(19) Deutsches Patent- und Markenamt

(10) **DE 197 82 077 B4** 2011.04.07

(12) # Patentschrift

(21) Deutsches Aktenzeichen: **197 82 077.8**
(86) PCT-Aktenzeichen: **PCT/US97/13918**
(87) PCT-Veröffentlichungs-Nr.: **WO 1998/019241**
(86) PCT-Anmeldetag: **06.08.1997**
(87) PCT-Veröffentlichungstag: **07.05.1998**
(43) Veröffentlichungstag der PCT Anmeldung in deutscher Übersetzung: **23.09.1999**
(45) Veröffentlichungstag der Patenterteilung: **07.04.2011**

(51) Int Cl.[8]: ***G11C 29/42*** (2006.01)

Innerhalb von drei Monaten nach Veröffentlichung der Patenterteilung kann nach § 59 Patentgesetz gegen das Patent Einspruch erhoben werden. Der Einspruch ist schriftlich zu erklären und zu begründen. Innerhalb der Einspruchsfrist ist eine Einspruchsgebühr in Höhe von 200 Euro zu entrichten(§ 6 Patentkostengesetz in Verbindung mit der Anlage zu § 2 Abs. 1 Patentkostengesetz).

(30) Unionspriorität:
08/740,247 **25.10.1996** **US**

(73) Patentinhaber:
Intel Corporation, Santa Clara, Calif., US

(74) Vertreter:
BOEHMERT & BOEHMERT, 28209 Bremen

(72) Erfinder:
Leemann, Daniel H., Folsom, Calif., US

(56) Für die Beurteilung der Patentfähigkeit in Betracht gezogene Druckschriften:

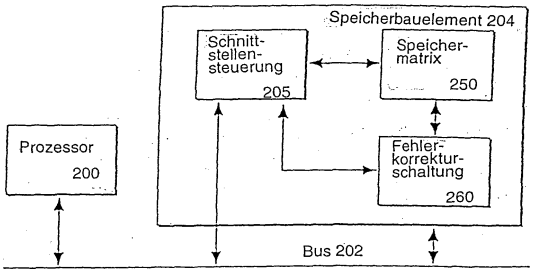
US	**56 21 682**	**A**
US	**54 50 363**	**A**
EP	**07 04 854**	**B1**
EP	**07 09 776**	**A1**
JP	**07-2 34 823**	**A**

FUJIWARA, E.

KITAKAMI, M.: "A Class of Error-Locating Codes for Byte-Organized Memory Systems", IEEE Transactions on Information Theory, Vol. 40, No. 6, November 1994, S. 1857-1865

(54) Bezeichnung: **Verfahren und Vorrichtung zum Korrigieren eines Mehrpegelzellenspeichers durch Verwendung fehlerlokalisierender Codes**

(57) Hauptanspruch: Verfahren zum Lesen von Daten aus einem Speicher mit Mehrpegel-Speicherzellen, die jeweils wenigstens drei Ladungszustände speichern können, wobei:
a) die Daten und ein zugehöriger Fehlerlokalisierungscode gelesen werden (502), wobei der Fehlerlokalisierungscode ein Identifizieren desjenigen Blocks oder derjenigen Blöcke der Daten, der bzw. die jeweils einer fehlerhafte Daten enthaltenden Mehrpegel-Speicherzelle zugeordnet sind, gestattet; und
b) dann, wenn der gelesene Fehlerlokalisierungscode anzeigt, dass ein gelesener, einem bestimmten Ladungspegel einer Speicherzelle entsprechender Block der Daten fehlerhaft ist, dieser Block der Daten bei der Ausgabe derart modifiziert wird, dass er dem nächsthöheren Ladungspegel der Speicherzelle entspricht.



Beschreibung

GEBIET DER ERFINDUNG

[0001] Die vorliegende Erfindung bezieht sich auf das Gebiet der Speicherbauelemente. Insbesondere bezieht sich die Erfindung auf das Korrigieren eines Fehlers in Daten, die aus einem Mehrpegelzellenspeicher gelesen wurden.

HINTERGRUND DER ERFINDUNG

[0002] Ein Mehrpegelzellenspeicher besteht aus Mehrpegelzellen, von denen jede in der Lage ist, mehrere Ladungszustände oder -pegel zu speichern. Jeder der Ladungszustände ist einem Bitmuster des Speicherelements zugeordnet.

[0003] Fig. 1 zeigt eine Darstellung einer bekannten Mehrpegelzelle, die vier Ladungszustände speichert: Die Pegel 0 – 3. Pegel 3 hält eine höhere Ladung als Pegel 2; Pegel 2 hält eine höhere Ladung als Pegel 1; Pegel 1 hält eine höhere Ladung als Pegel 0. Eine Referenzspannung trennt die verschiedenen Ladungszustände. V_{ref2} trennt Pegel 3 und Pegel 2. V_{ref1} trennt Pegel 2 und Pegel 1. V_{ref0} trennt Pegel 1 und Pegel 0.

[0004] Jedem Ladungszustand ist ein Speicherelementbitmuster zugeordnet. Bei einer Implementierung ist das Speicherelementbitmuster ”00” dem Pegel 3, das Speicherelementbitmuster ”10” dem Pegel 2, das Speicherelementbitmuster ”01” dem Pegel 1 und das Speicherelementbitmuster ”11” dem Pegel 0 zugeordnet.

[0005] Ein Mehrpegelzellenspeicher ist in der Lage, mehr als ein Datenbit auf der Grundlage der Anzahl der Ladungszustände, die er speichern kann, zu speichern. Beispielsweise kann ein Mehrpegelzellenspeicher, der vier Ladungszustände speichern kann, zwei Bits Daten speichern; ein Mehrpegelzellenspeicher, der acht Ladungszustände speichern kann, kann drei Bits Daten speichern; ein Mehrpegelzellenspeicher, der 16 Ladungszustände speichern kann, kann vier Bits Daten speichern. Für jeden der n-Bit-Mehrpegelzellenspeicher können verschiedene Speicherelementbitmuster so implementiert werden, daß sie jedem der unterschiedlichen Ladungszustände zugeordnet sind.

[0006] Die Anzahl der Ladungszustände, die in einer Mehrpegelzelle speicherbar sind, ist jedoch nicht auf Potenzen von zwei eingeschränkt. Beispielsweise speichert eine Mehrpegelzelle mit drei Ladungszuständen 1,5 Bits Daten. Wenn diese Mehrpegelzelle mit zusätzlicher Decodierlogik kombiniert und mit einer zweiten ähnlichen Mehrpegelzelle gekoppelt wird, werden drei Bits Daten als Ausgangssignal der Zwei-Zellen-Kombination zur Verfügung gestellt. Es sind verschiedene andere Mehrzellenkombinationen ebenfalls möglich.

[0007] Ein Beispiel eines Mehrpegelzellenspeichers ist in dem U.S.-Patent Nr. 5,450,363 mit dem Titel ”Grat” Coding for a Multilevel Cell Memory System” von Christopherson et al. beschrieben, das für die Anmelderin der vorliegenden Anmeldung ausgegeben worden ist. Das '363-Patent beschreibt eine Implementierung eines Mehrpegelzellenspeichers. Mehrpegelzellenspeicher können in dynamischen Speichern mit wahlfreiem Zugriff (DRAM) und verschiedenen Arten von Nur-Lese-Speichern (ROM), wie beispielsweise einem löschbaren und programmierbaren Nur-Lese-Speicher (EPROM), einem elektrisch löschbaren und programmierbaren Nur-Lese-Speicher (EEPROM) und einem Flash-EPROM, verwendet werden.

[0008] Fig. 2 zeigt eine bekannte Darstellung eines Prozessors **100** und eines Mehrpegelzellenspeichers **104**. Ein Prozessor **100** ist mit einem Bus **102** und dem Speicher **104** gekoppelt. Der Speicher **104** enthält eine Schnittstellensteuereinrichtung **105** und eine Mehrpegelzellenspeichermatrix **150**. Der Prozessor **100** ist über den Bus **102** mit der Schnittstellensteuereinrichtung **105** gekoppelt. Der Prozessor **100** ist darüber hinaus über den Bus **102** mit der Mehrpegelzellenspeichermatrix **150** gekoppelt. Die Schnittstellensteuereinrichtung **105** stellt die erforderlichen Operationen zur Verfügung, um die Mehrpegelzellenspeichermatrix **150** zu steuern.

[0009] Bei einem Ausführungsbeispiel sind die Schnittstellensteuereinrichtung **105** und die Mehrpegelzellenspeichermatrix **150** auf einem einzigen Chip einer integrierten Schaltung angeordnet.

Fehlercodes

[0010] Daten werden üblicherweise einer speziellen Speichermatrix zur Verfügung gestellt und aus der Speichermatrix in einer Einheit wiedergewonnen, die eine vorgegebene Anzahl von Bits aufweist und die im folgenden als ein ”Datenwort” bezeichnet wird. Wenn die Daten in der Speichermatrix gespeichert werden, wird manchmal ein Codierer verwendet, um Fehlercodes zu erzeugen. Die Fehlercodes können so gespeichert werden, daß sie mit ihrem zugehörigen Datenwort wiedergewonnen werden können. Wenn die Datenworte aus der Speichermatrix gewonnen werden, verwendet die Fehlerdecodierschaltung die den Datenworten zugeordneten, zuvor gesicherten Fehlercodes zum Erfassen, Lokalisieren und/oder Korrigieren von Fehlern, die in den Datenworten, wie sie aus der Speichermatrix gelesen wurden, gefunden wurden, wie es im Stand der Technik gut bekannt ist.

[0011] Es gibt unterschiedliche Arten von Fehlercodes, einschließlich Fehlererfassungscodes, Fehlerlokalisierungscodes und Fehlerkorrekturcodes. Fehlererfassungscodes erfassen, ob ein Fehler aufgetreten ist, können aber den Fehler nicht korrigieren. Fehlerlokalisierungscodes erfassen einen Fehler und sind in der Lage, den Fehler in einem Block von Bits zu lokalisieren, können aber nicht die genaue Position des fehlerhaften Bits innerhalb des Blocks von Bits bestimmen. Fehlerkorrekturcodes erfassen Einzelbitfehler in einem Datenwort und sind in der Lage, die Fehler zu korrigieren. Eine Art eines Fehlerkorrekturcodes, Hamming-Code genannt, ist beispielsweise in der Lage, Einzelbitfehler zu korrigieren, ist darüber hinaus in der Lage, Doppelbitfehler in einem Datenwort zu erfassen (aber nicht zu korrigieren).

