes interessierendes Signal) mit erwarteten Ergebnissen vergleichbar ist oder nicht vergleichbar ist, wird gemäß der verwendeten Konvention ein entsprechendes Bit an dieser Adresse in dem ECR entweder gesetzt oder gelöscht. So organisiert weist der ECR keinen Mehrbitwert für jede Adress-/Kanalkombination auf und kann stattdessen nur ein einziges Bit an Informationen für jede derartige Kombination speichern, egal wie oft auf diese Kombination während eines Tests zugegriffen wird. Eine Teststrategie ist daran beteiligt, was das Bit bedeutet und wie es aufrechterhalten wird. Das Bit kann die Dichotomie „nie versagt/zumindest einmal versagt" für einen gesamten Mehrzugriffstest darstellen, oder dasselbe kann nur das Resultat des letzten Zugriffs (d. h. Test) darstellen, selbst wenn dieses sich von früheren Tests unterscheidet. Falls Quantitätsinformationen über Fehler für eine bestimmte Adresse/Kanal gewünscht sind, muss eine zusätzliche Ressource (ein Zähler) zugeteilt werden, um dieselben aufzuzeichnen.

[0013] Herkömmliche Speichertester verwendeten einen SRAM für ihre ECRs. Auf einen SRAM wird unter Verwendung einer einzigen vereinheitlichten Adresse zugegriffen, und derselbe ist schneller als ein DRAM, wenn derselbe willkürlich adressiert wird, derselbe ist jedoch auch beträchtlich teurer. Der kostengünstigere DRAM ist intern organisiert, um das langwierige Vorladen einer adressierten „Zeile" mit RAS (Zeilenadressübernahmesignal) gefolgt von einem Spezifizieren einer adressierten „Spalte" mit CAS (Spaltenadressübernahmesignal) zu erfordern. Ein DRAM ist oft ausreichend schnell, wenn, nachdem eine Zeile vorgeladen wurde, ein weiteres Adressieren auf Spalten entlang dieser Zeile beschränkt werden kann (d. h, weitere Instanzen von CAS, aber keine von RAS). Eine derartige algorithmische Einschränkung der Testeroperation (die in die Fähigkeit eingreift, das DUT willkürlich zu adressieren) ist jedoch oft nicht akzeptabel, und obwohl es manchmal nützlich ist, besteht kein Verlass, dass dasselbe eine ECR-Operation hoher Geschwindigkeit liefert. Es wäre erwünscht, wenn durch ein Verwenden eines DRAM die Größe des ECR vergrößert und seine Kosten verringert werden könnten, wobei diese Vorzüge realisiert werden könnten, wenn es einen Weg gäbe, DRAMs mit einer willkürlichen Adressierung mit der gleichen Rate zu betreiben, die gewöhnlich von den teureren SRAMs erwartet wird.

[0014] Als ein Verbraucher von im Handel erhältlichen Teilen besteht keine Möglichkeit, bestehende einzelne DRAM-Teile um eine Größenordnung oder mehr schneller zu machen. Die einzige Option besteht darin, mehr DRAM zu verwenden, bis zu dem Punkt, wo dabei so viel ausgegeben wird wie für eine gewünschte Menge von SRAM. Dies ist attraktiv, da ein SRAM beträchtlich teurer ist als ein DRAM. Man denkt an Multiplexen, siehe z. B. US 5,905,909, aber ein n-Teil-Multiplexschema erzeugt eine zugehörige n-fache Steigerung der Anzahl von verwendeten Speicherbussen. Bei z. B. 50 bis 60 Anschlussstiften pro Bus wäre ein Zehn-Weg-Multiplexer ein Albtraum, schon allein um die benötigen physische Ausgangsverzweigung zu realisieren. Falls doch ein Weg gefunden wird, all diesen Speicher aufzustapeln und in denselben mit einer hohen Geschwindigkeit zur Verwendung als einem Fehlererfassungs-RAM zu schreiben, wäre es außerdem auch erwünscht, in der Lage zu sein, denselben für andere Verwendungen ohne Weiteres umzukonfigurieren, z. B. wenn bekannt ist, dass die Direktzugriffsgeschwindigkeit niedriger ist, oder wenn es erwünscht ist, mit einer hohen Geschwindigkeit sowohl lesen als auch schreiben zu können unter Verwendung von einfachen Verfahren, die den Teilen eigen sind, und vorausgesetzt, dass der Hauptmodus des Adressierens auf Veränderungen bei der Spaltenadresse beschränkt ist. Welche Lösung gibt es?

Zusammenfassung der Erfindung

[0015] Das Problem des Erhöhens der Geschwindigkeit einer DRAM-Operation zur Verwendung bei einem Fehlererfassungs-RAM kann durch eine Kombination eines Verschachtelns von Signalen für unterschiedliche

Speicherbänke in einer Gruppe derselben und eines Multiplexens zwischen diesen Bankgruppen gelöst werden. Ein Dreiwegmultiplexen zwischen drei Gruppen von jeweils vier Bänken kombiniert mit einem flexiblen Vierfachverschachtelungsschema für Signalverkehr zu einer Gruppe erzeugt eine Steigerung der Betriebsgeschwindigkeit, die sich einem Faktor von zwölf nähert, während nur drei Speicherbusse erforderlich sind. Eine zyklische Suchstrategie (Round-Robin-Strategie) zum Auswählen der nächsten Gruppe für den Multiplexer ist einfach und stellt sicher, dass der Verschachtelungsmechanismus für jede Gruppe die Zeit hat, die derselbe benötigt, um seine aktuell zugewiesene Aufgabe abzuschließen. Ungeachtet dessen, ob die nächste Adresse in einer Gruppe die gleiche ist wie die vorhergehende Adresse, auf die in dieser Gruppe zugegriffen wurde, benachbart oder fast benachbart ist oder sich weit von derselben entfernt befindet, werden alle verschachtelten Zugriffe in einer Gruppe bei einer nächsten Bank (in dieser Gruppe) durchgeführt, die ebenfalls durch eine Round-Robin-Auswahl ausgewählt wird, anstatt unnötigerweise eine Echtzeithochgeschwindigkeitsadressanalyse durchzuführen in einem Versuch, eine In-Bank-Lokalität zu erreichen. Bei dieser Konfiguration stellt jede der zwölf Bänke den gesamten verfügbaren Adressraum dar, und jeder einzelne Schreibzyklus kann damit enden, auf eine beliebige der zwölf Bänke zuzugreifen. Eine Implikation besteht darin, dass beim Abschluss des Testens alle zwölf Bänke untersucht werden müssen, um herauszufinden, welche Fehler während des Testens des DUT aufgetreten sind, da die Historie einer beliebigen interessierenden Adresse oder Sammlung von Adressen über alle zwölf Bänke verteilt ist. Das heißt, um zu bestimmen, welche Kanäle an einer Adresse bestanden oder versagt haben, ist es notwendig, Leseoperationen bei jeder der zwölf Bänke durchzuführen (wo in jeder Bank wird durch die Adresse bestimmt) und die Bedeutung der zwölf so erzeugten Bitsammlungen auszuwerten. Ein bestimmter Kanal wird somit durch zwölf Bits dargestellt (ein Bit von jeder Bank, und wobei dessen Bitposition in dem Wort für diese Bank durch den Kanal bestimmt wird).

[0016] Es wäre jedoch umständlich, einzeln (sozusagen manuell) alle zwölf Bänke untersuchen zu müssen, um Fehlerinformationen zu entdecken, deshalb wurde ein Hilfsmechanismus bereitgestellt, um automatisch Ergebnisse aller zwölf Bänke während eines ECR-Lesezyklus an einer Adresse in ein vereinigtes Ergebnis „zusammenzusetzen" (zusammenzuführen). Das heißt, es sei angenommen, dass eine Null in einer Adress-/Kanalkombination das Fehlen einer Vergleichbarkeit darstellt. Dann ist das i-te Bit des zusammengesetzten Ergebnisses eine Null, falls und nur falls zumindest eines der zwölf Wörter (eines von jeder der zwölf Bänke) ein i-tes Bit aufwies, das Null war. Eine Zusammensetzung erfolgt für alle Kanäle gleichzeitig, jedoch auf einer Adresse-für-Adresse-Basis. Falls es erwünscht ist, kann das zusammengesetzte Ergebnis für eine zukünftige Referenz in eine ausgewählte Bank oder vielleicht gleichzeitig in alle Bänke gespeichert werden. Wenn dies in einer Schleife enthalten ist, die diese Operation über einen ganzen Bereich von interessierenden Adressen durchführt (z. B. den gesamten Adressraum, der getestet wurde), wird dies sehr praktisch für nachfolgende Fehleranalysemechanismen. Falls die zusammengesetzten Daten in allen Bänken gespeichert wurden, können dieselben mit voller Geschwindigkeit in einer willkürlichen Reihenfolge zurückgelesen werden. Es gibt auch Mechanismen, um das Verfolgen der Integrität der zusammengesetzten Ergebnisse zu unterstützen, so dass, wenn es zu einem weiteren Testen kommt (d. h. eine Schreiboperation vorliegt, die nicht gleichzeitig auf alle zwölf Bänke gemeinsam gerichtet ist), es möglich ist, festzustellen, dass ein weiterer Zusammensetzungsschritt (fast sicher) benötigt wird. Diese Mechanismen können Register umfassen, die höchste und niedrigste Adressen, in die geschrieben worden ist, diverse Flags und Betriebssystemebeneninformationen darüber, welches Programm einen Bereich des ECR-Adressraums „besitzt", verfolgen, so dass eine Zusammensetzungsverwaltung flexibel ist und nach Wunsch minimal oder umfassend sein kann.

[0017] Der ECR ist auch in vier Speichersätze aufgeteilt, von denen zwei „interne" SRAMs sind und von denen zwei „externe" DRAMs sind. Natürlich sind alle diese Speicher innerhalb des Speichertesters; die Begriffe „intern" und „extern" beziehen sich eher auf eine Integrationsebene. Die SRAMs sind integrierte Teile von VLSI-(Very Large Scale Integration – Integration sehr großen Umfangs)Schaltungen, die dem ECR zugeordnet sind, während die DRAMs einzelne gehäuste Teile sind, die benachbart zu den VLSI-Elementen befestigt sind. Die SRAM-Menge ist ziemlich klein (z. B. etwa 1 Megabit pro Speichersatz), während die DRAM-Menge beträchtlich und auswählbar ist (z. B. in dem Bereich von 128 bis 1.024 Megabit pro Speichersatz). Die SRAM-Speichersätze sind immer vorhanden und können für jeden beliebigen geeigneten Zweck verwendet werden, wie z. B. ein Speichern des erwarteten Inhalts eines DUT, das ein ROM (Nur-Lese-Speicher) ist. Die DRAM-Speichersätze sind tatsächlich optional und werden normalerweise zum Erzeugen einer Spur zur nachfolgenden Analyse, die zu einer Reparatur führt, verwendet, obwohl es viele andere Verwendungen gibt. Der Tester setzt keine Unterscheidung zwischen den SRAM- und DRAM-Speichersätzen bezüglich verschiedener Zwecke, zu denen dieselben verwendet werden können, durch. Diese Unterscheidungen ergeben sich hauptsächlich aufgrund der Größe. Die SRAM-Speichersätze sind klein, während die DRAM-Speichersätze groß sind. Die Person oder Personen, die die Testprogrammierung erzeugen, treffen die Entscheidungen darüber, wie die verschiedenen Speichersätze verwendet werden sollen. Da jedoch ein SRAM bereits mit hoher Geschwindigkeit wahlfrei adressierbar ist, umfasst derselbe nicht den Multiplex-/Verschachtelungsmechanismus

für mehrere Bänke; jeder SRAM-Speichersatz ist also einfach eine einzelne Bank. Somit ist derselbe immer zusammengesetzt und benötigt keinen separaten Zusammensetzungsmechanismus.

[0018] Jeder der vier Speichersätze weist seine eigene Steuerung auf, und ihre Operation ist konfigurierbar, um unterschiedliche Modi einer ECR-Operation zu unterstützen. Ein Aspekt davon betrifft den Typ von speicherbezogenen Transaktionen, die die Speichersatzsteuerungen unterstützen. Es stimmt, dass auf der Speicherseite einer Speichersatzsteuerung einzelne Speicherzyklen in ihrer Beschaffenheit als „Leseoperation" oder „Schreiboperation" klassifizierbar sind. Auf ihrer Systemseite erkennt eine Speichersatzsteuerung jedoch mehrere unterschiedliche Speichertransaktionsstile. Diese umfassen: (A) Eine Überlagerungsschreiboperation, die „klebende Nullen" für Daten implementiert, die zu unterschiedlichen Zeiten geschrieben werden. Falls in eine beliebige Bitposition an einer Adresse zu einer beliebigen Zeit während dieses Modus eine Null geschrieben wird, erzeugt diese Bitposition an dieser Adresse eine Null, wenn dieselbe gelesen wird, selbst wenn Schreiboperationen einer Eins in diese Bitposition an dieser Adresse nach der Schreiboperation einer Null vorlagen. (B) Eine Überschreibschreiboperation, die eine strenge Ersetzung von adressierten Daten durch gelieferte Daten darstellt (d, h. eine reguläre Schreiboperation). (C) Eine Systemschreiboperation, die die gleichen Daten in alle Bänke schreibt, falls der Speichersatz extern ist. (D) Eine Analyseleseoperation, die von allen Bänken zusammensetzt, falls der Speichersatz extern ist. (E) Eine Pufferspeicherleseoperation, die Daten von einem Ausgewählten zurückliest, falls der Speichersatz extern ist. Diese Speichertransaktionsstile sind jeweils sowohl mit einem internen SRAM als auch einem externen DRAM ausführbar. Der einzige wirkliche Unterschied besteht darin, wie lange die Erledigung benötigt, und in der Erkenntnis, dass, falls sich die vorhergehenden Beschreibungen auf „alle Bänke" oder „eine ausgewählte Bank" beziehen und der Zielspeichersatz ein interner SRAM ist, dann dieser Zielspeichersatz ein Speicher einer Bank ist, die gleichzeitig sowohl „alle" Bänke als auch die „ausgewählte" Bank ist. Es ist dann offensichtlich, dass, obwohl alle Speichersätze (zumindest im Prinzip) durch die Benutzungssoftware gleichwertig behandelt werden können, als ob dieselben nur SRAM wären, es Gründe gibt, Unterschiede bei der internen Operation der verschiedenen Speichersatzsteuerungen zu erwarten.

[0019] Es gibt einige zusätzliche Konfigurationseigenschaften, die den DRAM-Speichersätzen zugeordnet sind. Für die externen DRAM-Speichersätze ermöglicht der im Vorhergehenden erwähnte Multiplex- und Verschachtelungsmodus einen vollen Direktzugriff bei Geschwindigkeiten von bis zu 100 MHz. Falls bekannt ist, dass Geschwindigkeiten 300 MHz nicht überschreiten, dann kann die interne Operation der externen DRAM-Speichersätze des ECR konfiguriert sein, um im Ausgleich für die niedrigere Geschwindigkeit die dreifache Tiefe zu liefern. Dies wird erreicht durch ein Entfernen des Multiplexens zwischen Gruppen zugunsten eines bloßen Verschachtelns bei einer größeren Gruppe; Bankfreigabebits, die als ein Teil des Multiplexens verwendet wurden, können nun als reguläre Adressbits verwendet werden, um die Größe des Adressraums der einen verbleibenden Gruppe zu vergrößern. Falls schließlich das Testen des DUT mit dem „linearen" Zugriffsmodus (ein RAS, viele CAS) zusammenpasst, ist eine zwölffache Steigerung der Speichertiefe verfügbar, selbst wenn das DUT mit der höchsten Geschwindigkeit getestet wird, mit der der Tester wirksam sein kann. Dies beseitigt das Verschachtelungsschema zugunsten eines Adressierens immer nur in jeweils einer einzigen Bank, und ist möglich aufgrund der besonderen Beschaffenheit von DRAMs, wenn dieselben mit einer linearen Adressierung verwendet werden.

[0020] Eine weitere flexible Neukonfiguration, die möglich ist, besteht darin, die externen DRAM-Speichersätze in einen Speichersatz zu kombinieren, der die doppelte Tiefe jedes nichtkombinierten Satzes aufweist, unabhängig von anderen (z. B. den geschwindigkeitsbezogenen) Operationsmodi. Dies kann auch für die internen SRAM-Speichersätze erfolgen.

[0021] Gemäß einem ersten Ausführungsbeispiel der vorliegenden Erfindung wird ein Verfahren zum Durchführen einer Speicheroperation bei einem DRAM geliefert, wie dasselbe in dem angehängten Anspruch 1 dargelegt ist, und gemäß einem zweiten Ausführungsbeispiel der vorliegenden Erfindung wird ein Verfahren zum Durchführen einer Speicheroperation bei einem DRAM geliefert, wie dasselbe in dem angehängten Anspruch 6 dargelegt ist.

Kurze Beschreibung der Zeichnungen

[0022] Fig. 1 ist ein vereinfachtes Blockdiagramm eines umfassend neu konfigurierbaren Testers für nichtflüchtige Speicher, der gemäß der Erfindung aufgebaut ist;

[0023] Fig. 2 ist eine vereinfachte Blockdiagrammvergrößerung des DUT-Testers **6** von Fig. 1;

[0024] Fig. 3 ist ein vereinfachtes Funktionsblockdiagramm des ECR-(Fehlererfassungs-RAM-)Mechanismus, der in dem Blockdiagramm von Fig. 2 erscheint;

[0025] Fig. 4 ist ein genaueres Blockdiagramm des ECR-Mechanismus von Fig. 3;

[0026] Fig. 5 ist ein vereinfachtes Blockdiagramm eines DRAM-Speichersatzsteuerungsmechanismus, der in den Blockdiagrammen der Fig. 3 und Fig. 4 erscheint;

[0027] Fig. 6 ist ein Blockdiagramm eines Master-DRAM-Steuerungsmechanismus, der in dem Diagramm von Fig. 5 erscheint;

[0028] Fig. 7 ist ein Blockdiagramm einer Zusammensetzungsschaltung, die in dem Blockdiagramm von Fig. 6 erscheint;

[0029] Fig. 8 ist ein Blockdiagramm eines Slave-SDRAM-Steuerungsmechanismus, der in dem Blockdiagramm von Fig. 5 erscheint;

[0030] Fig. 9 ist ein vereinfachtes Blockdiagramm einer Gruppe von SDRAM, die durch die Steuerungen der Fig. 6 und Fig. 8 gesteuert werden; und

[0031] Fig. 10 ist ein vereinfachtes Blockdiagramm, das sich auf die Operation eines „ZUSAMMENGESETZT"-Flags bezieht.

Beschreibung eines bevorzugten Ausführungsbeispiels

[0032] Es sei nun Bezug genommen auf Fig. 1, in der ein vereinfachtes Blockdiagramm **1** eines Testsystems für nichtflüchtige Speicher gezeigt ist, das gemäß den Prinzipien der Erfindung aufgebaut ist. Insbesondere kann das gezeigte System gleichzeitig mit jeweils 64 Testpunkten bis zu 36 einzelne DUTs (Testobjekte) gleichzeitig testen, wobei eine Neukonfiguration vorgesehen ist, um zu ermöglichen, dass Elemente einer Sammlung von miteinander zu verbindenden Testressourcen DUTs testen, die mehr als 64 Testpunkte aufweisen. Diese Testpunkte können Orte an einem Abschnitt eines Integrierte-Schaltung-Wafers sein, der noch nicht vereinzelt und gehäust wurde, oder es kann sich um die Anschlussstifte eines gehäusten Teils handeln. Der Begriff „Testpunkt" bezieht sich auf einen elektrischen Ort, an den ein Signal angelegt werden kann (z. B. Leistungsversorgungen, Takte, Dateneingaben) oder an dem ein Signal gemessen werden kann (z. B. eine Datenausgabe). In diesem Text wird der Industriekonvention gefolgt, die Testpunkte als „Kanäle" zu bezeichnen. Die „Sammlung von miteinander zu verbindenden Testressourcen", auf die im Vorhergehenden Bezug genommen wurde, kann so verstanden werden, dass es sich dabei um 36 Teststellen handelt, wobei jede Teststelle eine Teststellensteuerung (**4**), einen (64-Kanal-) DUT-Tester (**6**) und eine (64-Kanal-) Sammlung von Anschlussstiftelektronik (**9**) umfasst, die eine tatsächliche elektrische Verbindung mit einem DUT (**14**) herstellt. In dem Fall, bei dem ein Testen des DUT 64 oder weniger Kanäle erfordert, ist eine einzige Teststelle ausreichend, um Tests bei diesem DUT durchzuführen, und es heißt z. B., dass die Teststelle #1 (wie dieselbe in Fig. 1 erscheint) eine „Einstellenteststation" bildet oder als eine solche wirksam ist. Andererseits werden, wenn irgendeine Form der im Vorhergehenden erwähnten Neukonfiguration vorliegt, zwei (oder mehr) Teststellen miteinander „verbunden", um als eine größere gleichwertige Teststelle zu fungieren, die 128 Kanäle aufweist. Dementsprechend und erneut mit Bezugnahme auf ein Beispiel, das in Fig. 1 gezeigt ist, heißt es, dass die Teststellen #35 und #36 eine „Zweistellenteststation" bilden.

[0033] Um kurz den entgegengesetzten Fall zu betrachten, darf nicht angenommen werden, dass eine ganze Teststelle benötigt wird, um ein einziges DUT zu testen, oder dass eine einzelne Teststelle nur ein einziges DUT testen kann. Es sei angenommen, dass ein Wafer zwei (wahrscheinlich, aber nicht unbedingt benachbarte) Chips aufweist, wobei die Summe ihrer Testkanalanforderungen 64 Kanäle oder weniger beträgt. Beide DUTs können durch eine einzige Teststelle getestet werden. Dies wird ermöglicht durch die universelle Programmierbarkeit jeder Teststelle. Ein Testprogramm, das durch die Teststelle ausgeführt wird, kann derart geschrieben sein, dass ein Teil der Ressourcen der Teststelle verwendet wird, um eines der DUTs zu testen, während ein anderer Teil verwendet wird, um das andere DUT zu testen.

[0034] Schließlich ist anzunehmen, dass, falls ein drittes DUT vorläge, das die logische Vereinigung der ersten beiden wäre, es möglich wäre, dieses dritte DUT mit einer einzigen Teststelle zu testen, so dass es möglich sein sollte, seine „Komponenten-DUTs" somit ähnlich zu testen. Der einzige Unterschied ist ein individuelles Verfolgen, ob die zwei „Komponenten-DUTs" bestehen oder versagen, im Gegensatz zu einer vereinigten Ant-

wort für das „dritte" DUT (d. h. es besteht eine Frage, welcher Abschnitt des „dritten" DUT versagt hat). Diese „Einstellenmehrteststation"-Fähigkeit ist großenteils herkömmlich, und dieselbe wird hier zur Vervollständigung erwähnt, und um eine mögliche Verwirrung und Missverständnisse zu vermeiden, wenn dieselbe mit der Idee verglichen wird, zwei oder mehr Teststellen miteinander zu verbinden.

[0035] Ohne dieses Konzept der Neukonfiguration gäbe es keinen Unterschied zwischen einer Teststelle und einer Teststation und es könnte auf einen der Begriffe verzichtet werden. So aber ist ohne Weiteres ersichtlich, dass die Anzahl von Teststationen nicht gleich der Anzahl von Teststellen sein muss. In der Vergangenheit konnten sich die Anzahlen unterscheiden, weil Teststellen geteilt wurden, um mehr Teststationen zu erzeugen (DUTs, die nicht komplex genug sind, um eine ganze Teststelle zu verbrauchen). Nun kann der Unterschied jedoch auch darin begründet sein, dass Teststellen miteinander verbunden worden sind, um Mehrstellenteststationen zu bilden (DUTs, die für eine einzige Teststelle zu komplex sind).

