(19) Bundesrepublik Deutschland
Deutsches Patent- und Markenamt

(10) **DE 103 44 021 A1** 2004.07.15

(12) **Offenlegungsschrift**

(21) Aktenzeichen: **103 44 021.6**
(22) Anmeldetag: **16.09.2003**
(43) Offenlegungstag: **15.07.2004**

(51) Int Cl.[7]: **G11C 16/06**

(30) Unionspriorität:
315897 **09.12.2002** **US**

(71) Anmelder:
Samsung Electronics Co., Ltd., Suwon, Kyonggi, KR

(74) Vertreter:
Patentanwälte Ruff, Wilhelm, Beier, Dauster & Partner, 70174 Stuttgart

(72) Erfinder:
Kwon, Oh-Suk, Yongin, KR; Lim, Heung-Soo, Yongin, KR; Lee, June, Seoul/Soul, KR

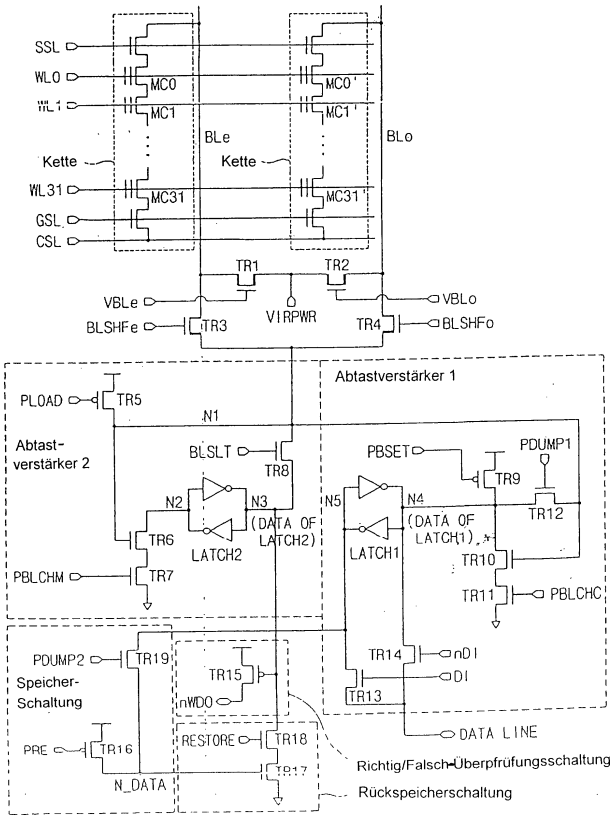
Die folgenden Angaben sind den vom Anmelder eingereichten Unterlagen entnommen

(54) Bezeichnung: **Nichtflüchtiger Speicherbaustein, Programmiervorrichtung und Programmierverfahren**

(57) Zusammenfassung: Die Erfindung bezieht sich auf einen nichtflüchtigen Speicherbaustein mit einem Speicherzellenfeld zum Speichern von Daten, einer Y-Gatterschaltung zum gesteuerten Durchleiten von in einer jeweiligen Gruppe der Speicherzellen gespeicherten Daten, einem Seitenpuffer, der über einen Abtastknoten (N1) zwischen dem Speicherzellenfeld und der Y-Gatterschaltung eingeschleift ist, wobei der Seitenpuffer einen ersten Abtastverstärker und einen zweiten Abtastverstärker umfasst, und einer Speicherschaltung zum Speichern von Daten des ersten Abtastverstärkers in Abhängigkeit von einem Speichersignal sowie auf eine zugehörige Programmiervorrichtung und auf ein zugehöriges Programmierverfahren.

Erfindungsgemäß umfasst der erste Abtastverstärker ein erstes Datenregister (LATCH1) und der zweite Abtastverstärker ein zweites Datenregister (LATCH2), wobei das zweite Datenregister (LATCH2) gemeinsam mit dem ersten Datenregister (LATCH1) funktionswirksam mit dem Abtastknoten (N1) gekoppelt ist.

Verwendung für nichtflüchtige Speicherbausteine, z. B. Flash-Speicher vom NAND-Typ.



Beschreibung

[0001] Die Erfindung betrifft einen nichtflüchtigen Speicherbaustein nach dem Oberbegriff des Patentanspruchs 1 und eine zugehörige Programmiervorrichtung sowie ein zugehöriges Programmierverfahren.

Stand der Technik

[0002] Gemäß dem gegenwärtigen Trend sind Halbleiterspeicherbausteine hoch integriert mit einer großen Speicherkapazität zur Unterstützung von Systemen mit einer hohen Betriebsgeschwindigkeit ausgeführt. Dieser Trend gilt sowohl für flüchtige Speicher, wie DRAMs und SRAMs, als auch für nichtflüchtige Speicher, wie Flash-Speicher.

[0003] Flash-Speicher sind generell in Flash-Speicher vom NOR-Typ oder in Flash-Speicher vom NAND-Typ unterteilt. Die Flash-Speicher vom NOR-Typ werden in Anwendungen benutzt, die Informationen mit einem kleinen Umfang nicht sequentiell und mit einer hohen Geschwindigkeit lesen, während die Flash-Speicher vom NAND-Typ in Anwendungen benutzt werden, die Informationen sequentiell lesen.

[0004] Flash-Speicherbausteine benutzen zum Speichern von Daten Speicherzellen, die Zellentransistoren umfassen. Jeder Zellentransistor hat eine Steuerelektrode und einen floatenden Gatebereich. Da der Flash-Speicherbaustein Information mittels Tunneln durch eine Isolationsschicht speichert, wird etwas Zeit benötigt, um die Information zu speichern.

[0005] Um Informationen mit einem großen Umfang in einer kurzen Zeitspanne zu speichern, benutzt der Flash-Speicher vom NAND-Typ ein Register, das auch als Seitenpufferschaltkreis bezeichnet wird. Große Datenumfänge werden für einen schnellen Speichervorgang im Speicherbereich von extern zugeführt und erst im Register gespeichert und von dort in die Speicherzellen übertragen.

[0006] In herkömmlichen Flash-Speichern vom NAND-Typ wird eine Datenseitengröße von 512 Byte nicht überschritten. Es wird angenommen, dass eine Programmierzeit oder eine Informationsspeicherzeit eines Flash-Speichers vom NAND-Typ etwa 200 bis 500 Mikrosekunden beträgt und ein Datenbyte von extern in einer Zeitdauer von 100 Nanosekunden in den Seitenpuffer geladen wird, so dass ungefähr 50 Mikrosekunden benötigt werden, um Informationen von 512 Byte in den Seitenpuffer zu laden.

[0007] **Fig.** 1 zeigt ein aus dem Stand der Technik bekanntes Ausführungsbeispiel, das weitgehend einem in **Fig.** 7 der Patentschrift US 5.831.900 dargestellten Beispiel entspricht, wobei zur vorliegenden Beschreibung zusätzliche Bezugszeichen eingeführt wurden.

[0008] Beim Speicherbaustein aus **Fig.** 1 werden Daten von einer Datenleitung IO in einen Zwischenspeicher **30** geladen, nachdem Seitenpuffer 20-i durch eine umliegende Schaltung zurückgesetzt wurden. Die in den Zwischenspeicher **30** geladenen Daten werden über einen Transistor Q4 in Speicherzellen **2–1**, **2–2**, **2–3** programmiert, häufig durch Empfangen eines zugehörigen Programmierbefehlssignals. Dieses Programmierverfahren wird normalerweise zum Programmieren von Flash-Speichern vom NAND-Typ verwendet.

[0009] Dieses Verfahren ist jedoch begrenzt. Bei diesem Programmiervorgang müssen Daten, die in den Zwischenspeicher geladen werden sollen, so lange warten, bis die vorher geladenen Daten den vorherigen Programmiervorgang abgeschlossen haben. Wie oben bereits ausgeführt wurde, werden Daten in Form von Byteeinheiten von beispielsweise 8 Bit in den Zwischenspeicher **30** geladen. Deshalb wird eine lange Zeitspanne benötigt, um beispielsweise Daten mit einem Umfang von 2048 Bytes auf eine Seite zu laden. Das liegt daran, dass der Zwischenspeicher **30** so lange Daten speichert, bis die Informationen des Registers in den zugehörigen Speicherzellen gespeichert sind.

[0010] Ein anderes Problem beim Stand der Technik ist das Rückkopierproblem. Manchmal wird ein Kopiervorgang von Daten einer ersten Seite auf eine zweite Seite benötigt. Ist es wünschenswert, den Kopiervorgang durchzuführen, nachdem die Daten von den Speicherzellen auf der ersten Seite im Zwischenspeicher **30** über einen Transistor Q7 zwischengespeichert wurden, dann werden die zwischengespeicherten Daten über den Transistor Q4 auf die zweite Seite programmiert. In diesem Fall werden programmierte, auf die zweite Seite kopierte Daten wegen dem Zwischenspeicher umgekehrt, d.h. ein Wert von „1" wurde zu einem Wert „0" und ein Wert „0" wurde zu einem Wert „1 ". Dieses Problem wird im Stand der Technik durch Verwendung von Flag- d.h. Markierzellen im Speicherzellenfeld berücksichtigt, deren Wert in Abhängigkeit davon aktualisiert wird, ob die Daten invertiert wurden oder nicht.

[0011] **Fig.** 2 zeigt ein Beispiel dieses Problems aus dem Stand der Technik anhand zweier Teilbilder, die weitgehend den **Fig.** 8 und 9 der Patentschrift US 5.996.041 entsprechen. In **Fig.** 2 sind Rückkopiervorgänge dargestellt. Daten der ersten Seite im Speicherzellenfeld werden in den Seitenpuffer geladen. Danach werden die Daten an einen anderen Platz im Speicherzellenfeld geladen, aber in invertierter Form. Das rechte Bit gehört zur Flagzelle, um anzuzeigen, dass die Daten in invertierter Form vorliegen.

[0012] Der Stand der Technik ist durch die mögliche Größe des Speicherbausteins begrenzt. Es wird beispielsweise angenommen, dass der Seitenpuffer temporär Informationen von 2048 Byte speichern kann und es ungefähr 200 Mikrosekunden dauert, bis die Informationen von 2048 Byte auf den Seitenpuffer als Informa-

tionen von einem Byte in Perioden von 100 Nanosekunden geladen sind. Deshalb ist die Ladezeit nahezu gleich der Informationsspeicherzeit oder der Programmierzeit von 200 bis 500 Mikrosekunden. Entsprechend wird die Informationsspeichercharakteristik des Flash-Speichers vom NAND-Typ durch die Ladezeit stark beeinflusst.

[0013] Mit höherer Integration von Flash-Speichern vom NAND-Typ müssen Daten in immer größerem Umfang verarbeitet werden, verglichen mit herkömmlichen Flash-Speichern. Die Daten sollten auch ohne Verschlechterung der Informationsspeichercharakteristik verarbeitet werden.

[0014] In **Fig.** 22 und 23 sind Beispiele aus einer Voranmeldung (US-Anmeldenummer 10/013191) zur vorliegenden Prioritätsanmeldung veranschaulicht. Wie aus den **Fig.** 22 und 23 ersichtlich ist, umfasst ein Seitenpuffer zwei Register mit Zwischenspeichern. Das erste Register hat einen ersten Zwischenspeicher LATCH1 und das zweite Register hat einen zweiten Zwischenspeicher LATCH2. Diese Struktur wird in der Voranmeldung detailliert beschrieben. Wie aus den **Fig.** 22 und 23 ersichtlich ist, werden zu programmierende Daten in einem Schritt F1 über eine Datenleitung in einen Knoten N4 des Zwischenspeichers LATCH1 geladen. Danach werden in einem Schritt F2 die Daten vom Knoten N4 im Zwischenspeicher LATCH1 in einen Knoten N3 im Zwischenspeicher LATCH2 übertragen oder geladen. Entsprechend dem Datenzustand am Knoten N3 werden die Daten während eines Schrittes F3 in eine erste Seite eines Speicherzellenfeldes programmiert. Haben die Daten am Knoten N3 einen Wert „0", d.h. einen Massepegel und Programmierzustand, dann werden die Speicherzellen programmiert. Haben die Daten am Knoten N3 andererseits einen Wert „1", d.h. einen Versorgungsspannungspegel (Vcc) und Programmiersperrzustand, dann werden die Speicherzellen nicht programmiert. Es sei angemerkt, dass eine Seite eine Gruppe von Speicherzellen umfasst, die von einer Wortleitung gesteuert werden.

[0015] Nach dem Programmieren müssen die Speicherzellen der Seite überprüft werden, ob sie erfolgreich programmiert wurden. Dieser Überprüfungsvorgang F4 aus **Fig.** 23 wird als Programmierprüflesevorgang bezeichnet. Beim Programmierprüflesevorgang wird der Zustand am Knoten N3 auf den Wert „0" zurückgesetzt, wenn die zu programmierende Zelle nicht programmiert ist, und auf den Wert „1" zurückgesetzt, wenn die zu programmierende Zelle programmiert ist. Die nicht programmierten Zellen müssen entsprechend dem oben beschriebenen Programmierverfahren erneut programmiert werden.

[0016] Sind alle Zellen programmiert, dann wird der Knoten N3 während eines Schritts F5 auf den Wert „1" gesetzt. Dies schließt den Programmiervorgang für die erste Seite des Speicherzellenfeldes ab.

[0017] Während des Programmiervorgangs der ersten Seite des Speicherzellenfeldes werden Daten der zweiten Seite simultan in den Knoten N4 des Zwischenspeichers LATCH1 geladen. Daraus resultiert, dass zwei Vorgänge gleichzeitig in einem vorgegebenen Seitenpuffer durchgeführt werden.

[0018] Das US-Patent 6.031.760 beschreibt in Verbindung mit der dortigen **Fig.** 5 ein bekanntes Speicherbauelement mit einem einzelnen Zwischenspeicher, wie es typisch für herkömmliche Speicherbausteine ist. Die beschriebene Schaltung hat einen einzelnen Abtastverstärker S/A, der nur einen einzelnen Zwischenspeicher LT umfasst.

Aufgabenstellung

[0019] Aufgabe der Erfindung ist es, einen nichtflüchtigen Speicherbaustein der eingangs genannten Art zur Verfügung zu stellen, der Daten in einem vergleichsweise großen Umfang ohne Verschlechterung der Informationsspeichercharakteristik verarbeiten kann, und eine zugehörige Programmiervorrichtung und ein zugehöriges Programmierverfahren anzugeben.

[0020] Die Erfindung löst diese Aufgabe durch einen nichtflüchtigen Speicherbaustein mit den Merkmalen des Patentanspruchs 1, durch eine Programmiervorrichtung mit den Merkmalen des Patentanspruchs 9 und durch ein Programmierverfahren mit den Merkmalen des Patentanspruchs 15.

[0021] Vorteilhafte Weiterbildungen der Erfindung sind in den abhängigen Ansprüchen angegeben.

Ausführungsbeispiel

[0022] Vorteilhafte, nachfolgend beschriebene Ausführungsformen der Erfindung sowie die zu deren besserem Verständnis oben erläuterten, herkömmlichen Ausführungsbeispiele sind in den Zeichnungen dargestellt. Es zeigen:

[0023] **Fig.** 1 ein Schaltbild eines herkömmlichen Speicherbausteins mit einem Seitenpuffer;

[0024] **Fig.** 2 eine schematische Darstellung eines Rückkopiervorgangs beim Stand der Technik, bei dem wegen Dateninvertierung ein Flagbit nötig ist;

[0025] **Fig.** 3 ein Blockschaltbild eines erfindungsgemäßen Halbleiterspeicherbausteins;

[0026] **Fig.** 4 ein Blockschaltbild eines Speicherzellenfeldes aus **Fig.** 3;

[0027] **Fig.** 5 ein Schaltbild eines Seitenregisters und einer Y-Gatterschaltung aus **Fig.** 3;

[0028] **Fig.** 6 ein Flussdiagramm eines erfindungsgemäßen Programmierverfahrens;

[0029] **Fig.** 7 ein Zeitablaufdiagramm von Befehlssignalen zur Ausführung des Verfahrens aus **Fig.** 6;
[0030] **Fig.** 8 eine Darstellung der Schaltung aus **Fig.** 5 mit zusätzlicher Verdeutlichung des Datenflusses bei Anlegen der Befehlssignale aus **Fig.** 7;
[0031] **Fig.** 9 ein Zeitablaufdiagramm von Befehlssignalen zur Ausführung eines Leseverfahrens im Baustein aus **Fig.** 3;
[0032] **Fig.** 10 eine Darstellung der Schaltung aus **Fig.** 5 mit zusätzlicher Verdeutlichung des Datenflusses bei Anlegen der Befehlssignale aus **Fig.** 9;
[0033] **Fig.** 11 ein Flussdiagramm eines erfindungsgemäßen Rückkopierverfahrens;
[0034] **Fig.** 12 ein Zeitablaufdiagramm von Befehlssignalen zur Ausführung des Rückkopierverfahrens beim Baustein von **Fig.** 3;
[0035] **Fig.** 13 eine schematische Darstellung zur Verdeutlichung von Daten, die von Speicherzellen in einen Seitenpuffer gemäß den Befehlssignalen aus einem ersten Teil von **Fig.** 12 übertragen werden;
[0036] **Fig.** 14 eine schematische Darstellung zur Verdeutlichung von Daten, die von Speicherzellen in einen Seitenpuffer gemäß den Befehlssignalen aus einem zweiten Teil von **Fig.** 12 übertragen werden;
[0037] **Fig.** 15 ein Flussdiagramm eines erfindungsgemäßen Löschverfahrens;
[0038] **Fig.** 16 ein Zeitablaufdiagramm von Befehlssignalen zur Ausführung des Löschverfahrens beim Baustein von **Fig.** 3;
[0039] **Fig.** 17 eine schematische Darstellung der Schaltung aus **Fig.** 5 mit zusätzlicher Verdeutlichung des Datenflusses bei Anlegen der Befehlssignale aus **Fig.** 16;
[0040] **Fig.** 18 eine schematische Darstellung zur Verdeutlichung von Speichergrößen für zwei alternative Speicherbausteinentwürfe;
[0041] **Fig.** 19 eine Tabelle zur Darstellung verschiedener Speicherbausteinentwürfe einschließlich der beiden Entwürfe aus **Fig.** 18;
[0042] **Fig.** 20 ein Blockschaltbild einer Anordnung eines jeweiligen Blocks;
[0043] **Fig.** 21 ein Diagramm zur Darstellung einer Zeitsequenz für einen erfindungsgemäßen Datenladevorgang, um eine höhere Kapazität zu erreichen;
[0044] **Fig.** 22 ein schematisches Schaltbild eines Speicherbausteins mit Dualregister aus einer Voranmeldung zur Prioritätsanmeldung der vorliegenden Anmeldung;
[0045] **Fig.** 23 ein Flussdiagramm eines Programmierverfahrens aus der Voranmeldung;
[0046] **Fig.** 24 ein Schaltbild eines typischen Speicherbausteins zur Darstellung eines von der Erfindung aufgegriffenen Problems;
[0047] **Fig.** 25 eine grafische Darstellung einer Spannungsverteilung über eine Mehrzahl von Speicherzellen zur Veranschaulichung eines von der Erfindung aufgegriffenen Problems;
[0048] **Fig.** 26 ein Schaltbild eines erfindungsgemäßen Speicherbausteins mit Dualregister;
[0049] **Fig.** 27 ein Flussdiagramm eines erfindungsgemäßen Programmierverfahrens für den Speicherbaustein von **Fig.** 26;
[0050] **Fig.** 28 ein Zeitablaufdiagramm zur Darstellung von verschiedenen Signalen und ihrer zeitlichen Beziehungen bei dem erfindungsgemäßen Programmierverfahren; und
[0051] **Fig.** 29 eine grafische Darstellung einer Spannungsverteilung über eine Mehrzahl von Speicherzellen zur Veranschaulichung der erfindungsgemäßen Lösung.
[0052] Unter Bezugnahme auf **Fig.** 3 wird nachfolgend ein erfindungsgemäßer Speicherbaustein **100** beschrieben. Der Speicherbaustein **100** kann beispielsweise ein Flash-Speicher vom NAND-Typ sein und umfasst ein Speicherzellenfeld **110** zum Speichern von Daten, einen Seitenregister- und Abtastverstärkerblock **120** und eine Y-Gatterschaltung **130**, um in einer jeweiligen Gruppe von Speicherzellen gespeicherte Daten torgesteuert weiterzuleiten. Der Seitenregister- und Abtastverstärkerblock **120** ist zwischen dem Speicherzellenfeld **110** und der Y-Gatterschaltung **130** eingeschleift und umfasst einen Seitenpuffer **122**. Der Seitenpuffer **122** umfasst Dualregister entsprechend der Erfindung, wie nachfolgend im Detail beschrieben wird.
[0053] Der Speicherbaustein **100** umfasst weitere Komponenten, wie X-Pufferspeicher und -Decoder, Y-Pufferspeicher und -Decoder, ein Befehlsregister, eine Steuerlogik und einen Spannungsgenerator zum Erzeugen einer erhöhten Spannung, E/A-Puffer und -Zwischenspeicher sowie globale Puffer. Die Komponenten tauschen Daten, Adressen und Befehlssignale aus, wie gezeigt und nachfolgend beschrieben.
[0054] **Fig.** 4 zeigt eine Abtastanordnung für das Speicherzellenfeld **110**. Es sind mehrere Bitleitungen dargestellt, die abwechselnd als BLe und BLo bezeichnet sind, wobei der Zusatz „e" geradzahlige und der Zusatz „o" ungeradzahlige Bitleitungen repräsentieren. Mehrere Speicherzellen (M1, M2,..., Mm) sind mit je einer Bitleitung verbunden.
[0055] Eine Gruppe von Speicherzellen, beispielsweise die Gruppe M1, wird jeweils von einer einzigen Wortleitung gesteuert, beispielsweise von der Wortleitung WL1. Die Speicherzellen dieser Gruppe werden hier als Seiteneinheit bezeichnet.
[0056] Unter Bezugnahme auf **Fig.** 5 werden der Seitenregister- und Abtastverstärkerblock **120** und die Y-Gatterschaltung **130** im Detail beschrieben. Die Y-Gatterschaltung **130** ist zwischen dem Seitenregister- und

Abtastverstärkerblock **120** und einer Datenleitung **131** angeordnet. Die Datenleitung **131** kann für Datenbits D0 bis D7 vorgesehen sein.