[0012] Eine Gruppe von fehlerlokalisierenden Codes ist in ”A Class of Error-Locating Codes for Byte-Organized Memory Systems” von E. Fujiwara und M. Kitakami (”Fujiwara”), IEEE Transactions on Information Theroy, Band 40, Nr. 6, November 1994, beschrieben. Fujiwara beschreibt fehlerlokalisierende Codes, die in einem fehlertoleranten Speichersystem verwendet werden. Wenn in einem Block des Speichers ein Fehler erfaßt wird, wird die fehlerhafte Speicherkarte lokalisiert und dann in eine Reservekarte umgeschaltet.

[0013] Fehlerlokalisierende Codes sind darüber hinaus in ”Error-Locating Codes – A New Concept in Error Control” von J. K. Wolf und B. Elspas, IEEE Transactions on Information Theory, April 1963, beschrieben.

[0014] In EP 0 704 854 B1 wird ein Speichergerät mit Fehlerdetektion und -korrektur und darüber hinaus ein Verfahren zum Löschen eines Speichergeräts offenbart. EP 0 709 776 A1 beschreibt ein Verfahren zum Feststellen und Korrigieren von Fehlern innerhalb von Multilevel-Speichern. In US 5,450,363 ist ein Speichersystem offenbart, welches eine Vielzahl von Speicherzellen aufweist. Außerdem sind dort eine Sensorschaltung und eine Umwandlungsschaltung zur Umwandlung eines Binärcodes in einen Graustufencode, so dass sich benachbarte Schwellenwerte lediglich um ein Bit unterscheiden, offenbart. In US 5,621,682 wird ein Speichersystem offenbart, bei dem ebenfalls Fehler auf der Basis von Binärcodes entsprechend den Prüf- und Informationsdaten verbessert wird. JP 07234823 A offenbart ein Speichersystem, welches in der Lage ist, Fehler in Daten zu erkennen und zu korrigieren, wozu nur wenig Redundanz erforderlich ist. E. Fujiwara: „A Class of Error-Locating Codes for Byte-Organized Memory Systems” beschreibt einen Fehlerfeststellungscode, der insbesondere für Multilevelspeicher genutzt werden kann.

ZUSAMMENFASSENDE DARSTELLUNG DER ERFINDUNG

[0015] Ein Verfahren und eine Vorrichtung korrigieren Daten, die aus einer Mehrpegelzelle eines Speichers gelesen wurden. Die Mehrpegelzelle ist in der Lage, drei oder mehr Ladungszustände zu speichern. Es wird bestimmt, daß ein erster Ladungszustand, der aus der Mehrpegelzelle gelesen worden ist, fehlerhaft ist. Es wird ein Ausgangssignal zur Verfügung gestellt, das dem entspricht, was zur Verfügung gestellt würde, wenn die Mehrpegelzelle einen zweiten Ladungszustand halten würde, wobei der zweite Ladungszustand mehr Ladung als der erste Ladungszustand aufweist.

[0016] Andere Merkmale und Vorteile der vorliegenden Erfindung sind aus den begleitenden Zeichnungen und aus der nun folgenden detaillierten Beschreibung ersichtlich.

KURZBESCHREIBUNG DER ZEICHNUNGEN

[0017] Fig. 1 zeigt eine Darstellung einer bekannten Mehrpegelzelle, die vier Ladungszustände speichert.

[0018] Fig. 2 zeigt eine bekannte Darstellung eines Prozessors und eines Mehrpegelzellenspeichers.

[0019] Fig. 3 zeigt eine Darstellung eines Prozessors und eines Mehrpegelzellenspeichers, der die vorliegende Erfindung benutzt.

[0020] Fig. 4 zeigt ein Ausführungsbeispiel einer Fehlerkorrekturschaltung, die zum Programmieren (d. h. Schreiben) von Daten in die Speichermatrix verwendet wird.

[0021] Fig. 5 zeigt ein Ausführungsbeispiel einer Fehlerkorrekturschaltung, die zum Lesen von Daten aus der Speichermatrix **250** verwendet wird.

[0022] Fig. 6 zeigt ein Ausführungsbeispiel einer Fehlerkorrekturschaltung, die einen Datenpuffer und einen Fehlercodierer und -decodierer (EED) enthält und die sowohl zum Schreiben in die Speichermatrix als auch zum Lesen aus der Speichermatrix verwendet werden kann.

[0023] Fig. 7 zeigt ein weiteres Ausführungsbeispiel einer Fehlerkorrekturschaltung zum Schreiben von Daten in die Speichermatrix.

[0024] Fig. 8 zeigt ein weiteres Ausführungsbeispiel einer Fehlerkorrekturschaltung zum Lesen von Daten aus der Speichermatrix.

[0025] Fig. 9 zeigt ein Ablaufdiagramm des Prozesses zum Erzeugen von Fehlerkorrekturcodes, wenn

ein Datenwort in den Mehrpegelzellenspeicher geschrieben wird.

[0026] Fig. 10 zeigt eine repräsentative Zeichnung eines Datenworts und seines zugeordneten Fehlerlokalisierungscodes.

[0027] Fig. 11 zeigt ein Beispiel eines 64-Bit-Datenworts zusammen mit dem ihm zugeordneten 8-Bit-Fehlerlokalisierungscode.

[0028] Fig. 12 zeigt ein Ablaufdiagramm des Prozesses zum Korrigieren von Fehlern, die durch die Fehlerlokalisierungscodes beim Lesen eines Datenworts aus einem Mehrpegelzellenspeicher erfaßt worden sind.

DETAILLIERTE BESCHREIBUNG

[0029] Es werden ein Verfahren und eine Vorrichtung zum Korrigieren von Fehlern in einem Mehrpegelzellenspeicher beschrieben. Das Verfahren verwendet Fehlerlokalisierungscodes, um Fehler in einem aus einer oder mehrereren Mehrpegelzellen gelesenen Datenwort zu korrigieren. Die Fehlerlokalisierungscodes zeigen einen Block von Bits innerhalb des Datenworts an, der einen Fehler enthält, geben aber nicht an, welches Bit fehlerhaft ist. Da der Block von Bits einer oder mehreren Mehrpegelzellen entspricht, identifizieren die Fehlerlokalisierungscodes eine fehlerhafte Mehrpegelzelle. Die Erfindung benutzt die Identifikation der fehlerhaften Mehrpegelzelle und kombiniert diese mit der Erkenntnis, daß ein Hauptausfallmechanismus einer Mehrpegelzelle der Ladungsverlust ist. Sobald eine bestimmte Mehrpegelzelle als inkorrekt gelesen identifiziert worden ist, kann der korrekte Zustand der Mehrpegelzelle als ein Ladungszustand vorhergesagt werden, der mehr Ladung aufweist, als der Ladungszustand, der aus der Mehrpegelzelle gelesen worden ist.

[0030] Während die Fehlerlokalisierungscodes bei Fujiwara (genannt im Hintergrund) verwendet wurden, um einen Speicherblock zu erfassen, der fehlerhaft war, so daß die den Speicherblock speichernde Speicherkarte gesperrt werden konnte, zieht die vorliegende Erfindung Nutzen aus dem Ausfallmechanismus der Mehrpegelzellen, um den Fehler zu korrigieren. Da der normale Ausfallmodus von Mehrpegelzellen auf der Grundlage des Ladungsverlustes nur eine Richtung hat und nicht zufällig ist, wie es bei Fujiwara beschrieben wurde, hat die vorliegende Erfindung entdeckt, daß die Vorhersage des richtigen Zustandes der fehlerhaften Mehrpegelzelle möglich ist.

[0031] Die Korrektur einer fehlerhaften Mehrpegelzelle wird durch Modifizieren der aus der fehlerhaften Mehrpegelzelle gelesenen Daten erreicht, so daß der Wert wiedergegeben wird, den sie hätte, wenn die fehlerhafte Mehrpegelzelle sich auf dem nächsthöheren Ladungszustand befinden würde. Die Fehlerlokalisierungscodes werden verwendet, um zu überprüfen, ob die Modifikation den Fehler korrigiert. Sofern die modifizierten Daten aus der fehlerhaften Mehrpegelzelle immer noch unrichtig sind, dann kann dieser Modifikationsprozeß wiederholt werden. Wenn mehrere Mehrpegelzellen fehlerhaft sind, so können die Daten aus einer Mehrpegelzelle konstant gehalten werden, während die anderen modifiziert werden oder umgekehrt, bis der Fehler korrigiert worden ist.

[0032] In der gesamten Beschreibung der Erfindung werden die Fehlerlokalisierungscodes, wie sie bei Fujiwara beschrieben worden sind, als ein Beispiel der Implementierung genannt. Jedoch können auch andere Algorithmen oder andere Arten von Implementierungen verwendet werden, um eine fehlerhafte Mehrpegelzelle zu identifizieren.

[0033] Die Fig. 3 bis Fig. 9 zeigen beispielhafte Implementierungen der Erfindung. Fig. 3 zeigt eine Darstellung eines Prozessors **200** und eines Mehrpegelzellenspeichers **204**. Der Prozessor **200** ist mit einem Bus **202** gekoppelt. Der Mehrpegelzellenspeicher **204** weist eine Schnittstellensteuereinrichtung **205**, eine Mehrpegelzellenspeichermatrix **250** und eine Fehlerkorrekturschaltung **260** auf. Die Schnittstellensteuereinrichtung **205** ist so eingekoppelt, daß sie die erforderlichen Operationen zum Steuern der Mehrpegelzellenspeichermatrix **250** und der Fehlerkorrekturschaltung **260** zur Verfügung stellt. Sowohl die Schnittstellensteuereinrichtung **205** als auch die Fehlerkorrekturschaltung **260** sind mit dem Bus **202** gekoppelt. Die Mehrpegelzellenspeichermatrix **250** ist mit der Fehlerkorrekturschaltung **260** gekoppelt. Der Prozessor **200** schreibt Daten zu dem Speicher **204** und liest Daten aus dem Speicher **204**.