[0036] Um fortzufahren, ist dann eine Testsystemsteuerung **2** durch einen Systembus **3** mit insgesamt 36 Teststellensteuerungen verbunden, deren Namen mit den Suffixen #1 bis #36 enden (**4a**–**4z**). (Es stimmt, dass die Indices a–z nur von 1–26, aber nicht bis 36 reichen. Diese geringfügige Täuschung scheint aber gegenüber numerischen Indizes bei numerischen Referenzschriftzeichen, was möglicherweise sehr verwirrend sein könnte, vorzuziehen zu sein.) Bei der Testsystemsteuerung **2** handelt es sich um einen Computer (z. B. einen PC, auf dem NT läuft), der ein geeignetes Testsystemsteuerprogramm ausführt, das sich auf die Aufgabe eines Testens von nichtflüchtigen Speichern bezieht. Das Testsystemsteuerprogramm stellt die höchste Abstraktionsebene in einer hierarchischen Arbeitsteilung (und Komplexitätsteilung) zum Erreichen des gewünschten Testens dar. Die Testsystemsteuerung bestimmt, welche Programme durch die unterschiedlichen Teststellen ausgeführt werden, und überwacht ein Robotersystem (nicht gezeigt), das die Testsonden und DUTs nach Bedarf bewegt. Die Testsystemsteuerung **2** kann auf Weisen funktionieren, die das Konzept unterstützen, dass einige Teststellen programmiert sind, um als Einstellenteststationen wirksam zu sein, während andere miteinander verbunden sind, um Mehrstellenteststationen zu bilden. Offensichtlich gibt es unter derartigen Umständen unterschiedliche Teile, die getestet werden, und es ist sehr erwünscht, dass unterschiedliche Tests für die unterschiedlichen Teile verwendet werden. Gleichermaßen besteht keine Anforderung, dass alle Einstellenteststationen den gleichen Teilstil testen, und es gibt auch keine derartige Anforderung für Mehrstellenteststationen. Dementsprechend ist die Testsystemsteuerung **2** programmiert, um die Befehle auszugeben, um das benötigte Teststellenverbinden zu erreichen, und dann die geeigneten Testprogramme für die verschiedenen verwendeten Teststationen aufzurufen. Die Testsystemsteuerung **2** empfängt auch Informationen über Ergebnisse, die von den Tests erhalten werden, so dass dieselbe die geeignete Maßnahme zum Verwerfen des schlechten Teils ergreifen kann, und so dass dieselbe Protokolle für die verschiedenen Analysen unterhalten kann, die verwendet werden können, um z. B. Herstellungsprozesse in einer Fabrikumgebung zu steuern.

[0037] Das Testsystem selbst ist ein ziemlich großes und komplexes System, und normalerweise verwendet dasselbe ein Roboterteilsystem, um Wafer auf eine Bühne zu laden, die dann sequentiell einen oder mehr zukünftige Chips unter Sonden positioniert, die mit der Anschlussstiftelektronik **9** verbunden sind, woraufhin diese zukünftigen Chips (der Wafer wurde noch nicht vereinzelt) getestet werden. Das Testsystem kann auch verwendet werden, um gehäuste Teile zu testen, die auf einen geeigneten Träger geladen worden sind. Es ist (wie es im Folgenden beschrieben wird) jeder verwendeten Teststation zumindest eine Teststellensteuerung zugeordnet, unabhängig davon, wie viele Teststellen verwendet werden, um diese Teststation zu bilden, oder wie viele Teststationen sich an einer Teststelle befinden. Eine Teststellensteuerung ist ein eingebettetes System, bei dem es sich um einen i960-Prozessor von Intel mit 36 bis 64 MB von kombiniertem Programm- und Datenspeicher handeln kann, auf dem ein proprietäres Betriebssystem namens VOS (Versa Test O/S) läuft, das auch bei früheren Produkten zum Testen von nichtflüchtigen Speichern verwendet wurde (z. B. Agilent V1300 oder V3300). Im Moment soll nur die Situation für Einstellenteststationen betrachtet werden. Zu Zwecken eines klaren Beispiels sei angenommen, dass eine Teststelle #1 als eine Teststation #1 fungiert, und dass dieselbe das Teil WHIZCO Nr. 0013 testen soll. Die Testvorgabe umfasst etwa 100 unterschiedliche Testtypen (Variieren und Überwachen von Spannungspegeln, Pulsbreiten, Flankenpositionen, Verzögerungen sowie ein großes Volumen von einfachem Speichern und dann Wiedergewinnen von ausgewählten Informationsmustern), und jeder Testtyp umfasst viele Millionen von einzelnen Speicherzyklen für das DUT. Auf der höchsten Ebene weisen die Bedienungspersonen des Testsystems die Testsystemsteuerung **2** an, die Teststation #1 zu verwenden, um das Testen von WHIZCO 0013s zu beginnen. Zur gegebenen Zeit weist die Testsystemsteuerung **2** die Teststellensteuerung #1 (4a) (bei der es sich um ein eingebettetes [Computer-]System handelt) an, das zugeordnete Testprogramm, z. B. TEST_WHIZ_13, auszuführen. Falls dieses Programm bereits in der Umgebung der Teststellensteuerung #1 verfügbar ist, dann wird dasselbe einfach ausgeführt. Ist dies nicht der Fall, dann wird dasselbe durch die Testsystemsteuerung **2** geliefert.

[0038] Nun könnte das Programm TEST_WHIZ_13 im Prinzip vollkommen in sich abgeschlossen sein. Wäre dies jedoch der Fall, dann wäre dasselbe fast sicher ziemlich groß, und es kann für den Prozessor des eingebetteten Systems in der Teststellensteuerung **4a** schwierig sein, schnell genug zu laufen, um die Tests mit der gewünschten Geschwindigkeit oder zumindest mit einer Rate, die von einem DUT-Speicherzyklus zum nächsten einheitlich ist, zu erzeugen. Dementsprechend werden Niedrigebenensubroutinentypaktivitäten, die Sequenzen von Adress- und zugeordneten Daten erzeugen, die geschrieben werden sollen oder von einer Leseoperation erwartet werden, nach Bedarf durch einen programmierbaren algorithmischen Mechanismus erzeugt, der in dem DUT-Tester **6** angeordnet ist, der aber synchron mit dem Programm wirksam ist, das durch das eingebettete System in der Teststellensteuerung **4** ausgeführt wird. Dies kann man sich vorstellen als ein Exportieren einer bestimmten niedrigebenensubroutineartigen Aktivität und der Aufgabe, DUT-Speicherzyklen einzuleiten, zu einem Mechanismus (dem DUT-Tester), der sich näher an der Hardwareumgebung des DUT **14** befindet. Allgemein liefert dann, immer wenn die Testsystemsteuerung **2** eine Teststellensteuerung mit einem Testprogramm ausstattet, dieselbe dem zugeordneten DUT-Tester auch geeignete Niedrigebenenimplementierungsroutinen (die vielleicht für den Speicher, der getestet wird, spezifisch sind), die benötigt werden, um die Gesamtaktivität zu erzielen, die durch die Programmierung für die Teststellensteuerung beschrieben oder benötigt wird. Die Niedrigebenenimplementierungsroutinen werden als „Muster" bezeichnet, und sie sind im Allgemeinen mit einem Namen versehen (genauso wie Funktionen und Variablen in Programmiersprachen hoher Ebene Namen aufweisen).

[0039] Jede Teststellensteuerung #n (**4**) ist mit ihrem zugeordneten DUT-Tester #n (**6**) durch einen Stellentestbus #n (**5**) gekoppelt. Die Teststellensteuerung verwendet den Stellentestbus **5**, um sowohl die Operation des DUT-Testers zu steuern als auch Informationen über Testresultate von demselben zu empfangen. Der DUT-Tester ist in der Lage, die verschiedenen DUT-Speicherzyklen, die in der Testvorgabe enthalten sind, mit hoher Geschwindigkeit zu erzeugen, und derselbe entscheidet, ob die Ergebnisse eines Lesespeicherzyklus wie erwartet sind. Grundsätzlich antwortet derselbe auf Befehle oder Operationscodes („mit Namen versehene Muster"), die von der Teststellensteuerung gesendet werden, durch ein Einleiten von entsprechenden nützlichen Sequenzen von Lese- und Schreib-DUT-Speicherzyklen (d. h. derselbe führt die entsprechenden Muster aus). Konzeptmäßig handelt es sich bei der Ausgabe des DUT-Testers **6** um Stimulusinformationen, die an das DUT anzulegen sind, und derselbe nimmt auch Antwortinformationen von demselben an. Diese Stimulus-/Antwortinformationen **7a** werden zwischen dem DUT-Tester **6a** und einer Anschlussstiftelektronik-#1-Anordnung **9a** geleitet. Die Anschlussstiftelektronikanordnung **9a** unterstützt bis zu 64 Sonden, die an das DUT **14** angelegt werden können.

[0040] Die im Vorhergehenden erwähnten Stimulusinformationen sind nur eine Sequenz von parallelen Bitmustern (d. h. eine Sequenz von „Sendevektoren" und erwarteten „Empfangsvektoren"), die gemäß den Spannungspegeln einer Familie von Logikvorrichtungen, die in dem DUT-Tester verwendet werden, ausgedrückt sind. Es besteht eine konfigurierbare Abbildung zwischen Bitpositionen in einem Stimulus/einer Antwort und den Sonden auf dem Chip, und diese Abbildung wird durch den DUT-Tester **6** verstanden. Die einzelnen Bits sind bezüglich ihrer Zeitgebung und Flankenplatzierung korrekt, aber zusätzlich zu der Abbildung können sie auch ein Spannungspegelverschieben benötigen, bevor dieselben an das DUT angelegt werden können. Gleichermaßen kann es sein, dass eine Antwort, die in dem DUT nach einem Stimulus entsteht, ein Puffern und (Rückwärts-)Pegelverschieben benötigt, bevor dieselbe als geeignet zum Zurückspeisen an den DUT-Tester betrachtet werden kann. Diese Pegelverschiebeaufgaben gehören zum Bereich der Anschlussstiftelektronik **9a**. Die Anschlussstiftelektronikkonfiguration, die zum Testen eines WHIZCO 0013 benötigt wird, funktioniert wahrscheinlich nicht für das Testen eines Teils von dem Unternehmen ACME Co., und vielleicht nicht einmal mit einem anderen Teil von WHIZ Co. Somit ist es ersichtlich, dass die Anschlussstiftelektronikanordnung ebenfalls konfigurierbar sein muss; eine derartige Konfigurierbarkeit ist die Funktion der PE-Konfig-Leitungen **8a**.

[0041] Im Vorhergehenden wurde ein kurzer Architekturüberblick abgeschlossen, wie eine einzelne Teststelle zum Testen eines DUT strukturiert ist. Nun erfolgt eine Zuwendung zu Aspekten, die sich ergeben, wenn viele Teststellen vorliegen, mit denen zu arbeiten ist. Als Einleitung wird ein bevorzugtes Ausführungsbeispiel zum Herstellen eines Testsystems mit mehreren Teststellen beschrieben. In vieler Hinsicht unterliegen einige der Informationen, die beschrieben werden, einer Wahlmöglichkeit basierend auf Marktuntersuchungen einer Kundenvorliebe und Kosten-Nutzen-Analysen. Wie dem auch sei, um ein solches herzustellen, müssen bestimmte Wahlen getroffen werden, und wenn dies einmal erfolgt ist, ergeben sich bestimmte Konsequenzen, die in dem gesamten System sichtbar sind. Es scheint nützlich zu sein, zumindest allgemein die gröberen Umrisse der Hardwareeigenschaften des Testsystems zu beschreiben. Auch wenn einige dieser Eigenschaften bedingt sind, hilft eine Kenntnis derselben trotzdem bei einem Verständnis von verschiedenen Beispielen, die verwendet werden, um die Erfindung zu veranschaulichen.

[0042] Zu Beginn seien vier ziemlich große Kartengestelle betrachtet. Jedes Kartengestell weist neben Leistungsversorgungen und Wasserkühlung (Lüfter können in einer Reinraumumgebung eine Verunreinigungsquelle sein) eine Hauptplatine, eine Frontplatine und eine Rückwandplatine auf. In jedes Kartengestell können bis zu neun Anordnungen platziert werden. Jede Anordnung umfasst eine Teststellensteuerung, einen DUT-Tester und eine Anschlussstiftelektronik. Es werden die allgemeinen Grundzüge beschrieben, wie Teststellensteuerungen miteinander verbunden werden, was einige Busse umfasst, die verwendet werden, um Verkettungen zu erzeugen.

[0043] Ein kurzer Exkurs bezüglich des Begriffs „Verkettung" ist vielleicht angebracht. Es seien Systemelemente A, B, C und D betrachtet. Es sei angenommen, dass dieselben in dieser Reihenfolge miteinander verkettet werden sollen. Man könnte sagen, dass ein Informations- oder Steuerungsweg besteht, der A verlässt und in B hineingeht, dass B selektiv Verkehr weiterleiten kann, der dann B verlässt und in C hineingeht, und dass C selektiv Verkehr weiterleiten kann, der dann in D hineingeht. Diese gleiche Art von Anordnungen kann auch für Verkehr in der anderen Richtung existieren. Verkettungen werden oft verwendet, um Prioritätsschemata zu erzeugen; hier werden sie verwendet, um Master/Slave-Beziehungen zwischen verschiedenen der Teststellensteuerungen zu erzeugen. Diese Verkettungsstilkommunikationsanordnungen werden mit dem Suffix „DSY" statt „BUS" bezeichnet. Somit kann auf eine Befehl/Daten-DSY anstelle eines Befehl/Daten-Busses Bezug genommen werden. Nun kann das Konzept, dass Informationen „in B hineingehen und selektiv weitergeleitet werden" andeuten, dass Verkehr auf einen separaten Satz von Leitern reproduziert wird, bevor derselbe weitergeleitet wird. So könnte es sein, aber aus Leistungsgründen ist es eher wie ein regulärer Bus, der adressierbare Entitäten aufweist. Mittels einer programmierbaren Adressabbildungsanordnung und der Fähigkeit, Abschnitte von nachgeschalteten Teststellensteuerungen „in Schlaf zu versetzen", kann bewirkt werden, dass der einzelne Bus logisch als eine Mehrzahl von Verkettungen erscheint (d. h. wirksam ist). Schließlich sei darauf hingewiesen, dass die Verkettungen Hochleistungsübertragungswege für Befehls- und Steuerinformationen sind, und dass, falls dies nicht der Fall wäre, nicht erwartet werden könnte, dass eine Master/Slave-Kombination (Mehrstellenteststation) so schnell wie eine einzelne Teststelle wirksam ist. Zugunsten der Verkettungsleistung verlassen die verschiedenen DSY nicht ihre jeweiligen Kartengestelle. Die Wirkung dieser Entscheidung besteht darin, Grenzen zu setzen, welche (und somit auch wie viele) Teststellen miteinander verbunden werden können. Im Prinzip besteht keine grundsätzliche Notwendigkeit für diese Einschränkung, und es besteht auch kein wirklicher Mangel an technischer Durchführbarkeit (es wäre möglich); es scheint einfach, dass, da sich bereits neun Teststellen in einem Kartengestell befinden, ein Erweitern der DSYs erhebliche zusätzliche Kosten für einen relativ geringen zusätzlichen Vorteil verursacht.

[0044] Um die Erörterung von Fig. 1 wiederaufzunehmen, seien die verschiedenen Teststellensteuerungen **4a**–**4z** betrachtet, die die vier Kartengestelle bestücken können, jedes mit neun Teststellensteuerungen. Diese seien als **4a**–**4f**, **4g**–**4m**, **4n**–**4t** und **4u**–**4z** bezeichnet. (Wie bereits erläutert, wird darüber hinweggesehen, dass es sich dabei nominell nur um 26 Indices handelt – der Leser ist eingeladen, sich vorzustellen, dass irgendwo noch weitere zehn Indexsymbole vorliegen.) Eine CMD/DAT-DSY **17a** (Befehl- & Daten-Verkettung) verbindet die Teststellensteuerungen **4a**–**4f**, die sich in einem Kartengestell befinden, während eine andere CMD/DAT-DSY **17b** die Teststellensteuerungen **4g**–**4m** in einem anderen Kartengestell verbindet. Die gleiche Anordnung existiert für die verbleibenden Kartengestelle und die Teststellensteuerungen **4n**–**4t** bzw. **4u**–**4z**. Es wurde bereits erwähnt, dass die DSY die Kartengestelle nicht verlassen, dahingehend dass ein „Ende" eines Busses, das tatsächlich die DSY bildet, ein Kartengestell nicht verlässt und der Kopf des nächsten Segments in einem anderen Kartengestell wird. Stattdessen geht der Systembus **3** von der Testsystemsteuerung **2** zu allen Teststellensteuerungen, und jede ist in der Lage, ein Master an dem Kopf eines DSY-Segments zu werden, das das Kartengestell nicht verlässt.

[0045] Die CMD/DAT-DSY **17a**–d, die erörtert worden sind, existieren zwischen den verschiedenen Teststellensteuerungen **4a**–**4z**. Es besteht eine ähnliche Anordnung für die SYNC/ERR-DSY **18a**–**18d** und die DUT-Tester **6a**–**6z**. Die Synchronisations- und Fehlerinformationen, die durch die SYNC/ERR-DSY **18** übermittelt werden, ermöglichen es den DUT-Testern, in Einklang wirksam zu sein. Diese beiden Verkettungen (**17** und **18**) tragen leicht unterschiedliche Typen von Informationen, jede besteht jedoch als ein Teil des gleichen allgemeinen Mechanismus zum Verbinden von ein oder mehr Teststellen miteinander zu einer Teststation.

[0046] Es folgt nun eine Erörterung von Fig. 2, bei der es sich um eine vereinfachte Blockdiagrammvergrößerung des DUT-Testers **6** von Fig. 1 handelt, von denen es 36 geben kann. Es ist momentan ausreichend, nur eine Instanz derselben zu beschreiben. Ein Blick auf Fig. 2 zeigt, dass dieselbe ziemlich gut mit Elementen bestückt ist; insbesondere für ein „vereinfachtes" Blockdiagramm. Einiges von dem, was sich in dem DUT-Tester **6** befindet und in dem Blockdiagramm dargestellt ist, ist funktionell ziemlich kompliziert und ist nicht in einer „Standard"-Form verfügbar. Es ist hier angemessen, auf zwei Punkte hinzuweisen. Erstens besteht der

Hauptzweck, Fig. 2 aufzunehmen, darin, die Grundeigenschaften einer wichtigen Betriebsumgebung innerhalb des Gesamttestsystems **1** für nichtflüchtige Speicher zu beschreiben. Die Erfindung(en), die im Detail in Verbindung mit Fig. 3 und nachfolgenden Figuren beschrieben sind, sind entweder Erweiterungen von Mechanismen, die in der folgenden Beschreibung von Fig. 2 dargelegt sind, oder es handelt sich um neue Mechanismen, deren Motivationsgrundlage in Fig. 2 zu finden ist. In jedem Fall ist, während dies geschrieben wird, nicht genau bekannt, welches davon dem Leser vorliegt. Das Ziel besteht momentan darin, einen vereinfachten, aber informativen Ausgangspunkt für zahlreiche unterschiedliche detaillierte Beschreibungen von verschiedenen bevorzugten Ausführungsbeispielen zu liefern, so dass jede derselben so knapp wie angemessen sein kann (im Gegensatz zu einer „Riesen"-Beschreibung, die alles über jede unterschiedliche Erfindung offenbart). Der zweite Punkt besteht darin, dass das erweiterte oder ausgebaute Material, obwohl dasselbe sich allgemein insgesamt in Übereinstimmung mit Fig. 2 befindet, Informationen enthalten kann, die nicht genau mit der vereinfachten Version „übereinstimmen". Das bedeutet nicht, dass ein Fehler vorliegt oder dass die Sachverhalte absolut unvereinbar sind; es tritt auf, weil es manchmal schwierig oder unmöglich ist, etwa derart zu vereinfachen, dass es das genaue Abbild in Miniaturausgabe darstellt. Die Situation ähnelt derjenigen bei Karten. Eine Straßenkarte von Colorado in Standardgröße zeigt, dass jemand, der auf der I-70 nach Osten fährt, bei Denver auf der I-25 nach Norden fahren kann. Das sieht wie ein Linksabbiegevorgang aus. Und obwohl es einmal tatsächlich ein Linksabbiegevorgang war, ist das nicht mehr so, und eine detaillierte Karte dieser Kreuzung zeigt eine Folge von zusammengehörigen Abzweigungen und dazwischenliegenden Straßenabschnitten. Niemand würde jedoch behaupten, dass die Straßenkarte der Standardgröße falsch ist; sie ist für ihre Abstraktionsebene korrekt. Auf ähnliche Weise und trotz ihres ziemlich beladenen Erscheinungsbildes ist Fig. 2 tatsächlich eine Vereinfachung, die auf einer mittleren Abstraktionsebene wirksam ist, aber einige scheinbare Linksabzweigungen sind nicht wirklich einfache Linksabzweigungen.

[0047] Wie es in Fig. 1 gezeigt ist, ist der Haupteingang zu dem DUT-Tester **6** eine Instanz des Teststellenbusses **5**, die von einer Teststellensteuerung **4** ausgeht, die der Instanz des interessierenden DUT-Testers **6** zugeordnet ist. Der Teststellenbus **5** ist mit einem Mikrosteuerungssequenzer **19** gekoppelt, der mit einem Spezialmikroprozessor verglichen werden kann. Derselbe ruft Anweisungen von einem Programm ab, das in einem Programmspeicher gespeichert ist, der entweder bezüglich des Mikrosteuerungssequenzers **6** intern (PGM SRAM **20**) oder extern (EXT.DRAM **21**) ist. Obwohl es scheint, dass diese beiden Speicher von etwas adressiert werden, bei dem es sich im Wesentlichen um eine logisch gemeinsame Adresse **63** handelt, die als ein Programmzähler (oder eine Anweisungsabrufadresse) dient, und obwohl beide eine Quelle einer auszuführenden Programmierung sein können, sei darauf hingewiesen, dass: (1) nur einer der Speicher Anweisungsabrufspeicherzyklen während einer beliebigen Zeitperiode durchführt; und (2) dieselben tatsächlich durch elektrisch unterschiedliche Signale adressiert werden. Der SRAM ist schnell und ermöglicht einen echten Direktzugriff, verbraucht aber wertvollen Platz innerhalb der Mikrosequenzsteuerung **19** (bei der es sich um eine große integrierte Schaltung (IC) handelt), so dass seine Größe begrenzt ist. Der externe DRAM kann in einstellbaren Mengen beträchtlicher Größe geliefert werden, ist jedoch nur schnell, wenn auf denselben in sequentiellen Stücken zugegriffen wird, die eine lineare Ausführung und keine Verzweigung umfassen. Eine Programmierung ist in dem SRAM **20** meistens diejenige, die stark algorithmisch ist, während der EXT.DRAM **21** am besten für Material geeignet ist, das nicht einfach durch algorithmische Prozesse erzeugt wird, wie z. B. Initialisierungsroutinen und zufällige oder unregelmäßige Daten.

[0048] Das Anweisungswort, das durch den Mikrosteuerungssequenzer **19** ausgeführt wird, ist ziemlich breit: 208 Bit. Es besteht aus 13 16-Bit-Feldern. Diese Felder stellen oft abgerufene Anweisungsinformationen für Mechanismen dar, die sich außerhalb des eigentlichen Mikrosteuerungssequenzers befinden. Derartige Felder sind für ihre zugeordneten Mechanismen reserviert. Ein Satz von ALU-ANWEISUNGEN **22** wird an eine Sammlung von acht 16-Bit-ALUs (arithmetic logic unit – Arithmetik-Logik-Einheit) **24** angelegt, während andere an verschiedene andere Mechanismen ausgegeben werden, die in dem DUT-Tester verteilt sind. Diese letztere Situation ist durch die Linien und die Legende **42** „VERSCHIEDENE STEUERWERTE & ANWEISUNGEN" dargestellt.

[0049] Die acht 16-Bit-ALUs (**24**) haben jede ein herkömmliches Repertoire von arithmetischen Anweisungen, die um zugeordnete 16-Bit-Ergebnisregister gebildet sind (jede ALU weist auch mehrere andere Register auf). Drei dieser Ergebnisregister und ihre zugeordneten ALUs dienen zum Erzeugen von X-, Y- und Z-Adresskomponenten **27**, die verschiedenartig zu einer vollständige Adresse kombiniert werden, die dem DUT geliefert werden soll. Zwei weitere der acht ALU/Register (DH & DL) sind bereitgestellt, um bei der algorithmischen Erzeugung von 32-Bit-Datenmustern **28** zu helfen, die zwischen einem höchstwertigen Abschnitt (DH) und einem niederstwertigen Abschnitt (DL) aufgeteilt sind. Die letzten drei ALU/Register (A, B, C) werden als Zähler verwendet und tragen zu der Erzeugung von verschiedenen PROGRAMMSTEUERFLAGS **25** bei, die bei einer Programmsteuerung und -verzweigung beim Abschluss einer programmatisch spezifizierten Anzahl von Itera-

tionen oder einer anderen numerischen Bedingung unterstützen. Diese PROGRAMMSTEUERFLAGS **25** werden zu dem Mikrosteuerungssequenzer **19** zurückgesendet, wo dieselben den Wert der Anweisungsabrufadresse auf eine Weise beeinflussen, die denjenigen bekannt ist, die ein Verständnis von Mikroprozessoren haben. Es gibt auch verschiedene ANDERE FLAGS **55**, die ebenfalls verwendet werden können, um eine Programmverzweigung zu bewirken. Diese gehen von verschiedenen der anderen Mechanismen in dem DUT-Tester **6** aus, die durch die unterschiedlichen Felder des abgerufenen Anweisungsworts gesteuert werden. Ein spezifisches zusätzliches Flag ist ausdrücklich als ein separates Element gezeigt: VEC_FIFO_FULL **26**. Bei einer anderen Zeichnung, die etwas weniger detailliert ist, könnte dasselbe mit den ANDEREN FLAGS **55** zusammengefasst werden. Es wurde abgesondert, um bei der Erläuterung eines Aspekts der Operation des Mikrosteuerungssequenzers **19** behilflich zu sein.