[0057] Die Y-Gatterschaltung **130** umfasst zwei NMOS-Transistoren **132** und **133**. Die Transistoren **132** und **133** werden durch Signale YA und YB gesteuert, die von Informationen aus einer Spaltenadresse abgeleitet werden können.

[0058] Der Seitenregister- und Abtastverstärkerblock **120** umfasst einen einzelnen Seitenpuffer **122**, der eine Abtastleitung **125** und einen Abtastknoten E umfasst. Eine oder mehrere Bitleitungen können mit dem Seitenpuffer am Knoten E verbunden sein. Beim Ausführungsbeispiel aus **Fig.** 5 sind zwei Bitleitungen BLe und BLo mit dem Knoten E verbunden.

[0059] Ein Transistor **141** umfasst einen Sourceanschluß, der mit einer korrespondierenden Bitleitung BLe verbunden ist, einen Drainanschluß, der mit einem Knoten verbunden ist, der ein Signal VIRPWR zur Verfügung stellt, und einen Gateanschluß, der so angeschlossen ist, dass er ein Gatesteuersignal VBLe empfängt.

[0060] Ein Transistor **142** umfasst einen Sourceanschluß, der mit einer korrespondierenden Bitleitung BL0 verbunden ist, einen Drainanschluß, der mit dem Knoten verbunden ist, der das Signal VIRPWR zur Verfügung stellt, und einen Gateanschluß, der so angeschlossen ist, dass er ein Gatesteuersignal VBLo empfängt.

[0061] Der Knoten, der das Signal VIRPWR zur Verfügung stellt, wird von einer ersten oder einer zweiten Versorgungsspannung geladen. Entsprechend versorgen die Transistoren **141** und **142** in Abhängigkeit von den Gatesteuersignalen VBLe und VBLo die Bitleitungen BLe und BLo mit der ersten oder der zweiten Versorgungsspannung.

[0062] Zusätzlich verbindet ein NMOS-Transistor **143** in Abhängigkeit von einem Signal BLSHFe die Bitleitung BLe mit dem Knoten E. Ein NMOS-Transistor **143** verbindet in Abhängigkeit von einem Signal BLSHFo die Bitleitung BLo mit dem Knoten E.

[0063] Deshalb wird der Seitenpuffer **122** über den Knoten E der Abtastleitung **125** mit den Bitleitungen BLe und BLo verbunden. Ein PMOS-Transistor **148** versorgt die Bitleitungen BLe und BLo über die Abtastleitung **125** während eines Lesevorgangs mit Strom. Der PMOS-Transistor **148** ist zwischen der Versorgungsspannung und der Abtastleitung **125** eingeschleift und wird entsprechend einem Steuersignal PLOAD leitend oder sperrend geschaltet.

[0064] Wichtig ist, dass der Seitenpuffer **122** zwei Register **150** und **170** umfasst. Der Stand der Technik stellt nur ein solches Register zur Verfügung. Beide Register **150** und **170** sind mit der Abtastleitung verbunden.

[0065] Das zweite Register **150** wird auch als Hauptregister bezeichnet und umfasst zwei NMOS-Transistoren **151**, **152**, zwei Inverter **153**, **154** und einen PMOS-Transistor **155**. Die Daten werden in einem aus den Invertern **153**, **154** gebildeten Hauptzwischenspeicher **156** gespeichert. Der PMOS-Transistor **155** bildet eine Vorladeschaltung für den Hauptzwischenspeicher **156**.

[0066] Das erste Register **170** wird auch als Hilfs- oder Nebenregister bezeichnet und umfasst zwei NMOS-Transistoren **171, 172**, zwei Inverter **173**, **174** und einen PMOS-Transistor **175**. Die Daten werden in einem aus den Invertern **173**, **174** gebildeten Nebenzwischenspeicher **176** gespeichert. Der PMOS-Transistor **175** bildet eine Vorladeschaltung für den Nebenzwischenspeicher **176**.

[0067] Das aus den beiden Registern **150** und **170** gebildete Dualregister des Seitenpuffers **122** der vorliegenden Erfindung bietet viele Vorteile. So können Funktionen besser ausgeführt werden als beim Stand der Technik, was eine Vergrößerung des Seitenpuffers rechtfertigt.

[0068] Es werden zusätzliche Strukturen zur Verfügung gestellt, die einen Datenaustausch zwischen den beiden Seitenpufferregistern **150**, **170**, dem Speicherzellenfeld **110** und der Y-Gatterschaltung **130** ermöglichen und steuern.

[0069] Ein NMOS-Transistor **181**, der von einem Steuersignal PDUMP gesteuert wird, wird leitend geschaltet, um Daten zwischen dem Nebenregister **170** und dem Hauptregister **150** zu übertragen. Abwechselnd wird der Transistor **181** sperrend geschaltet, um das Nebenregister **170** vom Hauptregister **150** elektrisch zu trennen. Die Übertragung wird in vorteilhafter Weise über die Abtastleitung **125** durchgeführt. Der NMOS-Transistor **181** wird auch als Trennschalter bezeichnet.

[0070] NMOS-Transistoren **182**, **183** werden zum Speichern von Informationen im Nebenregister **170** zur Verfügung gestellt. Dies wird als Reaktion auf extern eingegebene Signale DI bzw. nDI durchgeführt.

[0071] Ein NMOS-Transistor **184** verbindet oder trennt das Hauptregister **150** mit oder von einer ausgewählten der Bitleitungen BLe, BLo. Dies wird durchgeführt, wenn zu programmierende Informationen vom Hauptregister **150** zu der ausgewählten der Bitleitungen BLe, BLo übertragen werden.

[0072] Ein NMOS-Transistor **185** wird von einem Steuersignal PBDO gesteuert. Der Transistor **185** gibt während eines ausgewählten Zeitintervalls ausgelesene Informationen über die ausgewählte Bitleitung nach außerhalb des Seitenpuffers **122** aus.

[0073] Ein Transistor **186** ist vorgesehen, um einen Programmierzustand zu überprüfen und eine Information über einen erfolgreichen oder nicht erfolgreichen Programmiervorgang an einem Knoten B des Hauptregisters **150** zur Verfügung zu stellen.

[0074] Nachfolgend werden erfindungsgemäße Verfahren beschrieben. Unter Bezugnahme auf die **Fig.** 6, 7,

8 und 4 werden zunächst erfindungsgemäße Programmierverfahren beschrieben. Unter einem Programmiervorgang wird verstanden, dass Daten in Speicherzellen des Bausteins von außerhalb des Bausteins eingegeben werden.

[0075] **Fig.** 6 zeigt ein Flussdiagramm **600** eines erfindungsgemäßen Programmierverfahrens. Das Programmierverfahren des Flussdiagramms **600** kann z.B. durch die Schaltung aus **Fig.** 3 ausgeführt werden.

[0076] Entsprechend einem Schritt **610** werden erste externe Daten über eine Y-Gatterschaltung, wie die Schaltung **130**, geführt. Die ersten externen Daten werden in einen Seitenpuffer, wie den Seitenpuffer **122**, übertragen. Dies können Einzeldaten oder eine Mehrzahl von Daten sein. Es kann auch eine ganze Datenseite sein.

[0077] Entsprechend einem nächsten Schritt **620** werden die ersten Daten aus Schritt **610** in einem ersten Register des Seitenpuffers gespeichert. Das erste Register kann das Nebenregister **170** sein.

[0078] Entsprechend einem optionalen Schritt **630** kann ein Schalter aktiviert werden, um das erste Register mit einem zweiten Register zu verbinden. Das zweite Register kann das Hauptregister **150** sein. Der Schalter kann der NMOS-Transistor **181** sein, der vom Steuersignal PDUMP gesteuert wird.

[0079] Entsprechend einem nächsten Schritt **640** werden die im ersten Register gespeicherten Daten im zweiten Register gespeichert.

[0080] Entsprechend einem optionalen Schritt **650** kann der Schalter aktiviert werden, um das erste Register vom zweiten Register zu trennen.

[0081] Entsprechend einem nächsten Schritt **660** werden die ersten im zweiten Register gespeicherten Daten in einer Zelle des Speicherzellenfeldes gespeichert, was auch als Programmieren bezeichnet wird. Gleichzeitig werden zweite externe Daten vom ersten Register empfangen und gespeichert. Deshalb kann ein Informationsspeichervorgang ohne eine Verlängerung der Informationsladezeit durchgeführt werden.

[0082] Im Ausführungsbeispiel aus **Fig.** 3 werden die gleichzeitigen Vorgänge im Schritt **660** dadurch ermöglicht, dass das erste Register und das zweie Register getrennt sind. Andere Verfahren sind ebenfalls möglich.

[0083] Unter Bezugnahme auf die **Fig.** 7 und 8 wird ein erfindungsgemäßes Programmierverfahren detaillierter beschrieben. **Fig.** 7 zeigt Befehlssignale, die an die Schaltung aus **Fig.** 5 angelegt werden können. Die horizontale Achse ist in neun Zeitsegmente aufgeteilt, die jeweils mit **1**, **2**, ..., **9** bezeichnet sind.

[0084] **Fig.** 8 zeigt, wie Daten als Reaktion auf die Befehlssignale aus **Fig.** 7 in der Schaltung aus **Fig.** 5 übertragen werden. Auf **Fig.** 8 ist unter Benutzung der Zeitsegmente aus **Fig.** 7 gemeinsam mit **Fig.** 7 Bezug zu nehmen.

[0085] In einem ersten Zeitsegment **1** wird eine Datenleitung **131** auf Massespannung gelegt und der Transistor **175** wird durch ein Signal PBSET leitend geschaltet. Dies wird auch als Setzen des Seitenpuffers auf eine erste Seite bezeichnet.

[0086] Danach ist im Zeitsegment **2** ein Knoten D des Nebenzwischenspeichers **176** auf einem hohen Zustand und die NMOS-Transistoren **132** und **133** werden leitend geschaltet. Daten mit dem Wert „0" oder „1" auf der Datenleitung werden deshalb im Nebenzwischenspeicher **176** durch Anlegen von Phasen der Signale DI und nDI gespeichert werden. Dies wird auch als Datenladevorgang der ersten Seite bezeichnet und korrespondiert in etwa mit dem oben beschriebenen Schritt **610**.

[0087] Dann werden im Zeitsegment **3** die gespeicherten Daten vom Nebenregister **176** zur Abtastleitung **125** übertragen. Dies wird durch einen Wechsel des Steuersignals PDUMP auf einen logisch hohen Zustand erreicht. Vor dem Übertragen der Daten ins Hauptregister **150** werden die Abtastleitung **125** und ein Knoten A des Zwischenspeichers **156** durch die Transistoren **148** bzw. **155** vorgeladen.

[0088] Danach werden im Zeitsegment **4** die Signale auf Null gelegt. Dieser Vorgang wird auch HV-Freigabe genannt.

[0089] Danach wird im Zeitsegment **5** die zugehörige der Bitleitungen BLe, BLo durch Vorladen gesetzt.

[0090] Dann laufen in den Zeitsegmenten **7** und **8** zwei Abläufe entsprechend dem oben beschriebenen Schritt **660** gleichzeitig ab. Die zu programmierenden Daten werden vom Hauptregister **150** durch Aktivierung des Signals BLSLT zur ausgewählten Bitleitung BLe übertragen und von dort in die Speicherzelle. Zusätzlich werden die nächsten zu programmierenden Daten von außerhalb des Speicherbausteins im Nebenregister **170** gespeichert bzw. in selbiges geladen.

[0091] Generell wird ein Datenladevorgang in Form von Byte-Einheiten durchgeführt, während ein Programmiervorgang in Form von Seiteneinheiten durchgeführt wird. Bei einem Datenladevorgang werden Daten von der Datenleitung in das Nebenregister **170** übertragen, während bei einem Programmiervorgang Daten vom Hauptregister **150** in die Speicherzellen des Speicherzellenfelds **110** übertragen werden. Wie oben ausgeführt, bedeutet eine Seiteneinheit, dass eine Mehrzahl von Speicherzellen mit einer einzelnen Wordleitung verbunden und gesteuert sind.

[0092] Da zwei Abläufe gleichzeitig ablaufen, wird die Datenspeichercharakteristik auch bei einem großen Datenvolumen beibehalten. Deshalb ist die Vergrößerung des Seitenpuffers durch die Implementierung desselben mit dem Nebenregister **170** ohne weiteres gerechtfertigt.

[0093] Dann wird im Zeitsegment **8** der Lesevorgang verifiziert und im Zeitsegment **9** werden die Bitleitungen

für den nächsten Lade- oder Programmiervorgang wieder vorgeladen.

[0094] Unter Bezugnahme auf die **Fig.** 9 und 10 wird nun ein Lesevorgang für das Ausführungsbeispiel aus **Fig.** 3 im Detail beschrieben. Es wird vorausgesetzt, dass Daten aus einer der Speicherzellen des Speicherzellenfeldes **110** ausgelesen werden und Gatesteuersignale der auszulesenden Speicherzellen geeignete Spannungen an Wortleitungen anlegen.

[0095] **Fig.** 9 zeigt Befehlsignale, die an die Schaltung aus **Fig.** 5 angelegt werden können. Die horizontale Achse ist in sechs Zeitsegmente aufgeteilt, die mit **1**, **2**,..., **6** bezeichnet sind.

[0096] **Fig.** 10 zeigt, wie Daten in der Schaltung aus **Fig.** 5 als Reaktion auf die Befehlssignale aus **Fig.** 9 übertragen werden. Auf **Fig.** 10 wird im Zusammenhang mit **Fig.** 9 Bezug genommen, wobei die gleichen Zeitsegmente wie in **Fig.** 9 verwendet werden.

[0097] Kurz gesagt, wird ein Auslesevorgang direkt durch das Hauptregister **150** unter Umgehung des Nebenregisters **170** durchgeführt. Auf diese Weise behindert das Nebenregister **170** nicht den Datenlesevorgang, während es den Datenladevorgang und den Datenprogrammiervorgang unterstützt, wie oben beschrieben ist.

[0098] Um einen stabilen Lesevorgang auszuführen, werden die Bitleitungen BLe und BLo zuerst über die NMOS-Transistoren **141** und **142** dadurch, dass das Signal VIRPWR auf Null gelegt wird und die Steuersignale VBLe und VBLo aktiviert werden, im Zeitsegment **1** entladen.

[0099] Hierbei wechselt das Signal PBRST gleichzeitig von einem hohen logischen Pegel auf einen niedrigen logischen Pegel, so dass der Zustand des Hauptregisters **150** bzw. ein Eingang des Inverters **153** auf einen vorbestimmten Zustand gesetzt wird, d.h. auf einen logisch hohen Pegel.

[0100] Anschließend geht im Zeitsegment **2** das Signal PLOAD auf einen niedrigen Pegel, wodurch der PMOS-Ladetransistor **148** leitend geschaltet wird. Das Steuersignal BLSHFe des NMOS-Transistors **143** wird so eingestellt, dass seine Spannung der Summe einer Bitleitungsvorladespannung und einer Schwellwertspannung des NMOS-Transistors **143** entspricht. Nach dem Vorladen der Bitleitung BLe mit einer passenden Spannung geht das Signal BLSHFe auf den niedrigen Pegel der Massespannung.

[0101] Die vorgeladene Spannung der Bitleitung wird im Zeitsegment **3** entsprechend dem Zustand der ausgewählten Speicherzelle verändert. Beispielsweise wird für den Fall, dass die ausgewählte Speicherzelle eine Aus-Zelle ist, die vorgeladene Spannung der Bitleitung auf ihrem Pegel gehalten. Für den Fall, dass die ausgewählte Speicherzelle eine An-Zelle ist, wird die vorgeladene Spannung der Bitleitung abgesenkt.

[0102] Wird die Spannung des Signals BLSHFe auf eine Zwischenspannung zwischen der Vorladespannung und des vorherigen Pegels des Signals BLSHFe verändert, dann wird eine Spannung auf der Abtastleitung **125** durch Sperren des NMOS-Transistors **143** auf dem Pegel der Versorgungsspannung gehalten, wenn die ausgewählte Speicherzelle eine Aus-Zelle ist. Wenn nicht, wird hingegen die Spannung auf der Abtastleitung **125** mit der Spannung der Bitleitung BLe bzw. synchronisiert mit der Bitleitung BLe abgesenkt. An einem Mittelpunkt, an dem das Signal BLSHFe auf einen logisch niedrigen Pegel der Massespannung geht, geht das Signal PLOAD auf den Pegel der Versorgungsspannung.

[0103] Danach geht im Zeitsegment **4** das Gatesteuersignal PBLCHM des NMOS-Transistors **152** auf den logisch hohen Pegel der Versorgungsspannung und der NMOS-Transistor **151** wird leitend oder sperrend geschaltet, entsprechend dem Zustand der Abtastleitung **125**. Daraus resultiert, dass der Zustand der Abtastleitung **125** im Hauptregister **150** gespeichert wird.

[0104] Dann werden im Zeitsegment **6** die im Hauptregister **150** gespeicherten Daten über den NMOS-Transistor **185**, der vom Steuersignal PBDO gesteuert wird, und als nächstes über die Y-Gatterschaltung **130** zur Datenleitung übertragen.

[0105] Nun werden erfindungsgemäße Rückkopierverfahren beschrieben. Während der Durchführung eines Lesevorgangs kann es erforderlich sein, einen Seitenkopiervorgang durch Kopieren der von einer ersten Seite der Speicherzellen mit einer ersten Adresse ausgelesenen Daten auf eine zweite Seite der Speicherzellen mit einer zweiten Adresse durchzuführen.

[0106] Unter Bezugnahme auf **Fig.** 11 wird ein Flussdiagramm **1100** benutzt, um das erfindungsgemäße Rückkopierverfahren zu erklären. Das Verfahren des Flussdiagramms **1100** kann z.B. mit dem Baustein **100** aus **Fig.** 3 durchgeführt werden.

[0107] Entsprechend einem Schritt 1110 werden Daten einer ersten Zelle in einem ersten Register eines Seitenpuffers gespeichert. Dies kann durch Auslesen der Daten in das Nebenregister **170** durchgeführt werden. Das Auslesen kann wie oben beschrieben durchgeführt werden.

[0108] Entsprechend einem nächsten Schritt 1120 werden die im ersten Register gespeicherten Daten in einem zweiten Register des Seitenpuffers gespeichert. Dies kann durch eine Übertragung der ausgelesenen Daten zwischen dem Nebenregister **170** und dem Hauptregister **150** durchgeführt werden. Diese Übertragung kann optional eine Aktivierung eines Schalters umfassen, der das erste Register mit dem zweiten Register verbindet.

[0109] Entsprechend einem nächsten Schritt 1130 werden die Daten des zweiten Registers in einer zweiten Zelle des Speicherzellenfeldes gespeichert. Dies kann als Programmiervorgang durchgeführt werden, wie oben beschrieben ist.

[0110] Unter Bezugnahme auf die **Fig.** 12, 13 und 14 wird nun ein Rückkopiervorgang des Bausteins aus **Fig.** 3 detaillierter beschrieben. Es wird vorausgesetzt, dass Daten aus originalen Speicherzellen des Feldes **110** in den Seitenpuffer **122** ausgelesen und in andere Zellen des Feldes **110** zurückkopiert werden.

[0111] **Fig.** 12 zeigt Befehlssignale, die hierzu an die Schaltung aus **Fig.** 5 angelegt werden können. Die horizontale Achse ist in elf Zeitsegmente aufgeteilt, die entsprechend mit **1**, **2**,..., **11** bezeichnet sind. Die Daten werden erst aus den Zellen in den Seitenpuffer ausgelesen. Wie aus **Fig.** 12 ersichtlich ist, entsprechen die ersten vier Zeitsegmente **1**, **2**, **3**, **4** im Wesentlichen den ersten vier Zeitsegmenten aus **Fig.** 10 mit der Ausnahme, dass die Daten in das Nebenregister **10** anstatt ins Hauptregisters **150** ausgelesen werden.

[0112] **Fig.** 13 zeigt das Auslesen der Daten in den Seitenpuffer. Zudem wird ein Leerfeld an der Stelle dargestellt, an welcher der in **Fig.** 2 dargestellte Stand der Technik das zusätzliche Anzeigebit zur Anzeige der Polarität der gespeicherten Daten aufweist, d.h. ob die Daten invertiert sind oder nicht.

[0113] Wie wiederum aus **Fig.** 12 ersichtlich ist, werden die Daten dann vom Nebenregister **170** ins Hauptregister **150** des Seitenpuffers übertragen. Dies geschieht während der Zeitsegmente **5** und **6**.

[0114] Dann werden die Daten vom Hauptregister **150** während der Zeitsegmente **7**, **8**, **9**, **10**, **11** in andere Zellen des Speichers programmiert. Wie ersichtlich ist, sind die Befehlssignale während der Zeitsegmente **5** bis **11** im Wesentlichen die gleichen wie in **Fig.** 8.

[0115] **Fig.** 14 zeigt die rückprogrammierten Daten. Ersichtlich werden die Daten erfindungsgemäß in anderen Zellen gespeichert, ohne ihren Zustand, den sie in den ursprünglichen Zellen hatten, zu invertieren. Entsprechend besteht kein Bedarf für das Anzeigebit aus **Fig.** 2, wodurch zusätzlich Platz gespart wird.