[0034] Bei einem Ausführungsbeispiel sind die Schnittstellensteuereinrichtung **205**, die Mehrpegelzellenspeichermatrix **250** und die Fehlerkorrekturschaltung **260** auf einem einzigen Chip einer integrierten Schaltung angeordnet. Bei einem anderen Ausführungsbeispiel ist die Fehlerkorrekturschaltung **260** von der Speichermatrix **250** und der Schnittstellensteuereinrichtung **205** getrennt. Beispielsweise könnte sich die Fehlerkorrekturschaltung auf einem separaten Chip aufhalten.

[0035] Fig. 4 zeigt ein Ausführungsbeispiel einer Fehlerkorrekturschaltung **260**, die zum Programmieren (d. h. Schreiben) von Daten in die Speichermatrix **250** verwendet wird. Die Fehlerkorrekturschaltung **260** enthält einen Datenpuffer **302**, einen Fehlerpuffer **304** und einen Fehlercodierer **300**.

[0036] Der Bus **202** ist über den Datenpufferbus **308** mit dem Datenpuffer **302** gekoppelt. Der Datenpuffer **302** ist über den Speicherbus **312** mit der Speichermatrix **250** gekoppelt. Der Datenpuffer ist über den

Bus **314** mit dem Fehlercodierer **300** gekoppelt. Der Fehlercodierer **300** ist über den Bus **316** mit dem Fehlerpuffer **304** gekoppelt, und der Fehlerpuffer **304** ist über den Fehlerbus **318** mit der Speichermatrix **250** gekoppelt.

[0037] Daten werden über den Datenpufferbus **308** in den Datenpuffer **302** geschrieben. Jede Übertragung von Daten über dem Datenpufferbus **308** ist üblicherweise ein Bruchteil der Breite des Datenpuffers **302**. Beispielsweise kann der Datenpufferbus **308** 8 Bits oder 16 Bits breit sein und der Datenpuffer **302** kann 64 Bits breit sein.

[0038] Wenn die gesamte Breite des Datenpuffers **302** mit Daten aus dem Bus **202** vollgeschrieben worden ist, wird das 64-Bit-Datenwort in dem Datenpuffer **302** an die Speichermatrix **250** über den Speicherbus **312** übertragen. Der Fehlercodierer **300** erzeugt unter Verwendung der über den Bus **314** aus dem Datenpuffer **302** eingegebenen Daten einen Fehlerkorrekturcode, der dem Fehlerpuffer **304** zur Verfügung gestellt wird. Der Fehlerpuffer **304** liefert den Fehlerkorrekturcode über den Bus **318** an die Speichermatrix **250**.

[0039] Bei einem Ausführungsbeispiel der Erfindung erzeugt der Fehlercodierer **300** Fehlerlokalisierungscodes, welche eine Klasse von Fehlerkorrekturcodes sind, wie beispielsweise die in "A Class of Error-Locating Codes for Byte-Organized Memory Systems" von E. Fujiwara und M. Kitakami, IEEE Transactions on Information Theory, Band 40, Nr. 6, November 1994, beschriebenen. Fehlerlokalisierungscodes erfassen einen oder mehrere fehlerhafte Blöcke von Daten innerhalb eines Datenworts, zeigen aber nicht genau die Bits des Blocks an, die fehlerhaft sind. Darüber hinaus können Fehlerlokalisierungscodes Einzelbitfehler erfassen.

[0040] Bei einer Implementierung der vorliegenden Erfindung werden das von dem Datenpuffer **302** zur Verfügung gestellte Datenwort und der von dem Fehlerpuffer **304** zur Verfügung gestellte Fehlercode zusammenhängend in der Speichermatrix **250** gespeichert. Bei einer anderen Implementierung wird das Datenwort separat von dem Fehlercode innerhalb der Speichermatrix **250** gespeichert. Bei noch einer anderen Implementierung wird der Fehlercode in einem von seinem zugehörigen Datenwort getrennten Speicherbereich gespeichert.

[0041] Fig. 5 zeigt ein Ausführungsbeispiel einer Fehlerkorrekturschaltung **260**, die zum Lesen von Daten aus der Speichermatrix **250** verwendet wird. Die Fehlerkorrekturschaltung **260** enthält einen Datenpuffer **320**, einen Fehlerpuffer **322** und einen Fehlercodierer und -decodierer (EED) **324**.

[0042] Während einer Leseoperation wird ein Datenwort aus der Speichermatrix **250** über den Bus **330** dem Datenpuffer **320** zur Verfügung gestellt. In ähnlicher Weise wird der dem Datenwort zugeordnete Fehlerkorrekturcode dem Zählerpuffer **322** zur Verfügung gestellt. Bei einem Ausführungsbeispiel wird der Fehlerkorrekturcode aus der Speichermatrix **250** über den Bus **322** geliefert.

[0043] Der Datenpuffer **320** ist so eingekoppelt, daß er seine Daten dem EED **324** über den Bus **334** zur Verfügung stellt. Der Fehlerpuffer **322** ist so eingekoppelt, daß er seinen Fehlerkorrekturcode dem EED **324** über den Bus **336** zur Verfügung stellt. Bei Empfang der Eingangssignale von sowohl dem Datenpuffer **320** als auch dem Fehlerpuffer **322** bestimmt der EED **324**, ob es irgendwelche fehlerhaften Bits in dem Datenwort gibt, das aus der Speichermatrix gelesen worden ist. Wenn es irgendwelche fehlerhaften Bits in dem Datenwort gibt, dann legt der EED **324** die geeigneten Signale **344** an, um die fehlerhaften Bits zu korrigieren, wie es unter Bezugnahme auf die Fig. 11 und Fig. 12 beschrieben werden wird. Die Signale **340** können beliebige der Datenbits des Datenpuffers **320** einzeln ändern. Es kann erforderlich sein, daß der EED **324** verschiedene Datenwerte in dem Datenpuffer **320** iterativ testet, bevor er einen endgültigen Datenwert bestimmt, der keine fehlerhaften Bits mehr aufweist. Der Datenpuffer **320** ist über den Bus **338** mit dem Bus **202** gekoppelt.

[0044] Fig. 6 zeigt ein Ausführungsbeispiel einer Fehlerkorrekturschaltung **260** mit einem Datenpuffer **350** und einem Fehlercodierer und -decodierer (EED) **354**, die sowohl zum Schreiben in die Speichermatrix **250** als auch zum Lesen aus der Speichermatrix **250** verwendet werden kann. Der EED **354** umfaßt einen Codierer zum Erzeugen des fehlerlokalisierenden Codes aus dem Datenwort und einen Decodierer, welcher zum Korrigieren des Datenworts verwendet wird, wenn es aus der Speichermatrix gelesen wird.

[0045] Fig. 7 zeigt ein weiteres Ausführungsbeispiel einer Fehlerkorrekturschaltung **260** zum Schreiben von Daten in die Speichermatrix **250**. Die Fehlerkorrekturschaltung **260** gemäß Fig. 7 jedoch enthält keinen Datenpuffer oder Fehlerpuffer wie die in Fig. 4. Das Datenwort wird von dem Bus **202** zu der Speichermatrix **250** über den Bus **380** zur Verfügung gestellt. Das Datenwort wird darüber hinaus als Eingabe dem Codierer **384** über den Bus **382** zur Verfügung gestellt. Der Codierer **384** erzeugt fehlerlokalisierende Codes und liefert diese der Speichermatrix **250** über den Bus **386**. Dieses Ausführungsbeispiel kann benutzt werden, wenn der Codierer **384** ausreichend Zeit hat, um die Fehlerkorrekturcodes zu erzeugen, bevor die Eingabedaten an dem Codierer **384** geändert werden.

[0046] Fig. 8 zeigt ein weiteres Ausführungsbeispiel der Fehlerkorrekturschaltung **260** zum Lesen von Daten aus der Speichermatrix **250**. Die Fehlerschaltung **260** gemäß Fig. 8 enthält keinen Datenpuffer oder Fehlerpuffer wie die in Fig. 5.

[0047] Ein Datenwort wird von der Speichermatrix **250** zu der Fehlerkorrekturschaltung **260** über die Signalleitungen **388a**–z geliefert. Das Datenwort wird dem Codierer **390** über die Signalleitungen **389a**–z zur Verfügung gestellt. Der Codierer codiert das Datenwort erneut und stellt die Fehlerkorrekturcodes auf den Signalleitungen **393a**–m zur Verfügung. Die erneut codierten Fehlerkorrekturcodes werden mit den Fehlerkorrekturcodes verglichen, die zuvor gesichert worden sind und die auf den Signalleitungen **392a**–m zur Verfügung gestellt werden. Beispielsweise kann der Vergleich mit einer XOR-Schaltung durchgeführt werden. Das Ausgangssignal des Vergleichs wird einem Decodierer **394** zur Verfügung gestellt. Der Decodierer liefert die Signale **396a**–z, welche die ursprünglichen Signale **388a**–z modifizieren können, um in dem Datenwort erfaßte Fehler zu korrigieren. Das korrigierte Datenwort wird dem Bus **202** über die Signalleitungen **398** geliefert.

[0048] Bei einem alternativen Ausführungsbeispiel können die Programmierschaltung gemäß Fig. 7 und die Leseschaltung gemäß Fig. 8 derart kombiniert werden, daß sie sich einen Codierer teilen.

[0049] Fig. 9 zeigt ein Ablaufdiagramm, das die Schritte zeigt, die beim Erzeugen von Fehlerkorrekturcodes unternommen werden, wenn ein Datenwort in den Mehrpegelzellenspeicher geschrieben wird. Die folgende Beschreibung wird unter Bezugnahme auf die Programmierschaltung gemäß Fig. 4 gegeben. Das Ablaufdiagramm startet im Block **400**, von welchem sich die Operation zum Block **402** fortsetzt. Im Block **402** wird der Datenpuffer **302** mit Daten gefüllt, die ein Datenwort bilden. Das Schreiben in den Datenpuffer **302** kann mehrere Buszyklen beanspruchen, da der Datenbuspuffer **308** üblicherweise nicht so breit ist wie der Datenpuffer **302**.