[0050] VEC_FIFO_FULL hält (vorübergehend) eine weitere Programmausführung durch den Mikrosteuerungssequenzer **19** an. Es gibt viele Pipelinestufen zwischen den Anweisungen, die durch den Mikrosteuerungssequenzer **19** abgerufen werden, und dem Mechanismus, der schließlich Testvektoren abgibt, damit dieselben an das DUT angelegt werden. Zusätzlich besteht ein Teil des Gepäcks, das einen Vektor begleitet, während sich derselbe darauf hinbewegt, an das DUT angelegt zu werden, in Informationen, die die Rate einer schließlichen Vektoranlegung oder die Dauer jedes Vektors betreffen. Somit muss die Rate einer Vektoranlegung an das DUT nicht konstant sein, und insbesondere kann es länger dauern, eine Gruppe von Vektoren anzulegen, als es dauerte, sie zu erzeugen. Der Mikrosteuerungssequenzer führt einfach eine Programmierung mit seiner höchsten Rate aus. Aber natürlich muss die Rate eines „Vektorverbrauchs" durchschnittlich gleich der Rate einer „Vektorerzeugung" sein, ansonsten müsste die Pipeline nahezu ohne Einschränkung elastisch sein. Es befindet sich ein Vektor FIFO **45** an dem Ausgang der Adressabbildungsvorrichtung **29**, die im Folgenden erörtert ist, und derselbe dient als eine elastische Kapazität in der Pipeline. Das Signal VEC_FIFO_FULL wird verwendet, um zu verhindern, dass die begrenzte Anzahl von Stufen in der Pipeline überschritten wird, indem eine vorübergehende Einstellung der Erzeugung neuer Vektoren an dem Kopfende der Leitung bewirkt wird.

[0051] Des weiteren werden die (3 × 16 = 48 Bits von den) X-, Y- und Z-Adresskomponenten **27** an eine Adressabbildungsvorrichtung **29** angelegt, bei deren Ausgabe es sich um eine im Voraus ausgewählte, beinahe willkürliche Neuanordnung der Adresswerte in dem geordneten 48-Bit-Adressraum handelt. Als ein Ausgangspunkt zum Verständnis desselben sei für einen Moment angenommen, dass die Adressabbildungsvorrichtung **29** ein Speicher ist, der einen 48-Bit-Adressraum voll bestückt, und dass derselbe einen 48-Bit-Wert an jeder Adresse hält. (Es sei vorübergehend außer Acht gelassen, dass ein derartiger Speicher – zumindest heute – die Größe eines großen Kühlschranks hätte.) Mit einem solchen Speicher könnte eine Nachschlagetabelle implementiert werden, die eine beliebige angelegte Adresse in einen anderen, willkürlich ausgewählten 48-Bit-Wert abbilden könnte, der dann als eine Ersatzadresse verwendet werden könnte. Der Grund dafür, dass eine derartige Adressabbildung erwünscht ist, besteht darin, dass die X-, Y- und Z-Adresskomponenten im Allgemeinen in dem Kontext einer Innenarchitektur eines bestimmten DUT eine nützliche Bedeutung haben, die mit höchster Wahrscheinlichkeit nicht mit einem großen linearen Decodierer implementiert wird. Die Konzepte von Zeilen, Spalten und Schichten, Block oder Seiten können für den Testtechniker sehr nützlich sein, und Fehler, die an Orten auftreten, die physisch nahe beieinander liegen, können eine entsprechende Nähe bei ihren X-, Y- und Z-Adressen umfassen. Derartige Muster in den Testergebnissen können dabei wertvoll sein, zu erkennen, wo der Fehler liegt, und dabei zu versuchen, denselben zu beheben, ob auf einer Entwurfsebene oder auf einer Herstellungsebene der Neuprogrammierung eines Teils, um die Operation eines fehlerhaften Abschnitts mit der eines Reserveabschnitts zu überbrücken. Zwei Punkte ergeben sich aus diesem Gedankengang. Erstens ein Stutzen der 48 Bits herunter auf die tatsächliche Anzahl von Bits (z. B. 32 oder vielleicht 16), die an das DUT angelegt werden sollen. Es wird in Kürze kurz erwähnt, wie das Herunterstutzen erfolgt, und es handelt sich dabei hauptsächlich darum, eine bestimmte Anzahl von Bits von X, eine bestimmte Anzahl von Y und den Rest von Z zu nehmen. Das stimmt jedoch nicht ganz, und das ist der zweite Punkt, da bestimmte Adressen in einer Schaltungsanordnung liegen können, die ein Links-Rechts- (oder Links-Rechts- und Oben-Unten-)Spiegelbild eines anderen Abschnitts der Schaltungsanordnung ist. Dies hat den Effekt eines Neuanordnens, was die Bits bedeuten, bezüglich dessen, welche sequentiellen Adresswerte in dieser Schaltungsanordnung in physischer Reihenfolge sind. Diese Chipentwurfeigenschaft kann viele Male auftreten, und es kann durchaus der Fall sein, dass die Art, wie eine Gruppe von Bits, z. B. für Y, interpretiert wird, von dem zugehörigen Wert von anderen, z. B. Z, Bits abhängt. Die Adressabbildungsvorrichtung **29** ist bereitgestellt, um zu ermöglichen, dass die Roh-X-, Y- und Z-Adressen sozusagen „neu gepackt" werden, um dies zum Nutzen derer, die Speicher testen, die derartige Innenarchitekturanordnungen aufweisen, widerzuspiegeln. Bezüglich der tatsächlichen Vorgehensweise ist die Adressabbildungsvorrichtung **29** aus einer ziemlich großen Anzahl von miteinander verbundenen Multiplexern aufgebaut. Dieselbe kann nicht das vollkommen willkürliche Nachschlagetabellenverhalten eines voll bestückten Speicherdecodierschemas implementieren,

wie es im Vorhergehenden vorübergehend zu Erläuterungszwecken angenommen wurde. Dieselbe kann jedoch Teilfelder der X-, Y- und Z-Adresskomponenten nach Bedarf neu anordnen, insbesondere da es noch einen anderen Mechanismus gibt, der das Herunterstutzen von 48 Bits auf die tatsächlich benötigte Anzahl übernimmt. Die Adressabbildungsvorrichtung **29** enthält auch drei 16-Bit-(Adress-)Nachschlagetabellen, die es ihr ermöglichen, eine begrenzte willkürliche Abbildung innerhalb lokaler Bereiche durchzuführen.

[0052] Die abgebildete Adressausgabe **30** der Adressabbildungsvorrichtung **29** wird als eine Adresse an einen Hilfs-RAM **31** und an einen Fehlererfassungs-RAM **32** angelegt, die, obwohl dieselben gesonderte Funktionen aufweisen, trotzdem als auswählbare Partitionen in einem größeren Gesamt-RAM implementiert sein können. Die abgebildete Adressausgabe **30** wird auch als eine Eingabe an eine Adressbitauswählschaltung **37** angelegt, die im Folgenden beschrieben wird.

[0053] Es sei der Hilfs-RAM **31** betrachtet. Seine Funktion besteht darin, Datenmuster **33** und Adressen **34**, die an das DUT angelegt werden können, zu halten. Dabei handelt es sich um logisch separate Ausgaben von dem Hilfs-RAM **31**, da dieselben etwas unterschiedlich behandelt und an unterschiedlichen Stellen verwendet werden. (Der Hilfs-RAM **31** ist kein Zwei-„Tor-Speicher", sondern weist bevorzugt mehrere Bänke auf, deren Ausgaben an MUXs angelegt werden.) Dementsprechend kann es sein, dass gespeicherte Daten **33** in einer Bank oder einem Adressbereich des Hilfs-RAM **31** gehalten werden, während gespeicherte Adressen **34** in einem anderen gehalten werden. Es wurde auch kein expliziter Mechanismus zum Schreiben in den Hilfs-RAM **31** gezeigt. Dies wird durch eine adressierte Busoperation erreicht, die durch eine Teststellensteuerung **4** auf Veranlassung des Programms, das dieselbe ausführt, eingeleitet wird. (Es gibt sozusagen einen „unterirdischen" „Hilfsdienst"-Bus, der der „Ringbus" genannt wird [nicht gezeigt – da derselbe die Zeichnung stark überfüllen würde], der zu fast allen Elementen in Fig. 2 geht.) Der Fehlererfassungs-RAM **32** wird durch die gleiche Adresse adressiert, die an den Hilfs-RAM **31** angelegt wird, und derselbe speichert entweder Informationen über Fehler oder gewinnt dieselben wieder, wobei diese Operationen zusammen mit einer Nachdecodierschaltung durchgeführt werden, die später erörtert wird. Wie bei den Wegen **33** und **34** von dem Hilfs-RAM **31** sind die Wege **61** (in den Fehlererfassungs-RRM) und **62** (aus dem Fehlererfassungs-RAM) bevorzugt gemultiplexte Ausgaben von einem Mehrbankspeicher (dem Fehlererfassungs-RAM **32**) gemäß Konfigurationsinformationen, die durch den Ringbus (nicht gezeigt) verteilt werden.

[0054] Es sei darauf hingewiesen, dass der Daten-MUX **35** als Eingaben die GESPEICHERTE-DATEN-Ausgabe **33** von dem Hilfs-RAM **31** sowie Daten **28** von den Registern DH und DL in der Sammlung **24** von ALUs aufweist. Der Daten-MUX **35** wählt aus, welche dieser Eingaben (**28**, **32**) als seine Ausgabe **38** präsentiert werden, die dann als eine von zwei Vektorkomponenten an eine Sendevektorabbildungsvorrichtung-/Seriellumsetzer-/Empfangsvektorvergleichsdatenschaltung **40** angelegt wird (die andere Komponente ist die Ausgabe **39** der Adressbitauswählschaltung **37**). Der Daten-MUX **35** führt diese Auswahl gemäß Werten **36** durch, die in dem PGM SRAM **20** gespeichert sind.

[0055] Die Schaltung **40** kann drei Funktionen erfüllen: Vektorkomponenten (**38**, **39**) zu einer geordneten logischen Darstellung eines ganzen Vektors zusammensetzen, der an das DUT angelegt (gesendet) werden soll; eine willkürliche dynamische Entsprechung (Abbildung) zwischen den geordneten Bits der logischen Darstellung des Sendevektors und der tatsächlichen physischen Kanalanzahl der Anschlussstiftelektronik anlegen (d. h. welche Sondenspitze kontaktiert das DUT für dieses Signal (d. h. dieses Bit in dem Vektor)); und mit dem Kompilierer zusammenwirken bei der Teilung eines ganzen logischen Vektors in Stücke, die getrennt und in Reihenfolge (Serialisierung) angelegt werden sollen, für DUTs, die etwas Derartiges zulassen. Welche dieser Funktionen durchgeführt wird, wird durch Steuersignale von einem SRAM **41** bestimmt, der auch gemäß einem Feld in der 208-Bit-Anweisung adressiert wird, die durch den Mikrosteuerungssequenzer **19** abgerufen wird. Die Ausgabe der Schaltung **40** ist ein bis zu 64-Bit-Vektor **44**, der an einen Vektor FIFO **45** angelegt wird, der, wenn derselbe voll ist, das Signal VEC_FIFO_FULL **26** erzeugt, dessen Bedeutung und Verwendung im Vorhergehenden erörtert wurden. Der Vektor am oberen Ende des Vektors FIFO **45** wird auf den Empfang eines Signals VEC_FIFO_UNLOAD (VEC_FIFO_ENTLADEN) **47**, das bei einem Periodengenerator **49** entsteht (der im Folgenden erörtert wird), von demselben entfernt. Derartige entfernte Vektoren (**46**) werden an eine Zeitgebung-/Formatierungs- & Vergleichsschaltung **52** angelegt, die mit dem DUT über die zugeordnete Instanz der Anschlussstiftelektronik **9** verbunden ist. Das heißt, jede Instanz der Anschlussstiftelektronik **9** empfängt gesendete & empfangene Vektoren **7** und Anschlussstiftelektronikkonfigurationsinformationen **8** von ihrer zugeordneten Zeitgebungs-/Formatierungs- & Vergleichsschaltung **52**.

[0056] Die Zeitgebungs-/Formatierungs- & Vergleichsschaltung **52** weist einen internen SRAM **54** auf, der durch die gleiche Anweisungsadresse („A" in dem kleinen Kreis) adressiert wird wie der Programm-SRAM **20** des Mikrosteuerungssequenzers **19**. (Ein externer DRAM **53** kann anstelle des internen SRAM **54** verwendet

werden.) Der interne SRAM **54** (oder der externe DRAM **53**) hilft bei der Erzeugung von Treiber- und Vergleichszyklen. Treiberzyklen legen einen Sendevektor an das DUT an. Vergleichszyklen empfangen einen Vektor, der durch das DUT präsentiert wird, und untersuchen denselben, um zu bestimmen, ob derselbe mit im Vorhergehenden gelieferten Vergleichsdaten übereinstimmt. Sowohl die Treiber- als auch die Vergleichszyklen sind bezüglich ihrer Dauer, ob und wann eine Last angelegt wird, und wann Daten zwischengespeichert oder übernommen werden, einstellbar. Der Vergleich erzeugt einen 64-Bit-Wert **56**, der an eine Empfangsvektorrückwärtsabbildungsvorrichtung/Seriell-Parallel-Umsetzer **57** angelegt wird, dessen Funktion als die logische Inverse der Schaltung **40** betrachtet werden kann. (Die Operation der Schaltung **57** wird durch einen SRAM **58** gesteuert, der der Steuerung der Schaltung **40** durch den SRAM **41** entspricht.) Die Ausgabe **59** der Schaltung **57** wird wiederum an die Nachdecodierschaltung **60** angelegt. Momentan reicht es aus, zu sagen, dass die Nachdecodierschaltung **60** über programmatische Kriterien sowohl eingehende Fehlerinformationen **59** als auch (im Vorhergehenden) gespeicherte Fehlerinformationen **60** (die in dem Fehlererfassungs-RAM gespeichert sind) überprüfen kann, um komprimierte und leichter interpretierbare Fehlerinformationen zu erzeugen, die dann über den Weg **61** zurück in den Fehlererfassungs-RAM **32** gespeichert werden können. Ein Beispiel bestünde darin, einen Zählwert zu erzeugen, wie oft ein Fehler in einem bestimmten Adressbereich vorlag, wobei diese Information nützlich sein kann beim Entscheiden, wann versucht werden soll, eine chipinterne Reparatur durch ein Freigeben von Ersatzschaltungen zu unternehmen.

[0057] Es erfolgt nun eine Zuwendung zu dem Periodengenerator **49** und seinem zugeordneten Zeitgebungs-SRAM **51**. Diese sprechen auf ein 8-Bit-Signal T_SEL **43** an, das für jede 208-Bit-Anweisung, die durch den Mikrosteuerungssequenzer **19** abgerufen wird, eine Dauer für die zugeordnete Operation der Zeitgebungs-/Formatierungs- & Vergleichsschaltung **52** bestimmt. T_SEL **43** ist ein Element der verschiedenen Steuerwerte & Anweisungen **42**, die durch die unterschiedlichen Felder innerhalb der abgerufenen Anweisung dargestellt sind. Als ein 8-Bit-Wert kann dasselbe 256 unterschiedliche Elemente darstellen oder codieren. In diesem Fall handelt es sich bei diesen „Elementen" um 28-Bit-Werte, die in dem Zeitgebungs-SRAM **51** gespeichert sind und die durch T_SEL adressiert werden. Jeder adressierte 28-Bit-Wert (**23**) spezifiziert eine gewünschte Dauer mit einer Auflösung von 19,5 Pikosekunden. Die Sequenz von 28-Bit-Dauerwerten (**23**), auf die zugegriffen wird, ist in einem Perioden-FIFO **50** gespeichert, so dass die einzelnen Elemente dieser Sequenz synchron mit der Gewinnung ihres beabsichtigten entsprechenden Vektors, der in dem Vektor-FIFO **45** gespeichert ist, gewonnen und angelegt werden.

[0058] Ein Grobzeitgebungswertfeld in dem ältesten Eintrag in dem FIFO **50** übermittelt Dauerinformationen mit einer Auflösung von 5 ns und erzeugt daraus ein Signal VEC_FIFO_UNLOAD (VEC_FIFO_ENTLADEN) **47**, das den nächsten Sendevektor von dem Vektor-FIFO **45** zu der Zeitgebungs-/Formatierungs- & Vergleichsschaltung **52** überträgt. Ein Begleitsignal ZEITGEBUNGSREST **48** wird ebenfalls an die Schaltung **52** angelegt. Dort wird die schließliche Auflösung auf 19,5 Pikosekunden erreicht.

[0059] Es sei nun Bezug genommen auf Fig. 3, bei der es sich um ein vereinfachtes Blockdiagramm **64** des ECR **32** in dem Blockdiagramm von Fig. 2 handelt. Dasselbe empfängt eine abgebildete 48-Bit-Adresse **30** von der Adressabbildungsvorrichtung **29**, die an verschiedene Adressklassierer **77**, **78** und **79** angelegt wird. Die Adressklassierer sind Speichersätzen **73**–**76** zugeordnet, bei denen es sich jeweils um vollständige Speichermechanismen handelt, die einzeln zugeordnete ECR-Funktionen durchführen können. Zwei dieser Speichersätze (**73**, **74**) sind von einem externen DRAM, während zwei von einem internen SRAM sind. Bei den zwei externen DRAM-Speichersätzen ist immer die gleiche Adressklassiererfunktion in Kraft, und dieselben verwenden somit einen gemeinsamen Adressklassierer **77**. Die internen SRAM-Speichersätze **75** und **76** weisen jeder seinen eigenen zugeordneten Adressklassierer **78** bzw. **79** auf. Diese Adressklassierer können herkömmlich sein und können die Adresse gemäß Prinzipien und zu Zwecken, die in der Technik bekannt sind, verändern. Sie sind hier zur Vervollständigung gezeigt und um eine Kompatibilität zwischen dieser Anwendung und einer erwarteten, damit in Beziehung stehenden Anwendung zu fördern. Obwohl die Adressklassierer dazu dienen, eine nützliche Funktion durchzuführen, können sie hier ohne weiteres ignoriert werden, indem einfach angenommen wird, dass dieselben keine Veränderung bei der Adresse durchführen.

[0060] Jeder Speichersatz umfasst eine Speichersatzsteuerung; die externen DRAM-Speichersätze **73** und **74** weisen DRAM-Speichersatzsteuerungen **65** bzw. **66** auf, während die internen SRAM-Speichersätze **75** und **76** SRAM-Speichersatzsteuerungen **67** bzw. **68** aufweisen. Während des Testens eines DUT kommt die Adresse für Speichertransaktionen, die an einen beliebigen dieser Speichersätze gerichtet sind, von dem jeweils zugeordneten Adressklassierer bei der zugeordneten Speichersatzsteuerung an. Während des Testens eines DUT werden Fehlerdaten **61**, die von der Nachdecodierschaltung **60** ankommen und die in den ECR geschrieben werden sollen, zuerst an Datenklassierer **80**–**83** angelegt, von denen einer jedem Speichersatz zugeordnet ist. Die Funktion der Datenklassierer ist momentan nicht von Interesse und dieselben werden hier

hauptsächlich zur Vervollständigung gezeigt, und um eine Kompatibilität zwischen dieser Anwendung und einer erwarteten, damit in Beziehung stehenden Anwendung zu fördern. Wie die Adressklassierer können die Datenklassierer **80**–**83** hier ohne Weiteres ignoriert werden, indem angenommen wird, dass dieselben die Daten einfach ohne eine Modifizierung durchleiten. Die Adress- und Datenklassierer stellen Hochgeschwindigkeitswege für Adressen bzw. Daten dar, die mit den höchsten notwendigen Geschwindigkeiten wirksam sein sollen. Es wird in Kürze darauf eingegangen, dass der Ringbus (noch nicht gezeigt) eine andere Möglichkeit liefert, um Adressen und Daten an die Speichersätze zu übermitteln.

[0061] An diesem Punkt liegen vier Speichersatzsteuerungen (**65**–**68**) vor, die jede eingehende Adresse und Daten aufweisen. Jede dieser Speichersatzsteuerungen ist mit einem zugeordneten Speicher gekoppelt: Die DRAM-Speichersatzsteuerungen **73** und **74** sind mit externen DRAMs **69** bzw. **70** gekoppelt, während die SRAM-Speichersatzsteuerungen **75** und **76** mit internen SRAMs **71** bzw. **72** gekoppelt sind. Diese Anordnungen bilden die vier Speichersätze **73**–**76**, von denen zwei (**75**, **76**) mäßige Mengen von Hochgeschwindigkeits-SRAM aufweisen, und von denen zwei (**73**, **74**) große Mengen von langsamerem DRAM aufweisen. Das Hauptinteresse liegt momentan darin, wie die DRAM-Speichersätze so schnell wie die SRAM-Speichersätze gemacht werden können, und wie bestimmte Alternativen bezüglich der Konfiguration des DRAM abhängig von einer Benutzervorliebe und einer Testprogrammstrategie eingegliedert werden können. Somit wird sich herausstellen, dass die DRAM-Speichersatzsteuerungen **65** und **66** konfigurierbar sind, unterschiedliche Typen von Speichertransaktionen durchführen und nicht ganz gleich den einfacheren SRAM-Speichersatzsteuerungen **67** und **68** sind. Zur Verkürzung zeigt Fig. 3 nicht die Struktur, die diese Flexibilität liefert; es sei momentan nur erwähnt, dass jede Speichersatzsteuerung mit dem Ringbus (noch nicht gezeigt) verbunden ist, von dem dieselbe bezüglich des bestimmten Operationsmodus und der Konfiguration, die gewünscht sind, unterwiesen wird. Einige dieser Modi umfassen, wie Daten gespeichert werden, und einige hängen damit zusammen, wie dieselben wieder zurückerhalten werden. Das Hauptinteresse liegt bei den Modi und Konfigurationen der DRAM-Speichersätze. Abschließend sei darauf hingewiesen, dass jeder Speichersatz eine zugeordnete Datenausgabe (**62A**–D) aufweist, die zu dem Nachdecodiermechanismus **60** zur weiteren Verarbeitung gesendet wird.

[0062] Es sei nun Fig. 4 betrachtet, bei der es sich um ein detaillierteres Blockdiagramm **84** des ECR **32** handelt, der in Verbindung mit Fig. 3 beschrieben wurde. Derselbe ist ziemlich ähnlich, und ähnliche Elemente wurden mit gemeinsamen Bezugszeichen bezeichnet. Für die momentanen Zwecke ist es ausreichend, auf die zusätzlichen Unterschiede hinzuweisen, die in Fig. 4 vorhanden sind. Im Einzelnen sei darauf hingewiesen, dass jedem der Adressklassierer (**78**–**79**) ein zugeordneter MUX (**85**–**87**) vorgeschaltet ist. Diese MUXs helfen bei dem Prozess einer Adressweiterentwicklung und speziell beim Verringern der Größe der Adresse von 48 Bits auf 32. Daraus wird ersichtlich, dass die Situation bezüglich dieser MUXs derjenigen für die Adressklassierer und die Datenklassierer ähnlich ist: obwohl sie aus nützlichen Gründen vorhanden sind, werden dieselben momentan nicht berücksichtigt und diese MUXs werden hauptsächlich zur Vervollständigung gezeigt (und um die Adresse auf 32 Bits zu verringern!). Außerdem sei darauf hingewiesen, dass der Ringbus **85** mit jeder der Speichersatzsteuerungen (**65**–**68**) gekoppelt ist. Es sei darauf hingewiesen, dass die Datenausgaben (**62A**–D) von den Speichersatzsteuerungen, wenn sie zu der Nachdecodierschaltung **60** gelangen, an einen 4:1-MUX **94** angelegt werden, der gemäß einem Steuerregister **95**, das durch den Ringbus gesetzt ist, bestimmt, welche Ausgabe zur weiteren Verarbeitung ausgewählt wird.

[0063] Das Hauptinteresse in Fig. 4 liegt bei der Gesamtorganisation des Speichers, der durch die verschiedenen Speichersatzsteuerungen **65**–**68** gesteuert wird. In dem Fall von Speichersatz 2 (**75**) und Speichersatz 3 (**76**) ist dieser Speicher einfach ein SRAM, der als ein einziger (in seinem Speichersatz) Adressraum angeordnet ist, und der auf eine herkömmliche Weise wirksam ist. In dem Fall von Speichersatz 0 (**73**) und Speichersatz 1 (**74**) ist der Speicher jedoch für jeden jeweils drei Gruppen von vier Bänken, und das Format für eine Adresse hängt von den Modus- und Konfigurationsinformationen ab, die derzeit auf diese Speichersätze anwendbar sind.