[0116] Nun werden erfindungsgemäße Löschverfahren beschrieben. Generell werden durch einen Löschvorgang Daten entladen. In einem Flash-Speicher geht die Schwellwertspannung durch Anlegen einer höheren Spannung an die Speicherzellen auf einen Wert zwischen –1 V und –3 V. Dadurch werden Daten in Registern entladen.

[0117] In **Fig.** 15 wird ein Flussdiagramm **1500** benutzt, um einen erfindungsgemäßen Verifizierlesevorgang zu erklären, der nach einem Löschvorgang durchgeführt wird. Das Verfahren gemäß dem Flussdiagramm **1500** kann z.B. von der Schaltung aus **Fig.** 3 ausgeführt werden.

[0118] Entsprechend einem Schritt 1510 werden Daten aus einer ersten Speicherzelle durch ein erstes Register des Seitenpuffers entladen.

[0119] Entsprechend einem weiteren Schritt 1520 werden die im ersten Register des Seitenpuffers gespeicherten Daten durch ein zweites Register entladen.

[0120] Entsprechend einem optionalen Schritt 1530 werden die im ersten Register gespeicherten Daten z.B. durch den Transistor **186** im Vergleich zum Zustand der Speicherzelle auf „richtig" oder „falsch" geprüft.

[0121] Unter Bezugnahme auf die **Fig.** 16 und 17 wird nun ein Löschverfahren für den Baustein aus **Fig.** 3 beschrieben. **Fig.** 16 zeigt Befehlssignale, die an die Schaltung aus **Fig.** 5 angelegt werden können. Die horizontale Achse ist in sieben Zeitsegmente aufgeteilt, die mit **1**, **2**,..., **7** bezeichnet sind.

[0122] **Fig.** 17 zeigt, wie Daten als Reaktion auf die Befehlssignale aus **Fig.** 16 in der Schaltung aus **Fig.** 5 gelöscht werden. Auf **Fig.** 17 wird gemeinsam mit **Fig.** 16 Bezug genommen, wobei auf die Zeitsegmente aus **Fig.** 16 verwiesen wird.

[0123] Während der Zeitsegmente **1** und **2** wird ein Löschauslösebefehl empfangen. Während des Zeitsegments **3** werden die Bitleitungen BLe, BLo zum Entladen auf Masse gelegt. Während des Zeitsegments **4** wird der Verifizierungslesevorgang für eine erste Zelle durchgeführt. Während des Zeitsegments **5** wird ein Verifizierungslesevorgang für eine zweite Zelle durchgeführt.

[0124] Während des Zeitsegments **6** werden Daten durch das erste Register entladen. Die Daten umfassen Daten aus einer Speicherzelle und auch Daten aus dem Hauptregister **150** und aus dem Nebenregister **170** des Seitenpuffers. Während des Zeitsegments **7** wird eine verdrahtete ODER-Funktion ausgeführt und es werden Daten vom Knoten B des Hauptregisters **150** entladen.

[0125] Die Erfindung bietet den Vorteil, dass auch bei einer Vergrößerung der Seitengröße die Programmierzeit bzw. die Informationsspeicherzeit des Speichers nicht oder nur leicht erhöht wird. Zusätzlich erhöht sich die Zeitdauer zum Laden von Informationen in die Seitenpufferschaltung proportional zur Vergrößerung der Seite.

[0126] Unter Bezugnahme auf die **Fig.** 18, 19, 20, 21 werden Beispiele beschrieben, wie große Datenvolumen in den Speichern behandelt werden. Dadurch wird die Effektivität der Erfindung verdeutlicht.

[0127] **Fig.** 18 veranschaulicht anhand von zwei Fällen A und B, wie große Speichervolumina für die Kapazität eines Speicherbausteins gezählt werden.

[0128] Ein dreidimensionaler Block veranschaulicht die gesamte Speicherkapazität des Bausteins. Der dreidimensionale Block kann als Stapel von Einzelblöcken und die Einzelblöcke können als Stapel von Seiten angesehen werden. Jede Seite und damit auch jeder Einzelblock ist ein Byte (1B) breit. Ein Byte entspricht acht Bits, die mit I/O0 bis I/O7 bezeichnet sind.

[0129] Im Fall A i st eine Seite (52 + 16) 528 Byte lang. Unter der Voraussetzung, dass ein Einzelblock **32** Seiten umfasst, bildet eine Kapazität von 2048 Einzelblöcken einen Baustein mit 264 MBit.

[0130] Im Fall B ist es durch die Erfindung möglich, dass eine Seite (2048 + 64) 2112 Byte lang ist. Unter der Voraussetzung, dass ein Einzelblock vierundsechzig Seiten umfasst, bildet eine Kapazität von 1024 Einzelblöcken einen Speicherbaustein von 1 GBit.
[0131] **Fig.** 19 zeigt verschiedene Entwurfsmöglichkeiten für Speicherbausteine, einschließlich der Entwürfe für die Bausteine A und B aus **Fig.** 18.
[0132] **Fig.** 20 zeigt, wie ein Block aus zweiunddreißig Seiten, wie der Baustein A aus **Fig.** 18, zu einem Block mit vierundsechzig Seiten, wie der Baustein B aus **Fig.** 18, durch Bestimmung von Daten von aufeinander folgenden Seiten als „gerade" und „ungerade" rekonfiguriert werden kann.
[0133] Die Erfindung erreicht schnellere Ladezeiten als der Stand der Technik. Dies wird durch folgendes Beispiel veranschaulicht: Angenommen: T1 = Ladezeit von 1B = 1 μs
F2 = 1 Seite (für zwei Fälle von 528 B und 2112 B)
T3 = Programmierzeit = 200 μs
F4 = 1 Block (hier zweiunddreißig Seiten)
[0134] Dann wird die vom aus dem Stand der Technik bekannten Baustein benötigte Zeitdauer für eine Sequenz von Datenladevorgang, Programmiervorgang, Datenladevorgang, Programmiervorgang usw. durch folgende Gleichung 1 bestimmt:

$$\text{Gesamtzeit (Stand der Technik)} = [(T1 \bullet F2) + T3] \bullet F4 \quad (1)$$

[0135] D.h. es ergibt sich für einen Baustein mit 528 B eine Gesamtzeit von 8089,6 μs und für einen Baustein von 2112 B eine Gesamtzeit von 13158,4 μs. Entsprechend ist es nicht möglich, ein großes Informationsvolumen in einer kurzen Zeit in den Seitenpuffer zu laden, wodurch sich die Infomationsspeichercharakteristik verschlechtert.
[0136] Wie aus **Fig.** 21 ersichtlich ist, werden durch die Erfindung Daten effektiver geladen und programmiert. Die Gesamtzeit ergibt sich aus folgender Gleichung 2:

$$\text{Gesamtzeit (Erfindung)} = (T1 \bullet F2) + (T3 \bullet F4) \quad (2)$$

[0137] Dies ergibt für einen Baustein mit 2112 B eine Gesamtzeit von 6611,2 μs. Dies entspricht etwa der Hälfte der vergleichbaren Gesamtzeit aus Gleichung 1. Dadurch können nun Seitenpufferschaltungen mit einem großen Volumen, beispielsweise über 2048 B, benutzt werden.
[0138] Die **Fig.** 24 bis 29 beziehen sich auf ein anderes erfindungsgemäßes Ausführungsbeispiel, das nachfolgend im Detail beschrieben wird.
[0139] **Fig.** 24 zeigt ein Schaltbild eines Speicherzellenfeldes **100** in einem NAND-Flash-Speicherbaustein. Das Speicherzellenfeld **100** umfasst eine Mehrzahl von Ketten, die eine Mehrzahl von Speicherzellen umfassen. Jede der Ketten ist mit einer Bitleitung verbunden. Die Ketten sind parallel mit einer gemeinsamen Sourceleitung CSL verbunden. Die gemeinsame Sourceleitung CSL ist mit Masse verbunden.
[0140] In diesem NAND-Flash-Speicherbaustein werden alle mit einer Wortleitung verbundenen Speicherzellen gleichzeitig programmiert. In anderen Worten ausgedrückt, es werden alle Speicherzellen MC1 entsprechend dem Zustand der Bitleitung programmiert, wenn die Wortleitung WL1 freigeschaltet wird. Ist der Zustand der Bitleitung „0", dann werden die Speicherzellen programmiert. Ist der Zustand der Bitleitung „1 ", dann werden die Speicherzellen nicht programmiert.
[0141] Danach werden während eines Programmierverifizierungsvorgangs die Zustände der Speicherzellen in einem Datenknoten zwischengespeichert, beispielsweise im Knoten N3 des Zwischenspeichers LATCH2 aus **Fig.** 26. Ist der Zustand der Bitleitung „0", dann wurden nicht alle Speicherzellen im ersten Programmierschritt programmiert.
[0142] Normalerweise sind die Zellen nach mehreren Schritten des Programmiervorgangs erfolgreich programmiert. Da die Ankoppelverhältnisse der einzelnen Speicherzellen sich durch Toleranzen beim Herstellungsprozess unterscheiden können, werden nicht alle zu programmierenden Speicherzellen notwendigerweise während eines einzigen Zyklus oder Schritts des Programmiervorgangs programmiert, auch wenn die Bitleitung den Programmierzustand „0" hat. Allgemein werden vor dem Programmiervorgang alle Speicherzellen eines NAND-Flash-Speicherbausteins gelöscht. Entsprechend haben alle Speicherzellen eine negative Schwellwertspannung. Nach mehreren Programmierschritten auf der ersten Seite gehen alle Speicherzellen auf eine positive Schwellwertspannung, die über einer Verifizierungsspannung liegt. Bei einer gegebenen Seite mit einer Mehrzahl von Speicherzellen werden, wenn die erste Seite den ersten Programmierschritt beendet hat, während eines Programmierverifizierungsvorgangs alle Speicherzellen daraufhin überprüft, ob ihre Schwellwertspannung unter der Verifizierungsspannung liegt oder nicht. Die Verifizierungsspannung ist in **Fig.** 25 dargestellt. Hierbei liegen die meisten der Speicherzellen aus den oben genannten Gründen typischerweise unterhalb des Bereichs der Verifizierungsspannung, auch wenn ein Teil der Speicherzellen erfolgreich mit dem Wert „0" programmiert wurden.

[0143] Unter Bezugnahme auf **Fig.** 24 steigt während des Programmierverifizierungsvorgangs der Spannungspegel der gemeinsamen Sourceleitung CSL wegen Widerständen R0, R1, R2,..., Rm und Strömen Ic0, Ic1, Ic2, ..., Icm an. Dies ergibt sich aus dem Ohmschen Gesetz U = 1 • R. Dem Fachmann ist verständlich, dass die Widerstände R0, R1, R2,..., Rm parasitäre Widerstände der gemeinsamen Sourceleitung repräsentieren und die Ströme Ic0, Ic1, Ic2,..., Icm diejenigen Ströme repräsentieren, die von jeder Bitleitung zur gemeinsamen Sourceleitung fließen. Diese Ströme fließen durch Zellen, die im gelöschten Zustand verblieben sind oder die nicht vollständig programmiert sind.

[0144] Daraus resultiert, dass der Spannungspegel der gemeinsamen Sourceleitung CSL wegen des Stromflusses durch die Ketten ansteigt. Diese Fluktuation des Spannungspegels der gemeinsamen Sourceleitung CSL wird auch als CSL-Rauschen bezeichnet.

[0145] Dieses Phänomen tritt wegen des Zustands des Speicherbausteins nach dem ersten Programmierschritt leichter auf. Nach einigen Programmierschritten ist das Phänomen hingegen minimiert, weil der Stromfluss durch die Speicherzellen minimal ist.

[0146] Unter Bezugnahme auf **Fig.** 25 setzt der Zwischenspeicher LATCH2 während des Programmierverifizierungsvorgangs den Knoten N3 wegen des CSL-Rauschens auf den Programmierzustand „1", auch wenn die Schwellwertspannung der Speicherzelle aktuell unterhalb des Pegels der Verifizierungsspannung liegt. Daraus resultiert, dass die nicht ausreichend programmierte Speicherzelle fälschlicherweise und irreführend als ausreichend bzw. erfolgreich programmierte Speicherzelle gekennzeichnet wird.

[0147] Hat beispielsweise die Speicherzelle MCO nach dem ersten Programmierschritt eine Schwellwertspannung von 0,3 V und der Pegel auf der gemeinsamen Sourceleitung CSL ist 0,7 V, dann geht die Schwellwertspannung der Speicherzelle MCO während des Programmierverifizierungsvorgangs auf 0,7 V. Hat die Verifizierungsspannung einen Wert von 0,7 V, dann wird die Speicherzelle MCO im Seitenpuffer als programmierte Speicherzelle gekennzeichnet. Entsprechend geht der Knoten N3 des Zwischenspeichers LATCH2 auf den Zustand „1 ".

[0148] In anderen Worten ausgedrückt, der Knoten N3 des Zwischenspeichers LATCH2 ist auf dem hohen Zustand „1", obwohl die Speicherzelle MCO aus **Fig.** 24 nicht ausreichend programmiert ist. Wird die Speicherzelle im zweiten Schritt programmiert, dann wird die Schwellwertspannung der Speicherzelle MCO mit dem Wert 0,3 V nicht verändert, weil der Knoten N3 des Zwischenspeichers LATCH2 auf seinem Zustand „1" verbleibt. Es ist eine Absicht der Erfindung, dieses Problem zu lösen.

[0149] Eine weitere Absicht der Erfindung ist es, Speicherzellen, die nicht programmiert werden sollen, in einen Programmiersperrzustand zu versetzen, und Speicherzellen, die programmiert werden sollen, erneut zu programmieren, auch wenn fälschlicherweise während des Programmierverifizierungsvorgangs angezeigt wird, dass die Speicherzelle einen programmierten Zustand angenommen hat.

[0150] **Fig.** 26 veranschaulicht die vorliegende Erfindung durch eine schematische Darstellung. Wie aus **Fig.** 26 ersichtlich ist, umfasst das erfindungsgemäße Ausführungsbeispiel eine Speicherschaltung und eine Rückspeicherschaltung, wie sie in der älteren Anmeldung (US-Anmeldenummer 10/013191) nicht dargestellt ist.

[0151] Unter Bezugnahme auf **Fig.** 26 und 27 wird dieses Beispiel der vorliegenden Erfindung wie folgt erklärt. Wie aus **Fig.** 26 ersichtlich ist, umfasst ein Seitenpuffer einen ersten Abtastverstärker **1**, einen zweiten Abtastverstärker **2**, eine Richtig/Falsch-Überprüfungsschaltung, eine Speicherschaltung und eine Rückspeicherschaltung. Dem Fachmann ist ersichtlich, dass der Abtastverstärker **1** bzw. **2** in der älteren Anmeldung als Register bezeichnet ist.

[0152] Im Schritt F1 im Flussdiagramm aus **Fig.** 27 werden Daten, die programmiert werden sollen, und Daten, die programmiergesperrt werden sollen, in einen Knoten N4 eines Datenregisters LATCH1 geladen. Die zu programmierenden Daten haben einen Wert „0", d.h. Massepotential, und die programmiergesperrten Daten haben einen Wert „1", d.h. das Potential der Versorgungsspannung VDD.

[0153] Im Schritt F2 aus **Fig.** 27 werden die Daten mit dem Wert „0" und dem Wert „1" zu einem Knoten N_DATA entladen. Vor dem Schritt F2 wird der Knoten N_DATA entsprechend einem Signal PRE auf den Versorgungsspannungspegel VDD vorgeladen.

[0154] Im Schritt F3 werden die Daten am Knoten N4 durch einen Transistor TR12 zum Knoten N3 eines weiteren Datenregisters LATCH2 entladen. Der Phasenzustand der Daten am Knoten N3 ist der gleiche wie der Phasenzustand der Daten am Knoten N4 und ist invertiert zum Phasenzustand der Daten am Knoten N_DATA in der Speicherschaltung.

[0155] Im Schritt F4 werden die Speicherzellen entsprechend dem Zustand am Knoten N3 des weiteren Datenregisters LATCH2 programmiert. Ist der Zustand am Knoten N3 „0", dann wird die betreffende Speicherzelle programmiert. Ist der Zustand am Knoten N3 „1 ", dann wird die Speicherzelle nicht programmiert. Der Programmierzustand bedeutet, dass die Schwellwertspannung des Speichers auf einen Pegel über der Verifizierungsspannung geht, wobei die Verifizierungsspannung einen Zwischenpegel zwischen der Schwellwertspannung einer programmierten Speicherzelle und einer gelöschten Speicherzelle hat.

[0156] Im Schritt F5 wird der Knoten N3 entsprechend dem Zustand der Speicherschaltung zurückgespei-

chert. Ist der Zustand am Knoten N_DATA „1 ", dann wird der Knoten N3 auf den Wert „0" zurückgesetzt. Ist der Zustand am Knoten N_DATA „0", dann behält der Knoten N3 seine vorherigen Daten.

[0157] Im Schritt F6 wird der Programmierverifizierungsvorgang durchgeführt. In einem ersten Programmierverifizierleseschritt wird eine nicht ausreichend programmierte Speicherzelle im Datenregister LATCH2 als eine programmierte Zelle angezeigt. Aber nach mehreren Programmierschritten wird die Speicherzelle als Zelle angezeigt, die nicht programmiert ist, weil das CSL-Rauschen reduziert ist. Da der Knoten N3 entsprechend dem Zustand in der Speicherschaltung auf den Wert „0" zurückgesetzt ist, wird die nicht ausreichend programmierte Speicherzelle während des nächsten Programmierschritts programmiert.

[0158] Im Schritt F7 wird der Zustand des Knotens N3 im Datenregister LATCH2 in der Richtig/Falsch-Überprüfungsschaltung überprüft. Ist der Zustand am Knoten N3 „1 ", dann ist der Programmiervorgang beendet. Wenn nicht, dann kehrt das Verfahren zum Schritt F4 zurück.

[0159] **Fig.** 28 zeigt ein Zeitablaufdiagramm des erfindungsgemäßen Programmier- und Verifizierverfahrens. Die Schritte F1 bis F7 sind entlang der horizontalen Achse aufgetragen, während die verschiedenen Steuer- und Datensignale entlang der vertikalen Achse aufgetragen sind. Die Steuersignale umfassen Signale eines X-Decoders wie SSL, ausgewählte Wortleitung W/L (Sel.), nicht ausgewählte Wortleitung W/L, GSL und gemeinsame Sourceleitung CSL. Sie umfassen ebenfalls die Signale des Seitenpuffers wie Versorgungsspannung VIRPWR, Spannung einer geraden Bitleitung VBLe, Spannung einer ungeraden Bitleitung VBLo, Verschiebespannung einer geraden Bitleitung BLSHFe, Verschiebespannung einer ungeraden Bitleitung BLSHFo, Gattersteuersignal PBLCHM, PBLCHC, PLOAD, PBset, PDUMP1, ausgewählte Bitleitung BLSLT, Dateneingang DI, invertierter Dateneingang nDI, Vorladung PRE, RESET, PDUMP2 und DATA LINE. Diese Signale betreffen bekannte Signale oder sind in der o.g. älteren Anmeldung beschrieben.

[0160] Wie aus **Fig.** 28 ersichtlich, eilt entsprechend der Erfindung das Signal PDUMP2 aus Schritt F2 dem Signal PDUMP1 aus Schritt 3 vor, so dass der vorherige Zustand am Knoten N3 des Datenregisters LATCH2 temporär gespeichert wird, um den Zustand am Knoten N3 für den Fall wiederherzustellen, dass das Bit nochmals durch einen Rücksprung zum Schritt 4 programmiert werden muss, wie oben beschrieben ist.

[0161] Tabelle 1 zeigt typische Spannungswerte für die Programmier- und Verifiziervorgänge zum Programmieren der hier beschriebenen Speicherbausteine.

	WL (ausgewählt)	WL (nicht ausgewählt)	BL (programmiert)	BL (gesperrt)
Programmieren	18V	12V	0V	Vcc
Verifizieren	1V	4,5V	0,8V	0,8V

Tabelle 1

[0162] Die Wortleitungsspannung nimmt mit der programmierten Spannung schrittweise zu, wobei sich folgender Ablauf einstellt:

15,5 V → Verifizieren → 16 V → Verifizieren → 16,5 V → ... usw.

[0163] In Verbindung mit einem erfindungsgemäßen Ausführungsbeispiel ist die maximale Anzahl der Schritte bzw. die Anzahl der Perioden gleich zwölf und das Schrittspannungsinkrement beträgt 0,5 V/Schritt. Selbstverständlich können auch eine andere maximale Anzahl von Schritten und andere Werte für die Schrittspannung gewählt werden. In der Regel wird der Programmiervorgang innerhalb von fünf oder sechs Schritten abgeschlossen, so dass die Maximalzahl nicht erreicht wird.

[0164] **Fig.** 29 zeigt einen Kurvenverlauf einer Spannungsverteilung über eine Mehrzahl von Speicherzellen nach einem erfindungsgemäßen Programmiervorgang. Wie im Gegensatz zur **Fig.** 25 ersichtlich ist, steigt beim erfindungsgemäßen Programmierverfahren von **Fig.** 29 die Anzahl der erfolgreich programmierten Bits signifikant an, indem alle oder nahezu alle mit dem Datenwert „0" programmierten Zellen beim Programmieren effektiv auf eine höhere Schwellwertspannung angehoben werden, die über der Verifizierungsspannung liegt. Dies wird durch das Fehlen jeglicher Überlappung zwischen den mit dem Datenwert „0" programmierten Bits, die durch die Glockenkurve auf der rechten Seite des Kurvenverlaufs repräsentiert werden, und dem Pegel der Verifizierungsspannung verdeutlicht, der durch eine gestrichelte Linie repräsentiert wird.

