US 20060181934A1

(19) United States

(12) Patent Application Publication

Shappir et al.

(10) Pub. No.: US 2006/0181934 A1

(43) Pub. Date: Aug. 17, 2006

(54) METHODS FOR PREVENTING FIXED PATTERN PROGRAMMING

(75) Inventors: **Assaf Shappir**, Kiryat Ono (IL); **Shai Eisen**, Tel Aviv (IL); **Guy Cohen**, Yaad (IL); **Kobi Danon**, Rishon Lezion (IL)

Correspondence Address:
TIAJOLOFF & KELLY
CHRYSLER BUILDING, 37TH FL
405 LEXINGTON AVE
NEW YORK, NY 10174 (US)

(73) Assignee: **Saifun Semiconductors, Ltd.**, Netanya (IL)

(21) Appl. No.: **11/335,316**

(22) Filed: **Jan. 19, 2006**

Related U.S. Application Data

(60) Provisional application No. 60/644,569, filed on Jan. 19, 2005.

Publication Classification

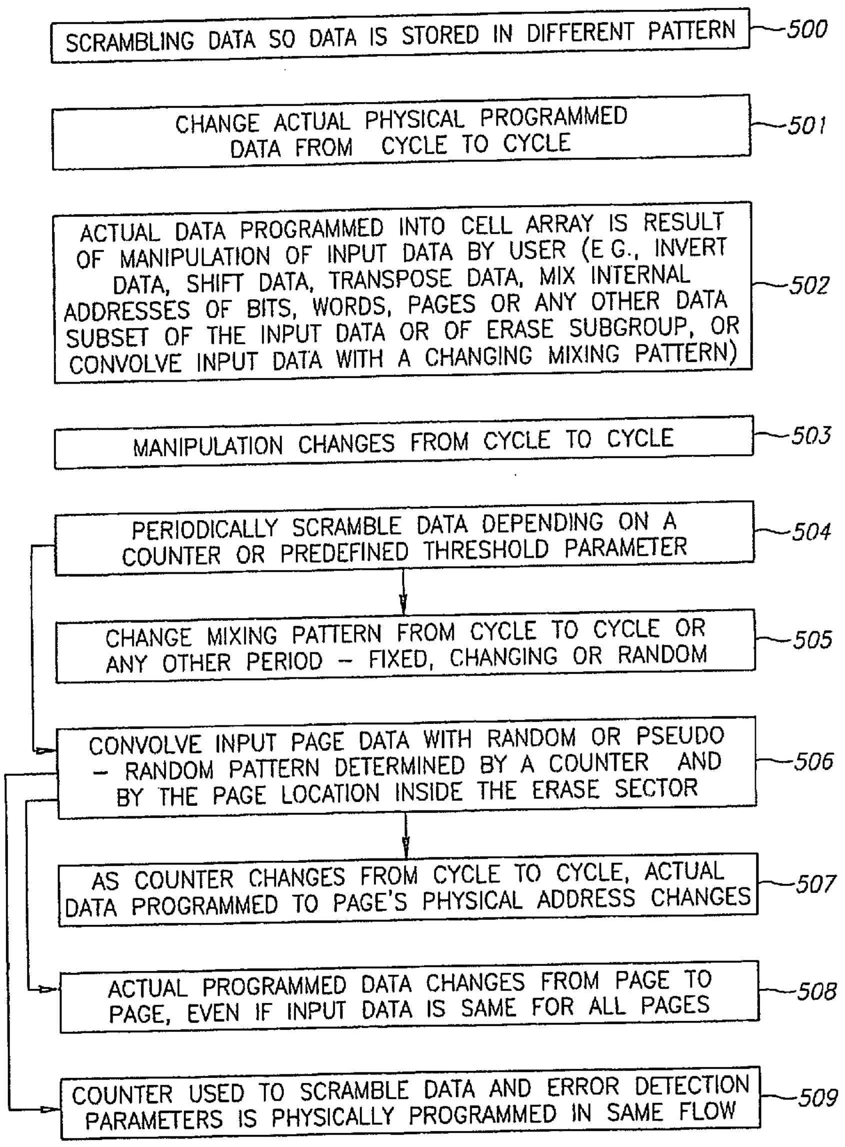
(51) **Int. Cl.**