[0064] So weist z. B. der Speichersatz 0 (**73**) drei Gruppen **88**, **89** und **90** auf, während der Speichersatz 1 (**74**) die Gruppen **91**, **92** und **93** aufweist. Bei einem Hochgeschwindigkeitsmodus zum wahlfreien Adressieren werden aufeinander folgende Speichertransaktionen automatisch zu unterschiedlichen Gruppen gesendet (Multiplexen), von denen jede ihren eigenen Hardwareweg für Adresse und Daten aufweist. Jede dieser Gruppen ist aus vier Bänken (vier Instanzen eines Adressraums) gebildet, für die die Speicheroperationen gemäß Prinzipien und Praktiken, die in der Technik bekannt sind, verschachtelt sein können. Im Einzelnen ist der Typ von DRAM, der momentan zur Verwendung bevorzugt wird, ein SDRAM, was eine bestimmte Strategie zum Verschachteln impliziert. Es sei darauf hingewiesen, dass andere Typen von DRAM existieren und dass andere Mechanismen zur Verschachtelung neben derjenigen, die im Folgenden beschrieben ist, möglich sind. Bei ei-

nem anderen Modus zum langsameren wahlfreien Adressieren sind die Gruppen adressierbar, anstatt automatisch ausgewählt zu werden. Bei diesem langsameren Modus werden zusätzliche Adressbits für die Gruppe verwendet, um den Hardwareweg auszuwählen. Bei diesem Modus spezifiziert eine Adresse eine Gruppe und eine Bankadresse innerhalb dieser Gruppe von verschachtelten Bänken. Bei einem weiteren Hochgeschwindigkeitsmodus mit einem gutes Verhalten aufweisenden Adressieren sind sowohl Multiplexen als auch Verschachteln abgeschaltet, und eine Adresse weist Gruppenauswählbits, Bankauswählbits und In-Bank-Adressbits auf. Bei einem Schmalwortoperationsmodus werden noch weitere zusätzliche Adressbits verwendet, um ein Feld in dem ganzen Wort zu spezifizieren, das das Ziel der Speichertransaktion ist.

[0065] Es erfolgt nun eine Zuwendung zu Fig. 5, bei der es sich um ein vereinfachtes Blockdiagramm **96** einer DRAM-Speichersatzsteuerung (**65**, **66**) handelt, die in den Fig. 3 und Fig. 4 erscheint. Dieselbe empfängt als Eingabe eine KLASSIFIZIERTE ADRESSE **106**, Modus- und Konfigurationsinformationen von dem Ringbus **85** und eine FEHLERDATENEINGABE **105** von dem zugeordneten Datenklassierer. Wie zuvor erzeugt dieselbe eine DATENAUSGABE (**62A**/B).

[0066] Es ist jedoch nun ersichtlich, dass andere Quellen von Adresse und Daten diese Größen über den Ringbus **85** liefern können. Das heißt, es gibt eine Busschnittstelle **97**, die den Ringbus mit der DRAM-Speichersatzsteuerung koppelt, und über diese Schnittstelle sind DATEN VON RINGBUS **99** und ADRESSE VON RINGBUS **100** verfügbar. Ein MUX **104** wählt aus, ob eine FEHLERDATENEINGABE **105** oder DATEN VON RINGBUS **99** als Daten **107** weitergeleitet werden, um daraufhin an den DATENEINGABEanschluss einer Master-DRAM-Steuerung **109** angelegt zu werden. Gleichermaßen wählt ein MUX **103** zwischen einer KLASSIFIZIERTEN ADRESSE **106** und einer ADRESSE VON RINGBUS **100** aus, um die Adresse **108** zu erzeugen, die an den ADRESSanschluss der Master-DRAM-Steuerung angelegt wird. Eine Sammlung von ein oder mehr Registern **98**, deren Inhalt durch einen Verkehr auf dem Ringbus gesetzt wird, erzeugt Steuersignale **101** und **102**, die die Wahlen anzeigen, die durch die MUXs **103** bzw. **104** vorzunehmen sind.

[0067] Es sei des weiteren darauf hingewiesen, dass die DATENAUSGABE (**62A**/B) zusätzlich an die Busschnittstelle **97** angelegt wird, wodurch es ermöglicht wird, dass die DATENAUSGABE über den Ringbus gesendet wird.

[0068] Eine Hauptfunktion der DRAM-Speichersatzsteuerung ist die Zuweisung oder Verteilung der verschiedenen Speichertransaktionen unter den drei Gruppen. Bei einem Hochgeschwindigkeitsoperationsmodus führt dieselbe diese Zuweisung auf eine Round-Robin-Art unter Verwendung (des Äquivalents) eines 1:3-MUX **125** durch. Der MUX **125** ist als eine gestrichelte Linie gezeigt, da es ersichtlich wird, dass, obwohl es tatsächlich einen MUX geben könnte, es bei dem vorliegenden bevorzugten Ausführungsbeispiel an diesem Ort keinen tatsächlichen MUX gibt. Stattdessen, und wie es in Verbindung mit Fig. 6 ersichtlich wird, gibt es mehrere Instanzen von adressierbaren Datenquellen unter der Steuerung eines fortgeschrittenen regelfolgenden Mechanismus (einer Zustandsmaschine).

[0069] Um die Erörterung von Fig. 5 abzuschließen, sei darauf hingewiesen, dass der 1:3-MUX **125** drei Slave-SDRAM-Steuerungen (**110**–**112**) treibt, wobei eine derartige Slave-SDRAM-Steuerung für jede von Gruppe 0, Gruppe 1 und Gruppe 2 vorhanden ist. Jede SDRAM-Slave-Steuerung weist als ihre Gruppe eine Sammlung von vier Bänken von SDRAM auf. Zum Beispiel ist die SDRAM-Steuerung **110** für die Gruppe 0 mit Bänken **113**, **114**, **115** und **116** gekoppelt. Auf ähnliche Weise weist die Gruppe 1 Bänke **117**–**120** auf, während die Gruppe 2 Bänke **121**–**124** aufweist. Das Ergebnis ist eine Gesamtzahl von zwölf Bänken für jeden DRAM-Speichersatz, von denen es zwei gibt.

[0070] Der SDRAM jeder Gruppe kann angeordnet sein, um in verschiedenen Modi oder Konfigurationen wirksam zu sein. Wenn derselbe zur Operation eines wahlfreien Adressierens mit der höchsten Geschwindigkeit konfiguriert ist, erfolgt ein Multiplexen zwischen Gruppen mit der höchsten Rate, und aufeinander folgende Speicheroperationen werden immer und automatisch an die nächste Gruppe in einer zyklischen Sequenz derselben gesendet. In einer Gruppe sind Speicheroperationen verschachtelt, um dieselben gleichmäßig unter den vier Bänken zu verteilen. Eine regelmäßige zyklische Sequenz ist hier ebenfalls bevorzugt. Das Verschachteln erzeugt eine vierfache Steigerung der Geschwindigkeit, die, wenn dieselbe mit einer dreifachen Steigerung kombiniert wird, die durch das Multiplexen geliefert wird, eine Geschwindigkeitssteigerung um einen Faktor von 12 ist. Diese Operationsweise behandelt jede Bank als einen vollen Adressraum ohne einen Versuch, vorzeitig zu steuern, welche der zwölf Bänke das Ziel für eine bestimmte Speichertransaktion ist. Das heißt, zu speichernde Daten können in einer beliebigen der zwölf Bänke enden, und eine einfache Leseoperation von dem Speicher kann Inhalte von einer beliebigen der zwölf Bänke wiedergewinnen. Es gibt keinen Grund dafür, zu erwarten, dass der Inhalt einer Adresse bei einer Bank der gleiche wie der Inhalt dieser glei-

chen Adresse bei einer anderen Bank ist. An diesem Punkt ist jedoch klar, dass Daten mit einer Rate gespeichert werden können, die zumindest zehnmal so hoch ist wie die Rate für eine einzelne Bank von DRAM. Dies geschieht natürlich auf Kosten dessen, dass, um Daten an einer Adresse auszulesen, der Inhalt dieser Adresse bei allen zwölf Bänken untersucht werden muss. (Streng genommen stimmt das nicht immer. Es gibt einen Operationsmodus, bei dem der Inhalt von nur vier Bänken untersucht werden muss. Darüber wird in Kürze mehr gesagt.)

[0071] An diesem Punkt kann ein Teil der Terminologie und kurze Beschreibungen für einige der verschiedenen Modi und Konfigurationen, die die DRAM-Speichersätze unterstützen können, dargelegt werden. Dies sind:

Wahlfrei 100 MHz (R100)

[0072] Schreiboperation voller Geschwindigkeit, die drei gemultiplexte Gruppen von jeweils vier verschachtelten Bänken verwendet, um wahlfrei adressierte Schreiboperationen in einen Adressraum zu erlauben, der in seiner Tiefe gleich einer Bank ist. Sowohl Multiplexen als auch Verschachteln sind in Verwendung. Daten werden in die „nächste" Bank in der „nächsten" Gruppe geschrieben, und eine eingehende Adresse hat keine Bits, um eine bestimmte Gruppe oder eine bestimmte Bank zu identifizieren.

Wahlfrei 33 MHz (R33)

[0073] Schreiboperation reduzierter Geschwindigkeit, die eine adressierte Gruppe von vier verschachtelten Bänken verwendet, um wahlfrei adressierte Schreiboperationen in einen Adressraum zu erlauben, der in seiner Tiefe gleich drei Gruppen ist, was einem drei Bänke tiefen Adressraum gleichwertig ist. Verschachteln ist in Verwendung, nicht jedoch Multiplexen. Daten werden in die „nächste" Bank in der adressierten Gruppe geschrieben, und eine Adresse weist Gruppenauswählbits, aber keine Bankauswählbits auf. Gruppenauswählbits sind höchstwertige Adressbits, so dass außer bei Gruppengrenzen aufeinander folgende Adressen zu aufeinander folgenden In-Bank-Orten gehören, jedoch bei einer Bank, die durch das automatische Verschachteln bestimmt ist.

Lokalisiert 100 MHz (L100)

[0074] Eine Operation mit voller Geschwindigkeit, die eine adressierte Bank innerhalb einer adressierten Gruppe verwendet, um sowohl Leseoperationen als auch Schreiboperationen zu erlauben, die minimale ZEILEN-Adressveränderungen bei einem Adressraum aufweisen, der in seiner Tiefe gleich zwölf Bänken ist. Eine eingehende Adresse weist Gruppenauswählbits, Bankauswählbits und Adressbits auf, die einen In-Bank-Ort spezifizieren. Kein Multiplexen, kein Verschachteln, und ein In-Bank-Adressieren soll Lokalitätsanforderungen erfüllen. Fehlende Lokalität wird automatisch erfasst und ein benötigtes erneutes Adressieren wird durchgeführt. Fehlende Lokalität ist nicht gravierend, aber falls beständig dagegen verstoßen wird, ist das Ergebnis eine verschlechterte, langsame Leistung.

Gestapelte Speichersätze

[0075] Bei jedem der oben Genannten können die Speichersätze 0 und 1 kombiniert werden, um die doppelte Tiefe von Adressraum zu präsentieren, wie es auch bei den Speichersätzen 2 und 3 möglich ist. Die eingehende Adresse weist Speichersatzauswählbits auf.

Schmalwörter

[0076] Bei jedem der oben genannten Modi kann ein Speichersatz konfiguriert sein, um eine Wortbreite aufzuweisen, die eine Potenz von 2 und kleiner oder gleich 32 ist. Ein derartiges Schmalwort ist ein Feld innerhalb der vollen Wortbreite eines adressierten Worts, ist an Potenz-von-Zwei-Grenzen angeordnet und verwendet zusätzliche Adressbits, um das Feld innerhalb des adressierten Worts zu lokalisieren. Funktioniert mit R100 und R33, aber nicht mit L100.

Zusammensetzung

[0077] Bei den oben genannten „wahlfreien" Zugriffsmodi wird die Notwendigkeit, mehrere Bänke zu lesen, wenn Ausgangsdaten für eine Leseoperation vorbereitet werden, durch eine Hardwarehilfe unterstützt, die die Ergebnisse an einer Adresse von den vier Bänken der geeigneten Gruppe für R33 und von allen zwölf Bänken

für R100 zusammenführt. Durch ein Einschließen einer Zusammensetzungsoperation in einer Schleife, die die In-Bank-Adresse durchwandert, während die Ergebnisse in allen Bänken oder in einer bekannten Bank gespeichert werden, kann ein ganzer Speicherbereich vorzeitig zusammengesetzt werden, um einen schnelleren Zugriff während der Analyse von Testergebnissen zu erlauben.

Zusammensetzungsintegrität

[0078] Eine Erfassung eines Verlusts einer Zusammensetzung bei einem zusammengesetzten Speicherbereich weist eine Hardwarehilfe auf.

Verborgenes Auffrischen

[0079] Die Auffrischoperation der DRAMs in den verschiedenen Bänken wird automatisch ohne ein Eingreifen in jegliche der oben genannten Modi oder Konfigurationen durchgeführt.

[0080] Im Einklang mit der obigen Sammlung von Fähigkeiten können einzelne Speichertransaktionen als zu einer der folgenden Kategorien gehörend beschrieben werden:

Überschreib-Schreiboperation (OWW)

[0081] Eine strenge Ersetzungsschreiboperation in alle (jede der) ein, vier oder zwölf Bänke an einer Adresse; vorhergehende Inhalte gehen verloren. Funktioniert mit R100 (12 Bänke), R33 (4 Bänke) und L100 (1 Bank), bewahrt jedoch nicht eine Nenngeschwindigkeitsoperation für wahlfreies Adressieren. Aufrechterhalten zur Kompatibilität mit Testprogrammen für ältere Speichertester. Verwendet klassifizierte Adresse und klassifizierte Daten.

Überlagerungs-Schreiboperation (OLW)

[0082] Eine Lesen-Modifizieren-Schreiben-Operation, die „klebende Nullen" unterstützt, bei der „Nächsten" von einer, vier oder zwölf Bänken. Eine Null in einer Bitposition wird nicht mit einer Eins überschrieben, aber eine Eins kann mit einer Null überschrieben werden. Funktioniert mit Nenngeschwindigkeit bei R100 (12 Bänke), R33 (4 Bänke) und L100 (1 Bank). Dies ist die hauptsächliche Art und Weise, um Daten während eines Testens zu schreiben, und ermöglicht das Einfangen eines Fehlers in einer Bitposition über ein wiederholtes Testen an einer DUT-Adresse. Darf nicht mit der Zusammensetzung verwechselt werden, da die Daten, außer dieselben werden in L100 geschrieben (eine belastende Anforderung), noch zusammengesetzt werden müssen. Verwendet eine klassifizierte Adresse und klassifizierte Daten.

Systemschreiboperation (SYW)

[0083] Schreibt in eine einzige Bank bei L100. Schreibt das gleiche in alle vier Bänke einer adressierten Gruppe bei R33 und in alle zwölf Bänke bei R100, bewahrt jedoch nicht eine Nenngeschwindigkeit. Quelle für die Adresse und die Daten ist der Ringbus.

Systemleseoperation (SYR)

[0084] Leseoperationen von einer einzigen Bank bei L100. Führt an der Adresse eine zusammengesetzte Leseoperation von allen vier Bänken einer adressierten Gruppe bei R33 und von allen zwölf Bänken bei R100 durch, bewahrt jedoch nicht die Nenngeschwindigkeit. Adressquelle und Datenbestimmungsort ist der Ringbus.

Analyseleseoperation (ANR)

[0085] Führt an der Adresse eine zusammengesetzte Leseoperation von allen vier Bänken einer adressierten Gruppe bei R33 und von allen zwölf Bänken bei R100 durch, bewahrt jedoch nicht die Nenngeschwindigkeit. Verwendet die klassifizierte Adresse und sendet die Daten über Wege **62A**–D an Nachdecodieren.

Pufferspeicherleseoperation (BMR)

[0086] Leseoperation bei voller Geschwindigkeit (100 MHz) an einer wahlfreien Adresse für die Bank, die die „Nächste" ist. Funktioniert mit L100 (die gleiche Bank ist immer die „Nächste"), R33 (die „Nächste" von vier

Bänken in der adressierten Gruppe) und R100 (die „Nächste" von vier Bänken in der „Nächsten" von drei Gruppen). Verwendet die klassifizierte Adresse und sendet die Daten über Wege **62A**–D an Nachdecodieren.

[0087] Es ist offensichtlich, dass sich einige der obigen Punkte aus dem Multiplex- und Verschachtelungsschema ergeben, das im Folgenden genauer beschrieben wird. Die Multiplex- und Verschachtelungsschemata sind natürlich auf die DRAM-Speichersätze beschränkt (die SRAM-Speichersätze sind ohnehin schnell). Dies bedeutet jedoch nicht, dass diese gleichen Fähigkeiten oder Operationsmodi nicht durch die SRAM-Speichersätze unterstützt werden können. Allgemein können Speichertransaktionen, die auf einen Speichersatz gerichtet werden können, auf einen beliebigen anderen gerichtet werden, wobei dies nur Größeneinschränkungen unterliegt. Ein SRAM-Speichersatz erfüllt jeden beliebigen Operationsstil, den ein DRAM-Speichersatz erfüllen würde. Der Unterschied besteht darin, wie die Speichersatzsteuerung intern die gewünschte Transaktion implementiert. Zum Beispiel muss ein SRAM-Speichersatz in dem Fall einer Analyseleseoperation (Zusammensetzen) nur die einfache Leseoperation durchführen, da seine Daten ohnehin bereits zusammengesetzt sind.

[0088] Diese verschiedenen Speichertransaktionsstile können nach Bedarf in einem Testprogramm kombiniert werden. Zum Beispiel kann eine Schleife nach einem Testen alle Daten in einem Adressbereich für einen bestimmten Speichersatz zusammensetzen. Dann können BMRs (mit hoher Geschwindigkeit) verwendet werden, um in einer wahlfreien Reihenfolge an die Daten zu gelangen. Dies funktioniert, da jede Bank an jeder Adresse (in dem zusammengesetzten Bereich) die gleichen Daten aufweist.

[0089] Die Erfahrung legt nahe, dass es eine erhebliche Zeitmenge dauern kann, die obigen Speichermodus-/Konfigurations- und Transaktionsinformationen zu verarbeiten. Die folgende Tabelle I wird als eine komprimierte Zusammenfassung angeboten, um bei dem Verarbeitungsprozess behilflich zu sein. Die Notation W 1/4 bezeichnet ein Schreiben in eine von vier Bänken; R bezeichnet eine Leseoperation.

TABELLE I

SPEICHERTRANSAKTIONEN

Quelle/ Best.	Operation	Name	Beschreibung	MODUS: L100	R33	R100
RINGBUS	SYSTEM-SCHREIBOP.	SYW	ALLE (1, 4, 12) BÄNKE	W 1/1	W 4/4	W 12/12
	SYSTEM-LESEOP.	SYR	ALLE (1, 4, 12) BÄNKE	R 1/1	R 4/4	R 12/12
HOCHG.ADR. UND DATEN	MUSTER-SCHREIBOP.	OWW	ALLE (1, 4, 12) BÄNKE	W 1/1	W 4/4	W 12/12
	MUSTER-LESEOP.	ANR	ALLE (1, 4, 12) BÄNKE	R 1/1	R 4/4	R 12/12
HOCHG.ADR. UND DATEN	MUSTER-SCHREIBOP.	OLW	NÄCHSTE (1, 4, 12) BÄNKE	W 1/1	W 1/4	W 1/12
	MUSTER-LESEOP.	BMR	NÄCHSTE (1, 4, 12) BÄNKE	R 1/1	R 1/4	R 1/12

[0090] Um mit der Erörterung fortzufahren, sei nun auf Fig. 6 Bezug genommen, bei der es sich um ein Blockdiagramm **126** der Master-DRAM-Steuerung **109** handelt, die in Fig. 5 erscheint. Angelegte SCHREIBDATEN **107** und ADRESSE **108** von der zugeordneten Speichersatzsteuerung sind mit jeweiligen FIFOs **127** und **128** gekoppelt. An der Ausgangsseite des FIFO **127** werden die SCHREIBDATEN **131** an weitere FIFOs angelegt, die den unterschiedlichen Gruppen zugeordnet sind. Bei diesen handelt es sich um FIFOs **137**, **139** und **141**. Ihre Ausgänge (**166**, **168** und **170**) sind die tatsächlichen Schreibdatenbusse für die jeweiligen Gruppen 0–2. Auf eine ähnliche Weise wird die Ausgabe des FIFO **128** an die FIFOs **138**, **140** und **142** angelegt, deren Aus-

gänge wiederum die Adressbusse (**167**, **169** und **171**) für diese Gruppen werden.

[0091] Die Master-DRAM-Steuerung **109** umfasst eine Zustandsmaschine **193**, die mit ADRESSE **132** sowie mit dem Ringbus **85** gekoppelt ist. Unter anderem können verschiedene Modussteuerregister **130** eingerichtet sein, um den gewünschten Modus und die gewünschte Konfiguration anzuzeigen. Die Zustandsmaschine **193** ist auch dafür zuständig, auszuwählen, welche Gruppe die nächste Speichertransaktion empfangen soll, gemäß dem geltenden Modus und der geltenden Konfiguration. Das heißt, dieselbe beachtet entweder ein Feld von Gruppenauswählbits bei einer eingehenden Adresse oder wählt automatisch die nächste Gruppe aus. Um eine Speicheroperation für eine Gruppe zu erzeugen, gibt dieselbe die geeigneten GRUPPENZYKLUSSTEUERsignale **133** aus. Diese werden für diese Gruppe in den FIFO zwischengespeichert (**143** für Gruppe 0, **144** für Gruppe 1 und **145** für Gruppe 2), von wo dieselben zur gegebenen Zeit auf den Speicher-ZYKLUSSTEUERUNG-Bussen (**172**, **173** und **174**) für diese Gruppen herauskommen. Andere Gründe dafür, dass die Zustandsmaschine **193** mit der ADRESSE **132** verbunden ist, werden während der folgenden Erörterung des Adressierschemas für die DRAM-Speichersätze des ECR ersichtlich.

[0092] Leseoperationen erzeugen GRUPPENLESEDATEN **134**, **135** und **136** für Gruppe 0, Gruppe 1 bzw. Gruppe 2. Diese Ergebnisse werden an eine Zusammensetzungsschaltung **146** sowie an einen MUX **148** angelegt. Der MUX **148** wählt zwischen einem der einzelnen GRUPPELESEDATEN und der zusammengesetzten Version von Daten für diese Adresse (ZUSAMMENGESETZTE GRUPPENLESEDATEN **147**) aus. Der kundige Leser wird bemerken, dass, wenn gerade eine Zusammensetzoperation stattfindet, jeder der GRUPPENLESEDATEN-Busse vier Datenwörter liefern muss, was die Notwendigkeit für vier aufeinander folgende LESEOPERATIONEN impliziert (was die ADRESS-FIFOs **138**, **140** und **142** und die ZYKLUSSTEUER-FIFOs **143**, **144** und **145** umfasst). Die Verwaltung einer derartigen Organisation unterliegt der Steuerung der Zustandsmaschine **193** und erfolgt ansprechend auf die Beschaffenheit der verschiedenen Speichertransaktionen, die spezifiziert werden können. In jedem Fall erscheinen die (durch den MUX **148** unter der Steuerung der Modussteuerregister **130**) ausgewählten Daten als GRUPPENLESEDATEN **149**, die dann entweder in den FIFO **150** zwischengespeichert werden, um LESEDATEN **62A**/B zu werden, oder angelegt werden, als ob dieselben SCHREIBDATEN **107** wären, und in (alle Bänke von dem) den Speicher zurückgeschrieben werden. Erneut ist es die Zustandsmaschine **193**, die diese verschiedenen Speichertransaktionen überwacht.