Patentansprüche

1. Nichtflüchtiger Speicherbaustein mit

– einem Speicherzellenfeld (**110**) zum Speichern von Daten,

– einer Y-Gatterschaltung (**130**) zum gesteuerten Durchleiten von in einer jeweiligen Gruppe der Speicherzel-

len gespeicherten Daten,
– einem Seitenpuffer (**120**), der über einen Abtastknoten (N1) zwischen dem Speicherzellenfeld (**110**) und der Y-Gatterschaltung (**130**) eingeschleift ist und einen ersten Abtastverstärker und einen zweiten Abtastverstärker umfasst, und
– einer Speicherschaltung zum Speichern von Daten des ersten Abtastverstärkers in Abhängigkeit von einem Speichersignal, **dadurch gekennzeichnet**, dass
– der erste Abtastverstärker ein erstes Datenregister (LATCH1) und der zweite Abtastverstärker ein zweites Datenregister (LATCH2) umfasst, wobei das zweite Datenregister (LATCH2) gemeinsam mit dem ersten Datenregister (LATSCH1) funktionswirksam mit dem Abtastknoten (N1) gekoppelt ist.

2. Nichtflüchtiger Speicherbaustein nach Anspruch 1, gekennzeichnet durch eine Richtig/Falsch-Überprüfungsschaltung, die funktionswirksam mit dem zweiten Abtastverstärker gekoppelt ist und ein Signal erzeugt, das anzeigt, ob eine Speicherzelle des Speicherzellenfelds (**110**) erfolgreich programmiert wurde.

3. Nichtflüchtiger Speicherbaustein nach Anspruch 1 oder 2, gekennzeichnet durch eine Rückspeicherschaltung, die funktionswirksam zwischen den zweiten Abtastverstärker und der Speicherschaltung eingeschleift ist, um den Inhalt des zweiten Datenregisters (LATCH2) entsprechend dem Inhalt der Speicherschaltung in Abhängigkeit von einem Zurückspeichersignal (RESTORE) zurückzusetzen.

4. Nichtflüchtiger Speicherbaustein nach einem der Ansprüche 1 bis 3, dadurch gekennzeichnet, dass die Speicherschaltung einen ersten Transistor (TR19) umfasst, dessen Gateanschluß vom Speichersignal getrieben wird, dessen Sourceanschluß oder Drainanschluß mit dem ersten Datenregister (LATCH1) verbunden ist und dessen verbleibender Drainanschluß oder Sourceanschluß mit einem Sourceanschluß oder Drainanschluß eines zweiten Transistors (TR16) an einem Datenknoten (N_DATA) verbunden ist, wobei der verbleibende Drainanschluß oder Sourceanschluß des zweiten Transistors (TR16) mit einer Referenzspannung verbunden ist und dessen Gateanschluß von einem Vorladesignal (PRE) getrieben wird und wobei der Datenknoten (N_DATA) ein Ausgang der Speicherschaltung ist.

5. Nichtflüchtiger Speicherbaustein nach einem der Ansprüche 2 bis 4, dadurch gekennzeichnet, dass die Richtig/Falsch-Überprüfungsschaltung einen ersten Transistor (TR15) umfasst, von dem ein Gateanschluß mit dem zweiten Datenregister (LATCH2) verbunden ist und ein zugehöriger Sourceanschluß oder Drainanschluß mit einer Referenzspannung verbunden ist und ein verbleibender Drainanschluß oder Sourceanschluß mit einem invertierten Überprüfungssignal verbunden ist.

6. Nichtflüchtiger Speicherbaustein nach einem der Ansprüche 3 bis 5, dadurch gekennzeichnet, dass die Rückspeicherschaltung einen ersten Transistor (TR18) umfasst, dessen Sourceanschluß oder Drainanschluß mit einem Ausgang des zweiten Datenregisters (LATCH2) verbunden ist und dessen verbleibender Drainanschluß oder Sourceanschluß mit einem Sourceanschluß oder Drainanschluß eines zweiten Transistors (TR17) verbunden ist, dessen verbleibender Drainanschluß oder Sourceanschluß mit einer Referenzspannung verbunden ist, wobei ein Gateanschluß des ersten Transistors vom Zurückspeichersignal (RESTORE) getrieben wird.

7. Nichtflüchtiger Speicherbaustein nach einem der Ansprüche 1 bis 6, dadurch gekennzeichnet, dass das erste Datenregister (LATCH1) mit dem zweiten Datenregister (LATCH2) über einen ersten Transistor (TR12) verbunden ist, dessen Gateanschluß durch ein erstes Entladesignal (PDUMP1) getrieben wird, wobei das zweite Datenregister (LATCH2) benutzt wird, um eine Speicherzelle in der Speicherpufferspeicherschaltung zu programmieren, und wobei das zweite Datenregister (LATCH2) nach einer solchen Programmierung mit den in der Speicherschaltung gespeicherten Daten zurückgeladen wird.

8. Nichtflüchtiger Speicherbaustein nach einem der Ansprüche 1 bis 7, dadurch gekennzeichnet, dass das Speicherzellenfeld aus nicht flüchtigen Speicherzellen aufgebaut ist und die Speicherschaltung invertierte Daten des ersten Datenregisters (LATCH1) speichert.

9. Programmiervorrichtung für einen Speicherbaustein mit
– einem Cache-Datenregister,
– einem Hauptdatenregister, in das Daten aus dem Cache-Datenregister zum Programmieren gespeichert werden,
– einer Speicherschaltung, in die Daten aus dem Cache-Datenregister zum Verifizieren gespeichert werden, und
– einer Richtig/Falsch-Überprüfungsschaltung zum Bestimmen, ob die Daten aus dem Hauptdatenregister er-

folgreich programmiert sind, dadurch gekennzeichnet, dass
– ausgewählte Speicherzellen programmiert werden und nach dem Programmiervorgang ein erfolgreiches Programmieren verifiziert wird und im Falle eines nicht erfolgreichen Programmierens von einer oder mehreren der ausgewählten Speicherzellen der Programmiervorgang mit einer schrittweise erhöhten Programmierspannung wiederholt wird, bis eine erfolgreiche Programmierung aller zu programmierenden Speicherzellen festgestellt wird.

10. Programmiervorrichtung nach Anspruch 9, gekennzeichnet durch eine Rückspeicherschaltung, die den Inhalt des Hauptdatenregisters entsprechend dem Inhalt der Speicherschaltung zurücksetzt.

11. Programmiervorrichtung nach Anspruch 9 oder 10, dadurch gekennzeichnet, dass die Speicherschaltung einen ersten Transistor umfasst, dessen Gateanschluß vom Speichersignal getrieben wird, dessen Sourceanschluß oder Drainanschluß mit einem Ausgang des Cache-Datenregister verbunden ist und dessen verbleibender Drainanschluß oder Sourceanschluß mit einem Sourceanschluß oder Drainanschluß eines zweiten Transistors an einem Datenknoten verbunden ist, wobei der verbleibende Drainanschluß oder Sourceanschluß des zweiten Transistors mit einer Referenzspannung verbunden ist und dessen Gateanschluß von einem Vorladesignal getrieben wird und wobei der Datenknoten ein Ausgang der Speicherschaltung ist, auf den die Richtig/Falsch-Überprüfungsschaltung reagiert.

12. Programmiervorrichtung nach einem der Ansprüche 9 bis 11, dadurch gekennzeichnet, dass die Richtig/Falsch-Überprüfungsschaltung einen ersten Transistor umfasst, von dem ein Gateanschluß mit dem Hauptdatenregister verbunden ist und ein Sourceanschluß oder Drainanschluß mit einer Referenzspannung verbunden ist und ein verbleibender Drainanschluß oder Sourceanschluß mit einem invertierten Überprüfungssignal verbunden ist.

13. Programmiervorrichtung nach einem der Ansprüche 9 bis 12, dadurch gekennzeichnet, dass die Rückspeicherschaltung einen ersten Transistor umfasst, dessen Sourceanschluß oder Drainanschluß mit einem Ausgang des Hauptdatenregisters verbunden ist und dessen verbleibender Drainanschluß oder Sourceanschluß mit einem Sourceanschluß oder Drainanschluß eines zweiten Transistors verbunden ist, dessen verbleibender Drainanschluß oder Sourceanschluß mit einer Referenzspannung verbunden ist, wobei ein Gateanschluß des ersten Transistors von einem Zurückspeichersignal getrieben wird.

14. Programmiervorrichtung nach einem der Ansprüche 9 bis 13, dadurch gekennzeichnet, dass das Cache-Datenregister mit dem Hauptdatenregister über einen ersten Transistor verbunden ist, dessen Gateanschluß durch ein erstes Entladesignal getrieben wird, wobei ein Ausgang des Hauptdatenregisters benutzt wird, um eine Speicherzelle in der Speicherpufferspeicherschaltung zu programmieren, und wobei das Hauptdatenregister nach einer solchen Programmierung mit den in der Speicherschaltung gespeicherten Daten zurückgeladen wird.

15. Programmierverfahren für einen nichtflüchtigen Speicherbaustein, gekennzeichnet durch folgende Schritte:
– Speichern von Daten in einem ersten Datenregister (LATCH1),
– Übertragen der invertierten Daten in eine Speicherschaltung,
– Entladen der Daten vom ersten Datenregister (LATCH1) in ein zweites Datenregister (LATCH2),
– Programmieren einer jeweiligen Speicherzelle entsprechend den Daten im zweiten Datenregister (LATCH2),
– Verifizieren der Speicherzellen bezüglich Speichern des Zustandes der Speicherzellen des zweiten Datenregisters (LATCH2) und
– Überprüfen des zweiten Datenregisters (LATCH2) durch eine Richtig/Falsch-Überprüfungsschaltung, ob die Speicherzelle programmiert ist oder nicht.

16. Programmierverfahren nach Anspruch 15, dadurch gekennzeichnet, dass nach der Programmierung der Inhalt des zweiten Datenregisters (LATCH2) entsprechend den invertierten Daten der Speicherschaltung zurückgesetzt wird.

17. Programmierverfahren nach Anspruch 16, dadurch gekennzeichnet, dass nach dem Zurücksetzen die Speicherzelle entsprechend den Daten im zweiten Datenregister (LATCH2) erneut programmiert wird.

18. Programmierverfahren nach Anspruch 17, dadurch gekennzeichnet, dass der Spannungspegel für die erneute Programmierung höher ist als der Spannungspegel für die vorherige Programmierung.

Es folgen 28 Blatt Zeichnungen

Anhängende Zeichnungen

Fig. 1

(Stand der Technik)

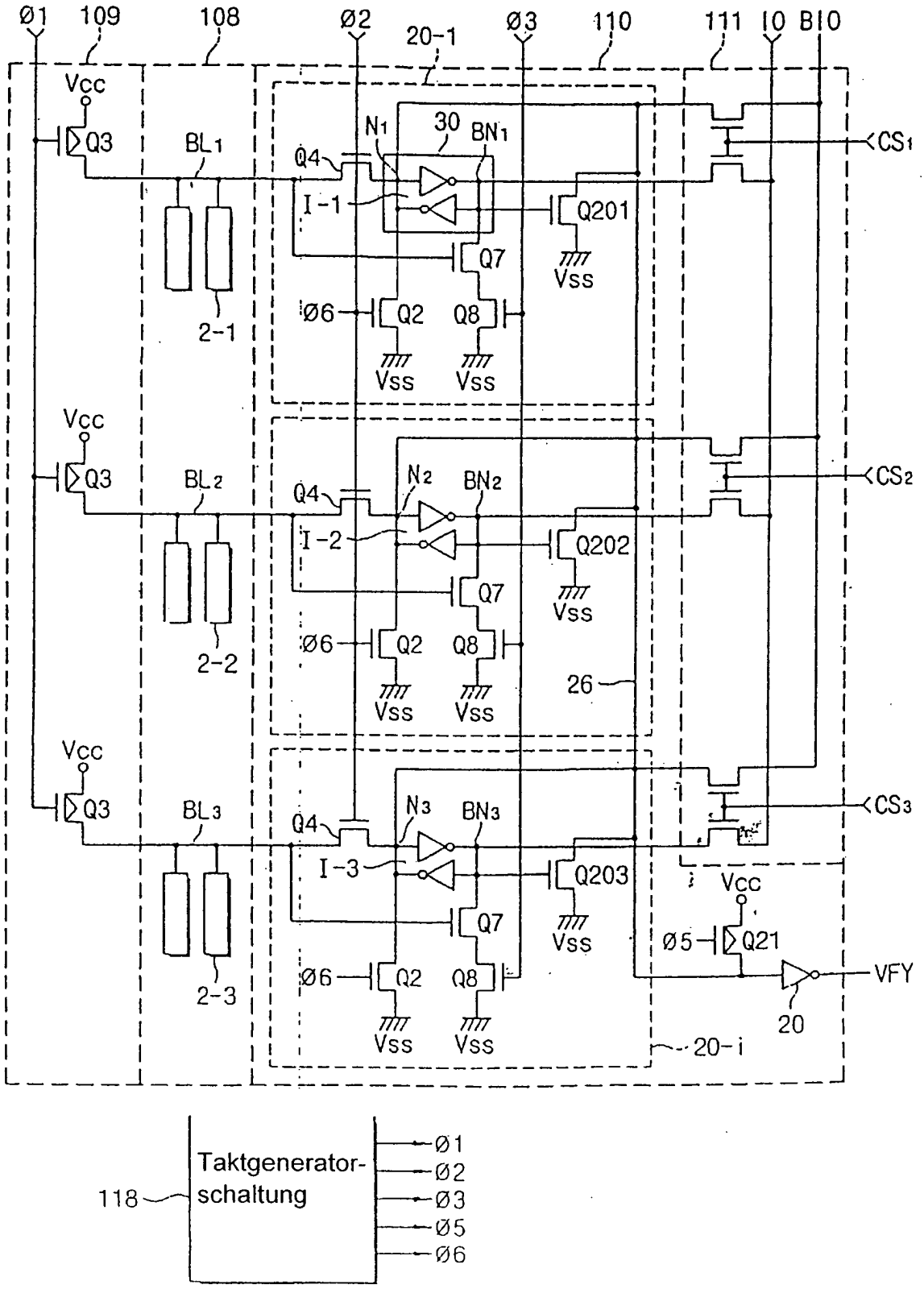


Fig. 2

(Stand der Technik)

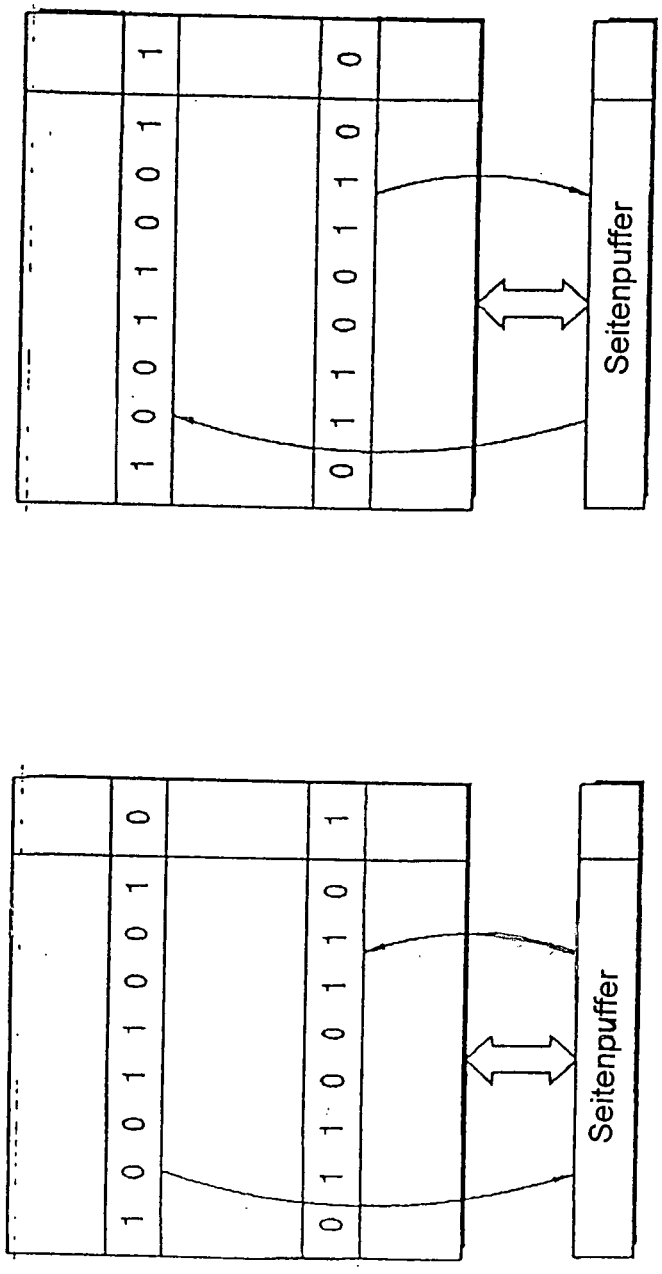


Fig. 3

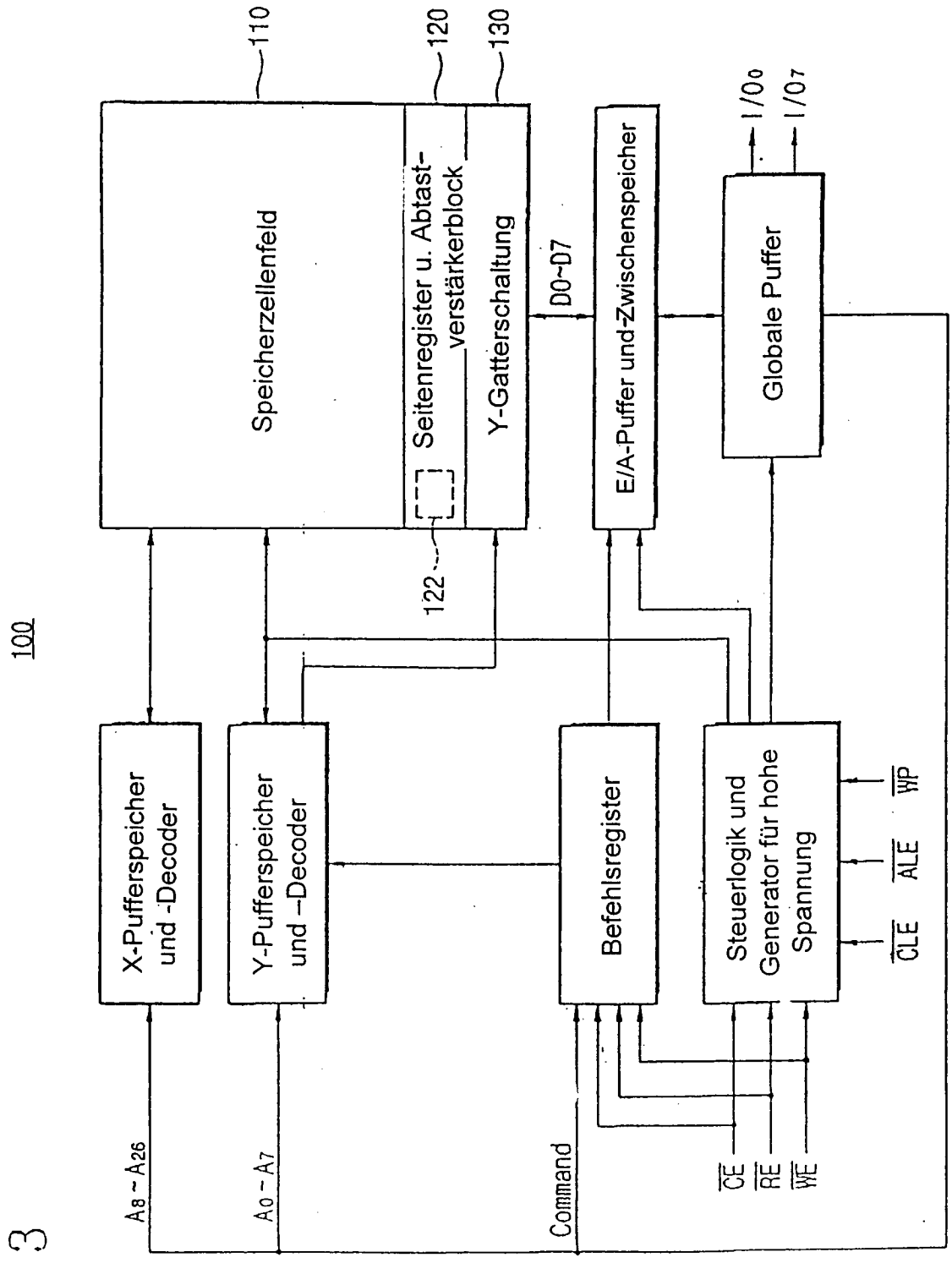
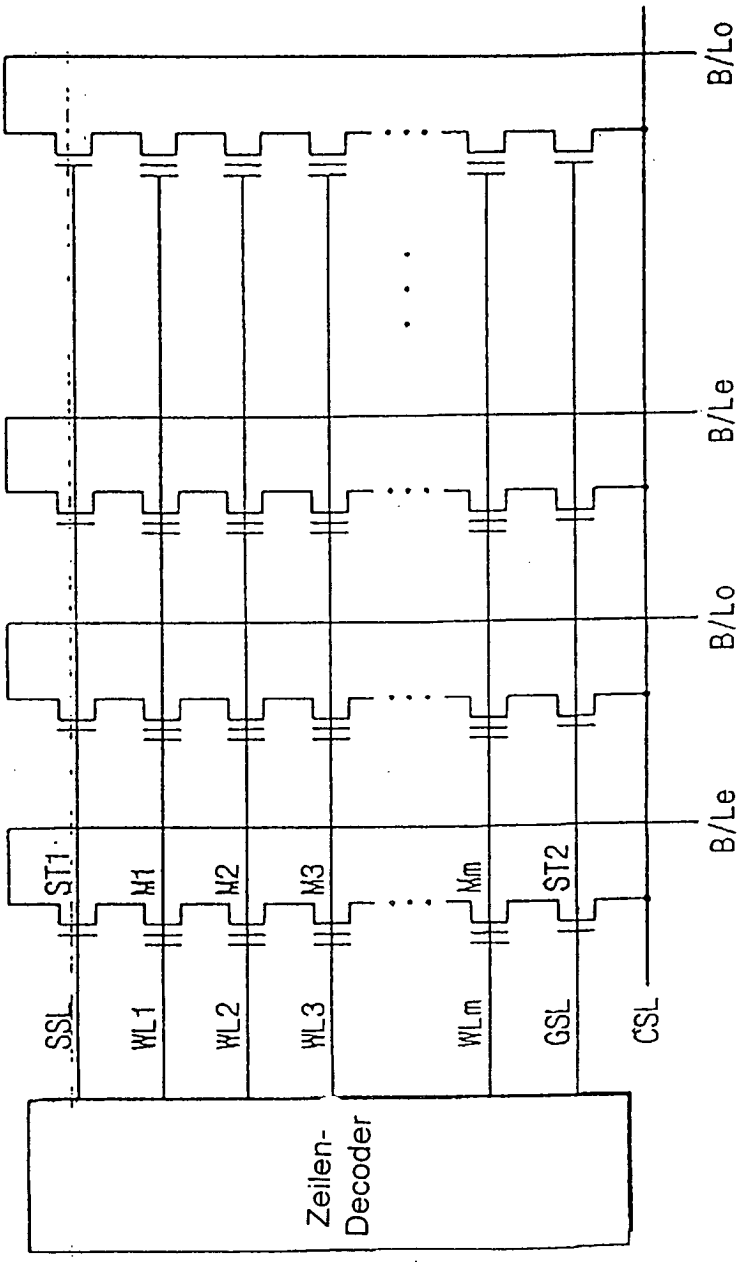