[0050] Die Operation setzt sich im Block **404** fort, bei welchem die Bits des Datenworts durch den Codierer **300** bearbeitet werden, um einen fehlerlokalisierenden Code zu erzeugen. Beim Block **406** werden das Datenwort und der fehlerlokalisierende Code in dem Mehrpegelzellenspeicher **204** gespeichert. Die Operation endet am Block **410**.

[0051] Fig. 10 zeigt eine repräsentative Zeichnung eines Datenworts **402** und seines zugehörigen fehlerlokalisierenden Codes **430**. Der fehlerlokalisierende Code **430** wird so decodiert, daß er auf eine fehlerhafte Gruppe von Bits, die auch ”Byte” genannt wird, in dem Datenwort **420** verweist.

[0052] Bei einem Ausführungsbeispiel entspricht jedes Byte exakt einer einzelnen Mehrpegelzelle, d. h., jedes Byte besteht aus derjenigen Anzahl von Bits, die in einer bestimmten Mehrpegelzelle gespeichert sind und jedes Byte ist an den Begrenzungen der Mehrpegelzelle ausgerichtet. In diesem Fall identifiziert der fehlerlokalisierende Code, wenn er ein fehlerhaftes Byte identifiziert, außerdem eine fehlerhafte Mehrpegelzelle. Da der Ausfallmechanismus der Mehrpegelzelle auf der Grundlage des Ladungsverlustes vorhersagbar ist, wird der richtige Zustand der fehlerhaften Mehrpegelzelle als Ladungszustand vorhergesagt, der mehr Ladung aufweist, als der aus der fehlerhaften Mehrpegelzelle gelesene Ladungszustand.

[0053] Bei einem anderen Ausführungsbeispiel überlappt das fehlerhafte Byte zwei Mehrpegelzellen, d. h., ein Teil des Bytes ist in einer Mehrpegelzelle und ein weiterer Teil des Bytes in einer weiteren Mehrpegelzelle gespeichert. Dies kann auftreten, wenn die Anzahl der Bits in dem Byte nicht gleich der Anzahl der in einer Mehrpegelzelle gespeicherten Bits ist oder wenn das Byte in Bezug auf die Mehrpegelzellenbegrenzungen nicht ausgerichtet ist. Wenn ein Überlappen von zwei Mehrpegelzellen auftritt, kann eine oder können beide dem fehlerhaften Byte zugeordnete Zelle(n) korrigiert werden, indem höhere Ladungszustände einer oder beider der zwei Mehrpegelzellen vorhergesagt werden. In ähnlicher Weise kann das Überlappen von mehr als zwei Mehrpegelzellen korrigiert werden, indem höhere Ladungszustände der einen oder mehreren der fehlerhaften Mehrpegelzellen vorhergesagt werden.

[0054] Fig. 11 zeigt ein Beispiel eines 64-Bit-Datenworts **450** zusammen mit dem ihm zugeordneten 8-Bit-Fehlerlokalisierungscode **460**. Der Fehlerlokalisierungscode **460** ist von einer Art eines Fehlerlokalisierungscodes, wie sie bei Fujiwara beschrieben sind. Er kann ein n-Bit-Byte erfassen, das fehlerhaft ist. Er kann darüber hinaus einen Einzelbitfehler erfassen. Fig. 11 zeigt ein Beispiel, bei dem n = 4 ist. Das Datenwort ist darüber hinaus in zwei 32-Bit-Blöcke unterteilt. Der 8-Bit-Fehlerlokalisierungscode **460** ist in der Lage, einen 4-Bit-Byte-Fehler in jedem der 32-Bit-Blöcke zu erfassen.

[0055] Somit entspricht dann, wenn das Datenwort in einem 4-Bit-Mehrpegelzellenspeicher (in welchem jede Zelle 16 Ladungszustände haben würde) gespeichert wird, jedes der 4-Bit-Bytes einer bestimmten Mehrpegelzelle. Sofern ein Fehler in einem der 4-Bit-Bytes erfaßt würde, so sind die aus der dem 4-Bit-Byte zugeordneten Mehrpegelzelle gelesenen Daten fehlerhaft. Unter Verwendung des Fehlerlokalisierungscodes kann eine fehlerhafte Mehrpegelzelle innerhalb jedes Blocks bestimmt werden.

[0056] Mehrere fehlerhafte Mehrpegelzellen können innerhalb des gleichen Datenworts bestimmt werden, indem der Fehlerlokalisierungscode **460** auf zwei getrennte fehlerhafte 4-Bit-Bytes des Datenworts **450** zeigt, wie es in Fig. 11 gezeigt ist. Die Genauigkeit der Fehlerkorrektur variiert in Abhängigkeit von der Anzahl der fehlerhaften Bits und n-Bit-Bytes, wie es in dem IEEE-Artikel von E. Fujiwara und M. Kitakami, der oben genannt wurde, beschrieben ist. Verschiedene Konfigurationen von n-Bit-Bytes und m-Bit-Blöcken können implementiert werden, um den gewünschten Grad der Fehlerlokalisierungs- und -korrekturleistung zu erreichen.

[0057] Fehlerlokalisierungscodes haben einen Vorteil gegenüber Fehlerkorrekturcodes, da weniger Bits für ihre Implementierung erforderlich sind. Dies reduziert die Kosten und die Größe der Speichermatrix. Wenn zum Vergleich ein 8-Bit-Hamming-Code (eine Art eines Fehlerkorrekturcodes) anstelle des Fehlerlokalisierungscodes **460** verwendet würde, dann könnte nur ein einzelner Bitfehler in dem Datenwort **450** korrigiert werden. Sofern ein Doppelbitfehler in dem Datenwort **450** aufträte, würde der Doppelbitfehler erfaßt werden, könnte aber nicht von dem Hamming-Code korrigiert werden. Somit stellt der Fehlerlokalisierungscode **460** ein besseres Verfahren zum Korrigieren von Fehlern in dem Datenwort als der Fehlerkorrektur-Hamming-Code zur Verfügung.

[0058] Da ein vorherrschender Ausfallmechanismus für Mehrpegelzellen darin besteht, daß die Mehrpegelzelle Ladung verliert, ist es möglich, sobald eine fehlerhafte Mehrpegelzelle bestimmt worden ist, vorherzusagen, daß die fehlerhafte Mehrpegelzelle einen höheren Ladungspegel haben sollte als der ihres gegenwärtigen Zustandes.

[0059] Fig. 12 zeigt ein Ablaufdiagramm des Prozesses des Korrigierens von durch die Fehlerlokalisierungscodes erfaßten Fehlern, wenn ein Datenwort aus einem Mehrpegelzellenspeicher gelesen wird. Die folgende Beschreibung wird unter Bezugnahme auf die Leseschaltung gemäß Fig. 5 ausgeführt. Das Ablaufdiagramm startet im Block **500**, von welchem es sich zum Block **502** fortsetzt. Beim Block **502** werden ein Datenwort und der ihm zugeordnete Fehlerlokalisierungscode in dem Mehrpegelzellenspeicher gelesen. Das Datenwort wird in dem Datenpuffer **320** und der Fehlerlokalisierungscode in dem Fehlerpuffer **322** gehalten.

[0060] Beim Block **504** werden die Datenbits und die Fehlerlokalisierungscodes durch den EED **324** verarbeitet, um zu bestimmen, ob es einen Fehler in dem Datenwort gab. Die Verarbeitung der Datenbits und des Fehlerlokalisierungscodes, die von dem EED **324** ausgeführt wird, kann durch gut bekannte Schaltungen implementiert werden.

[0061] Beim Block **506** wird dann, wenn kein Fehler erfaßt worden ist, die Operation beim Block **560** fortgesetzt, bei welchem die Operation beendet wird. Wenn jedoch ein Fehler erfaßt worden ist, setzt sich die Operation beim Block **510** fort.

[0062] Beim Block **510** wird dann, wenn der EED **324** einen Einzelbitfehler erfaßt hat, die Operation beim Block **512** fortgesetzt, bei welchem der EED **324** den Einzelbitfehler innerhalb des Datenpuffers **320** durch Anlagen einer der Signalleitungen **340** zum Ändern der richtigen Bitwerte des Datenpuffers **320** korrigiert. Vom Block **512** setzt sich die Operation beim Block **560** fort, bei welchem die Operation endet.

[0063] Wenn jedoch beim Block **510** der Decodierer einen Mehrbit-Byte-Fehler erfaßt, wie beispielsweise einen 4-Bit-Byte-Fehler, wie er unter Bezugnahme auf Fig. 11 beschrieben wurde, so wird die Operation beim Block **520** fortgesetzt. Beim Block **520** werden die eine oder die mehreren fehlerhaften Mehrpegelzellen, die Teile des Datenworts speichern, bestimmt. Auf diese fehlerhaften Mehrpegelzellen wird durch den EED **324** gezeigt. Beim Block **522** wird der gegenseitige Ladungszustand der fehlerhaften Zelle bzw. Zellen durch im Stand der Technik der Mehrpegelzellen gut bekannte Schaltungen erfaßt. Beim Block **524** wird das den fehlerhaften Zellen entsprechende Speicherelementbitmuster derart modifiziert, daß es das des nächsthöheren Ladungszustandes ergibt. Bei einer Implementierung wird direkt der nächsthöhere Ladungszustand aus dem aktuellen Ladungszustand als die höchstwahrscheinliche korrektive Maßnahme vorhergesagt. Bei einer anderen Implementierung kann ein Ladungszustand, der vom nächsthöheren Ladungszustand abweicht, wahrscheinlicher sein, um einen Fehler zu korrigieren. Beispielsweise kann es experimentell gefunden werden, daß der vorherrschende Fehlermodus für einen bestimmten Ladungszustand einer Mehrpegelzelle darin besteht, zwei Zustände nach unten zu springen. In diesem Fall kann der Fehler korrigiert werden, indem vorhergesagt wird, daß der zweitnächsthöhere Ladungszustand gegenüber dem gelesenen Ladungszustand die wahrscheinlichste korrektive Maßnahme ist. In ähnlicher Weise könnte dann, sofern es gefunden würde, daß ein bestimmter Ausfallmodus der einzige Ausfallmodus einer Mehrpegelzelle wäre, der Korrekturalgorithmus in geeigneter Weise implementiert werden, um nur die erforderlichen korrektiven Maßnahmen auszuführen.