[0093] Bevor die Slave-SDRAM-Steuerungen behandelt werden, erfolgt eine Zuwendung zu Fig. 7, bei der es sich um ein Blockdiagramm der ZUSAMMENSETZUNGS-Schaltung **146** handelt. Es sei daran erinnert, dass ihre Funktion darin besteht, von der gleichen Adresse für alle zwölf Bänke über drei Gruppen (Daten, die als R100 gespeichert sind) oder für alle vier Bänke einer Gruppe (Daten, die als R33 gespeichert sind) zu lesen und den Inhalt in ein Wort zusammenzuführen, wobei Nullen in einer Bitposition bewahrt werden, selbst wenn andere Wörter bei den zwölf (vier) eine Eins in dieser Bitposition aufweisen sollten. Zu diesem Zweck richtet es die Master-DRAM-Steuerung von Fig. 6 ein, dass alle vier Bänke bei jeder geeigneten Gruppe unter Verwendung der gleichen Adresse gelesen werden. Somit gilt es für einen R100-Fall, dass die Bänke 0, 3, 6 und 9 (in einer Sequenz von 0–11 Bänken, die bei diesem Beispiel für eine R100-Operation so benannt sind – in jeder Gruppe sind die Bänke darin einfach 0–3 benannt) in Sequenz bei GRUPPE-0-LESEDATEN **134** erscheinen; die Bänke 1, 4, 7 und 10 erscheinen in Sequenz bei GRUPPE-1-LESEDATEN **135**; und die Bänke 2, 5, 8 und 11 erscheinen in Sequenz bei GRUPPELESEDATEN **136**. Die Daten für die Bänke 0, 1 und 2 erscheinen gleichzeitig während eines Zyklus, während diejenigen für die Bänke 3, 4 und 5 gleichzeitig während des nächsten Zyklus erscheinen usw. Für jedes Bit in diesen DATEN **134–136** gibt es ein entsprechendes UND-Gatter (z. B. **151** & **152**), dessen Ausgabe nur wahr ist, wenn dieses Bit in allen drei Gruppen gesetzt ist, d.h. wenn bei dieser Adresse irgendwelche aufgezeichneten Fehler (Nullen) vorlägen, wäre die Ausgabe des UND-Gatters ebenfalls eine Null. Dies muss vier Mal durchgeführt werden (einmal für jede Stufe des Verschachtelns, von Bank zu Bank), und die Ausgabe des UND-Gatters muss in einem zugeordneten Zwischenspeicher (z. B. Zwischenspeicher **156** für UND-Gatter **151**, Zwischenspeicher **157** für UND-Gatter **152**) erfasst werden. Der Zustand des Zwischenspeichers wird an das UND-Gatter zurückgeführt, so dass, falls der Zwischenspeicher einmal eine Null erfasst, die Ausgabe des UND-Gatters Null bleibt, obwohl Bänke vorliegen, die eventuell noch geprüft werden müssen. (Im Folgenden wird darauf verzichtet, Bänke als 0–11 zu benennen, zugunsten eines nützlicheren Formats, das im Folgenden dargelegt ist und mit dem in Fig. 5 dargelegten übereinstimmt. Trotz ihrer nützlichen Andeutung haben die Namen 0–11 nur für eine R100-Operation Sinn, und es wäre verwirrend, diesen Stil auf eine R33-Operation auszudehnen.)

[0094] Das Schema, wie es bis zu diesem Punkt beschrieben ist, funktioniert, vorausgesetzt, dass die Zwischenspeicher gesetzt sind, bevor die Zusammensetzung beginnt. Dies benötigt jedoch Zeit, und es ist erwünscht, dass das Schema unabhängig von dem Anfangszustand der Zwischenspeicher **156–157** funktioniert. Dies wird eingerichtet, indem ein Signal ERSTER ZYKLUS **155** vorliegt, das als eine Eingabe an Zwei-Ein-

gangs-ODER-Gatter (**153**, **154**) angelegt wird, die auch die Zwischenspeicherausgabe als die andere Eingabe empfangen. Die Ausgaben der ODER-Gatter **153**–**154** werden als die „rückführenden" Eingaben an die UND-Gatter **151**–**152** angelegt. Das Signal ERSTER ZYKLUS **155** wird durch die Zustandsmaschine **193** in der Master-DRAM-Steuerung **109** erzeugt und ist nur während der Leseoperation der ersten Bänke jeder Gruppe WAHR. Bei den zwei DRAM-Speichersätzen (0 und 1) handelt es sich dabei um die Bänke 0:0:0, 0:1:0, 0:2:0, 0:3:0 und 1:0:0, 1:1:0, 1:2:0, 1:3:0 (das hier verwendete Format ist Speichersatz #: Gruppe #: Bank #).

[0095] Die Wirkung des Signals ERSTER ZYKLUS **155** besteht darin, den Zustand der Zwischenspeicher **156**–**157** während dieses ersten Zyklus zu „bedeutungslosen" Werten („don't cares") zu machen, und es den UND-Gattern **151**–**152** zu ermöglichen, die richtigen Ergebnisse zu erzeugen. Diese Ergebnisse setzen dann die Zwischenspeicher **156**–**157** richtig, woraufhin ERSTER ZYKLUS **155** für die Dauer der Zusammensetzungsoperation zu FALSCH übergeht.

[0096] Nachdem alle vier Banksätze gelesen worden sind, enthalten die 32 Zwischenspeicher **156**–**157** die ZUSAMMENGESETZTEN GRUPPENLESEDATEN **147**, die dann durch die Master-DRAM-Steuerung **109** von Fig. 6 in der im Vorhergehenden angezeigten Weise verwendet werden. Eine zusätzliche Information wird benötigt, um zu verstehen, wie dieser gleiche Mechanismus für die R33-Operation funktioniert. Ein nicht verwendeter oder inaktiver GRUPPE-N-LESEDATENBUS (dies wären während R33 zwei von **134**, **135** und **136**) erscheint als nur Einsen. Dies ermöglicht es, dass der gleiche Mechanismus, der für R100 funktioniert, auch für R33 richtig funktioniert.

[0097] Es sei nun Bezug genommen auf Fig. 8, bei der es sich um ein Blockdiagramm **158** einer Slave-SDRAM-Steuerung (**110**, **111** und **112** von Fig. 5) handelt. Ein zentrales Element der Slave-SDRAM-Steuerung ist eine Zustandsmaschine **161**, die einige Steuerregister **180** umfasst, die durch ein Koppeln mit dem Ringbus **85** gesetzt werden. Eine GRUPPE-N-ADRESSE (eines von **167**, **169** oder **171**) wird an einen FIFO **159** angelegt, von wo dieselbe durch ein Register **160** erfasst wird und auch mit der Zustandsmaschine **161** gekoppelt wird. (Es sei darauf hingewiesen, dass der Wert für N 0, 1 oder 2 ist.) Von dem Register **160** wird die GRUPPE-N-ADRESSE **170** an die SDRAM-Chips angelegt, die die interessierende Gruppe bilden. Die Zustandsmaschine **161** empfängt auch GRUPPENZYKLUSSTEUER-Informationen (eines von **172**, **173** oder **174**) von der Master-SDRAM-Steuerung **109**. Daraus kann die Zustandsmaschine **161** zusätzlich dazu, dass dieselbe weiß, welcher Operationsmodus und welche Konfiguration gilt, die geeignete Sequenz von SDRAM-Steuersignalen **176** (umfasst RAS, CAS, Chipfreigabe usw.) für die interessierende Gruppe erzeugen. Es ist die Zustandsmaschine **161**, die tatsächlich das Verschachteln erzielt, durch die Weise, in der dieselbe diese Steuersignale **176** erzeugt.

[0098] Die Zustandsmaschine **161** enthält auch einen Auffrischzeitnehmer (nicht explizit gezeigt), der, wenn derselbe abläuft (normalerweise nach etwa 40 µs), weitere Operationen von außen blockiert, während irgendein Teil einer Auffrischung durchgeführt wird. Eine Auffrischung wird eine Zeile nach der anderen für alle Spalten in der Zeile durchgeführt. Jeder nächste Auffrischungsteil behandelt die nächste Zeile. Um diese Operationsweise zu erleichtern, sind die Slave-DRAM-Steuerung und ihre äußere Umgebung pipelineartig angeordnet (alle diese FIFOs), und die systemspezifische Rate einer Slave-SDRAM-Steuerungsoperation beträgt 143 MHz, so dass die etwa 7% ihrer Zeit, die dem Auffrischen gewidmet werden, derselben trotzdem Zeit lassen, mit einer Gesamtrate von 100 MHz zu antworten.

[0099] In der Zwischenzeit werden die GRUPPE-N-SCHREIBDATEN (eines von **166**, **168** oder **170**) an einen FIFO **162** angelegt, dessen Ausgabe daraufhin als eine Eingabe an einen (2:1) X 32-MUX **163** angelegt wird. Das Ausgangssignal des MUX **163** wird mit den Signalen gekoppelt, bei denen es sich um die GRUPPE-N-DATEN **178** für die interessierende Gruppe handelt. An diesen Leitungen **178** erscheinen sowohl zu schreibende Daten als auch Daten, die gelesen worden sind. Zu schreibende Daten kommen von dem MUX **163** und stammen entweder über den FIFO **162** von den GRUPPE-N-SCHREIBDATEN oder von Daten, die gerade gelesen und in einem Register **164** gespeichert worden sind. Ein Weg **179** stellt diesen letzteren Fall dar, der auftritt, wenn eine Operation des Lesen/Modifizieren/Schreiben-Stils durchgeführt werden soll. (Der „Modifizierungs"-Teil kann komplex sein, wird mit dem MUX **163** durchgeführt und interessiert hier nicht.) Daten, die zur Verwendung außerhalb der Slave-SDRAM-Steuerung gelesen wurden, werden ferner in ein Register **165** zwischengespeichert, von wo dieselben GRUPPE-N-LESEDATEN werden (eines von **134**, **135** oder **136**).

[0100] Die verschiedenen Operationsmodi und Konfigurationen der Speichersätze werden durch ein Adressierschema erleichtert, das durch die folgenden Tabelle II–Tabelle X veranschaulicht ist. Das Gezeigte bezieht sich auf SDRAM-Speichersätze. Das Adressierschema für die SRAM-Speichersätze ist ähnlich, aber auch etwas einfacher (da dieselben keine Gruppen und Bänke aufweisen).

TABELLE II

SYMBOLDEFINITION

Symbol	Bedeutung
M	Speichersatzauswählbit zum Stapeln von Speichersätzen
G	Gruppenauswählbits zum Stapeln von Gruppen in einem Speichersatz
B	Bankauswählbits zum Stapeln von Bänken in einer Gruppe
R	Zeilenadressbits
C	Spaltenadressbits
F	Feldauswählbits zur Schmalwortoperation
E	Chipfreigabe

TABELLE III

UNTERSTÜTZTE TEILE

Daten-bits	Zeilen-bits	Spalten-bits	Bank-bits	Organisation
512M	13	10	2	(8M Adressen X 16 Datenbits) X 4 Bänke
256M	13	9	2	(4M Adressen X 16 Datenbits) X 4 Bänke
128M	12	9	2	(2M Adressen X 16 Datenbits) X 4 Bänke
64M	12	8	2	(1M Adressen X 16 Datenbits) X 4 Bänke

TABELLE IV

ABTASTTESTPROGRAMMABBILDUNGEN

Bit	3 3 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0
Pos	1 0 9 8 7 6 5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0
Vorgabeadresse von Abbildungsvorrichtung	<----------X----------> <----------Y---------->
Typische Adresse von Abbildungsvorrichtung	<------Z------> <----------X----------> <----------Y---------->

TABELLE V

512M-SDRAM-EINZELSPEICHERSATZ (R100‡ ODER R33)

Bit	33	22	22	22	22	22	21	11	11	11	11	10	00	00	00	00	00	00
Pos	10	98	76	54	32	10	98	76	54	32	10	98	76	54	32	10	98	76	54	32	10
Mode																					
1bit	G	G	E	R	R	R	R	R	R	R	R	R	R	R	C	C	C	C	C	C	C	C	C	C	C	C	F	F	F	F	F
2bit		G	G	E	R	R	R	R	R	R	R	R	R	R	R	C	C	C	C	C	C	C	C	C	C	C	F	F	F	F	F
4bit			G	G	E	R	R	R	R	R	R	R	R	R	R	R	C	C	C	C	C	C	C	C	C	C	F	F	F	F	F
8bit				G	G	E	R	R	R	R	R	R	R	R	R	R	C	C	C	C	C	C	C	C	C	C	F	F	F	F	F
16bit					G	G	E	R	R	R	R	R	R	R	R	R	R	C	C	C	C	C	C	C	C	C	C	F	F	F	F	F
32bit						G	G	E	R	R	R	R	R	R	R	R	R	R	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C

‡ G IST WÄHREND EINER R100-OPERATION ABWESEND ODER WIRD NICHT BEACHTET

TABELLE VI

GESTAPELTE 512M-SDRAM-SPEICHERSÄTZE (R100‡ ODER R33)

Bit	33	22	22	22	22	22	21	11	11	11	11	10	00	00	00	00	00	00
Pos	10	98	76	54	32	10	98	76	54	32	10	98	76	54	32	10	98	76	54	32	10
Mode																					
1bit	G	G	E	M	R	R	R	R	R	R	R	R	R	R	C	C	C	C	C	C	C	C	C	C	C	C	F	F	F	F	F
2bit		G	G	E	M	R	R	R	R	R	R	R	R	R	R	C	C	C	C	C	C	C	C	C	C	C	F	F	F	F	F
4bit			G	G	E	M	R	R	R	R	R	R	R	R	R	R	C	C	C	C	C	C	C	C	C	C	F	F	F	F	F
8bit				G	G	E	M	R	R	R	R	R	R	R	R	R	C	C	C	C	C	C	C	C	C	C	F	F	F	F	F
16bit					G	G	E	M	R	R	R	R	R	R	R	R	R	C	C	C	C	C	C	C	C	C	C	F	F	F	F	F
32bit						G	G	E	M	R	R	R	R	R	R	R	R	R	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C

‡ G IST WÄHREND EINER R100-OPERATION ABWESEND ODER WIRD NICHT BEACHTET

TABELLE VII

1-BIT-MODUS-EINZELSPEICHERSATZ‡

Bit	33	22	22	22	22	22	21	11	11	11	11	10	00	00	00	00	00	00
Pos	10	98	76	54	32	10	98	76	54	32	10	98	76	54	32	10	98	76	54	32	10
Type																					
512M	G	G	E	R	R	R	R	R	R	R	R	R	R	R	C	C	C	C	C	C	C	C	C	C	C	C	F	F	F	F	F
256M		G	G	E	R	R	R	R	R	R	R	R	R	R	R	C	C	C	C	C	C	C	C	C	C	C	F	F	F	F	F
128M			G	G	E	R	R	R	R	R	R	R	R	R	R	R	C	C	C	C	C	C	C	C	C	C	F	F	F	F	F
64 M				G	G	E	R	R	R	R	R	R	R	R	R	R	R	C	C	C	C	C	C	C	C	C	F	F	F	F	F

‡ G IST WÄHREND EINER R100-OPERATION ABWESEND ODER WIRD NICHT BEACHTET

TABELLE VIII

32-BIT-MODUS-EINZELSPEICHERSATZ ǂ

Bit	33	22	22	22	22	22	11	11	11	11	11	10	00	00	00	00
Pos	10	98	76	54	32	10	98	76	54	32	10	98	76	54	32	10
Type																
512M	GG	ER	RR	RR	RR	RR	RR	RR	RR	RR	RR	CC	CC	CC	CC	CC
256M	GG	ER	RR	RR	RR	RR	RR	RR	RR	RR	RR	CC	CC	CC	CC	CC
128M	GG	ER	RR	RR	RR	RR	RR	RR	RR	RR	RR	CC	CC	CC	CC	CC
64 M	GG	ER	RR	RR	RR	RR	RR	RR	RR	RR	RR	CC	CC	CC	CC	CC

ǂ G IST WÄHREND EINER R100-OPERATION ABWESEND ODER WIRD NICHT BEACHTET

TABELLE IX

EINZELSPEICHERSATZ L100 (IMMER 32 BIT)

Bit	33	22	22	22	22	22	11	11	11	11	11	10	00	00	00	00
Pos	10	98	76	54	32	10	98	76	54	32	10	98	76	54	32	10
Type																
512M	GG	BB	ER	RR	RR	RR	RR	RR	RR	RR	RR	RC	CC	CC	CC	CC
256M	GG	BB	ER	RR	RR	RR	RR	RR	RR	RR	RR	RC	CC	CC	CC	CC
128M	GG	BB	ER	RR	RR	RR	RR	RR	RR	RR	RR	RC	CC	CC	CC	CC
64 M	GG	BB	ER	RR	RR	RR	RR	RR	RR	RR	RR	RC	CC	CC	CC	CC

TABELLE X

GESTAPELTE SPEICHERSÄTZE L100 (IMMER 32 BIT)

Bit	33	22	22	22	22	22	11	11	11	11	11	10	00	00	00	00
Pos	10	98	76	54	32	10	98	76	54	32	10	98	76	54	32	10
Type																
512M	GG	BB	EM	RR	RR	RR	RR	RR	RR	RR	RR	RC	CC	CC	CC	CC
256M	GG	BB	EM	RR	RR	RR	RR	RR	RR	RR	RR	RC	CC	CC	CC	CC
128M	GG	BB	EM	RR	RR	RR	RR	RR	RR	RR	RR	RC	CC	CC	CC	CC
64 M	GG	BB	EM	RR	RR	RR	RR	RR	RR	RR	RR	RC	CC	CC	CC	CC

[0101] Tabelle II identifiziert die Bedeutungen von verschiedenen Symbolen, die in den Tabellen V–X verwendet werden. Das heißt, dieselbe teilt mit, welche Interpretation den verschiedenen Bitpositionen in einer Adresse unter unterschiedlichen ECR-Operations- und Konfigurationsmodi zu geben ist. Die Adressierschemata, die in den Tabellen V–X dargelegt sind, sollten so betrachtet werden, dass sie das sind, was an die Zustandsmaschinen in den Master- und Slave-Speichersteuerungen anstatt direkt an die Speicherteile selbst angelegt wird.

[0102] Tabelle III zeigt verschiedene Speicherteile, mit denen der ECR bestückt sein kann, und gibt Informationen über ihre erwartete Organisation. Insbesondere und mit Bezugnahme auf das vereinfachte Gruppen-SDRAM-Blockdiagramm **181** von Fig. 9 sei darauf hingewiesen, dass es sich dabei um Teile (**182**–**185**) mit 16 Datenbits handelt, so dass, um ein Wort voller Breite (32 Bits) zu erhalten, zwei Teile zusammen adressiert werden (**182** & **183**, **184** & **185**), wobei ihre Ausgangsbits in ein großes Feld **178** vereinigt werden. Es sei ebenfalls darauf hingewiesen, dass jeder Teil vier Bänke aufweist, von denen jede eine separate Instanz des Adressraums des Teils implementiert (die während einer R100- und R33-Operation ausgenutzt wird). Eine Alternativansicht besteht darin, die zugeordneten zwei Bankauswählbits als weitere Adressbits aufzufassen, was für eine L100-Operation nützlich ist. Es sei darauf hingewiesen, dass es zwei Paare von Speicherteilen gibt,

wobei jedes Paar durch eine gemeinsame Instanz eines CHIPAUSWÄHLsignals ausgewählt wird: Signal **188** für Teile **182** und **183**, und Signal **187** für Teile **184** und **185**. Diese CHIPAUSWÄHLsignale funktionieren aus einer Adressiersicht, als ob dieselben mit Zeilenadressbits äquivalent wären. Bei dieser Ansicht werden die Teile **182** und **184** kombiniert, um einen Speicher für die Hälfte der 32 Datenbits zu liefern, während die Teile **183** und **185** die andere Hälfte speichern. Somit ist, wenn die Teile 512Mbit-Teile sind, die gezeigte Anordnung eine 16M-Adresse an jeder von vier Bänken mal 32-Bit-Datenwortspeicher. Das E-Bit (von dem es eines pro Adresse in einem Speichersatz gibt) wird in komplementäre CHIPAUSWÄHLsignale (**187**, **188**) verwandelt, die jeder Gruppe in dem Speichersatz gemeinsam sind. Es sei darauf hingewiesen, dass **Fig. 9** nur eine von drei Gruppen in einem von zwei DRAM-Speichersätzen behandelt.

[0103] Tabelle IV ist nützlich, um Beziehungen zu erkennen, die zwischen der Logik des Testprogramms (die auf DUT-Architektur und interne Organisation fokussiert ist) und den Modi und Konfigurationen für den ECR und seine Speichersätze existieren können. Ein wichtiges zu berücksichtigendes Konzept besteht darin, dass sowohl das DUT als auch der ECR die gleiche geordnete Bitsequenz als eine Adresse auffassen können, selbst wenn dieselben jedes eine sehr unterschiedliche interne Organisation und Operationsweise aufweisen. Somit zeigt Tabelle IV nur zwei von sehr vielen möglichen unterschiedlichen DUT-bezogenen Interpretationen von 32 geordneten Adressbits. Die Tabellen IV–IX zeigen ECR-bezogene Interpretationen dieser gleichen 32 Adressbits, und die zwei Interpretationen haben sehr wenig miteinander zu tun. Auf einer sehr hohen Ebene ist die Hauptsache, die vom Schreiber eines Testprogramms angestrebt wird, die Fähigkeit, dass eine Adresse in dem ECR vorliegt, die der Adresse entspricht, die an das DUT angelegt wird, mit dem Wissen, dass „er" (d. h. der Speichertester unter programmatischer Steuerung) Daten in den ECR schreiben und dieselben später zur Analyse wieder zurückerhalten kann. Einerseits sucht der Testtechniker nach Möglichkeiten, um eine Bedeutung zu erfassen, die sich auf die DUT-Architektur bezieht. Andererseits beziehen sich die ECR-Adressierbits auf seine interne Operation, unabhängig von irgendeinem neumodischen DUT. Um ein einfaches Beispiel vorzuschlagen, gibt es kein Konzept einer Z-Adresse in dem ECR, wie kann also eine Möglichkeit eingerichtet werden, dass ein ECR-Speichersatz eine Adresse oder einen Adressbereich aufweist, die bzw. der einer bestimmten Z-Adresse oder einem Bereich von Z-Adressen in dem DUT entspricht? Im Moment wird dieses Spannungsverhältnis nicht beachtet und geraten anzunehmen, dass die Adressklassierer und Datenklassierer ihre Inhalte einfach unverändert durchleiten, und dass, solange der ECR in einen geeigneten Operationsmodus gesetzt wird (ausreichende Geschwindigkeit, genug Adressen), jede Adresse, die gut genug für das DUT ist, auch gut genug für den ECR ist. (Es stimmt, dass etwas verpasst wird, wenn immer so gedacht wird. Für den Moment interessieren jedoch andere Themen, und deshalb wird zur Verkürzung zumindest vorübergehend so gedacht.)

[0104] Tabelle V zeigt, wie 512M-SDRAM-Teile adressiert werden, wenn dieselben als ein Teil eines einzelnen Speichersatzes konfiguriert sind. Es sei der Fall betrachtet, bei dem die Wortbreite volle 32 Bits beträgt (unterste Zeile der Tabelle). Dies kann unter zwei unterschiedlichen Operationsmodi geschehen: R100 (das volle Unheil eines wahlfreien Adressierens bei 100 MHz unter Verwendung von Multiplexen und Verschachteln) und R33 (wahlfreies Adressieren bei 33 MHz mit Verschachteln, aber ohne Multiplexen). In jedem Fall wird der systemspezifische Adressraum von 8M durch die 13 R's und die 10 C's geliefert. Daraus werden 16M Adressraum (bei jeder von vier Bänken), wenn das Chipfreigabebit E eingeschlossen wird. Bei dem R100 wird kein separates benutzereingeleitetes Gruppenadressieren (die G-Bits) verwendet: die Zustandsmaschine der Master-DRAM-Steuerung richtet einfach die nächste Speichertransaktion auf den Bus der nächsten Gruppe, ohne dass eine Notwendigkeit irgendwelcher Adressierbits besteht, die der Gruppe entsprechen. Dies folgt direkt daraus, dass jede Gruppe ihren eigenen Bus aufweist; Adressen an einem Bus sind gegenüber denjenigen an einem anderen Bus in einer vollkommen separaten Instanz eines Adressraums. Daher die Fußnote in Tabelle V.

[0105] Aber es sei angenommen, dass der Operationsmodus R33 ist. Nun gibt es kein Multiplexen, obwohl immer noch (gleichzeitig!) wahlfrei adressierte Speichertransaktionen auf jede der drei Gruppen gerichtet werden können, allerdings mit der geringeren Rate von 33 MHz. Falls nun jede Gruppe mit Gruppenadressierbits (die G's) ausgestattet wird, können die drei Gruppen gestapelt werden, um einen kombinierten Adressraum zu bilden, der die dreifache Tiefe aufweist (48M Adressierbarkeit). Bei diesem Modus antwortet nur die adressierte Gruppe auf die Speichertransaktion. Aus diesem Grund (unter anderem) ist die Zustandsmaschine (**161**) der Slave-SDRAM-Steuerung mit der GRUPPE-N-ADRESSE gekoppelt: diese Slave-SDRAM-Steuerung kann für die adressierte Gruppe sein oder auch nicht.