100
110
120
130
Speicherzellenfeld
Seitenregister u. Abtast-verstärkerblock
Y-Gatterschaltung
122
D0~D7
E/A-Puffer und Zwischenspeicher
Globale Puffer
I/O0
I/O7
X-Pufferspeicher und -Decoder
Y-Pufferspeicher und –Decoder
Befehlsregister
Steuerlogik und Generator für hohe Spannung
WP
ALE
CLE
CE
RE
WE
A8 ~ A26
A0 ~ A7
Command

Fig. 4


B/Lo
B/Le
B/Lo
B/Le
ST1
M1
M2
M3
Mm
ST2
SSL
WL1
WL2
WL3
WLm
GSL
CSL
Zeilen-
Decoder

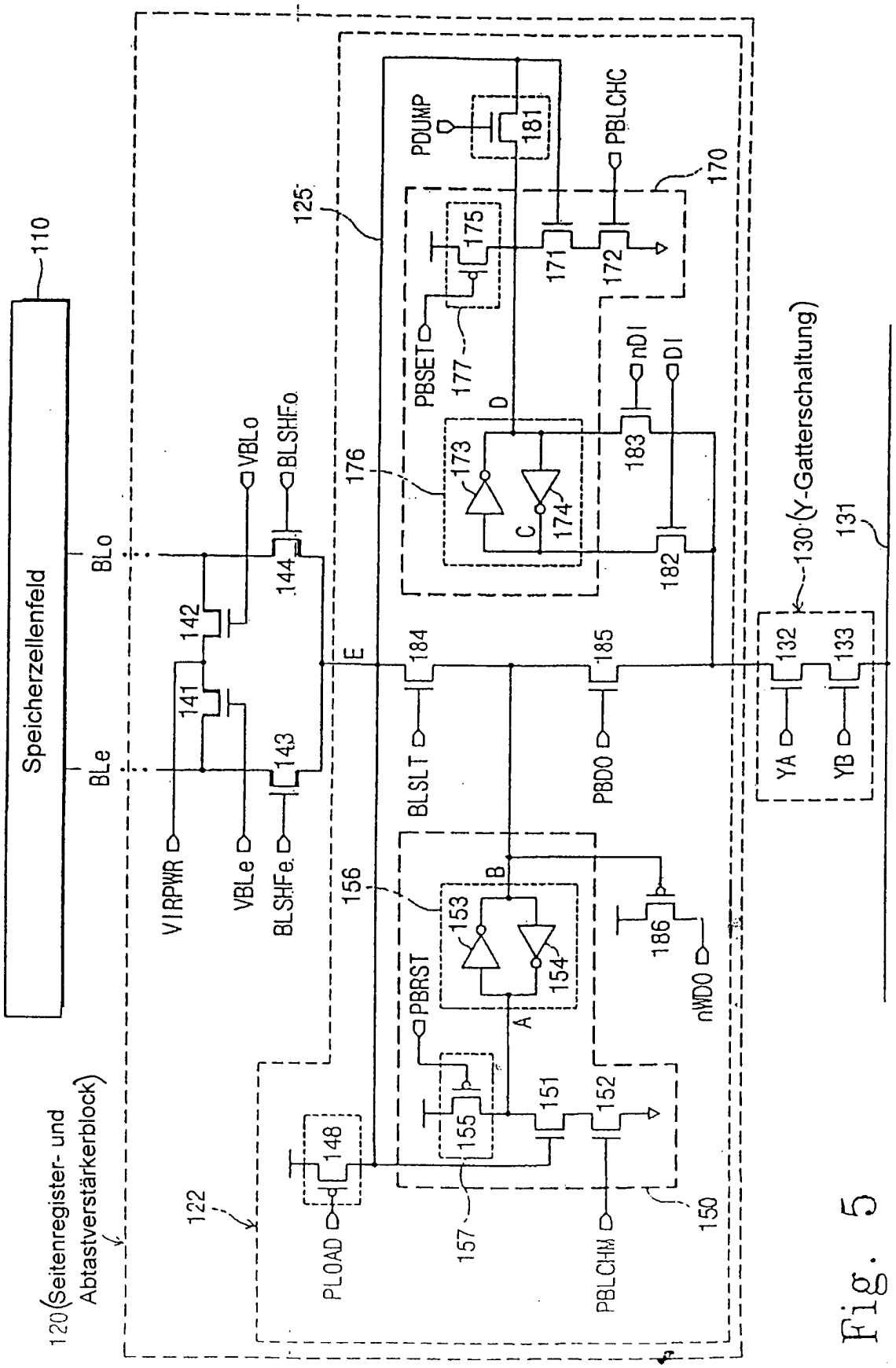

Speicherzellenfeld
110
BLo
BLe
VIRPWR
VBLe
BLSHFe
VBLo
BLSHFo
142
141
143
144
125
176
156
E
184
185
BLSLT
PBDO
PBRST
PDUMP
PBSET
PBLCHC
181
175
177
171
172
170
173
174
183
182
nDI
DI
D
C
B
A
153
154
155
151
152
186
nWDO
148
PLOAD
PBLCHM
157
150
122
120 (Seitenregister- und Abtastverstärkerblock)
130 (Y-Gatterschaltung)
131
132
133
YA
YB

Fig. 5

Fig. 6

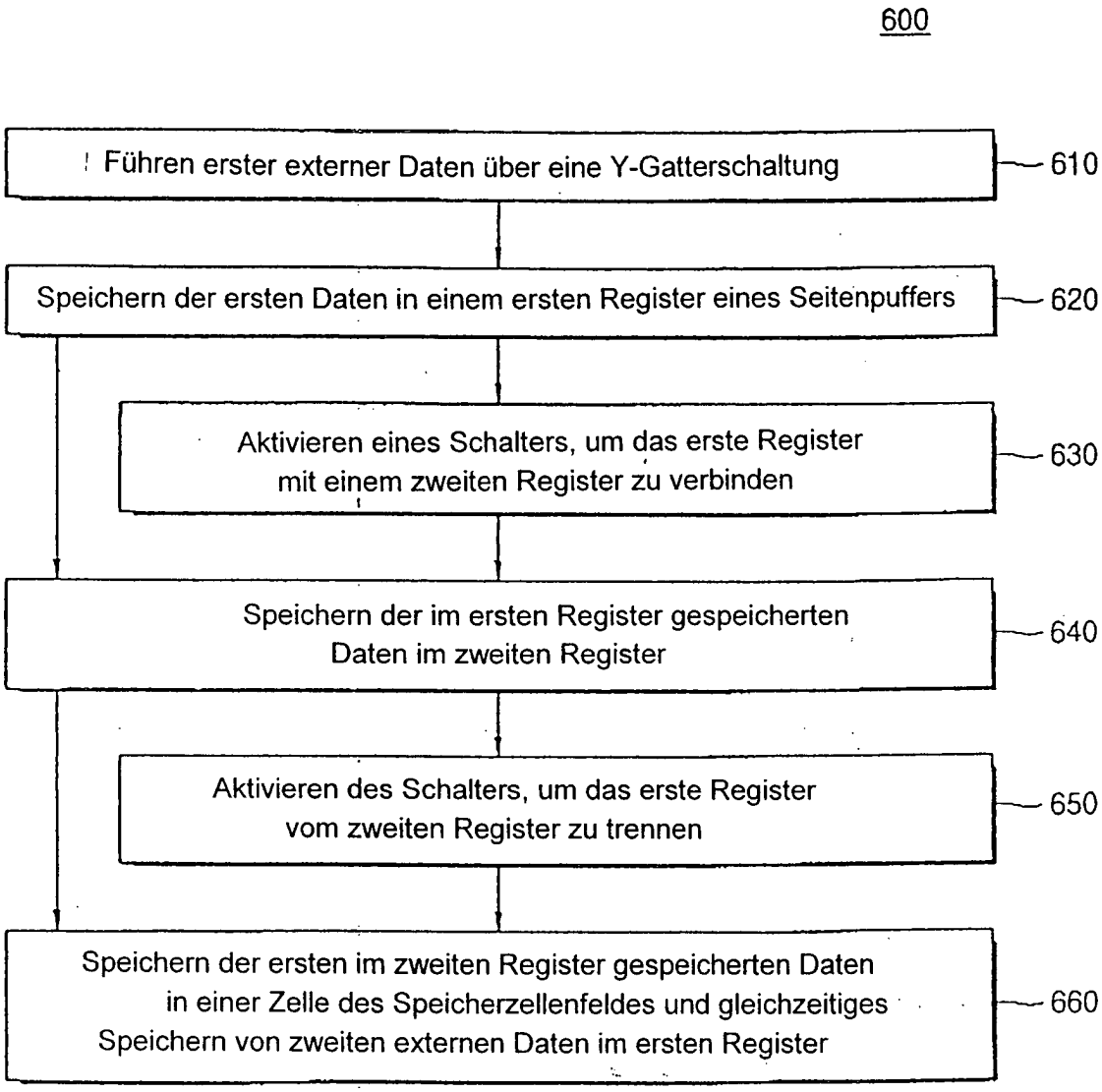

600
Führen erster externer Daten über eine Y-Gatterschaltung
610
Speichern der ersten Daten in einem ersten Register eines Seitenpuffers
620
Aktivieren eines Schalters, um das erste Register mit einem zweiten Register zu verbinden
630
Speichern der im ersten Register gespeicherten Daten im zweiten Register
640
Aktivieren des Schalters, um das erste Register vom zweiten Register zu trennen
650
Speichern der ersten im zweiten Register gespeicherten Daten in einer Zelle des Speicherzellenfeldes und gleichzeitiges Speichern von zweiten externen Daten im ersten Register
660

Fig. 7

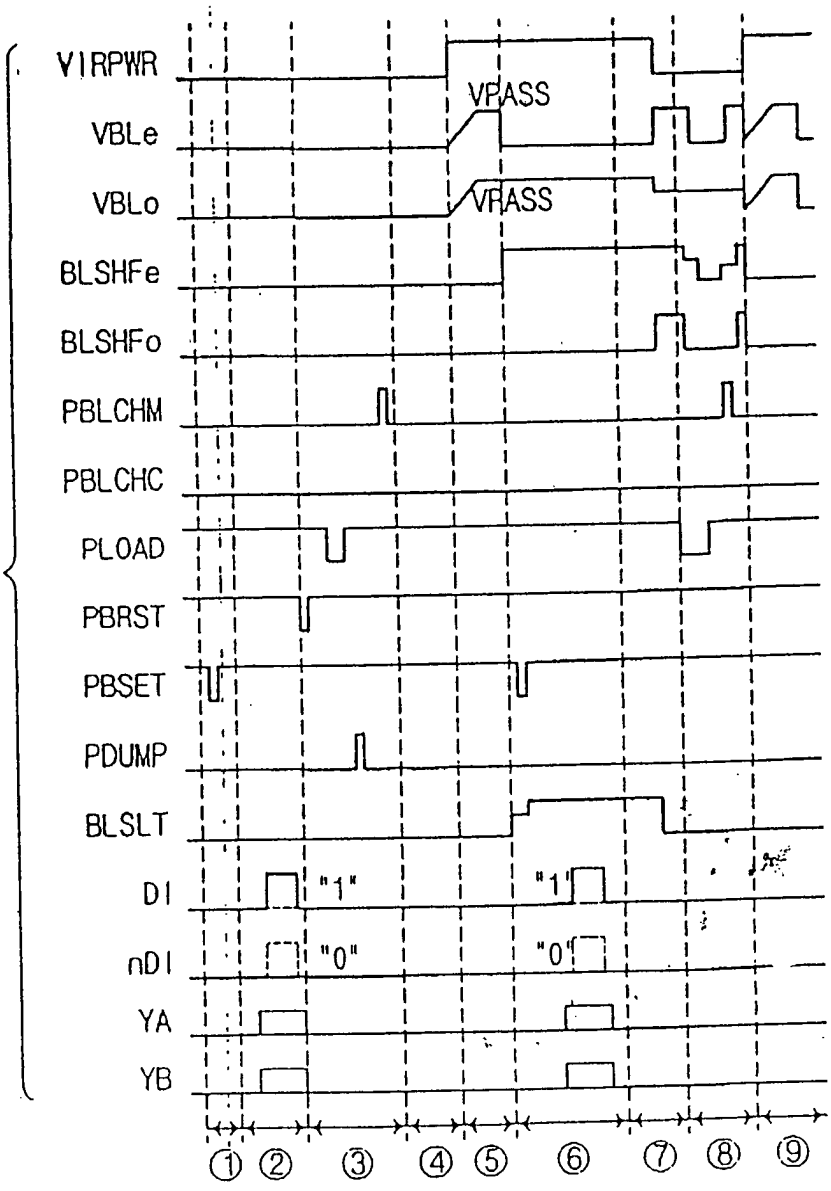

VIRPWR
VPASS
VBLe
VBLo
VRASS
BLSHFe
BLSHFo
PBLCHM
PBLCHC
PLOAD
PBRST
PBSET
PDUMP
BLSLT
DI
"1"
"1"
nDI
"0"
"0"
YA
YB
①
②
③
④
⑤
⑥
⑦
⑧
⑨

Fig. 8

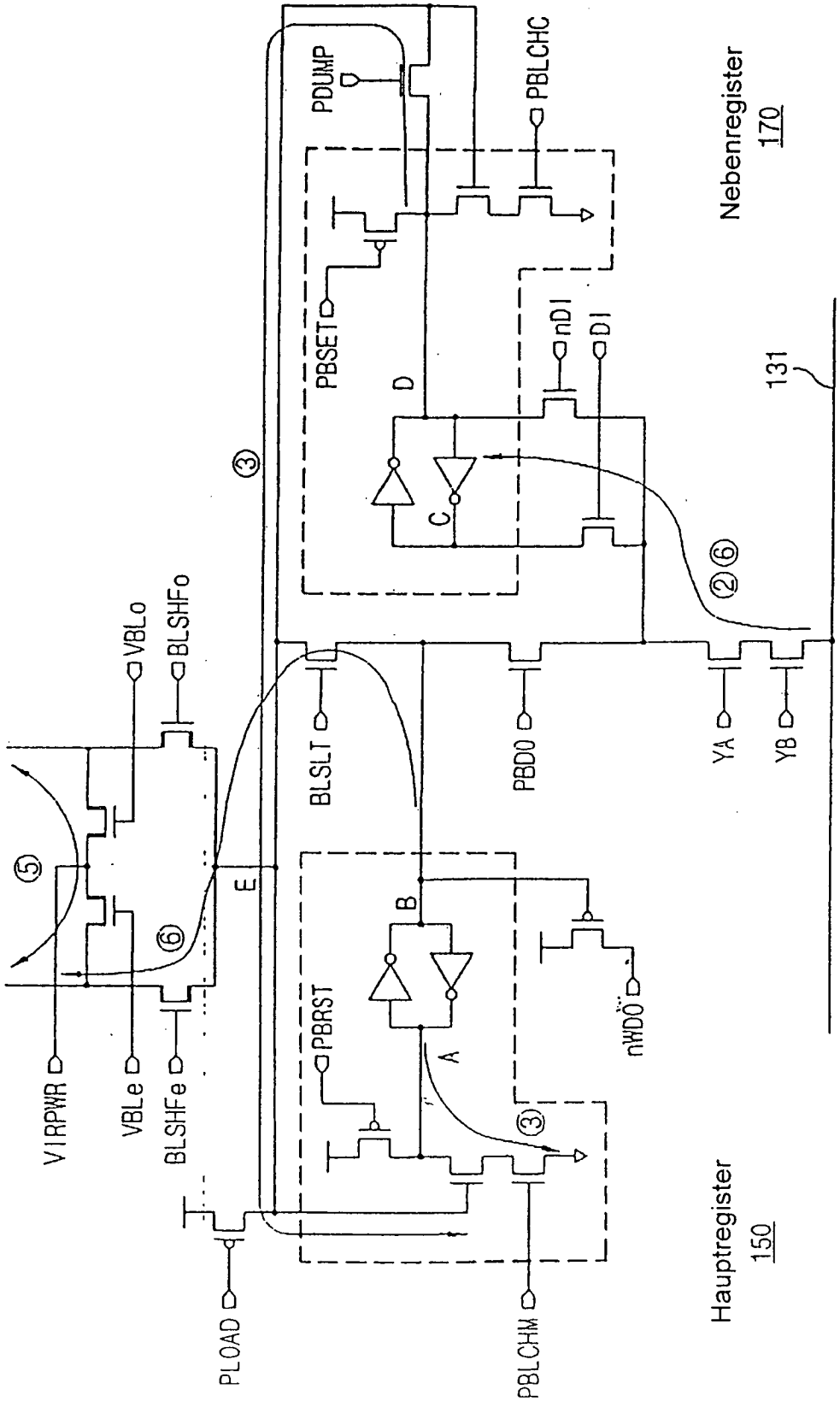

VIRPWR
VBLe
BLSHFe
VBLo
BLSHFo
PLOAD
PBRST
BLSLT
PDUMP
PBSET
PBLCHC
PBLCHM
PBDO
nWDO
nDI
DI
YA
YB
131
Nebenregister
170
Hauptregister
150
A
B
C
D
E