[0064] Beim Block **526** werden das modifizierte Datenwort und der Fehlerlokalisierungscode durch den EED **324** verarbeitet, um nachzusehen, ob der Fehler durch die im Block **524** durchgeführte Modifikation des Datenworts korrigiert worden ist.

[0065] Wenn es immer noch einen Fehler gibt und die der fehlerhaften Mehrpegelzelle oder -zellen zu-

geordneten Bits erneut auf einen höheren Ladungszustand modifiziert werden können, dann wird die Operation vom Block **530** zurück zum Block **524** fortgesetzt. Wenn jedoch sowohl die Bedingung, daß es noch einen Fehler gibt, als auch die Bedingung, daß das Speicherelementbitmuster, das einer fehlerhaften Mehrpegelzelle zugeordnet ist, nicht mehr auf einen höheren Ladungszustand modifiziert werden kann, erfüllt sind, dann wird die Operation beim Block **540** fortgesetzt.

[0066] Beim Block **540** wird dann, wenn es noch einen Fehler in dem Datenwort gibt, die Operation beim Block **550** fortgesetzt, bei welchem ein Fehler durch ein Flag angezeigt wird. Der Prozessor **200** oder die andere Einrichtung, die das Datenwort anfordern, werden darüber informiert, daß ein Fehler aufgetreten ist.

[0067] Wenn jedoch beim Block **540** kein Fehler mehr erfaßt wird, dann wird die Operation **560** fortgesetzt. Beim Block **560** endet das Ablaufdiagramm. Der Datenpuffer **320** hält ein Datenwort, bei welchem Fehler korrigiert worden sind. Bei einem Ausführungsbeispiel eines Mehrpegelzellenspeichers wird die fehlerhafte Mehrpegelzelle mit dem korrigierten Datenwort, das in dem Datenpuffer **320** gehalten wird, neu überschrieben. Bei einem anderen Ausführungsbeispiel jedoch wird eine Korrektur der Speichermatrix nicht ausgeführt. Dies beruht auf der Komplexität des Hinzufügens zusätzlicher Schaltungen, um zur Speichermatrix zurückzuschreiben. Zusätzlich würde bei einem Flash-Speicher-Ausführungsbeispiel grundsätzlich ein gesamter Block zu löschen sein, um ein Datenwort, das vorher bereits zu dem Flash-Speicher geschrieben worden ist, neu zu schreiben.

Patentansprüche

1. Verfahren zum Lesen von Daten aus einem Speicher mit Mehrpegel-Speicherzellen, die jeweils wenigstens drei Ladungszustände speichern können, wobei:
a) die Daten und ein zugehöriger Fehlerlokalisierungscode gelesen werden (**502**), wobei der Fehlerlokalisierungscode ein Identifizieren desjenigen Blocks oder derjenigen Blöcke der Daten, der bzw. die jeweils einer fehlerhafte Daten enthaltenden Mehrpegel-Speicherzelle zugeordnet sind, gestattet; und
b) dann, wenn der gelesene Fehlerlokalisierungscode anzeigt, dass ein gelesener, einem bestimmten Ladungspegel einer Speicherzelle entsprechender Block der Daten fehlerhaft ist, dieser Block der Daten bei der Ausgabe derart modifiziert wird, dass er dem nächsthöheren Ladungspegel der Speicherzelle entspricht.

2. Das Verfahren nach Anspruch 1, das ferner die Schritte aufweist:
(c) Verwenden des Fehlerlokalisierungscodes, um Zielbits des identifizierten Blocks oder der identifizierten Blöcke der Daten, die fehlerhaft sind, zu bestimmen, wobei die Zielbits weiter anzeigen, dass die Ziel-Mehrpegelzelle einen ersten Ladungszustand hält; und
(d) Modifizieren der Zielbits, so dass sie dem nächsthöheren Ladungspegel entspricht.

3. Das Verfahren nach Anspruch 2, ferner umfassend den Schritt des:
(e) Bestimmens, ob die Zielbits, die in dem Schritt (d) modifiziert worden sind, noch immer fehlerhaft sind, und, sofern die Zielbits noch immer fehlerhaft sind, zum anschließenden Modifizieren der Zielbits derart, dass sie dem Halten eines dritten Ladungspegels durch die Ziel-Mehrpegelzelle entsprechen, wobei der dritte Ladungspegel mehr Ladung hält als der zweite Ladungspegel.

4. Das Verfahren nach Anspruch 2, wobei die Zielbits darüber hinaus einer zweiten Ziel-Mehrpegelzelle zugeordnet sind, und wobei die Zielbits anzeigen, dass die zweite Ziel-Mehrpegelzelle einen vierten Ladungspegel hält, wobei das Verfahren ferner den Schritt aufweist:
(e) Modifizieren der Zielbits derart, dass sie dem Halten eines fünften Ladungspegels durch die zweite Ziel-Mehrpegelzelle entsprechen, wobei der fünfte Ladungspegel mehr Ladung hält als der vierte Ladungspegel.

5. Das Verfahren nach Anspruch 1, das ferner folgende Schritte aufweist:
(c) Verwenden der Fehlerlokalisierungsbits, um zu bestimmen, dass ein Abschnitt der vorgegebenen Anzahl von Datenbits unrichtig ist, wobei der Abschnitt der vorgegebenen Anzahl von Datenbits einer ersten Mehrpegelzelle entspricht, die gegenwärtig einen ersten Ladungspegel hält und
(d) Modifizieren des Abschnitts der vorgegebenen Anzahl von Datenbits, die der ersten Mehrpegelzelle zugeordnet sind, so dass sie dem nächsthöheren Ladungspegel entsprechen.

6. Das Verfahren nach Anspruch 5, ferner aufweisend die Schritte:
(e) Überprüfen, ob ein Fehler in der vorgegebenen Anzahl von Datenbits vorhanden ist, die in dem Schritt (d) modifiziert worden sind; und
(f) Modifizieren des Abschnitts der vorgegebenen Anzahl von Datenbits, die der ersten Mehrpegelzelle zugeordnet sind, derart, dass sie einem nächsthöheren Ladungspegel entsprechen, bis der Fehler korrigiert worden ist.

7. Eine Schaltung zum Durchführen eines Verfahrens nach einem der vorhergehenden Ansprüche, aufweisend:

eine Speichermatrix, die eine Mehrzahl von Mehrpegelzellen aufweist, die in der Lage sind, zumindest jeweils zwei Bits Daten zu speichern; und
eine Fehlerschaltung, die so eingekoppelt ist, dass sie die Daten aus der Speichermatrix empfängt, wobei die Fehlerschaltung einen Decodierer aufweist, der versucht, ein fehlerhaftes Speicherelementbitmuster, das durch eine der Mehrpegelzellen zur Verfügung gestellt worden ist, zu korrigieren, in dem der Decodierer auf der Grundlage eines dem fehlerhaften Speicherelementbitmuster zugeordneten Ladungszustandes das fehlerhafte Speicherelementbitmuster modifiziert.

8. Die Schaltung nach Anspruch 7, ferner aufweisend:
einen Datenpuffer, der das fehlerhafte Speicherelementbitmuster speichern kann, wobei der Datenpuffer eine Eingabe aus dem Decodierer zum Modifizieren einzelner, in dem Datenpuffer gespeicherter Bits gestattet.

Es folgen 10 Blatt Zeichnungen

Anhängende Zeichnungen

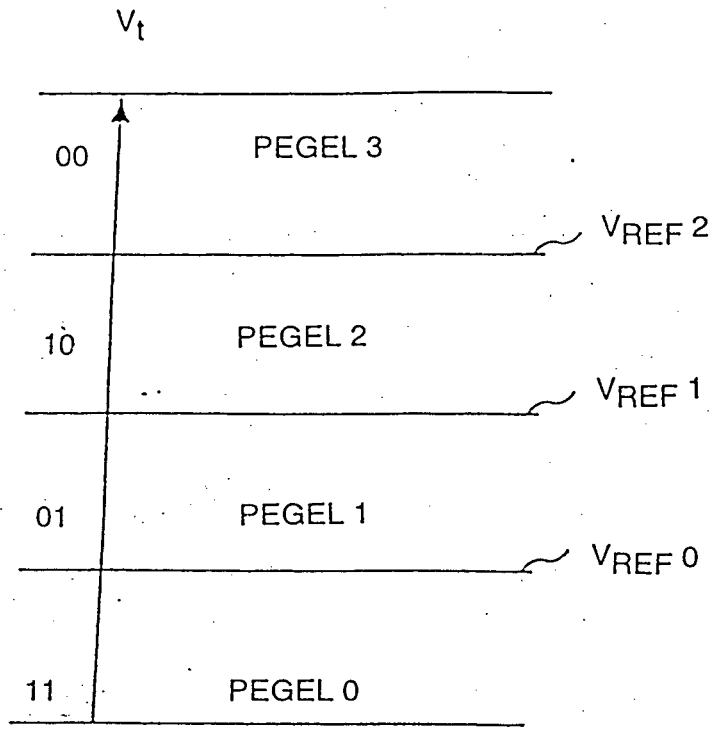


FIG. 1 (Stand der Technik)