[0106] Das Adressierschema, das in Tabelle V gezeigt ist, unterstützt auch die Schmalwortkonfigurationen. Erneut ist es ein Fall eines Lieferns einer zusätzlichen Adressierbarkeit, um die Unterteilung in kleine Felder innerhalb des vollen Wortes zu erklären. Dies sind die F-Bits, die sich zwischen keinem (volles 32-Bit-Wort) bis

fünf (32 Ein-Bit-Felder) bewegen. Diesen Operationsmodus zu implementieren ist ein weiterer Grund, warum die Zustandsmaschine **161** der Slave-SDRAM-Steuerung die GRUPPE-N-ADRESSE empfängt. Es ist auch ein Grund, warum der MUX **163** und der Datenweg **179** (siehe Fig. 8) bereitgestellt sind. Es sei darauf hingewiesen, dass die F-Bits nicht selbst zu den DRAM-Teilen gehen; sie werden wie für ein volles 32-Bit-Wort adressiert. Die Slave-SDRAM-Steuerung liefert diese zusätzliche Schmalwortfähigkeit, und die F-Bits verschwinden bei der SDRAM-Steuerung, um durch das geeignete Steuerungsverhalten ersetzt zu werden.

[0107] Tabelle VI ähnelt Tabelle V. Während Tabelle V Gruppen behandelte, die gestapelt werden können oder auch nicht, behandelt Tabelle VI Speichersätze, die gestapelt werden. Dieselbe verwendet das 512M-Teil als ein Beispiel; reduzierte Versionen von Tabelle VI existieren für die kleineren Speicherteile (genauso wie reduzierte Versionen von Tabelle V). Auf diese anderen Tabellen wurde zur Verkürzung verzichtet. Um zwei Speichersätze zu stapeln, müssen dieselben jeder gleich dem anderen konfiguriert sein, und es muss ein zusätzliches Bit (das M-Bit) bereitgestellt werden, um den Adressraum zu verdoppeln, so dass derselbe zwei Speichersätzen anstelle von nur einem entspricht. Dieses zusätzliche Bit wird durch das Testprogramm gesteuert. Die Master-DRAM-Steuerung antwortet darauf (dies ist der Grund, warum ihre Zustandsmaschine **193** von Fig. 6 mit der ADRESSE **132** gekoppelt ist). Das M-Bit verschwindet an diesem Punkt als ein Adressierbit, um durch die Anwesenheit oder Abwesenheit einer Speicheraktivität ersetzt zu werden, abhängig davon, ob der Speichersatz, der diese Instanz der Zustandsmaschine **193** aufweist, der adressierte Speichersatz ist.

[0108] Es seien nun die Tabellen VII und VIII betrachtet. Dieselben können im Wesentlichen davon abgeleitet werden, was bis zu diesem Punkt erörtert worden ist. Das heißt, die oberste Zeile von Tabelle VII ist gleich der obersten Zeile von Tabelle V, und die oberste Zeile von Tabelle VIII ist gleich der untersten Zeile von Tabelle V. Der Unterschied besteht darin, dass es sich bei den vertikalen Achsen der Tabellen VI und VII um Teilgröße handelt, während es sich bei der vertikalen Achse von Tabelle IV um Schmalwortmodus handelt.

[0109] Die Tabellen IX und X behandeln die L100-Konfiguration. Es sei daran erinnert, dass es sich dabei um den „linearen" oder Lokalitätsmodus des Adressierens handelt (minimale Veränderungen der Zeilenadresse). Hier gibt es kein Multiplexen und kein Verschachteln. Wo vorher zwölf separate Instanzen eines Adressraums vorlagen, gibt es nun einen, dieser ist jedoch zwölfmal so tief. Dies ergibt die Notwendigkeit von weiteren vier Adressbits, die an die Master-DRAM- und Slave-SDRAM-Speichersteuerungen anzulegen sind. Diese zusätzlichen Bits sind GG- und BB-Bits, die in beiden Tabellen gezeigt sind. Wie zuvor verschwinden diese Bits sozusagen in die Steuerungen, um durch die entsprechende Funktionalität ersetzt zu werden, was alles durch intelligente Zustandsmaschinen und separate Busse für die Speichersammlungen, die die Gruppen sind, ermöglicht wird. Der Unterschied zwischen den zwei Tabellen ist das M-Bit, das wie im Vorhergehenden im Zusammenhang mit Tabelle VI beschrieben wirksam ist. Es sei auch darauf hingewiesen, dass die L100-Operation das Konzept des Schmalwortmodus ausschließt: es gibt auf der Systemebene nicht genug Adressbits, um denselben zu unterstützen.

[0110] Einige Worte über die Zustandsmaschinen in den Speichersteuerungen sind angebracht. Zunächst sei die Zustandsmaschine **193** in der Master-DRAM-Steuerung **109** betrachtet. Ihre Hauptaspekte sind folgende: (A) Falls es sich bei dem Modus der Speichersatzoperation um R100 handelt, dann wird ein Multiplexen der Gruppen untereinander benötigt. Dieselbe weiß, welche Gruppe in der Round-Robin-Sequenz von Gruppen die „Nächste" ist, und führt die Speichertransaktion an der Sammlung von Bussen (GRUPPE-N-ADRESSE, SCHREIBDATEN, ZYKLUSSTEUERUNG & LESEDATEN, N = 0, 1) für diese nächste Gruppe aus. (B) Falls es sich bei dem Modus der Speichersatzoperation um einen anderen als R100 handelt, dann wird kein Multiplexen verwendet. In einem derartigen Fall wird die zu verwendende Gruppe durch eingehende Adressbits bestimmt, wie es im Vorhergehenden in Verbindung mit den Tabellen V–X beschrieben ist, und die Speichertransaktion wird an der Sammlung von Bussen (ADRESSE, SCHREIBDATEN, ZYKLUSSTEUERUNG & LESEDATEN) durchgeführt, wie durch die Gruppenadresse ausgewählt. Nun kann nicht mit Sicherheit angenommen werden, dass die Fälle (A) und (B) immer Schreiboperationen sind, obwohl ohne Weiteres zu erkennen ist, dass bei der R100- und R33-Operation keinerlei Garantie besteht, dass eine Leseoperation bei einer Adresse die letzten Daten erzeugt, die in diese Adresse geschrieben wurden. (Genau aus diesem Grund wurde die ZUSAMMENSETZ-Operation entwickelt.) Trotzdem sind Leseoperationen möglich; es sei angenommen, dass der Speicher zusammengesetzt worden ist. Dann gibt es kein Problem. Außerdem sagt Fall (B) nur „NICHT R100" und kann während einer L100-Operation erhalten. Hier gibt es wieder kein Multiplexen, und da es auch kein Verschachteln gibt, kann tatsächlich erwartet werden, dass sich Leseoperationen vorhersagbar verhalten.

[0111] Es darf jedoch nicht angenommen werden, dass die Fälle (A) und (B) die einzigen Fälle sind. Es gibt auch einen Fall (C) einer Zusammensetzung. Eine ZUSAMMENSETZ-Operation erzeugt ihre eigene spezielle Aktivität abhängig davon, ob R100 oder R33 gilt (L100-Schreiboperationen verursachen keine Notwendigkeit

einer Zusammensetzung vor Leseoperationen). In dem R100-Fall muss es vier aufeinander folgende Leseoperationen bei der gleichen Adresse geben (um die Verschachtelung zu veranlassen, alle Bänke durchzugehen). Diese vier Leseoperationen werden gleichzeitig bei jeder Gruppe durchgeführt. Dann muss es vier Schreiboperationen geben (die bei jeder Gruppe gleichzeitig durchgeführt werden), um das zusammengesetzte Ergebnis zurück in alle zwölf Bänke zu bekommen. All diese Aktivität kann das Ergebnis eines einzigen Befehls sein, obwohl dieselbe nicht mit einer 100-MHz-Rate pro Adresse vonstatten geht. (Der Grund hierfür liegt darin, dass es (mit den SDRAM-Teilen, die verwendet werden) keine Möglichkeit gibt, gleichzeitig in alle vier Bänke in einer Gruppe zu schreiben.)

[0112] Eine Zusammensetzung ist auch in dem R33-Operationsmodus möglich. Der Unterschied ist gering, nämlich dass die zwei nicht adressierten Gruppen während der vier Leseoperationen und Schreiboperationen, die benötigt werden, um die Verschachtelung durchzugehen, „heruntergefahren" werden müssen. Während der Leseoperationen müssen die zwei nicht adressierten der GRUPPE-N-LESEDATENBUSSE **134**, **135** und **136** alle Einsen vorliegen haben, während der adressierte wie gewöhnlich wirksam ist. Dies erzeugt das geeignete Ergebnis in der ZUSAMMENSETZEINRICHTUNG **146**, das dann in die Gruppe, die zusammengesetzt wurde, zurückgeschrieben wird. Es ist nur in die vier Bänke der Gruppe zu schreiben, da die entsprechenden Adressen in den anderen Gruppen wirklich ziemlich unterschiedliche Orte in dem verwendeten Adressraum (gestapelte Gruppen) sind. Ein Zusammensetzen über Gruppen bei R33 wäre wie ein Addieren von Kontonummern für unterschiedliche Kreditkarten; das Ergebnis ist keine nützliche Kontonummer! Die Zustandsmaschine in der Master-DRAM-Steuerung kann all dies einrichten durch ein Bestimmen, welche Gruppen GRUPPENZYKLUSSTEUERinformationen über ihre zugeordneten Busse erhalten. Es wird auch angenommen, dass die GRUPPE-N-LESEDATEN-Busse Einsen präsentieren, wenn dieselben inaktiv sind, oder dass dieselben anderweitig veranlasst werden können, in einen logisch hohen Zustand zu gehen. Ist das nicht der Fall, dann wird eine zusätzliche Steuerung über die Eingaben zu den UND-Gattern **151**–**154** benötigt, um die Bits von Gruppen, die nicht an der Zusammensetzung teilnehmen sollen, auszublenden.

[0113] Es sei nun die Zustandsmaschine **161** der Slave-SDRAM-Steuerung von **Fig. 8** betrachtet. Es folgen ihre Hauptzuständigkeiten. Erstens unterhält dieselbe einen Auffrischzeitnehmer. Wenn der Zeitnehmer auf Null heruntergeht, gibt dieselbe einen Auffrischzyklus an den SDRAM aus, den dieselbe steuert. Während dieser Zeit müssen jegliche eingehende Speichertransaktionen in der Pipeline abgehalten werden. Alle Spalten in einer spezifischen Zeile werden durch den ausgegebenen Auffrischzyklus aufgefrischt. Die Zustandsmaschine **161** weiß, welche Zeile als nächste zu behandeln ist. Falls gerade keine Auffrischoperation vonstatten geht, können regelmäßige Speichertransaktionen durchgeführt werden. Falls eine Verschachtelung vorliegt (R100 oder R33), dann wird die Transaktion bei der nächsten Bank durchgeführt. Falls keine Verschachtelung vorliegt, dann wird die Transaktion bei der adressierten Bank durchgeführt. In jedem Fall ist es die Aufgabe der Zustandsmaschine, das Senden der richtigen Sequenz von Speicherzyklussteuersignalen (**176**) zu verwalten. Dies umfasst alle möglichen Verschachtelungen für unterschiedliche Umstände.

[0114] Das Verschachteln, das durch die Slave-SDRAM-Steuerung durchgeführt wird, wird von Fachleuten in der Technik der SDRAM-Verwendung ohne Weiteres verstanden. Es werden nun die Tabellen XI–XV präsentiert, die eine komprimierte Version der Entsprechung zwischen einigen der verschiedenen interessierenden Speichertransaktionen und ihren zugeordneten Verschachtelungsschemata darstellen.

TABELLE XI

DEFINITIONEN

a =	Zeile (& Zeilenadresse) aktivieren
r =	Lesen (& Spaltenadresse)
w =	Schreiben (& Spaltenadresse)
p =	Vorladen (& Bankauswahl)
- =	Taktzyklus
Bn =	Verkehr für Bank # bei AC oder D
D =	(separater) Datenbus; i = (Eingangs-) Schreibdaten, o = (Ausgangs-)Lesedaten
AC =	(separater) Adress- & Steuerbus

TABELLE XII

SDRAM-STEUERUNG

	OPERATIONEN			
SDRAM-ANSCHLUSSSTIFTE	p	a	r	w
Zeilenadressauswahl (RAS)	1	1	0	0
Spaltenadressauswahl (CAS)	1	0	1	1
Schreiboperation/Leseoperation	0	0	0	1

TABELLE XIII

ÜBERLAGERUNGSSCHREIBOPERATION UND ÜBERSCHREIBSCHREIBOPERA TION

```
B0 p----a----r--o-wp----a----r--o-w
B1     p----a----r--o-wp----a----r--o-w
B2         p----a----r--o-wp----a----r--o-w
B3             p----a----r--o-wp----a----r--o-w
D               o-i-o-i-o-i-o-i-o-i-o-i-o-i-o-i
AC p---pa--par-parwparwparwparwparw-arw--rw----
```

16 Zyklen für eine OLW in einer Gruppe

TABELLE XIV

ANALYSELESEOPERATION ODER PUFFERSPEICHERLESEOPERATION

```
B0 p---a---r--op---a---r--o
B1    p---a---r--op---a---r--o
B2       p---a---r--op---a---r--o
B3          p---a---r--op---a---r--o
D             o--o--o--o--o--o--o--o
AC p--pa-pa-parparparparpar-ar--r---
```

12 Zyklen für eine ANR oder BMR in einer Gruppe

TABELLE XV

SCHNELLZUSAMMENSETZUNG

```
B0 p--a------r--o----w---r--o----w---r--o----w
B1   p--a-----r--o----w---r--o----w---r--o----w
B2     p--a----r--o----w---r--o----w---r--o----w
B3       p--a---r--o----w---r--o----w---r--o----w
D  ------------oooo-iiii---oooo-iiii---oooo-iiii---
C  p-papapa-arrrr----wwwwrrrr----wwwwrrrr----wwww
```

12 Zyklen, um R33-Daten bei vier Bänken für 1 Adresse in 1 Gruppe oder für R100-Daten bei 12 Bänken für 1 Adresse in 3 Gruppen (d bis d, r bis r usw.) zusammenzusetzen

[0115] Wer mit SDRAM vertraut ist, erkennt, dass die Inhalte der Tabellen XI und XII herkömmlich sind. Kurz zusammengefasst weisen diese SDRAM-Teile einen Datenbus (D) auf, der von dem Adress-/Steuerbus (AC) getrennt ist. Das Vorladen umfasst eine Bankauswahl. Der Grundoperationszyklus ist p (Vorladen), a (Zeile auswählen) und dann entweder r (Lesen) oder w (Schreiben), die beide eine Spaltenauswahl umfassen, gefolgt von Daten (i oder o) auf D. In dem Format, das für die Tabellen XIII–XV ausgewählt ist, sind die Zeilen, die mit B0–B3 etikettiert sind, keine separaten Sammlungen von elektrischen Signalen. Alles was auf diesen

Zeilen der Tabellen gezeigt ist, geschieht tatsächlich auf dem Datenbus (D) oder auf dem Adress-/Steuerbus (C). Es wird so gezeigt, um aus Übersichtlichkeitsgründen Signalverkehr zu trennen, während gleichzeitig ein derartiger Verkehr bezüglich der Zeit in Ausrichtung gehalten wird und auch ein lästiger Gebrauch von Indices vermieden wird.

[0116] Tabelle XIII zeigt das Verschachtelungsschema, das für die Überlagerungsschreiboperation (OLW) und die Überschreibschreiboperation (OWW) verwendet wird. Es handelt sich dabei um eine ziemlich einfache Anwendung des Verschachtelungskonzepts, und es ist ersichtlich, dass dieselbe 16 Taktzyklen erfordert, um eine OLW für vier Bänke in einer Gruppe durchzuführen. Eine weitere OLW könnte jedoch gleichzeitig in einer anderen Gruppe vonstatten gehen.

[0117] Tabelle XIV zeigt das Verschachtelungsschema, das für eine Analyseleseoperation (ANR) oder eine Pufferspeicherleseoperation (BMR) verwendet wird. Dasselbe erfordert zwölf Taktzyklen, um eine ANR oder eine BMR für vier Bänke in einer Gruppe durchzuführen. Natürlich könnte die gleiche Operation auch gleichzeitig in anderen Gruppen stattfinden.

[0118] Es seien nun einige weitere Aspekte der Zusammensetzoperation betrachtet. Das Testprogramm könnte die Ergebnisse an einer einzigen Adresse zusammensetzen. Dies geschähe mit einer ANR und würde zwölf Taktzyklen erfordern, ob es nun für R33-Daten oder für R100-Daten durchgeführt würde. Diese zwölf Taktzyklen bewerkstelligen jedoch nicht, dass die zusammengesetzten Daten irgendwo gespeichert werden. Um dies zu erreichen, wäre zusätzliche Zeit erforderlich. Falls nun die zusammengesetzten Daten nur für einen Durchgang benötigt werden und nicht gehalten werden müssen, oder wenn auf zusammenzusetzende aufeinander folgende Adressen wahlfrei zugegriffen werden soll, dann muss eine ANR verwendet werden. Um die zusammengesetzten Ergebnisse zu speichern, könnte jede ANR von einer OLW gefolgt sein, auf Kosten von 28 Taktzyklen pro Adressen. Dies ermöglicht nachfolgende Hochgeschwindigkeitszugriffe, falls die zusammengesetzten Daten in alle Bänke zurückgeschrieben werden, von denen dieselben zusammengesetzt wurden. Es geht nicht so sehr darum, dass dies nicht funktionieren würde (es funktioniert), es jedoch dabei zu belassen, bedeutet, eine Möglichkeit zu verpassen, die gleichen zusammengesetzten Ergebnisse wesentlich schneller zu bekommen (in nur zwölf Zyklen pro Adresse), wenn ein aufeinander folgender Adressbereich zusammengesetzt werden soll. Diese Hochgeschwindigkeitszusammensetzung wird mit einer Operation durchgeführt, die Schnellzusammensetzung (FCP) genannt wird. In der Programmierumgebung ist FCP eine Anweisung, die von Parametern begleitet wird, die den Speichersatz und den Adressbereich darin, der zusammengesetzt werden soll, anzeigen.

[0119] Das Verschachtelungsschema für FCP ist in Tabelle XV gezeigt. Es ist auch bei den vier Bänken in einer Gruppe wirksam und kann gleichzeitig in unterschiedlichen Gruppen für R100-Daten oder in einer einzigen Gruppe für R33-Daten durchgeführt werden. In jedem Fall erfordert FCP nur zwölf Taktzyklen pro Adresse und umfasst eine Schreiboperation, so dass zusätzliche Durchgänge bei den zusammengesetzten Daten durchgeführt werden können. Diese zusätzlichen Durchgänge können bei hoher Geschwindigkeit erfolgen.

[0120] Was FCP schnell macht, ist erstens, dass dieselbe auf die gleiche Weise wie L100 wirksam ist. Das heißt, sie nutzt Lokalität, wobei es sich um die Fähigkeit handelt, oft die Notwendigkeit zu vermeiden, ein weiteres Vorladen (p) und ein weiteres Zeile Aktivieren (a) auszugeben, und einfach die Spaltenauswahl während der nachfolgenden r's und w's zu ändern. Natürlich muss die Slave-SDRAM-Steuerung von Zeit zu Zeit ein weiteres (p) und (a) ausgeben. Die Notwendigkeit dazu kann sich entweder daraus ergeben, dass die Zeilenauswahl sich nicht mit der nächsten Adresse geändert hat, oder dass die Zeit, die seit dem letzten Zeile aktivieren (a) verstrichen ist, dies erfordert. Hauptsächlich erfolgt jedoch die große Mehrheit von FCPs in zwölf Taktzyklen. Zweitens ist FCP schnell, weil sie sowohl die Leseoperation als auch die Schreiboperation unter Verwendung von nur einer Instanz eines Adressierens für jeden Ort in einer Bank erledigt. Dies ist eine Folge davon, dass eine einzige vereinigte Operation anstelle von zwei, von denen jede ihr eigenes Adressieren durchführt, vorliegt.

[0121] Schließlich sei **Fig. 10** betrachtet, bei der es sich um ein vereinfachtes Blockdiagramm **189** handelt, wie ein ZUSAMMENGESETZT-Flag (CMP_FLG_MS#N) **190** gesteuert werden kann. Es gibt ein derartiges Flag für jeden DRAM-Speichersatz und es wird verwendet, um die Zusammensetzungsintegrität eines Speicherbereichs anzuzeigen. Die Idee besteht darin, dass, falls das Flag gesetzt ist, es sicher ist, den zugeordneten Speicherbereich als einen zu behandeln, der zusammengesetzt worden ist, und dass derselbe zusammengesetzt bleibt. Eine Schreiboperation in diesen Speicherbereich wird die Zusammensetzung bei der Adresse, in die geschrieben wird, möglicherweise vernichten (was wahrscheinlich auch geschieht), und wird verwendet, um das Flag zu löschen. Das Flag selbst ergibt sich aus dem Zustand eines Flip-Flops oder Zwi-

schenspeichers **191**, der durch ein Signal **193** gesetzt wird, das das logische ODER (erzeugt durch das ODER-Gatter **194**) von folgendem ist: (1) einer expliziten Anweisung **197**, den Zwischenspeicher zu setzen (SET_CMP_FLG_MS#N), die über den Ringbus ausgegeben werden kann; und (2) eines Signals **196** (FCP_MS#N), das anzeigt, dass eine FCP-Operation für den zugeordneten Speichersatz durchgeführt worden ist. Option (2) ermöglicht es dem Testprogramm, das Flag gesetzt zu bekommen, selbst wenn die FCP eventuell nicht verwendet worden ist, um die Zusammensetzung zu erreichen. Jede OLW, die in dem Speichersatz durchgeführt wird (OLW_MS#N **195**), ist eine potentielle Bedrohung für die Integrität der zusammengesetzten Ergebnisse und wird verwendet, um das Flag zu löschen. Der Zustand des Flags kann unter Verwendung des Ringbusses geprüft werden.

Patentansprüche

1. Ein Verfahren zum Durchführen von Speicheroperationen bei einem DRAM (**73**) für Informationswörter, die jeweiligen Adressen in einem Adressraum zugeordnet sind, wobei das Verfahren folgende Schritte aufweist:
(a) Organisieren von (n × m)-vielen Bänken (**113**–**124**) des DRAM in n-viele Gruppen (**88**–**90**) von m-vielen Bänken pro Gruppe, wobei jede Bank einen adressierbaren Ort für jede Adresse in dem Adressraum aufweist;
(b) sequentielles Richten jeder nächsten Speicheroperation auf die nächste Gruppe in einer geordneten zyklischen Sequenz derselben;
(c) in jeder Gruppe Auswählen jeder Bank darin in einer geordneten zyklischen Sequenz;
(d) in jeder Gruppe und für aufeinanderfolgende Speicheroperationen, die durch Schritt (b) auf diese Gruppe gerichtet sind, sequentielles Verschachteln dieser aufeinanderfolgenden Speicheroperationen unter den m-vielen Bänken der Gruppe gemäß der geordneten zyklischen Sequenz von Schritt (c); und
(e) für jede ausgewählte Bank in einer Gruppe und für verschachtelte Speicheroperationen, die durch Schritt (d) auf diese Bänke gerichtet sind, Durchführen der nächsten aufeinanderfolgenden Speicheroperation von Schritt (d) an der Adresse in dem Adressraum.

2. Ein Verfahren gemäß Anspruch 1, bei dem die Speicheroperation ein Schreiben ist, und das ferner den Schritt eines Erhaltens der zu schreibenden Informationen von Tests aufweist, die an einem adressierbaren Testobjekt (**14**) durchgeführt werden.

3. Ein Verfahren gemäß Anspruch 2, bei dem das Testobjekt (**14**) ein Speicher ist und die Bits in dem zu schreibenden Wort Kanäle in einem Speichertestsystem darstellen, und das ferner den Schritt eines Adressierens von adressierbaren Orten in den (n × m)-vielen Bänken mit Adressen, die von Adressen abgeleitet sind, die an das Testobjekt angelegt werden, aufweist.

4. Ein Verfahren gemäß Anspruch 1, bei dem die Speicheroperation ein Lesen ist, und das ferner die Schritte eines Lesens von einem adressierbaren Ort an einer gleichen Adresse in allen (n x m)-vielen Bänken (**113**–**124**), um (n × m)-viele Wörter zu erzeugen, eines Zusammenführens (**146**) der (n × m)-vielen Wörter in ein Endwort, eines Nehmens des Endwortes als das Ergebnis der Speicheroperation und eines Schreibens des Endwortes in alle (n × m)-vielen Bänke an der gleichen Adresse aufweist.

5. Ein Verfahren gemäß Anspruch 4, das ferner die Schritte eines Setzens eines Flags (**191**) nahe dem Zeitpunkt, wenn das Endwort in alle (n × m)-vielen Bänke (**113**–**124**) geschrieben wird, und eines Löschens des Flags auf eine nachfolgende Instanz von Schritt (b) hin, bei der die nächste aufeinanderfolgende Speicheroperation eine Schreiboperation ist, aufweist.