Fig. 9

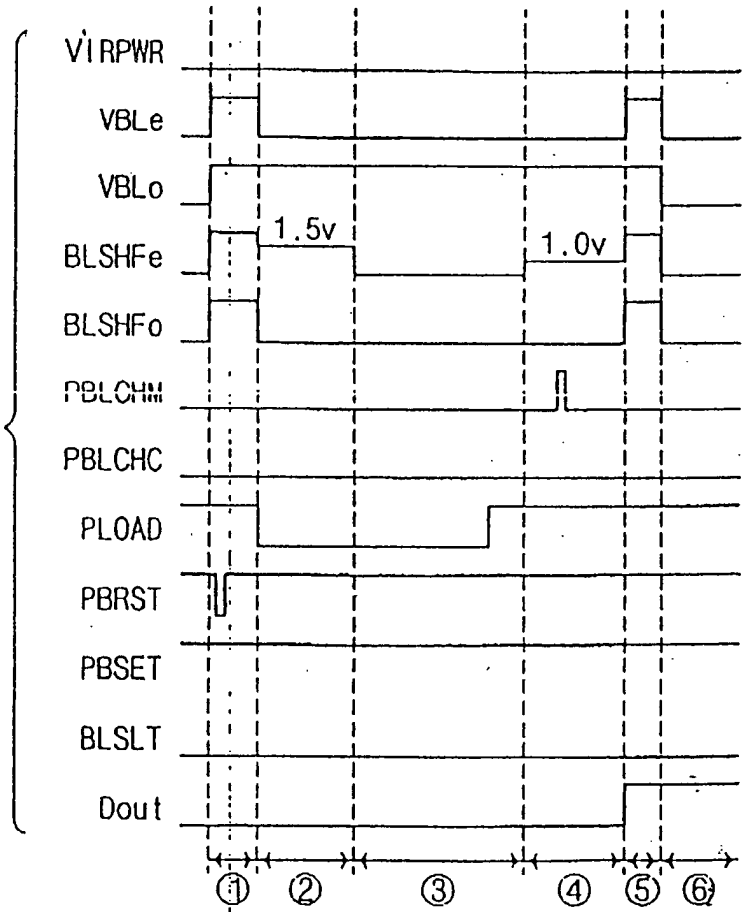

VIRPWR
VBLe
VBLo
BLSHFe
BLSHFo
PBLCHM
PBLCHC
PLOAD
PBRST
PBSET
BLSLT
Dout
1.5v
1.0v
①
②
③
④
⑤
⑥

Fig. 10

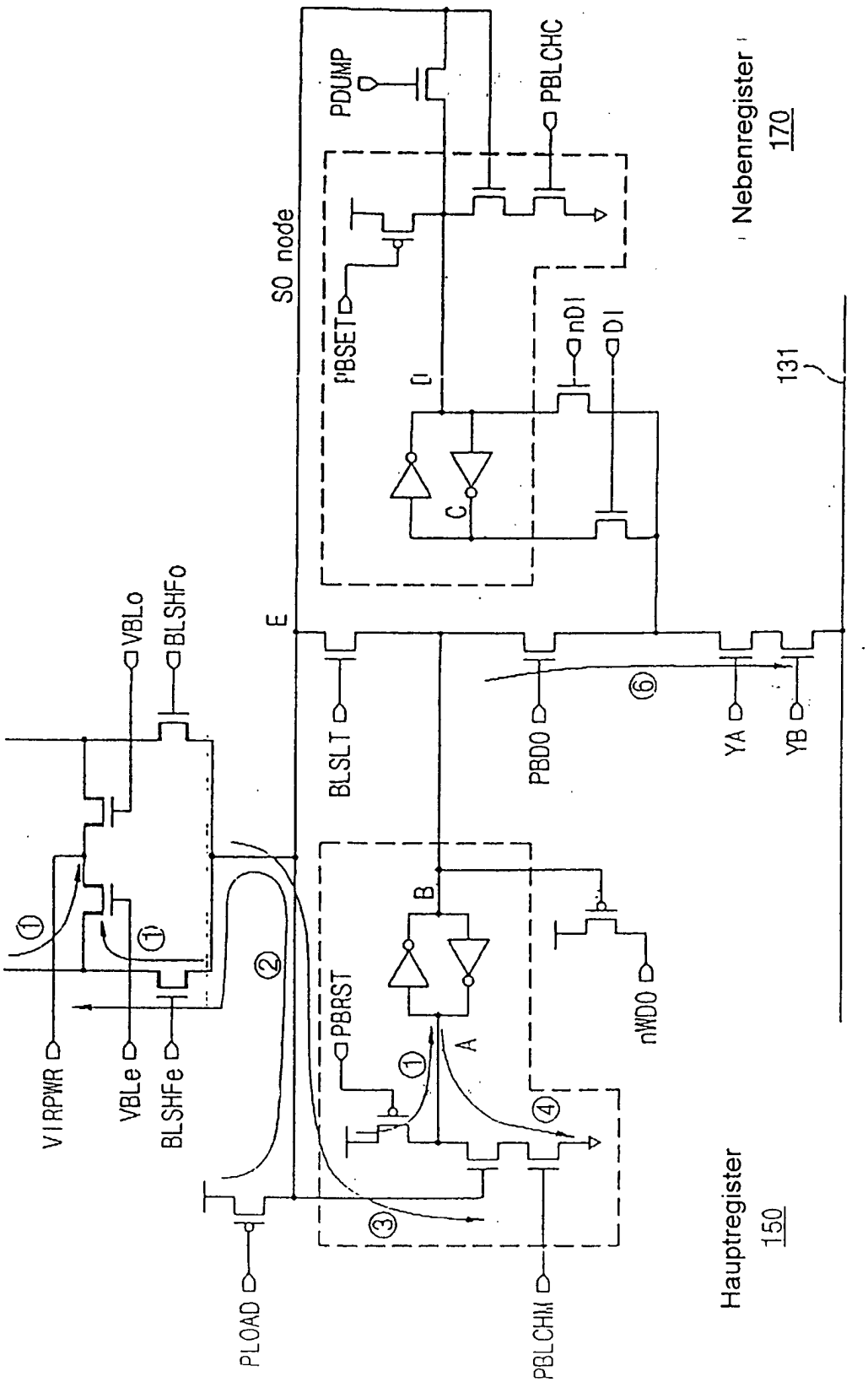

PDUMP
PBLCHC
SO node
Nebenregister
170
PBSET
nDI
DI
131
VBLo
BLSHFo
E
BLSLT
PBDO
YA
YB
VIRPWR
VBLe
BLSHFe
PBRST
nWDO
PLOAD
PBLCHM
Hauptregister
150

Fig. 11

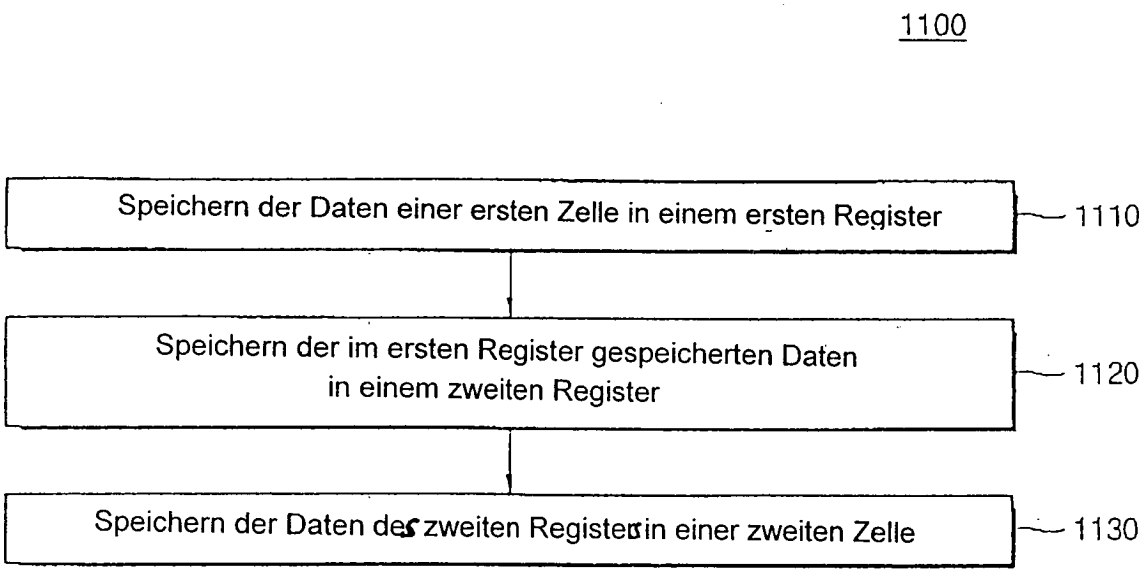

1100
Speichern der Daten einer ersten Zelle in einem ersten Register
1110
Speichern der im ersten Register gespeicherten Daten in einem zweiten Register
1120
Speichern der Daten des zweiten Registers in einer zweiten Zelle
1130

Fig. 12

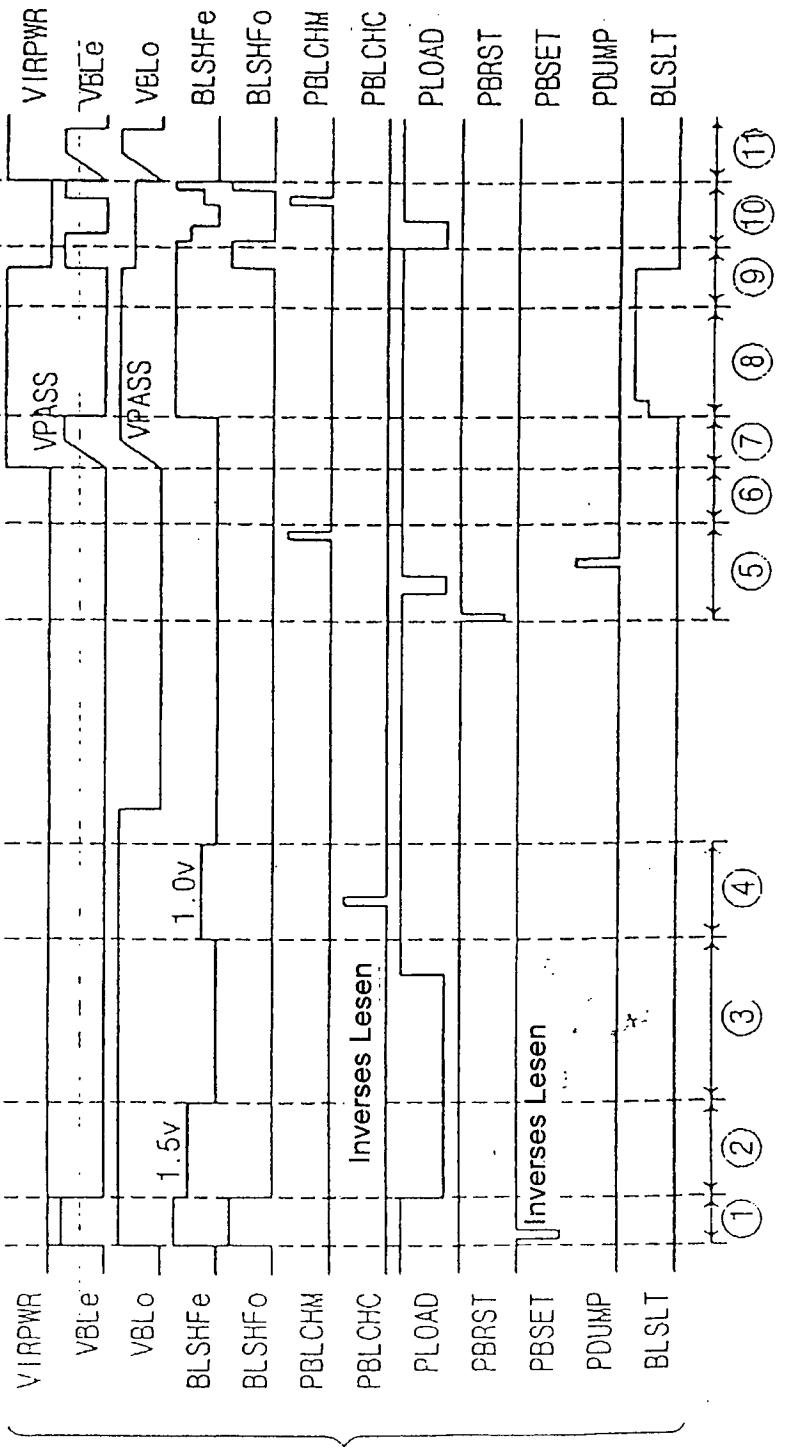

VIRPWR
VBLe
VBLo
BLSHFe
BLSHFo
PBLCHM
PBLCHC
PLOAD
PBRST
PBSET
PDUMP
BLSLT
VPASS
VPASS
1.5v
1.0v
Inverses Lesen
Inverses Lesen
1
2
3
4
5
6
7
8
9
10
11

Fig. 13

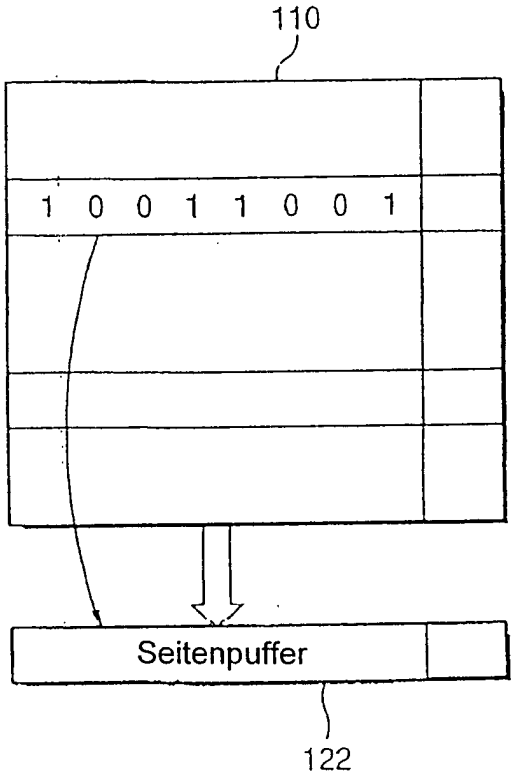


Fig. 14

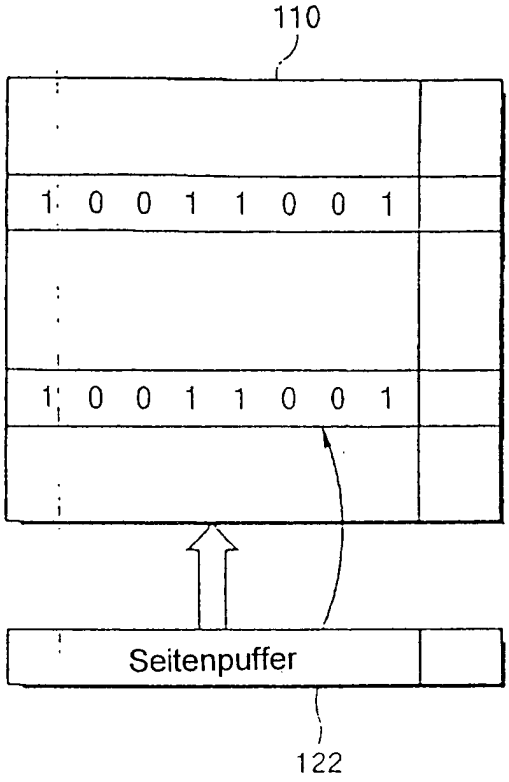

110
1 0 0 1 1 0 0 1
1 0 0 1 1 0 0 1
Seitenpuffer
122

Fig. 15

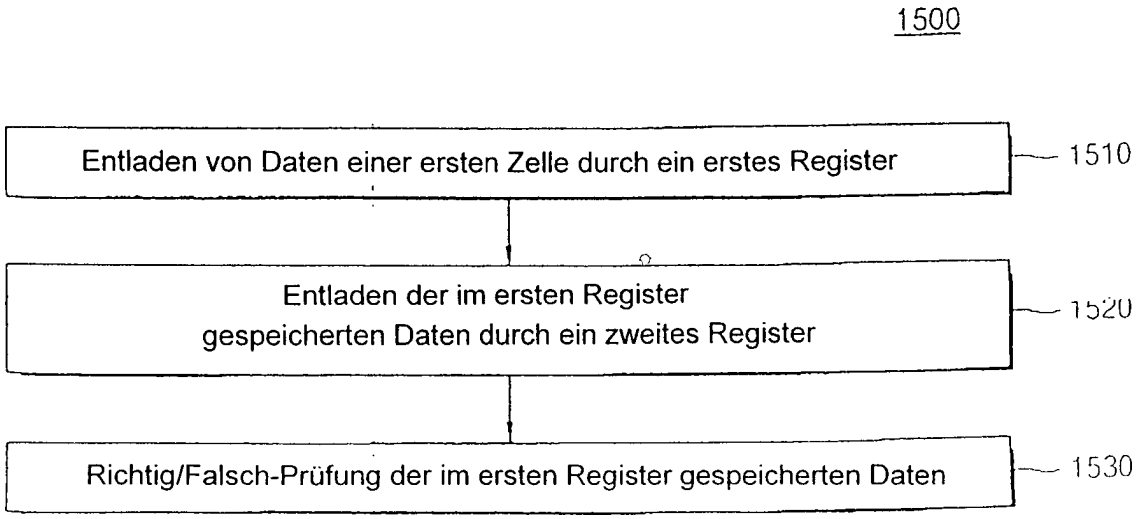

1500
Entladen von Daten einer ersten Zelle durch ein erstes Register
1510
Entladen der im ersten Register gespeicherten Daten durch ein zweites Register
1520
Richtig/Falsch-Prüfung der im ersten Register gespeicherten Daten
1530

Fig. 16

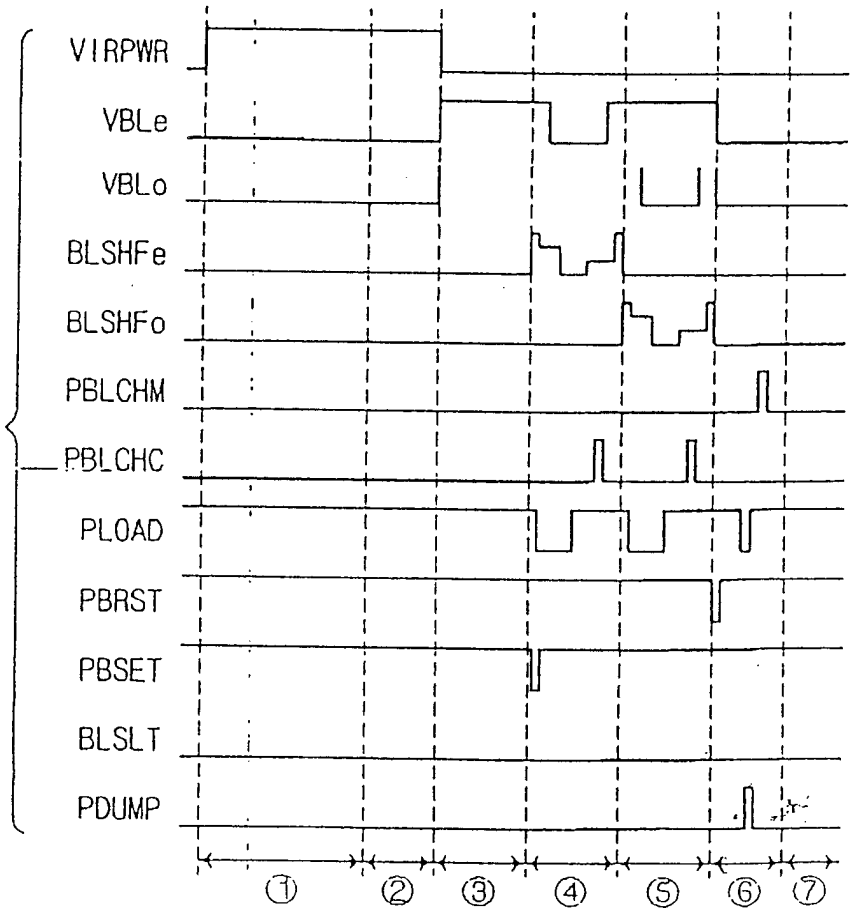

VIRPWR
VBLe
VBLo
BLSHFe
BLSHFo
PBLCHM
PBLCHC
PLOAD
PBRST
PBSET
BLSLT
PDUMP
①
②
③
④
⑤
⑥
⑦

Fig. 17

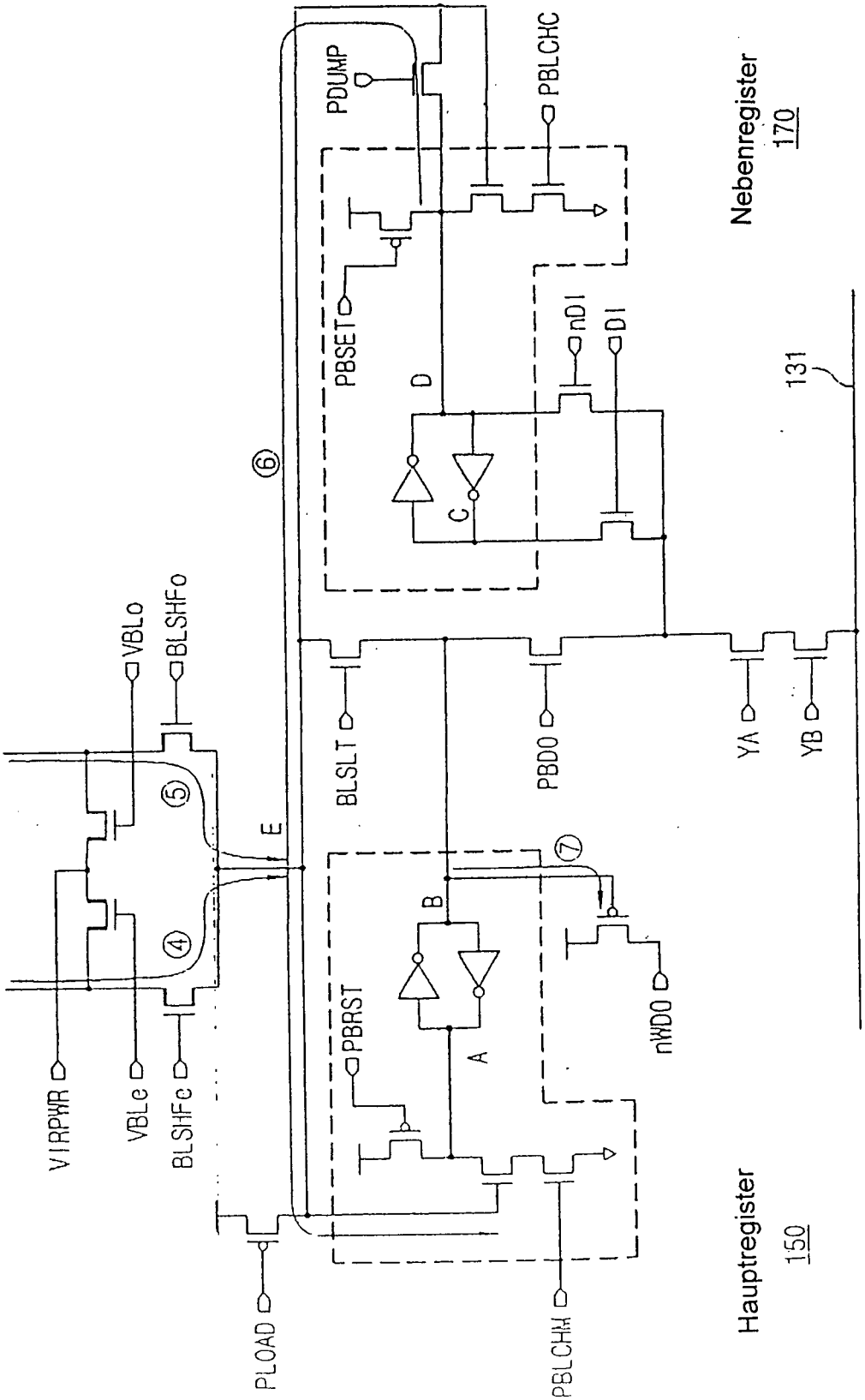

Nebenregister
170
Hauptregister
150
131
PDUMP
PBLCHC
PBSET
nDI
DI
D
C
VBLo
BLSHFo
BLSLT
PBDO
YA
YB
E
B
A
PBRST
nWDO
VIRPWR
VBLe
BLSHFe
PLOAD
PBLCHM
⑤
⑥
⑦
④