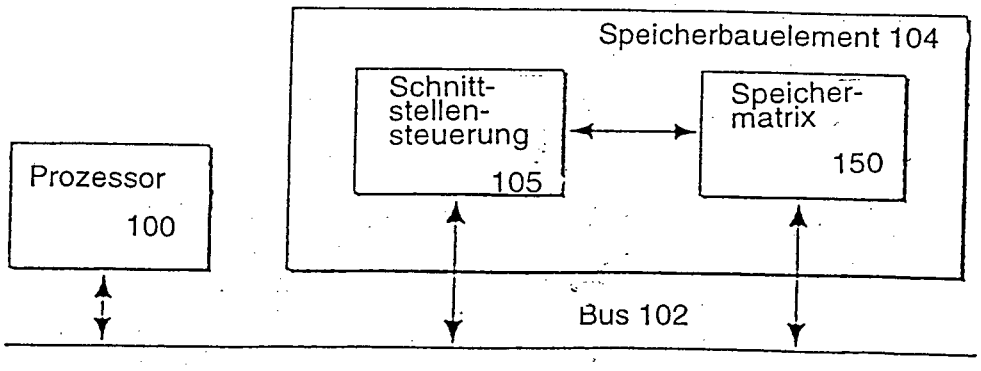


FIG. 2 (Stand der Technik)

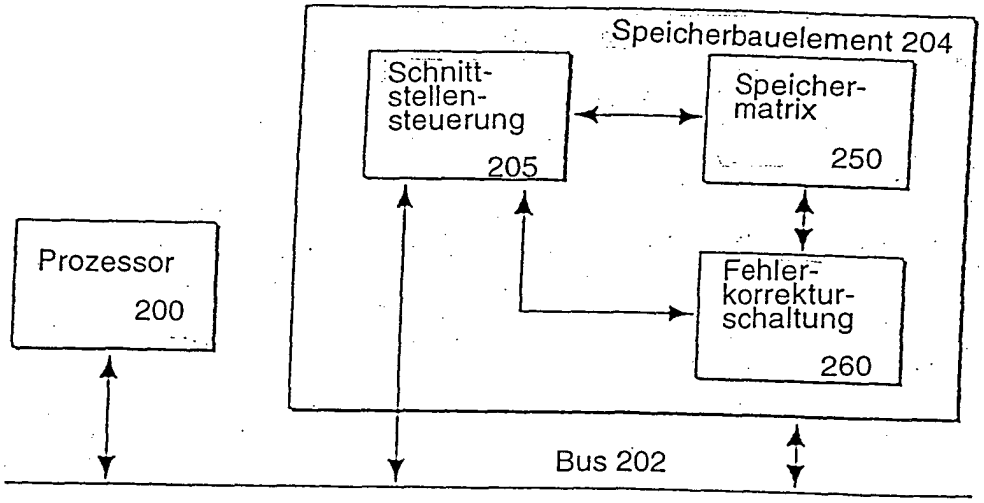

Speicherbauelement 204
Schnitt-
stellen-
steuerung
205
Speicher-
matrix
250
Prozessor
200
Fehler-
korrektur-
schaltung
260
Bus 202

FIG. 3

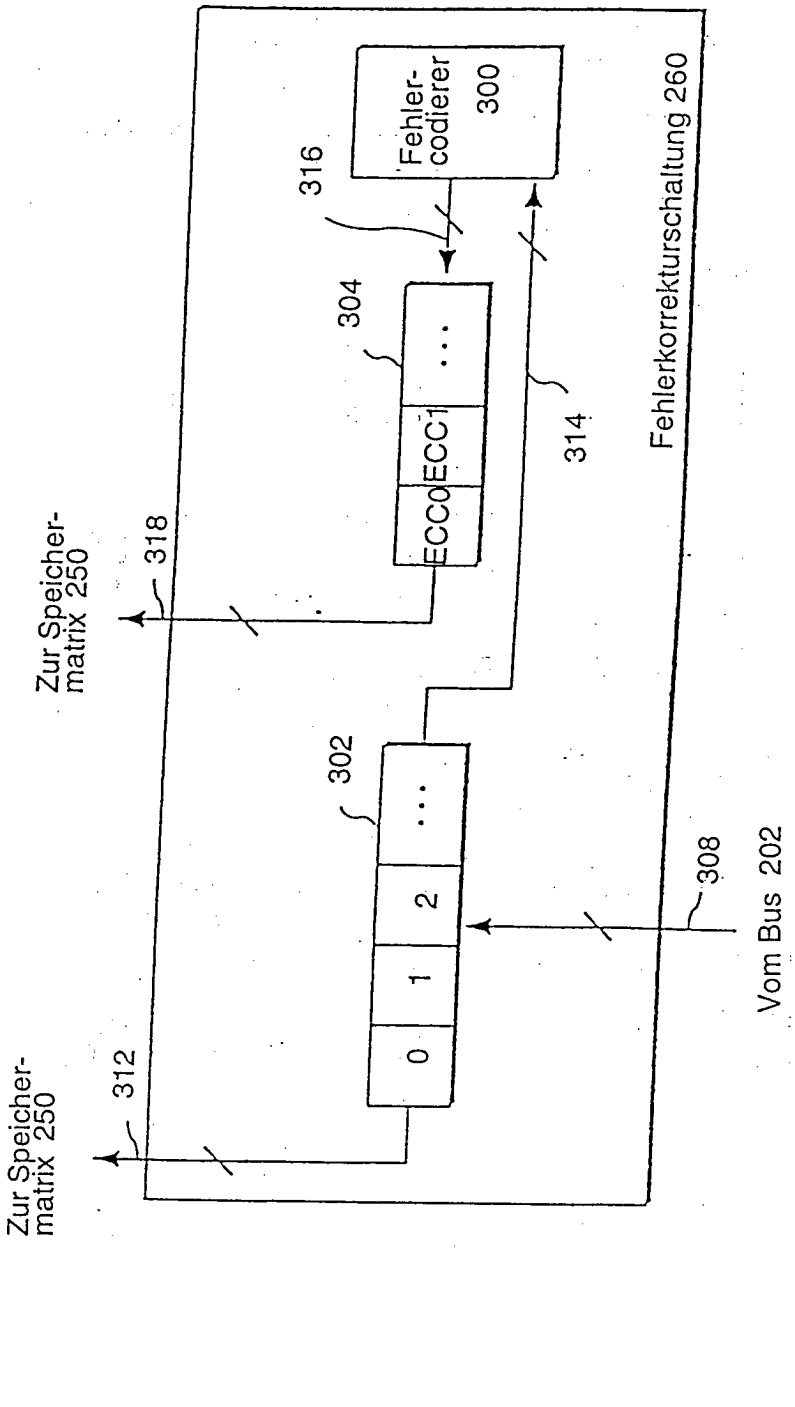

Fehlerkorrekturschaltung 260
Fehler-codierer 300
316
304
ECC0
ECC1
...
314
318
Zur Speicher-matrix 250
302
0
1
2
...
308
Vom Bus 202
312
Zur Speicher-matrix 250

FIG. 4

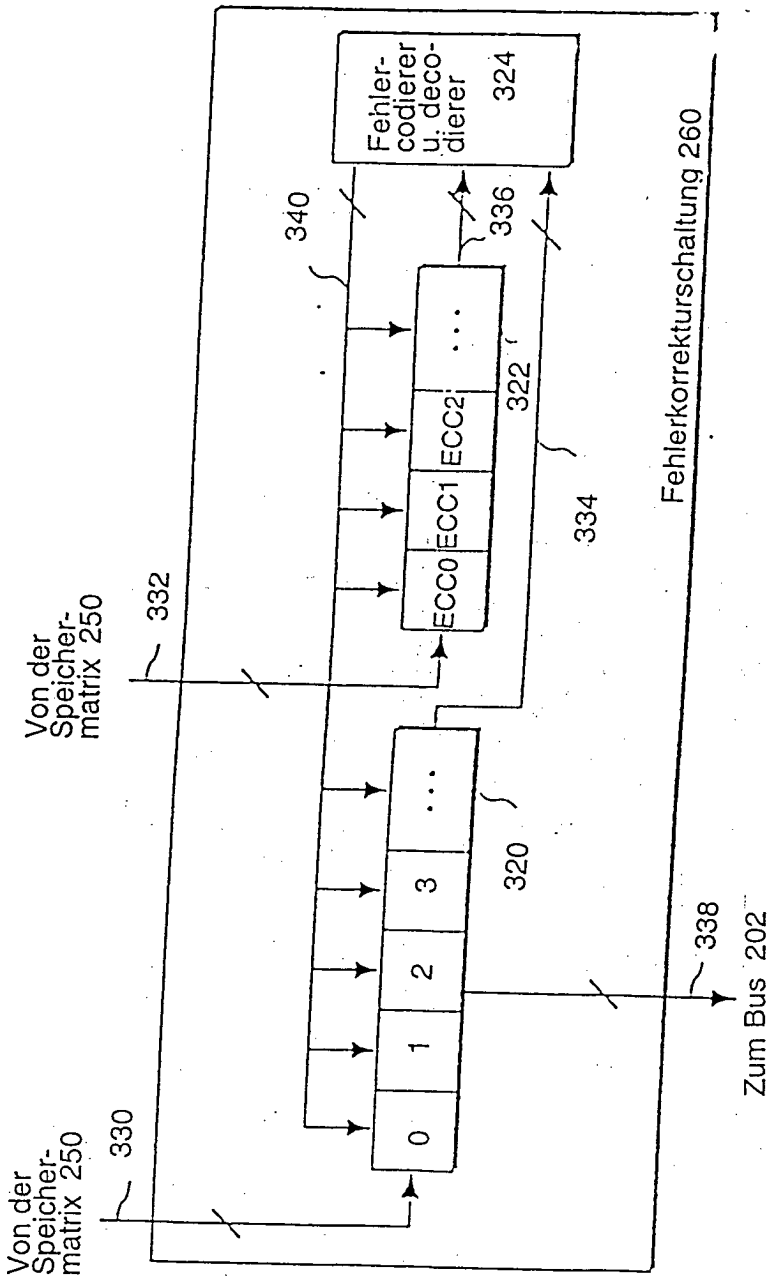

Fehlerkorrekturschaltung 260
Von der Speicher-matrix 250
330
0
1
2
3
...
320
Von der Speicher-matrix 250
332
ECC0
ECC1
ECC2
...
322
334
340
Fehler-codierer u. deco-dierer
324
336
338
Zum Bus 202