6. Ein Verfahren zum Durchführen von Speicheroperationen bei einem DRAM (**73**) für Informationswörter, die jeweiligen Adressen in einem Adressraum zugeordnet sind, der einen Gruppenauswählabschnitt und einen In-Bank-Adressabschnitt aufweist, wobei das Verfahren folgende Schritte aufweist:
(a) Organisieren von (n × m)-vielen Bänken (**113**–**124**) des DRAM in n-viele Gruppen (**88**–**90**) von m-vielen Bänken pro Gruppe, wobei jede Gruppe durch den Gruppenauswählabschnitt (**167**, **169**, **171**) auswählbar ist und jede Bank in einer Gruppe Orte aufweist, die durch den In-Bank-Adressabschnitt (**176**) adressierbar sind;
(b) Richten jeder nächsten Speicheroperation auf die Gruppe, die durch den Gruppenauswählabschnitt identifiziert ist;
(c) in jeder Gruppe Auswählen jeder Bank darin in einer geordneten zyklischen Sequenz;
(d) in jeder Gruppe und für aufeinanderfolgende Speicheroperationen, die durch Schritt (b) auf diese Gruppe gerichtet sind, sequentielles Verschachteln (**161**) dieser aufeinanderfolgenden Speicheroperationen unter den m-vielen Bänken der Gruppe, wenn jede Bank gemäß der geordneten zyklischen Sequenz von Schritt (c) ausgewählt wird; und

(e) für jede ausgewählte Bank in einer Gruppe und für verschachtelte Speicheroperationen, die durch Schritt (d) auf diese Bänke gerichtet sind, Durchführen der nächsten aufeinanderfolgenden Speicheroperation von Schritt (d) an dem Ort der ausgewählten Bank durch den In-Bank-Adressabschnitt.

7. Ein Verfahren gemäß Anspruch 6, bei dem die Speicheroperation ein Schreiben ist, und das ferner den Schritt eines Erhaltens der zu schreibenden Informationen von Tests aufweist, die an einem adressierbaren Testobjekt (**14**) durchgeführt werden.

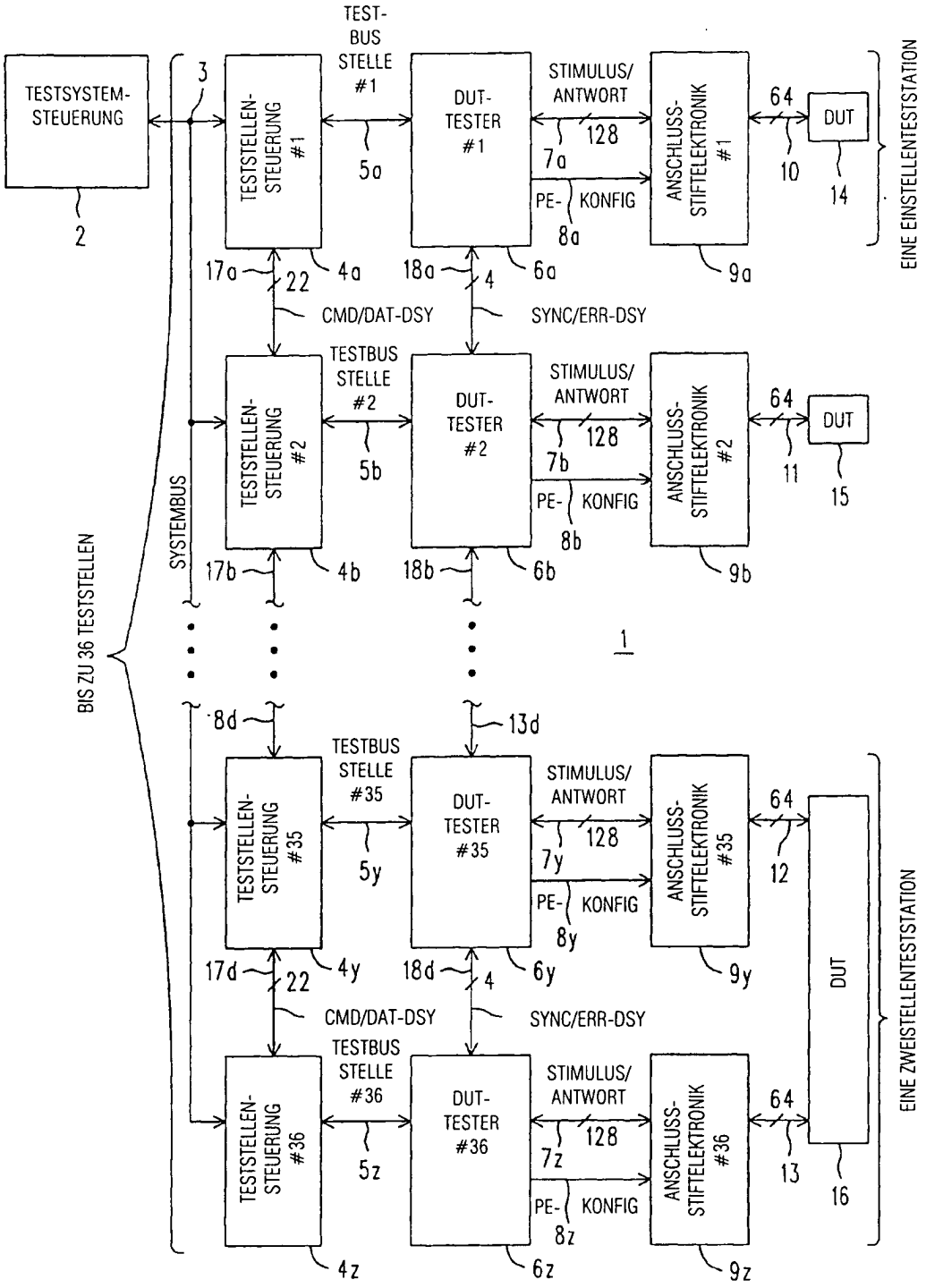
8. Ein Verfahren gemäß Anspruch 7, bei dem das Testobjekt (**14**) ein Speicher ist und die Bits in dem zu schreibenden Wort Kanäle in einem Speichertestsystem darstellen, und das ferner den Schritt eines Adressierens von adressierbaren Orten in den n-vielen Gruppen (**88**–**90**) von m-vielen Bänken mit Adressen, die von Adressen abgeleitet sind, die an das Testobjekt angelegt werden, aufweist.

9. Ein Verfahren gemäß Anspruch 6, bei dem die Speicheroperation ein Lesen ist, und das ferner die Schritte eines Lesens von einem adressierbaren Ort an einer gleichen Adresse in allen m-vielen Bänken der Gruppe, die durch den Gruppenauswählabschnitt identifiziert ist, um m-viele Wörter zu erzeugen, eines Zusammenführens (**146**) der m-vielen Wörter in ein Endwort, eines Nehmens des Endwortes als das Ergebnis der Speicheroperation und eines Schreibens des Endwortes in alle m-vielen Bänke an der gleichen Adresse der so identifizierten Gruppe aufweist.

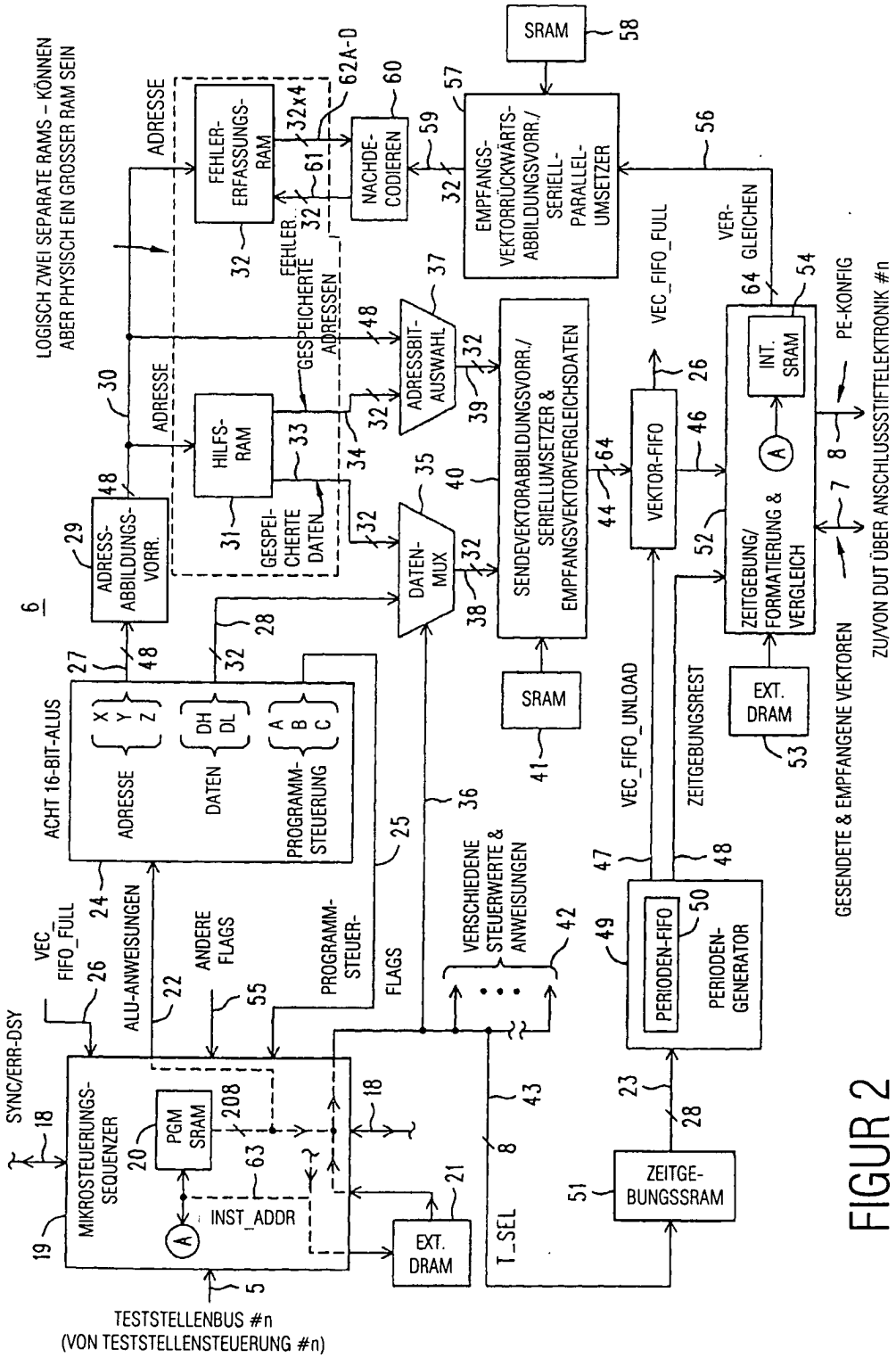
10. Ein Verfahren gemäß Anspruch 9, das ferner die Schritte eines Setzens eines Flags (**191**) nahe dem Zeitpunkt, wenn das Endwort in alle m-vielen Bänke geschrieben wird, und eines Löschens des Flags auf eine nachfolgende Instanz von Schritt (b) hin, bei der die nächste aufeinanderfolgende Speicheroperation eine Schreiboperation ist, aufweist.

Es folgen 10 Blatt Zeichnungen

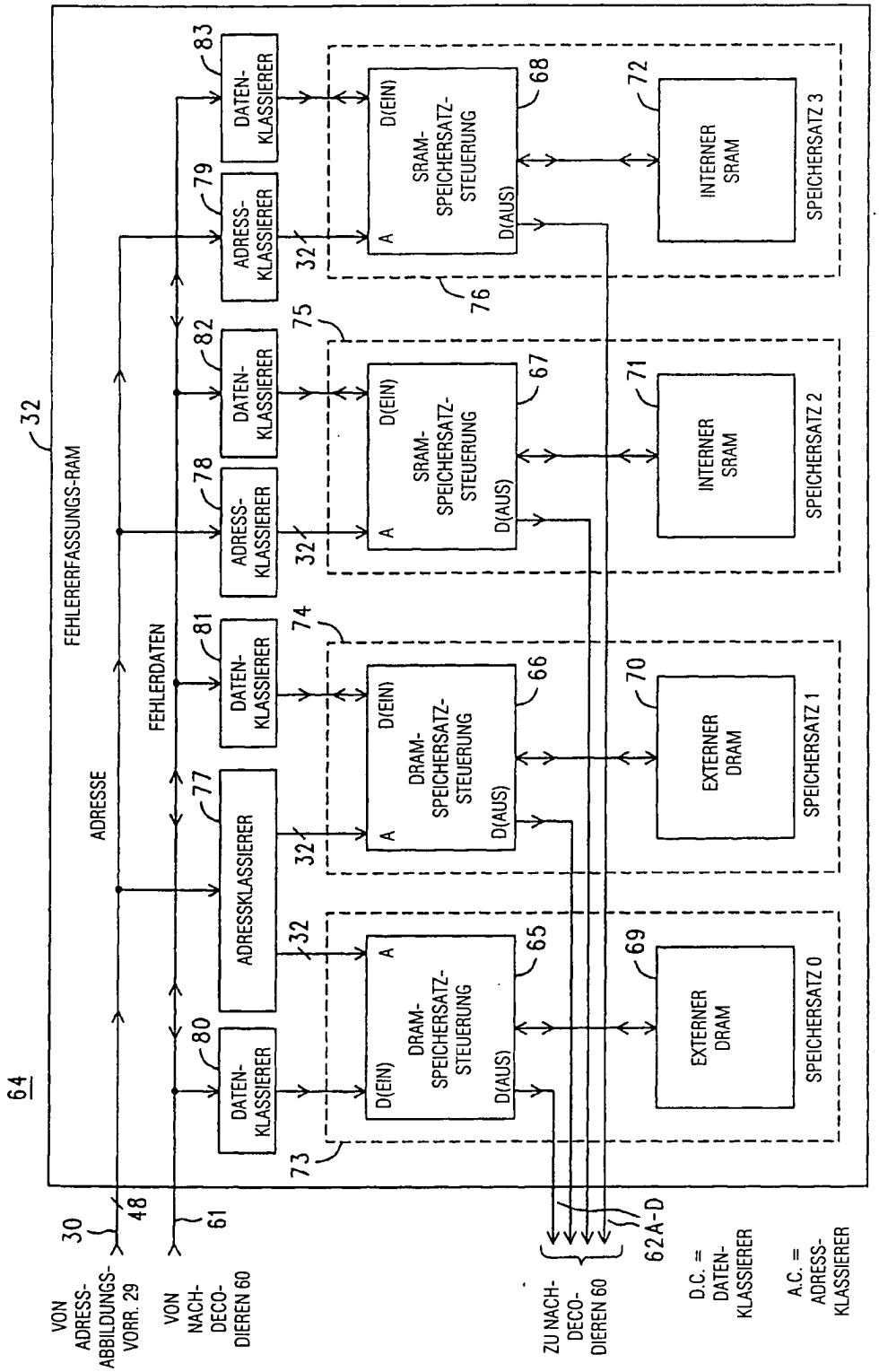
Anhängende Zeichnungen



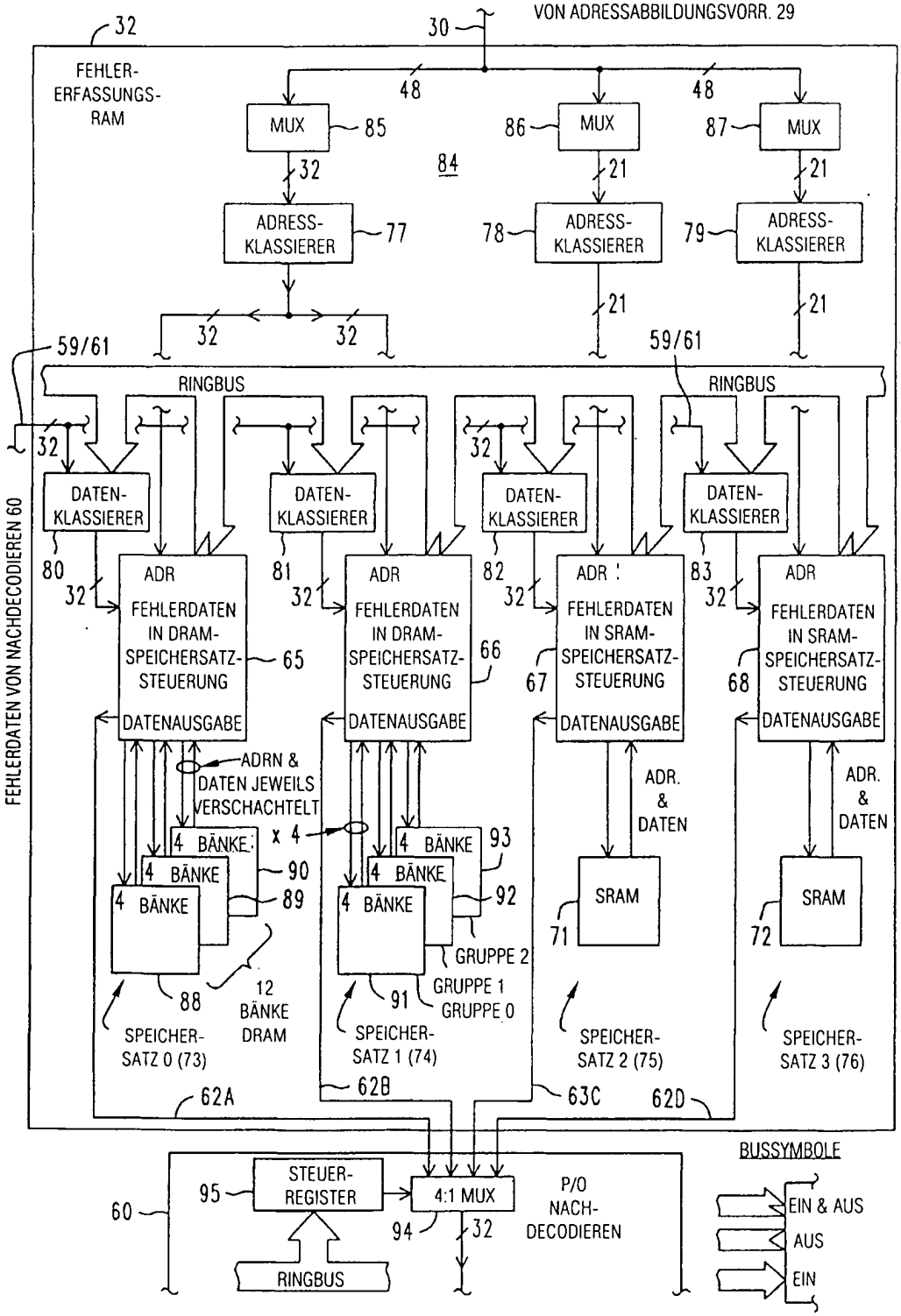
FIGUR 1


LOGISCH ZWEI SEPARATE RAMS – KÖNNEN ABER PHYSISCH EIN GROSSER RAM SEIN
6
ACHT 16-BIT-ALUS
ADRESSE
DATEN
PROGRAMM-STEUERUNG
X
Y
Z
DH
DL
A
B
C
ADRESSE
ADRESS-ABBILDUNGS-VORR.
HILFS-RAM
FEHLER-ERFASSUNGS-RAM
32x4
NACHDE-CODIEREN
GESPEICHERTE ADRESSEN
GESPEI-CHERTE DATEN
FEHLER
ADRESSBIT-AUSWAHL
DATEN-MUX
SENDEVEKTORABBILDUNGSVORR./ SERIELLUMSETZER & EMPFANGSVEKTORVERGLEICHSDATEN
EMPFANGS-VEKTORRÜCKWÄRTS-ABBILDUNGSVORR./ SERIELL-PARALLEL-UMSETZER
SRAM
VEKTOR-FIFO
VEC_FIFO_FULL
VER-GLEICHEN
INT. SRAM
PE-KONFIG
ZEITGEBUNG/ FORMATIERUNG & VERGLEICH
EXT. DRAM
SRAM
VEC_FIFO_UNLOAD
ZEITGEBUNGSREST
VERSCHIEDENE STEUERWERTE & ANWEISUNGEN
PERIODEN-FIFO
PERIODEN-GENERATOR
ZEITGE-BUNGSSRAM
EXT. DRAM
T_SEL
MIKROSTEUERUNGS-SEQUENZER
PGM SRAM
INST_ADDR
SYNC/ERR-DSY
VEC_ FIFO_FULL
ALU-ANWEISUNGEN
ANDERE FLAGS
PROGRAMM-STEUER-FLAGS
TESTSTELLENBUS #n
(VON TESTSTELLENSTEUERUNG #n)
GESENDETE & EMPFANGENE VEKTOREN
ZU/VON DUT ÜBER ANSCHLUSSSTIFTELEKTRONIK #n

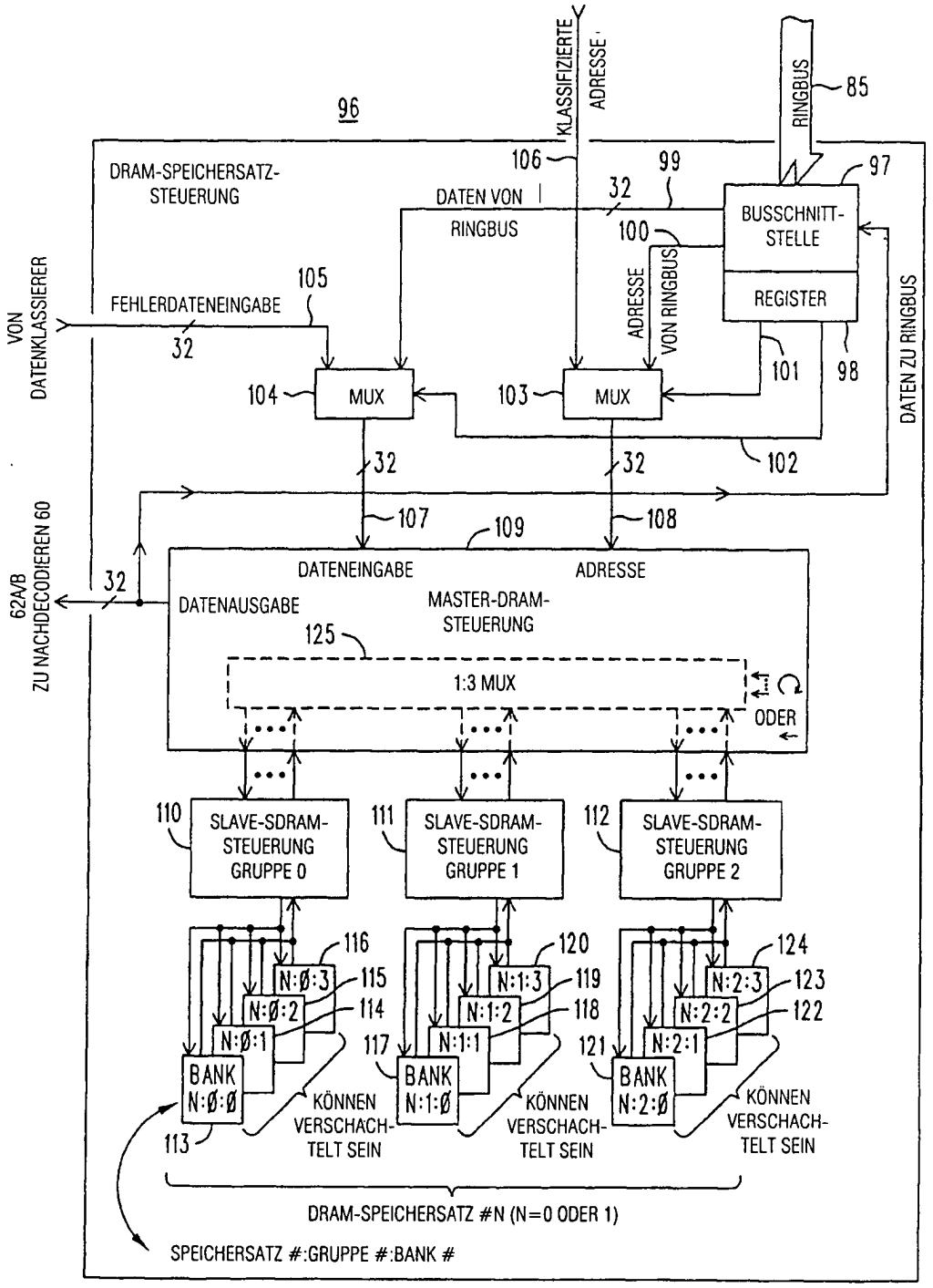
FIGUR 2


64
32
FEHLERERFASSUNGS-RAM
ADRESSE
FEHLERDATEN
30
48
61
VON ADRESS-ABBILDUNGS-VORR. 29
VON NACH-DECO-DIEREN 60
ADRESSKLASSIERER
DATEN-KLASSIERER
ADRESS-KLASSIERER
DRAM-SPEICHERSATZ-STEUERUNG
SRAM-SPEICHERSATZ-STEUERUNG
D(EIN)
D(AUS)
A
EXTERNER DRAM
INTERNER SRAM
SPEICHERSATZ 0
SPEICHERSATZ 1
SPEICHERSATZ 2
SPEICHERSATZ 3
62A-D
ZU NACH-DECO-DIEREN 60
D.C. = DATEN-KLASSIERER
A.C. = ADRESS-KLASSIERER