Fig. 18

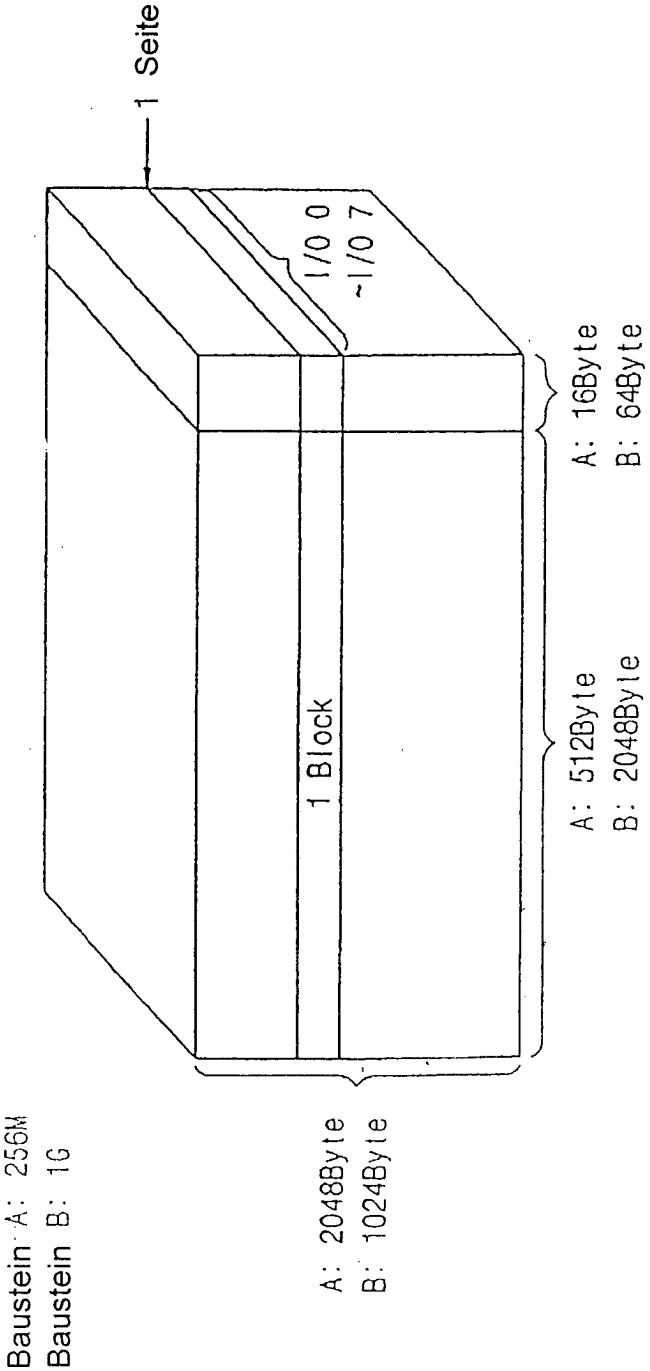

1 Seite
I/O 0
~I/O 7
1 Block
A: 16Byte
B: 64Byte
A: 512Byte
B: 2048Byte
A: 2048Byte
B: 1024Byte
Baustein A: 256M
Baustein B: 1G

Fig. 19

						Baustein A			Baustein B		
	4M (521K*8)	8M (1M*8)	16M (2M*8)	32M (4M*8)	64M (8M*8)	128M (16M*8)	256M (32M*8)	512M (64M*8)	1G (128M*8)	2G (256M*8)
Seite	32Byte	264B (256+8)	264B (256+8)	528B (512+16)	528B (512+16)	528B (512+16)	528B (512+16)	528B (512+16)	2112B (2048+64)	2112B (2048+64)
BLOCK	4Kbyte	16P	16P	16P	16P	32P	32P	32P	64P	64P
Baustein	128B	256B	512B	512B	1024B	1024B	2048B	4096B	1024B	2048B

Fig. 20

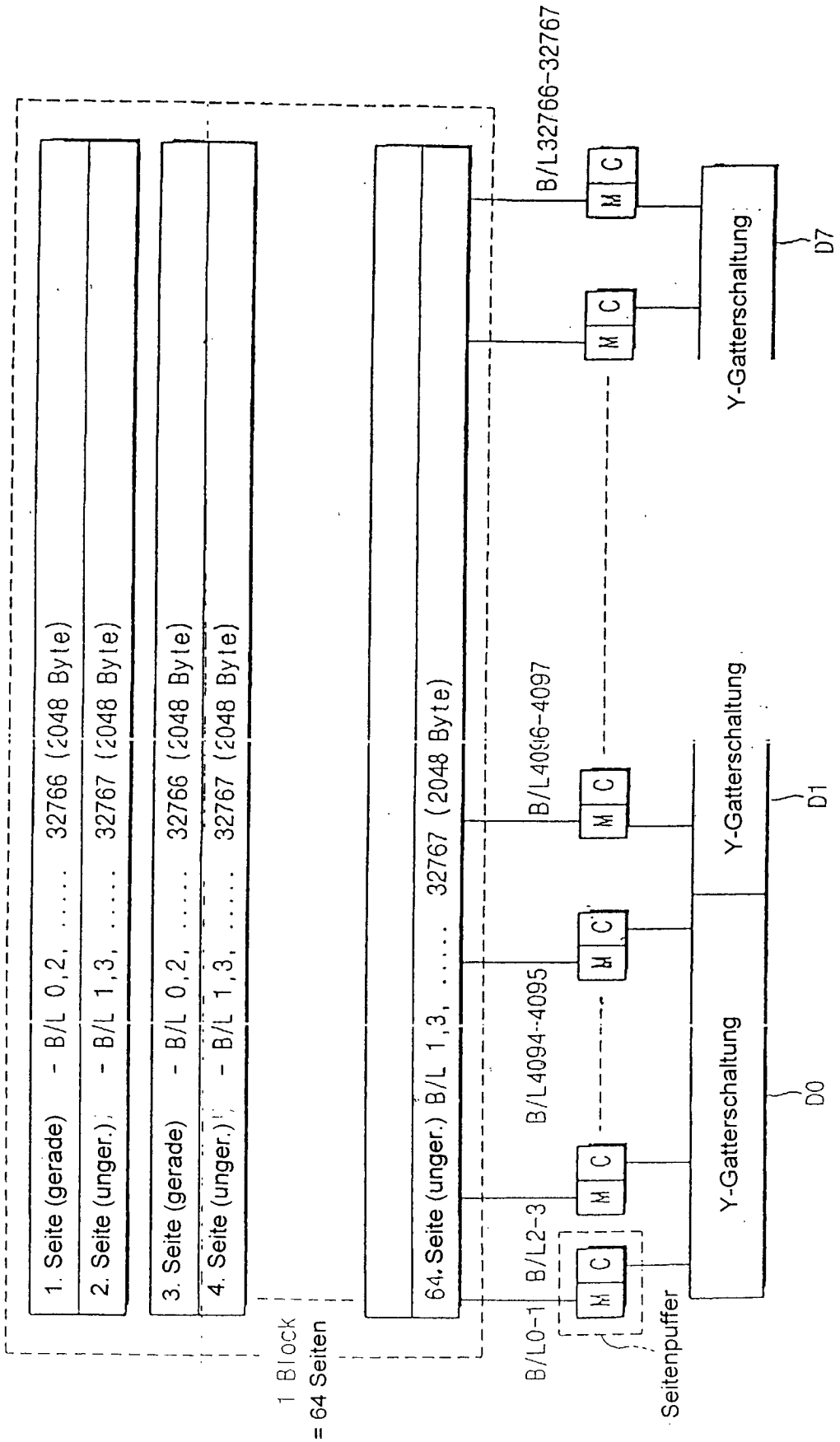

1. Seite (gerade) - B/L 0,2, ... 32766 (2048 Byte)
2. Seite (unger.) - B/L 1,3, ... 32767 (2048 Byte)
3. Seite (gerade) - B/L 0,2, ... 32766 (2048 Byte)
4. Seite (unger.) - B/L 1,3, ... 32767 (2048 Byte)
64. Seite (unger.) B/L 1,3, ... 32767 (2048 Byte)
1 Block
= 64 Seiten
B/L0-1
B/L2-3
B/L4094-4095
B/L4096-4097
B/L32766-32767
M C
Seitenpuffer
Y-Gatterschaltung
Y-Gatterschaltung
Y-Gatterschaltung
D0
D1
D7

Fig. 21

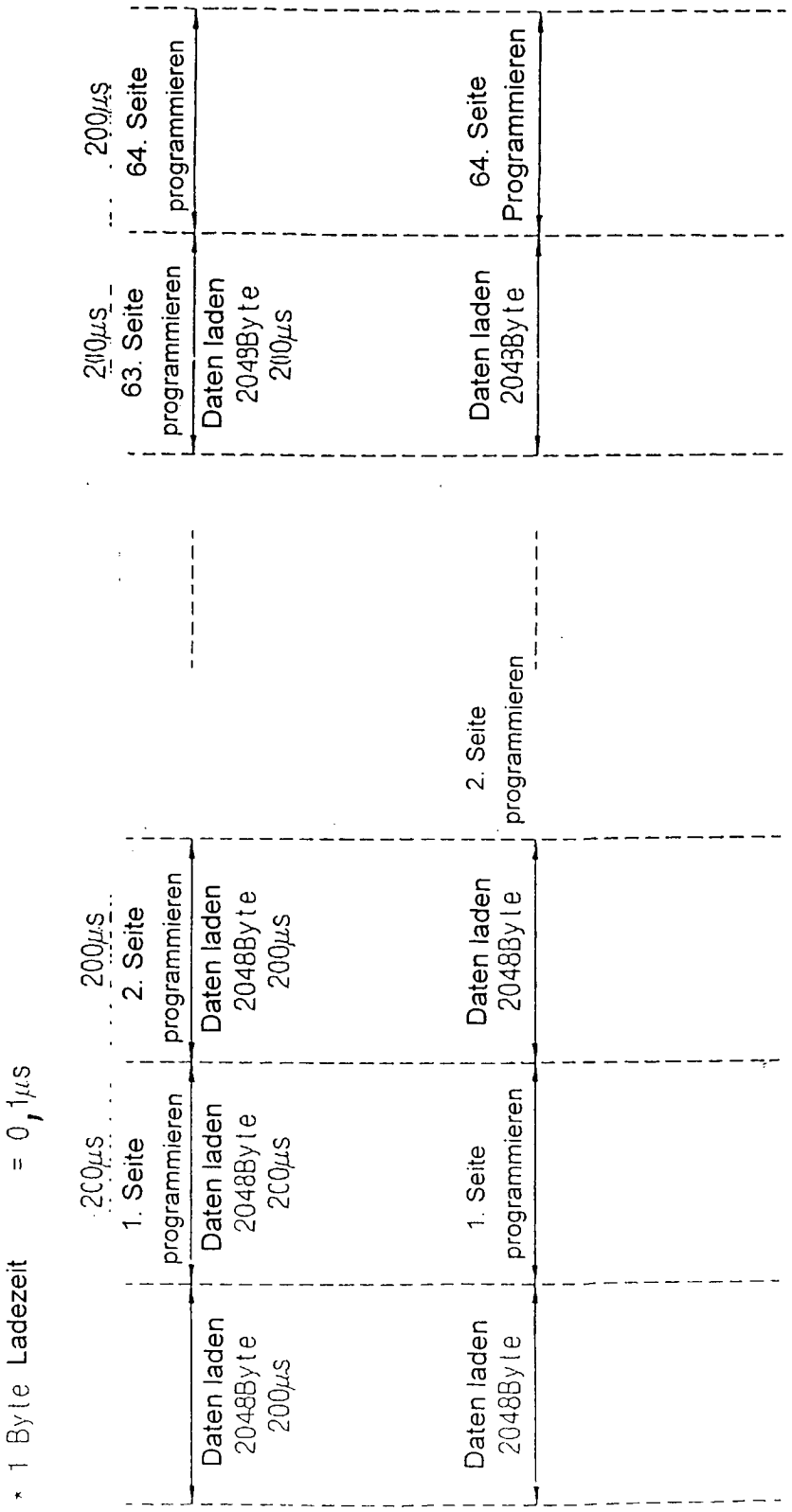

* 1 Byte Ladezeit = 0,1µs
200µs
1. Seite programmieren
Daten laden 2048Byte 200µs
Daten laden 2048Byte
200µs
2. Seite programmieren
Daten laden 2048Byte 200µs
Daten laden 2048Byte
2. Seite programmieren
210µs
63. Seite programmieren
Daten laden 2049Byte 210µs
Daten laden 2043Byte
200µs
64. Seite programmieren
64. Seite Programmieren

Fig. 22

(Stand der Technik)

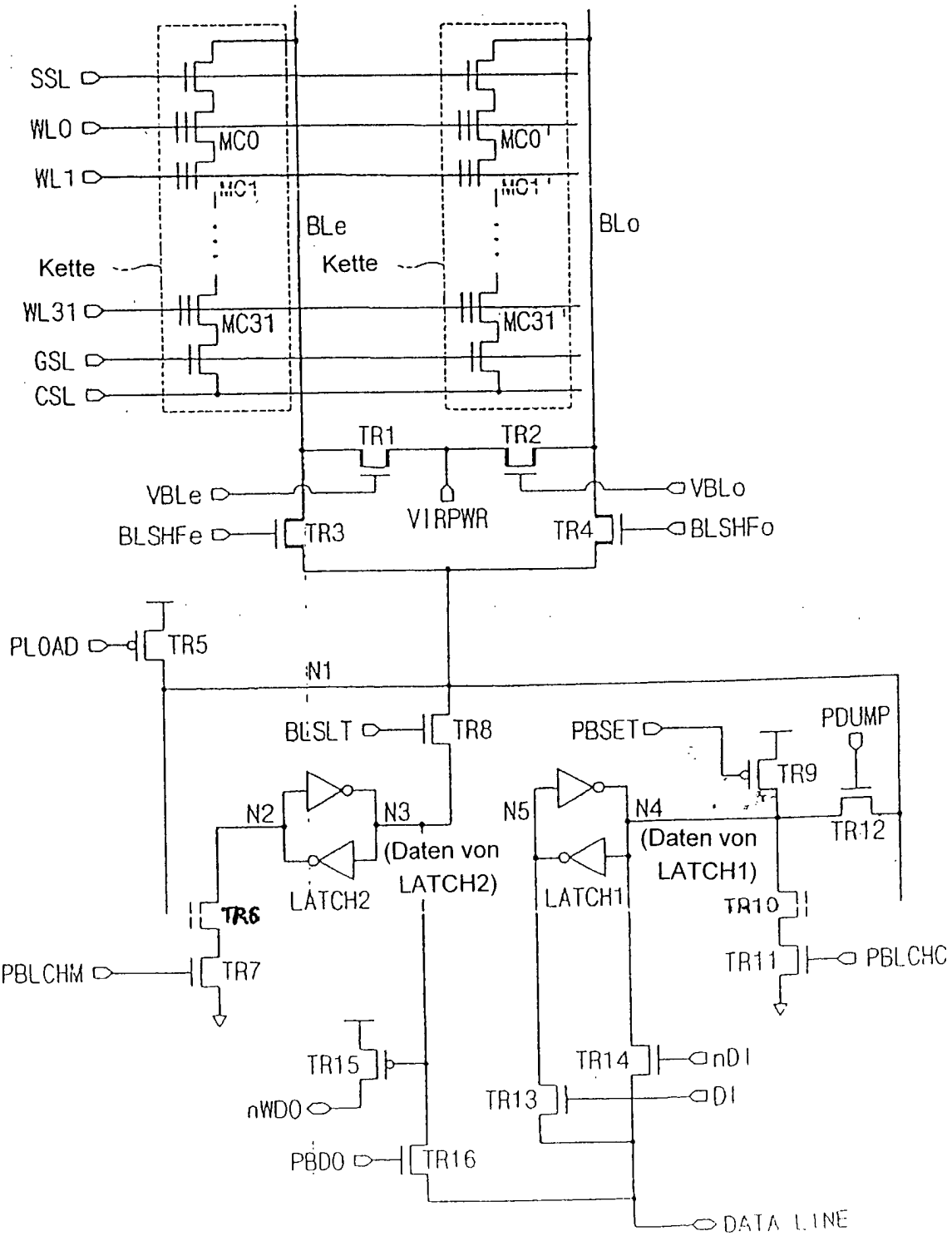


Fig. 23

(Stand der Technik)

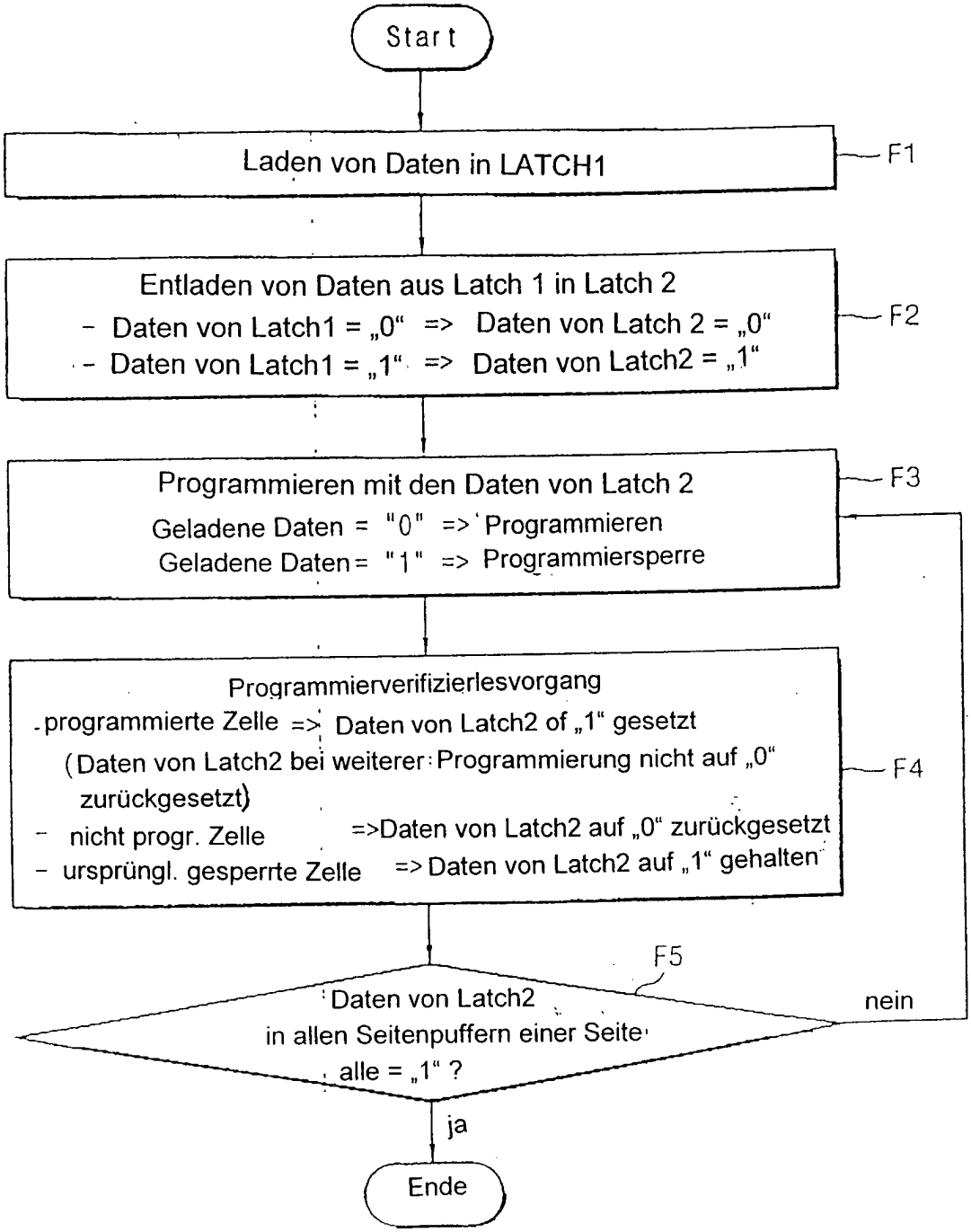


Fig. 24

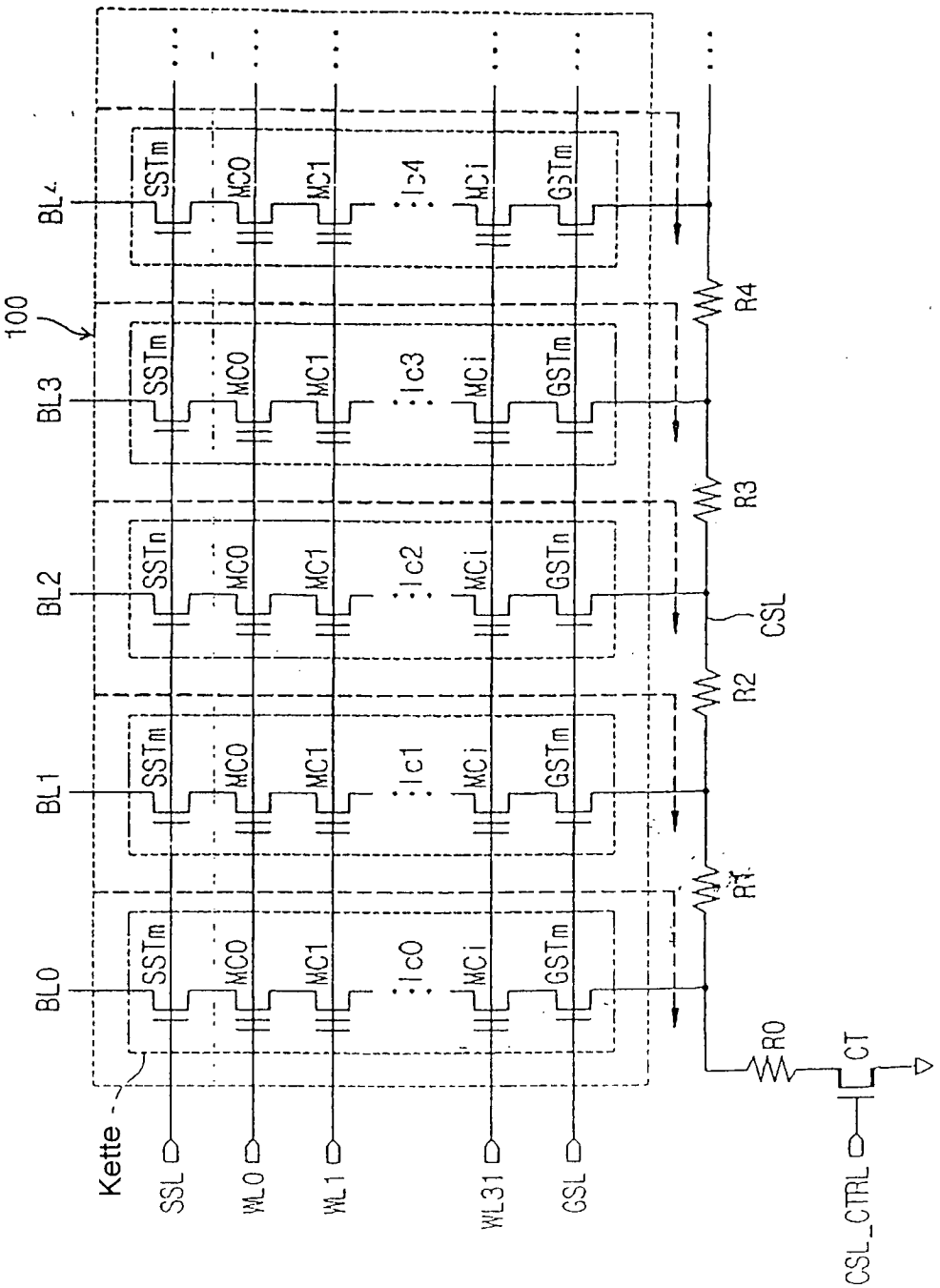

100
BL0
BL1
BL2
BL3
BL4
SSTm
SSTn
MC0
MC1
MCi
GSTm
GSTn
Ic0
Ic1
Ic2
Ic3
Ic4
R0
R1
R2
R3
R4
CT
CSL
Kette
SSL
WL0
WL1
WL31
GSL
CSL_CTRL