FIG. 5

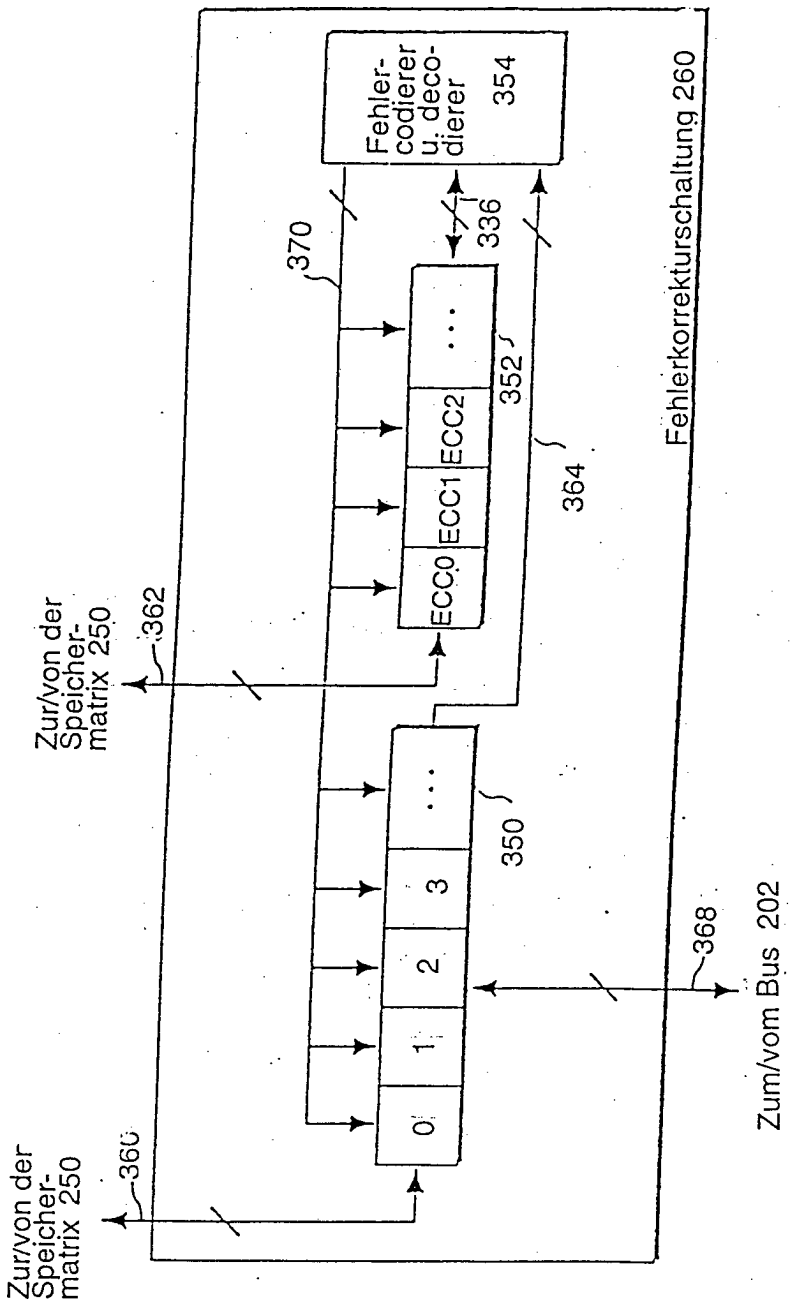

Fehlerkorrekturschaltung 260
Fehler-codierer u. deco-dierer
354
370
336
352
364
ECC0
ECC1
ECC2
...
0
1
2
3
...
350
362
Zur/von der Speicher-matrix 250
360
Zur/von der Speicher-matrix 250
368
Zum/vom Bus 202

FIG. 6

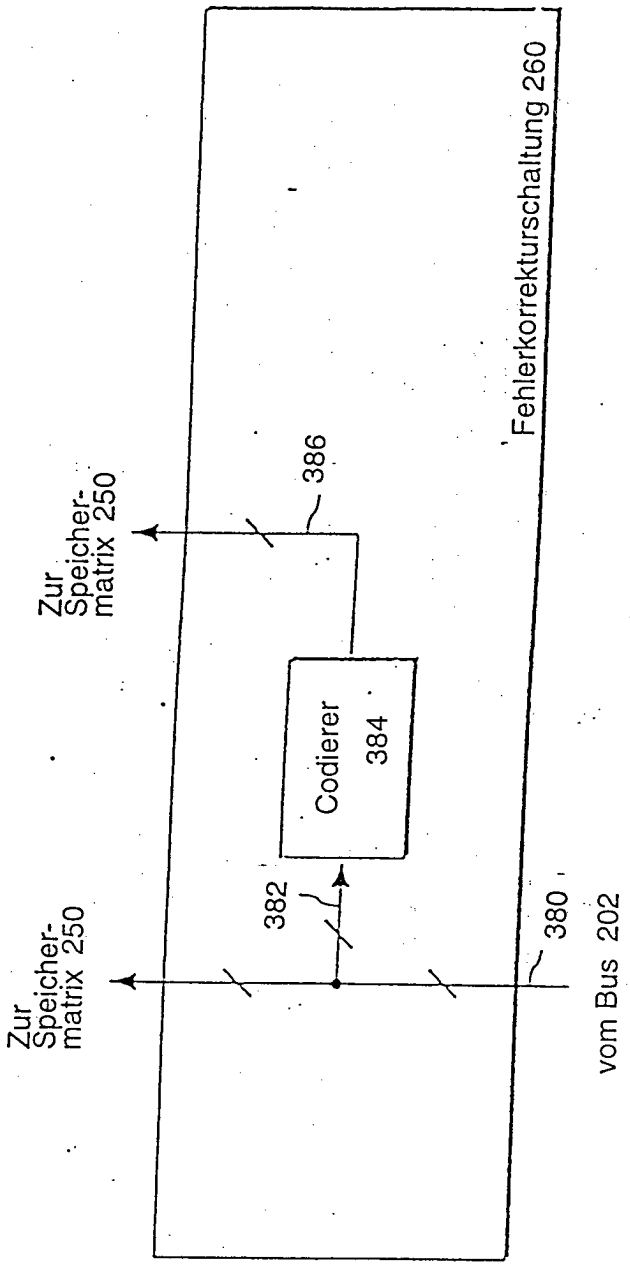

Fehlerkorrekturschaltung 260
386
Zur Speicher-matrix 250
Codierer 384
382
380
Zur Speicher-matrix 250
vom Bus 202

FIG. 7

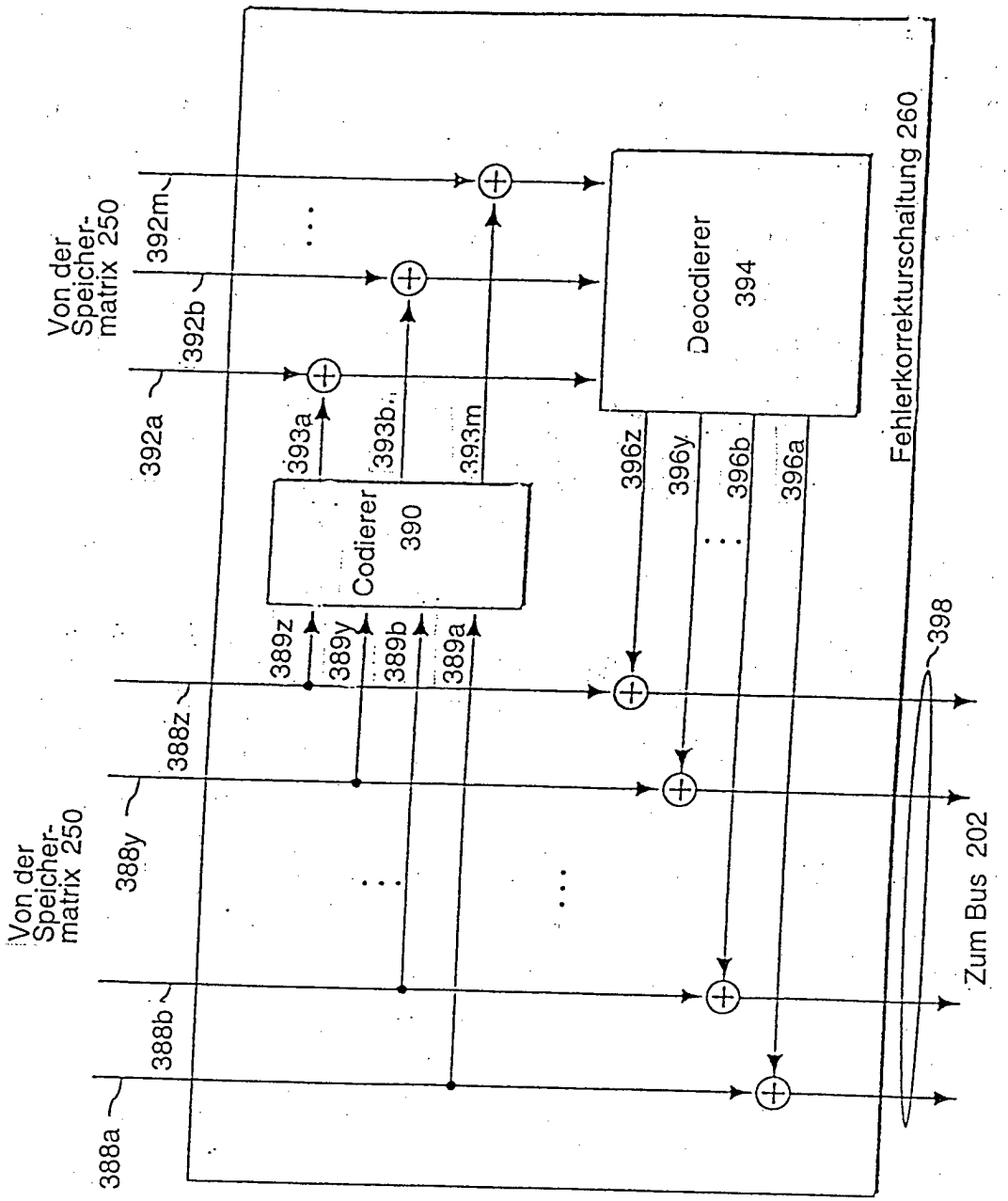

Von der Speicher-matrix 250
392m
392b
392a
393a
393b
393m
Codierer 390
389z
389y
389b
389a
388z
388y
388b
388a
Von der Speicher-matrix 250
Deocdierer 394
396z
396y
396b
396a
Fehlerkorrekturschaltung 260
398
Zum Bus 202