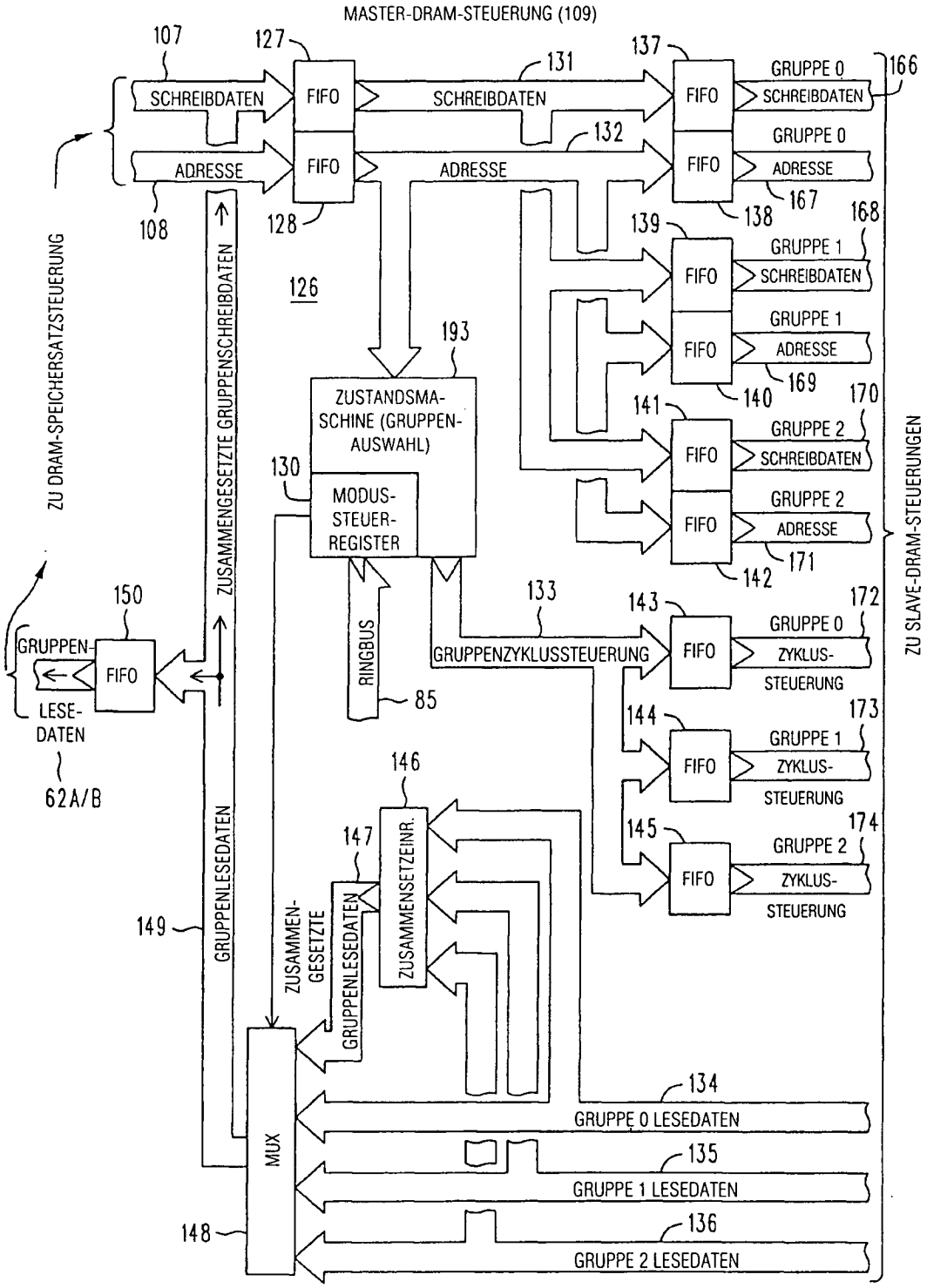
FIGUR 3


VON ADRESSABBILDUNGSVORR. 29
30
32
FEHLER-ERFASSUNGS-RAM
48
48
MUX
85
86
MUX
87
MUX
84
32
21
21
ADRESS-KLASSIERER
77
78
ADRESS-KLASSIERER
79
ADRESS-KLASSIERER
32
32
21
21
59/61
59/61
RINGBUS
RINGBUS
FEHLERDATEN VON NACHDECODIEREN 60
DATEN-KLASSIERER
DATEN-KLASSIERER
DATEN-KLASSIERER
DATEN-KLASSIERER
80
81
82
83
ADR
FEHLERDATEN IN DRAM-SPEICHERSATZ-STEUERUNG
65
ADR
FEHLERDATEN IN DRAM-SPEICHERSATZ-STEUERUNG
66
ADR
FEHLERDATEN IN SRAM-SPEICHERSATZ-STEUERUNG
67
ADR
FEHLERDATEN IN SRAM-SPEICHERSATZ-STEUERUNG
68
DATENAUSGABE
DATENAUSGABE
DATENAUSGABE
DATENAUSGABE
ADRN & DATEN JEWEILS VERSCHACHTELT
x 4
4 BÄNKE
4 BÄNKE
4 BÄNKE
4 BÄNKE
4 BÄNKE
90
89
93
92
ADR. & DATEN
ADR. & DATEN
SRAM
71
SRAM
72
GRUPPE 2
GRUPPE 1
GRUPPE 0
88
91
12 BÄNKE DRAM
SPEICHER-SATZ 0 (73)
SPEICHER-SATZ 1 (74)
SPEICHER-SATZ 2 (75)
SPEICHER-SATZ 3 (76)
62A
62B
63C
62D
BUSSYMBOLE
60
95
STEUER-REGISTER
4:1 MUX
P/O NACH-DECODIEREN
94
32
EIN & AUS
AUS
EIN
RINGBUS

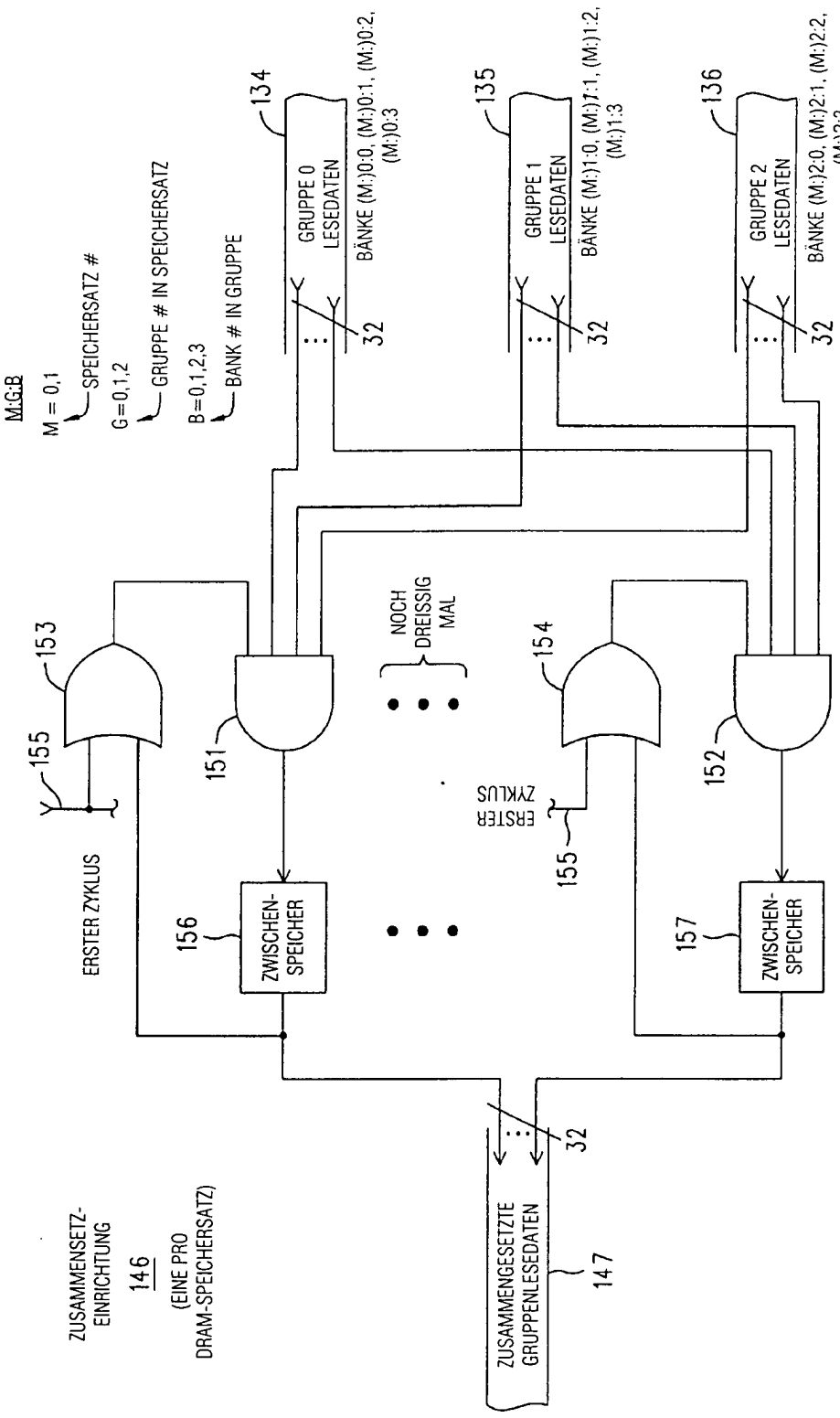
FIGUR 4


96
KLASSIFIZIERTE ADRESSE
RINGBUS
85
DRAM-SPEICHERSATZ-STEUERUNG
106
99
97
DATEN VON RINGBUS
32
BUSSCHNITT-STELLE
100
REGISTER
VON DATENKLASSIERER
105
FEHLERDATENEINGABE
32
ADRESSE VON RINGBUS
101
98
DATEN ZU RINGBUS
104
MUX
103
MUX
32
32
102
107
109
108
62A/B ZU NACHDECODIEREN 60
32
DATENEINGABE
ADRESSE
DATENAUSGABE
MASTER-DRAM-STEUERUNG
125
1:3 MUX
ODER
110
SLAVE-SDRAM-STEUERUNG GRUPPE 0
111
SLAVE-SDRAM-STEUERUNG GRUPPE 1
112
SLAVE-SDRAM-STEUERUNG GRUPPE 2
116
115
114
120
119
118
124
123
122
N:Ø:3
N:Ø:2
N:Ø:1
N:1:3
N:1:2
N:1:1
N:2:3
N:2:2
N:2:1
117
121
BANK N:Ø:Ø
BANK N:1:Ø
BANK N:2:Ø
113
KÖNNEN VERSCHACHTELT SEIN
KÖNNEN VERSCHACHTELT SEIN
KÖNNEN VERSCHACHTELT SEIN
DRAM-SPEICHERSATZ #N (N=0 ODER 1)
SPEICHERSATZ #:GRUPPE #:BANK #

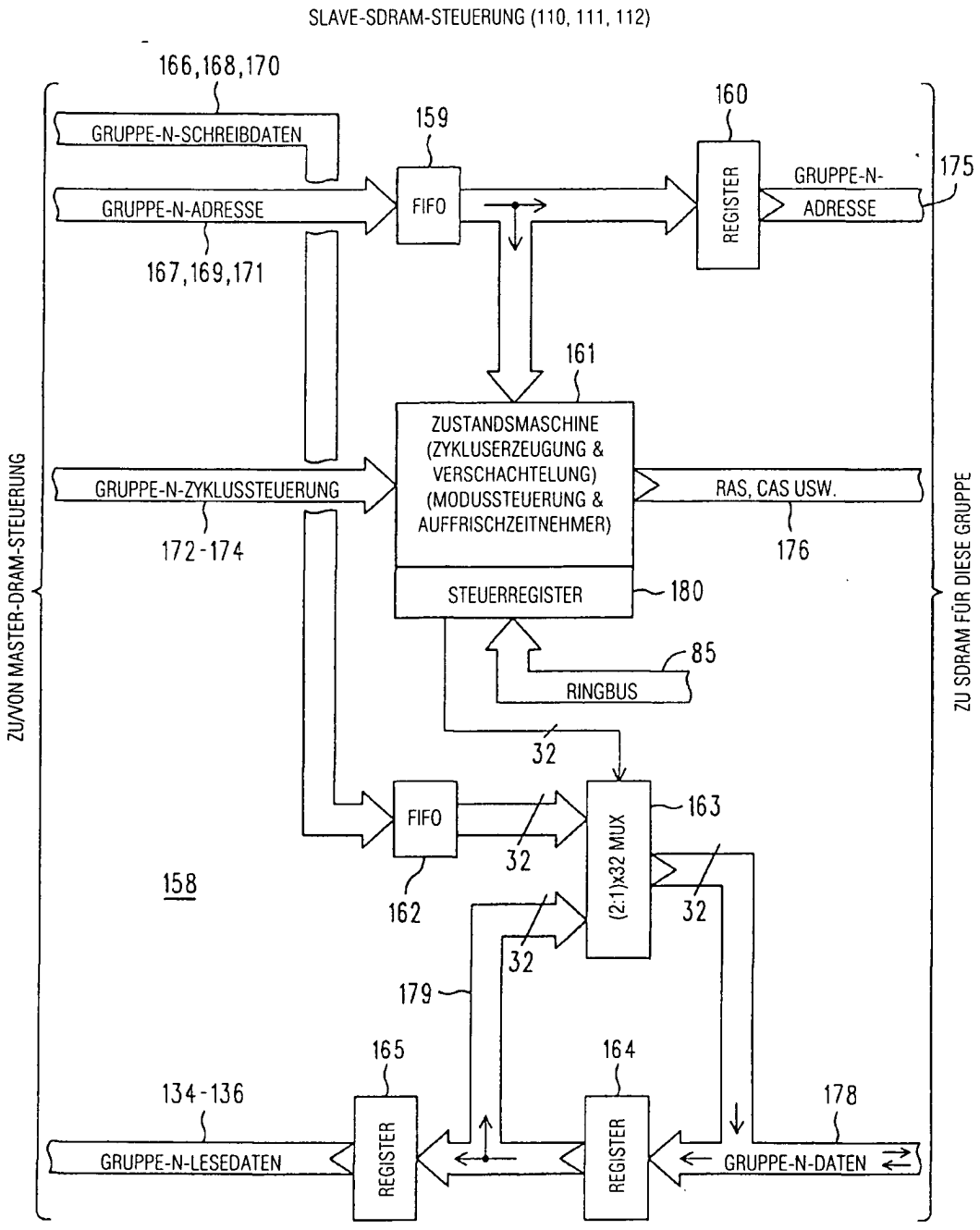
FIGUR 5


MASTER-DRAM-STEUERUNG (109)
ZU DRAM-SPEICHERSATZSTEUERUNG
ZU SLAVE-DRAM-STEUERUNGEN
SCHREIBDATEN
ADRESSE
FIFO
ZUSAMMENGESETZTE GRUPPENSCHREIBDATEN
ZUSTANDSMA-SCHINE (GRUPPEN-AUSWAHL)
MODUS-STEUER-REGISTER
RINGBUS
GRUPPENZYKLUSSTEUERUNG
GRUPPE 0
GRUPPE 1
GRUPPE 2
ZYKLUS-STEUERUNG
GRUPPEN-LESE-DATEN
62A/B
GRUPPENLESEDATEN
ZUSAMMEN-GESETZTE GRUPPENLESEDATEN
ZUSAMMENSETZEINR.
MUX
GRUPPE 0 LESEDATEN
GRUPPE 1 LESEDATEN
GRUPPE 2 LESEDATEN
107
108
126
127
128
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
166
167
168
169
170
171
172
173
174
193
85

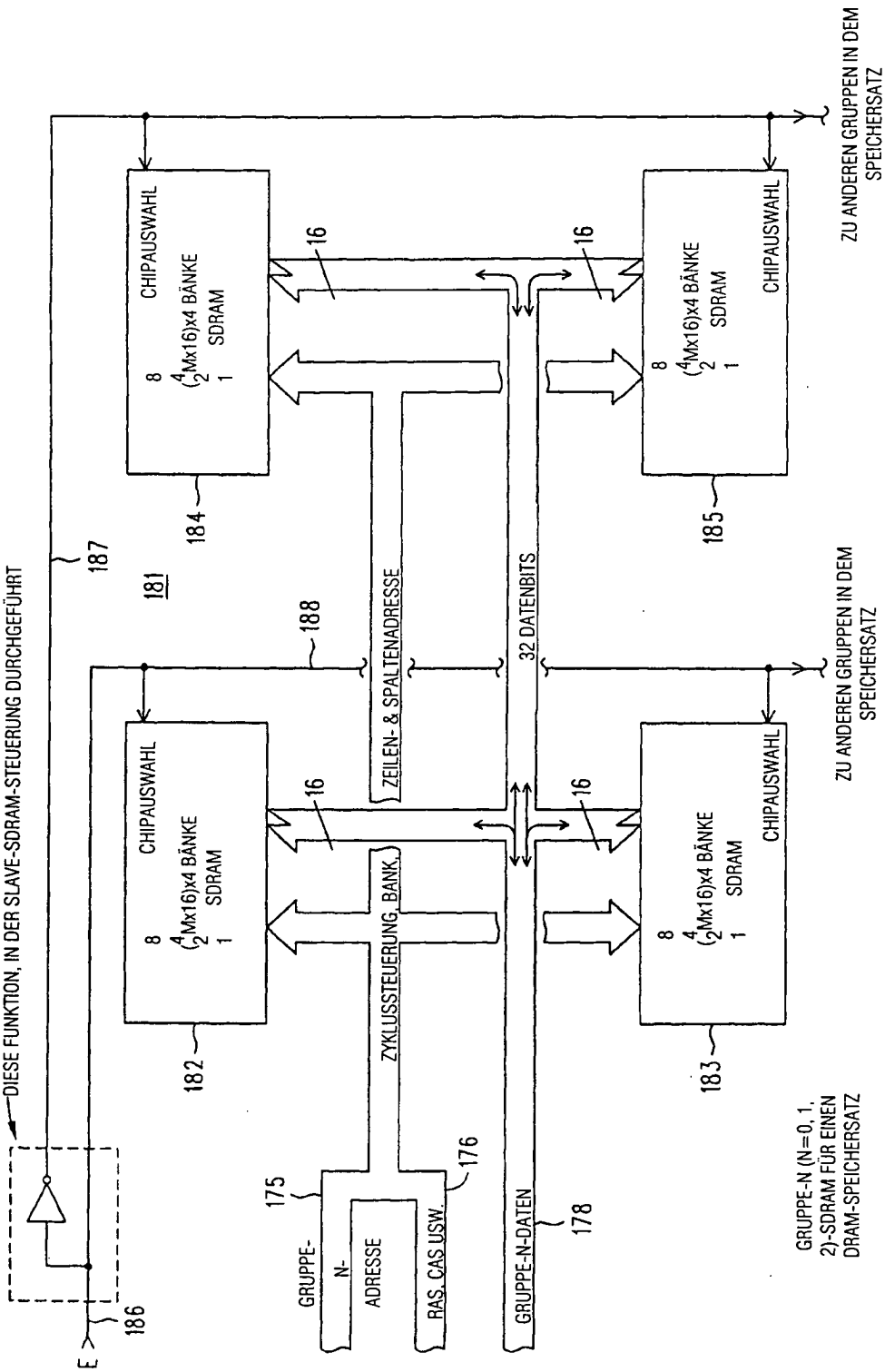
FIGUR 6


M:G:B
M = 0,1
SPEICHERSATZ #
G=0,1,2
GRUPPE # IN SPEICHERSATZ
B=0,1,2,3
BANK # IN GRUPPE
134
GRUPPE 0 LESEDATEN
BÄNKE (M:)0:0, (M:)0:1, (M:)0:2, (M:)0:3
135
GRUPPE 1 LESEDATEN
BÄNKE (M:)1:0, (M:)1:1, (M:)1:2, (M:)1:3
136
GRUPPE 2 LESEDATEN
BÄNKE (M:)2:0, (M:)2:1, (M:)2:2, (M:)2:3
32
153
154
155
151
152
NOCH DREISSIG MAL
ERSTER ZYKLUS
ERSTER ZYKLUS
156
157
ZWISCHEN-SPEICHER
ZWISCHEN-SPEICHER
ZUSAMMENSETZ-EINRICHTUNG
146
(EINE PRO DRAM-SPEICHERSATZ)
ZUSAMMENGESETZTE GRUPPENLESEDATEN
147

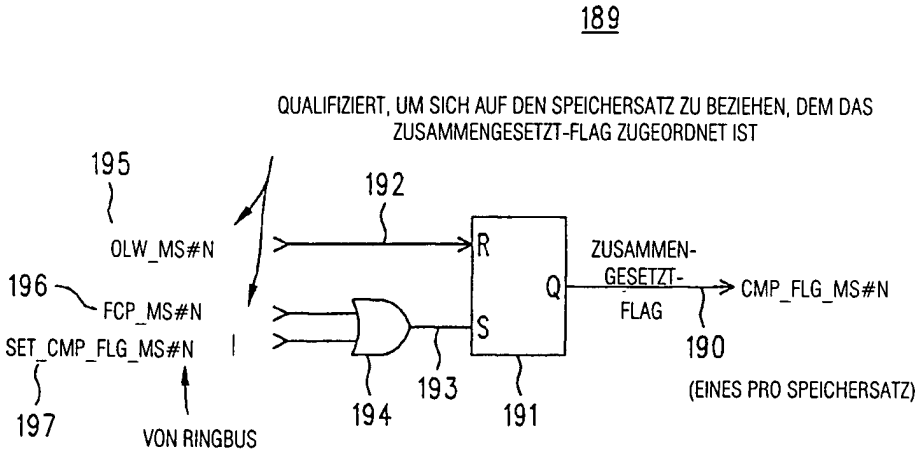
FIGUR 7


SLAVE-SDRAM-STEUERUNG (110, 111, 112)
166,168,170
GRUPPE-N-SCHREIBDATEN
GRUPPE-N-ADRESSE
167,169,171
159
FIFO
160
REGISTER
GRUPPE-N-ADRESSE
175
161
ZUSTANDSMASCHINE (ZYKLUSERZEUGUNG & VERSCHACHTELUNG) (MODUSSTEUERUNG & AUFFRISCHZEITNEHMER)
GRUPPE-N-ZYKLUSSTEUERUNG
172-174
RAS, CAS USW.
176
STEUERREGISTER
180
85
RINGBUS
32
FIFO
32
(2:1)x32 MUX
163
32
158
162
32
179
165
164
134-136
178
REGISTER
REGISTER
GRUPPE-N-LESEDATEN
GRUPPE-N-DATEN
ZU/VON MASTER-DRAM-STEUERUNG
ZU SDRAM FÜR DIESE GRUPPE

FIGUR 8


DIESE FUNKTION, IN DER SLAVE-SDRAM-STEUERUNG DURCHGEFÜHRT
E
186
187
181
188
CHIPAUSWAHL
8
4
2
1
(4Mx16)x4 BÄNKE SDRAM
184
185
182
183
16
ZEILEN- & SPALTENADRESSE
32 DATENBITS
ZYKLUSSTEUERUNG, BANK,
GRUPPE-N-ADRESSE
RAS, CAS USW
GRUPPE-N-DATEN
175
176
178
ZU ANDEREN GRUPPEN IN DEM SPEICHERSATZ
ZU ANDEREN GRUPPEN IN DEM SPEICHERSATZ
GRUPPE-N (N=0, 1, 2)-SDRAM FÜR EINEN DRAM-SPEICHERSATZ

FIGUR 9


189
QUALIFIZIERT, UM SICH AUF DEN SPEICHERSATZ ZU BEZIEHEN, DEM DAS ZUSAMMENGESETZT-FLAG ZUGEORDNET IST
195
OLW_MS#N
196
FCP_MS#N
SET_CMP_FLG_MS#N
197
VON RINGBUS
192
R
Q
S
ZUSAMMEN-GESETZT-FLAG
CMP_FLG_MS#N
190
(EINES PRO SPEICHERSATZ)
193
194
191

FIGUR 10