Fig. 25

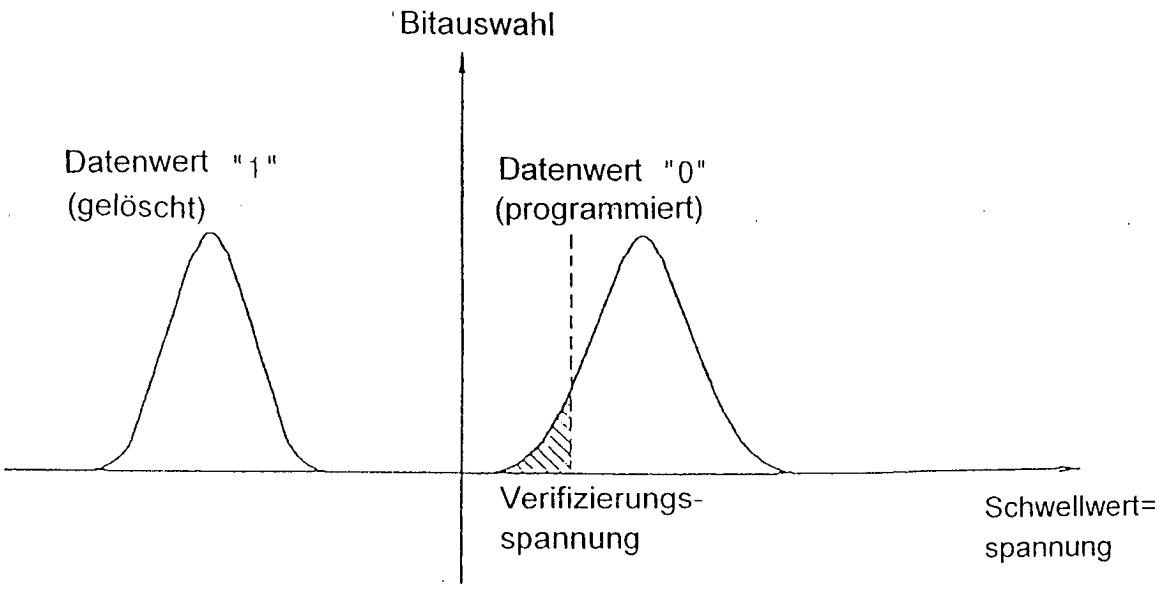

Bitauswahl
Datenwert "1"
(gelöscht)
Datenwert "0"
(programmiert)
Verifizierungs-
spannung
Schwellwert=
spannung

Fig. 26

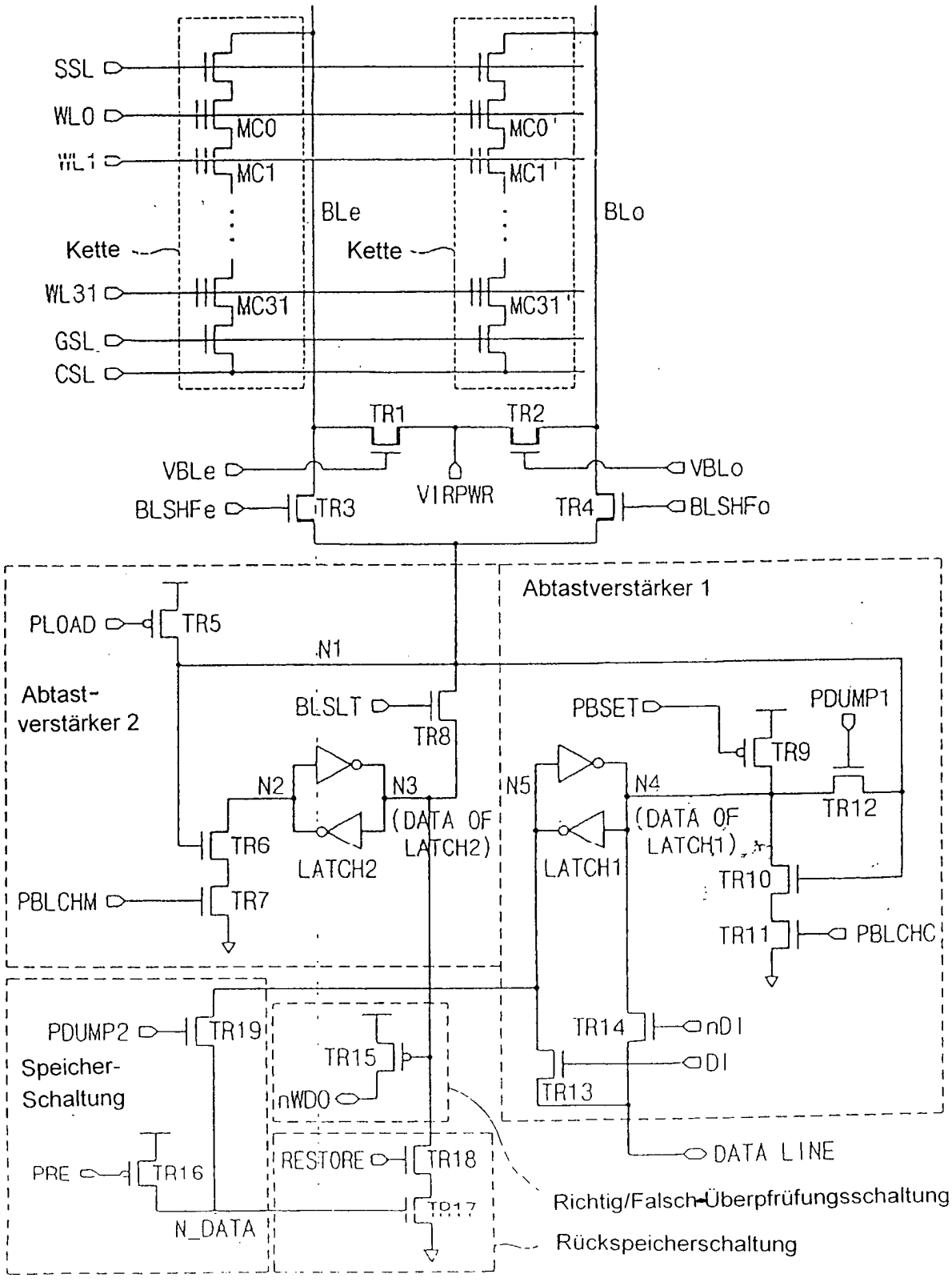

SSL
WL0
WL1
MC0
MC1
BLe
BLo
Kette
WL31
MC31
GSL
CSL
TR1
TR2
VBLe
VBLo
VIRPWR
BLSHFe
TR3
TR4
BLSHFo
Abtastverstärker 1
PLOAD
TR5
N1
Abtast-
verstärker 2
BLSLT
TR8
PBSET
PDUMP1
TR9
N2
N3
N5
N4
(DATA OF
LATCH2)
(DATA OF
LATCH1)
TR12
TR6
LATCH2
LATCH1
TR10
PBLCHM
TR7
TR11
PBLCHC
PDUMP2
TR19
TR15
TR14
nDI
Speicher-
Schaltung
nWDO
DI
TR13
DATA LINE
PRE
TR16
RESTORE
TR18
TR17
N_DATA
Richtig/Falsch-Überpfrüfungsschaltung
Rückspeicherschaltung

Fig. 27

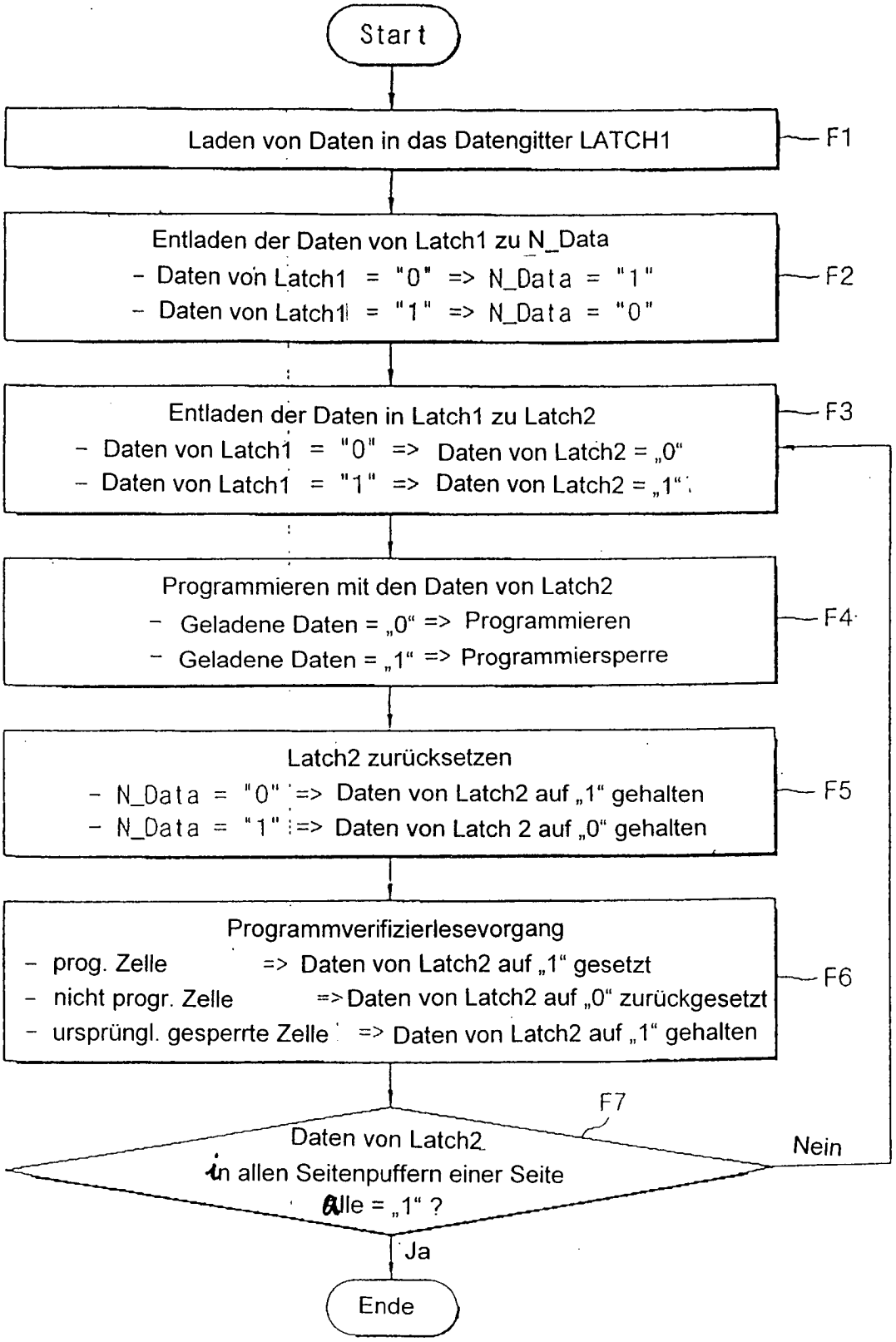

Start
Laden von Daten in das Datengitter LATCH1
F1
Entladen der Daten von Latch1 zu N_Data
- Daten von Latch1 = "0" => N_Data = "1"
- Daten von Latch1 = "1" => N_Data = "0"
F2
Entladen der Daten in Latch1 zu Latch2
- Daten von Latch1 = "0" => Daten von Latch2 = „0"
- Daten von Latch1 = "1" => Daten von Latch2 = „1"
F3
Programmieren mit den Daten von Latch2
- Geladene Daten = „0" => Programmieren
- Geladene Daten = „1" => Programmiersperre
F4
Latch2 zurücksetzen
- N_Data = "0" => Daten von Latch2 auf „1" gehalten
- N_Data = "1" => Daten von Latch 2 auf „0" gehalten
F5
Programmverifizierlesevorgang
- prog. Zelle => Daten von Latch2 auf „1" gesetzt
- nicht progr. Zelle => Daten von Latch2 auf „0" zurückgesetzt
- ursprüngl. gesperrte Zelle => Daten von Latch2 auf „1" gehalten
F6
F7
Daten von Latch2
in allen Seitenpuffern einer Seite
alle = „1" ?
Nein
Ja
Ende

Fig. 28

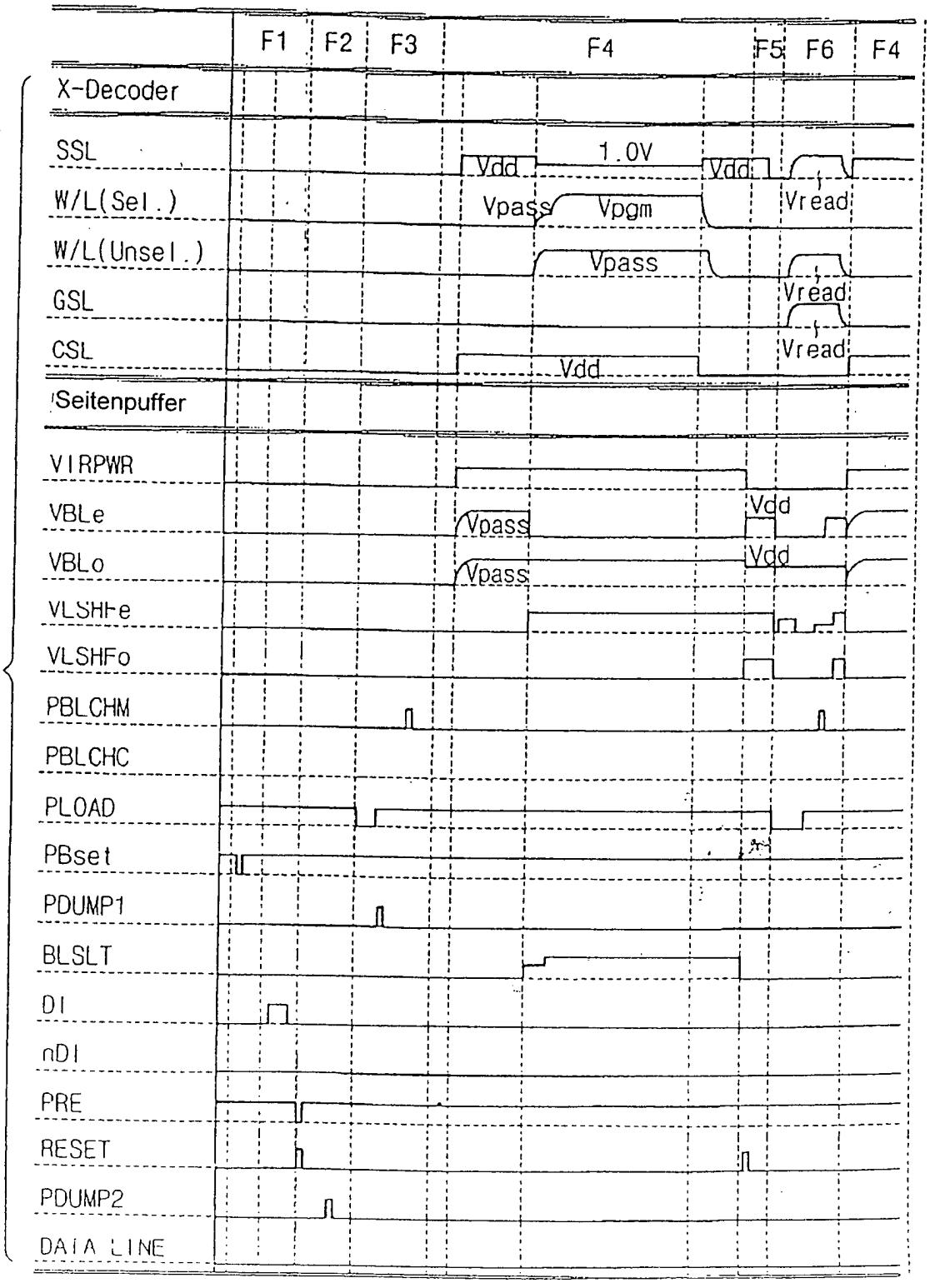
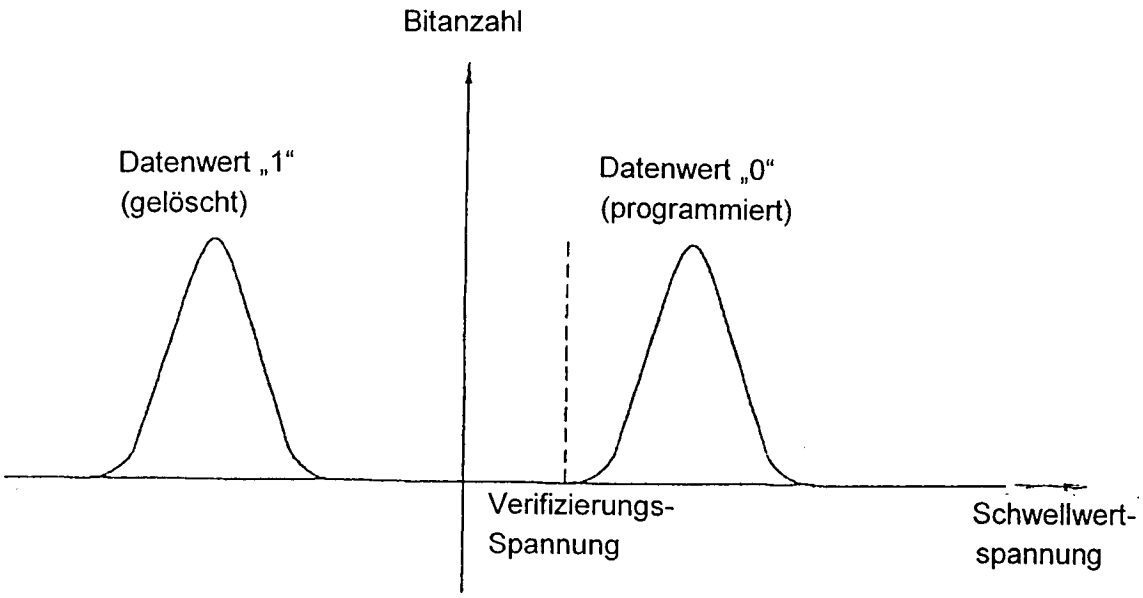

F1
F2
F3
F4
F5
F6
F4
X-Decoder
SSL
Vdd
1.0V
Vdd
W/L(Sel.)
Vpass
Vpgm
Vread
W/L(Unsel.)
Vpass
GSL
Vread
CSL
Vdd
Vread
Seitenpuffer
VIRPWR
VBLe
Vpass
Vdd
VBLo
Vpass
Vdd
VLSHFe
VLSHFo
PBLCHM
PBLCHC
PLOAD
PBset
PDUMP1
BLSLT
DI
nDI
PRE
RESET
PDUMP2
DATA LINE

Fig. 29


Bitanzahl
Datenwert „1“
(gelöscht)
Datenwert „0“
(programmiert)
Verifizierungs-
Spannung
Schwellwert-
spannung