FIG. 8

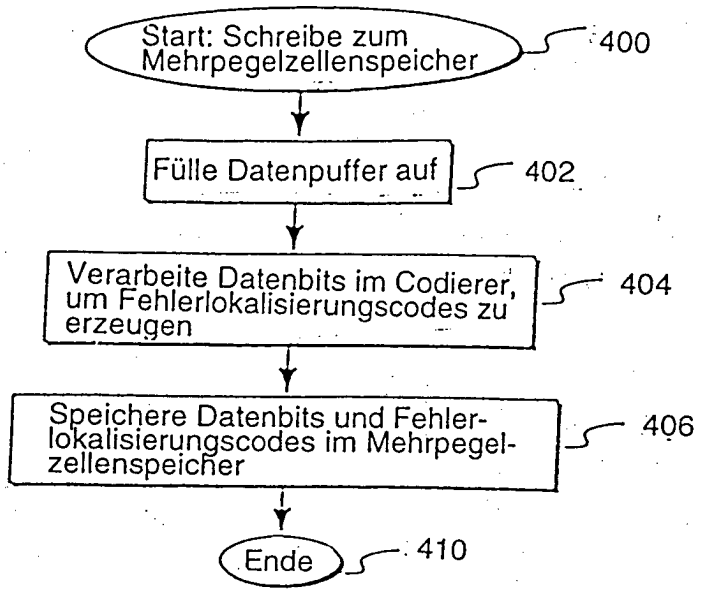

Start: Schreibe zum Mehrpegelzellenspeicher
400
Fülle Datenpuffer auf
402
Verarbeite Datenbits im Codierer, um Fehlerlokalisierungscodes zu erzeugen
404
Speichere Datenbits und Fehlerlokalisierungscodes im Mehrpegelzellenspeicher
406
Ende
410

FIG. 9

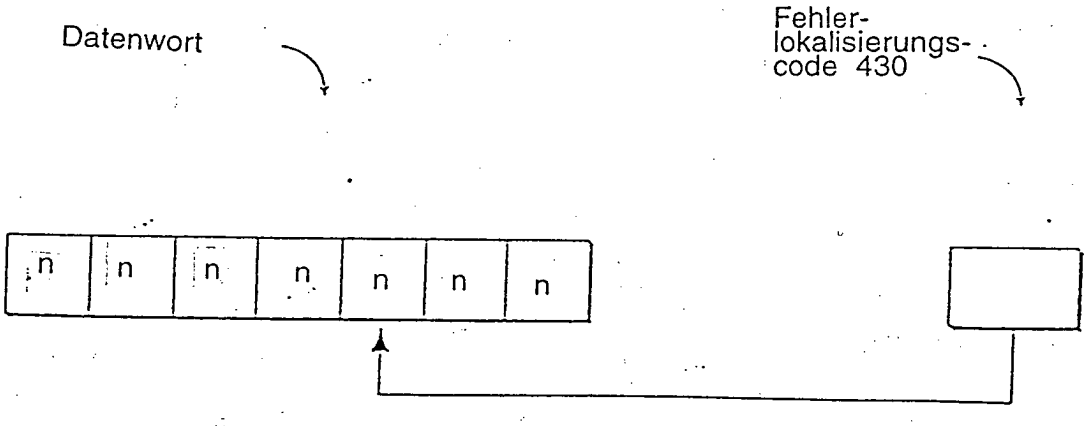

Datenwort
Fehlerlokalisierungscode 430
n
n
n
n
n
n
n

FIG. 10

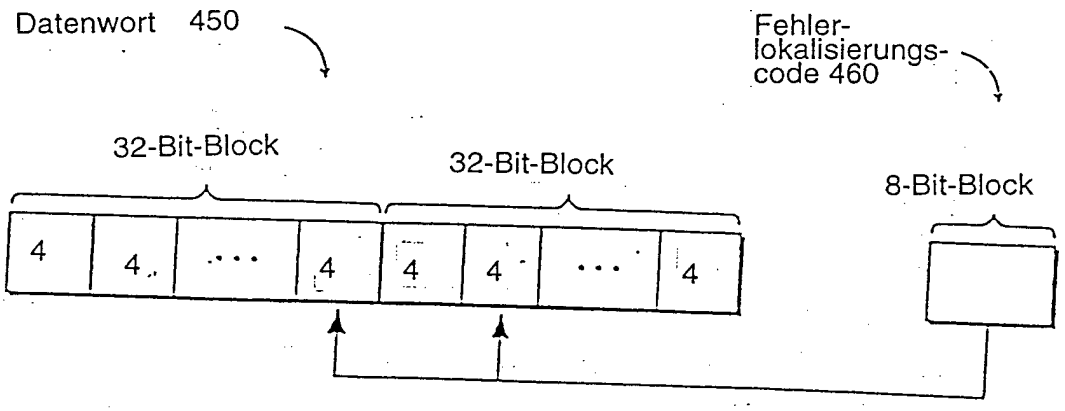

Datenwort 450
Fehler-
lokalisierungs-
code 460
32-Bit-Block
32-Bit-Block
8-Bit-Block
4
4
4
4
4
4
4

FIG. 11

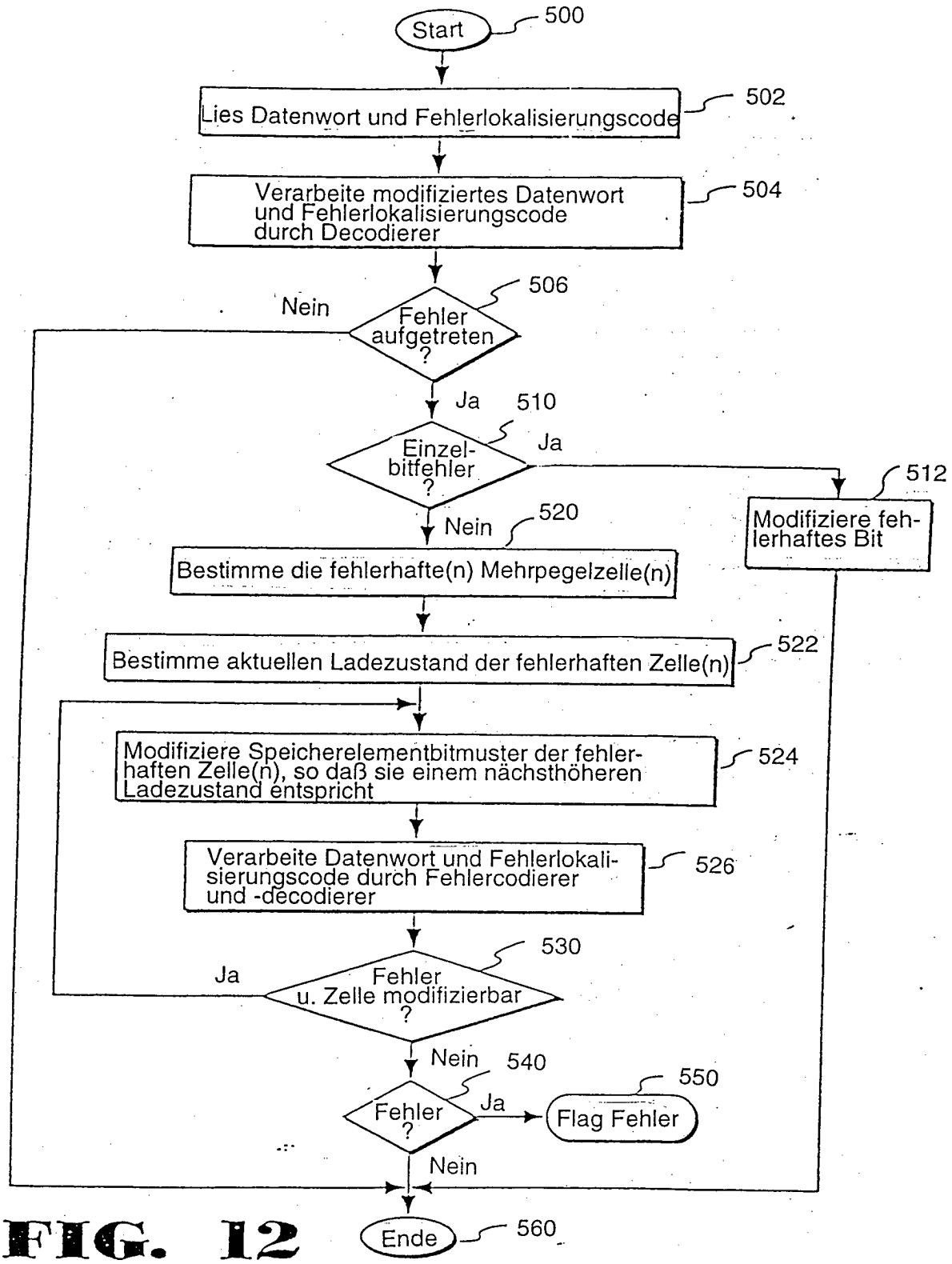

Start
500
Lies Datenwort und Fehlerlokalisierungscode
502
Verarbeite modifiziertes Datenwort und Fehlerlokalisierungscode durch Decodierer
504
Fehler aufgetreten ?
506
Nein
Ja
Einzel-bitfehler ?
510
Ja
512
Modifiziere fehlerhaftes Bit
Nein
520
Bestimme die fehlerhafte(n) Mehrpegelzelle(n)
Bestimme aktuellen Ladezustand der fehlerhaften Zelle(n)
522
Modifiziere Speicherelementbitmuster der fehlerhaften Zelle(n), so daß sie einem nächsthöheren Ladezustand entspricht
524
Verarbeite Datenwort und Fehlerlokalisierungscode durch Fehlercodierer und -decodierer
526
Fehler u. Zelle modifizierbar ?
530
Ja
Nein
Fehler ?
540
Ja
Flag Fehler
550
Nein
Ende
560

FIG. 12