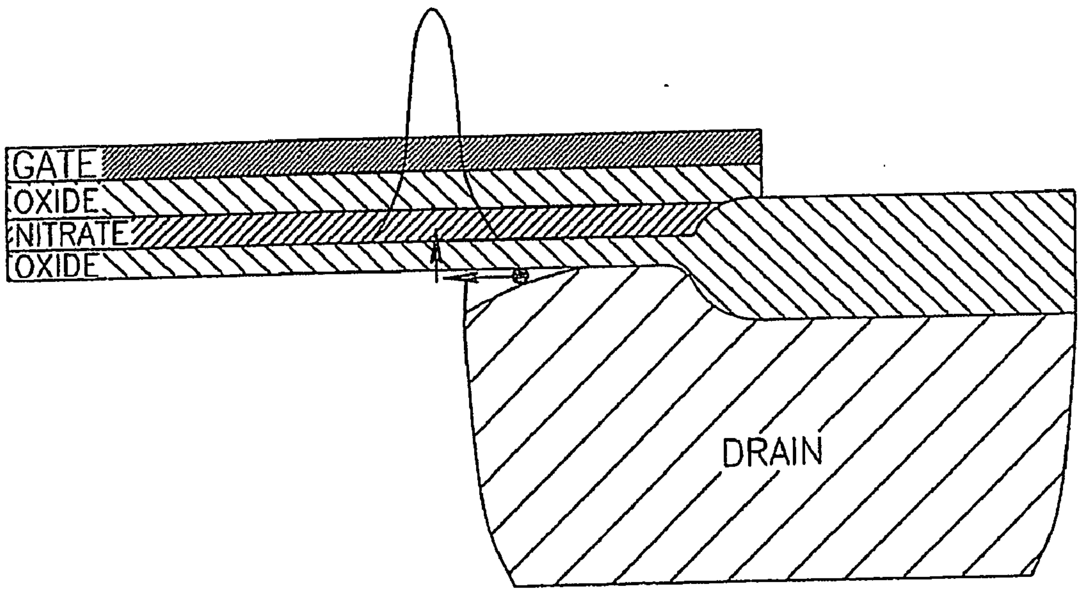
G11C 16/04	(2006.01)
G06F 12/00	(2006.01)

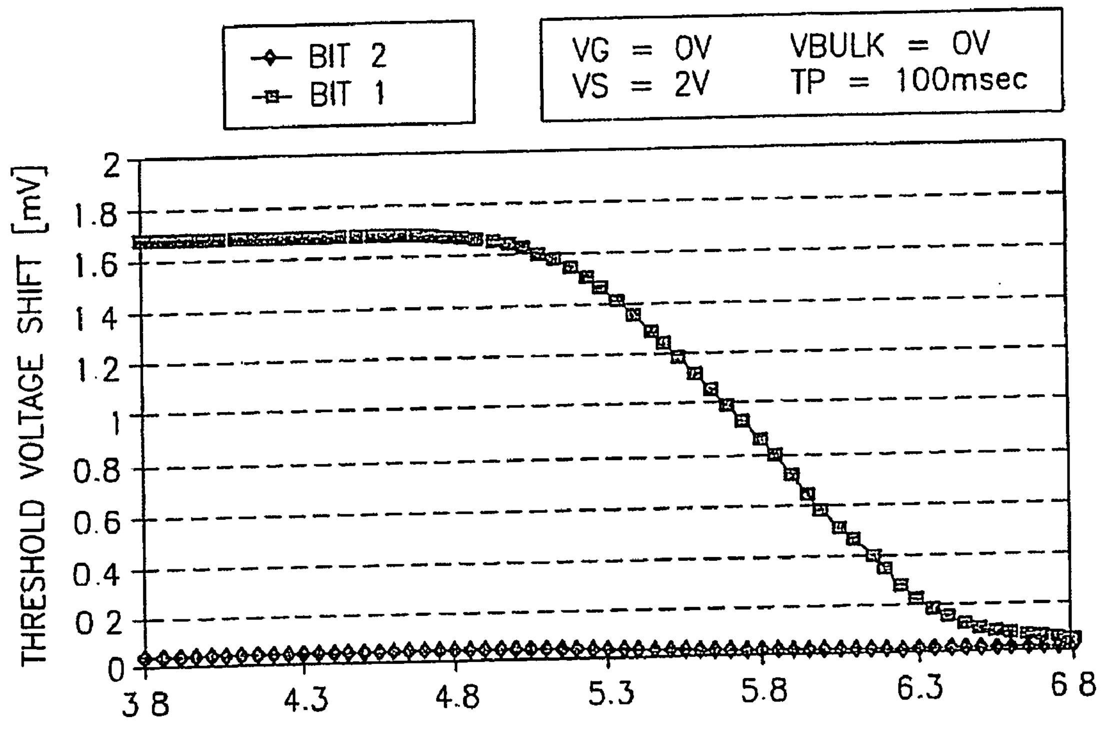
(52) **U.S. Cl.** **365/185.33**; 711/103

(57) **ABSTRACT**

A method for preventing fixed pattern programming, the method including programming data into a pattern of memory cells in a memory array, and preventing fixed pattern programming by periodically scrambling the data so that the data is stored in a different pattern of memory cells in the memory array.


SCRAMBLING DATA SO DATA IS STORED IN DIFFERENT PATTERN
500
CHANGE ACTUAL PHYSICAL PROGRAMMED DATA FROM CYCLE TO CYCLE
501
ACTUAL DATA PROGRAMMED INTO CELL ARRAY IS RESULT OF MANIPULATION OF INPUT DATA BY USER (E.G., INVERT DATA, SHIFT DATA, TRANSPOSE DATA, MIX INTERNAL ADDRESSES OF BITS, WORDS, PAGES OR ANY OTHER DATA SUBSET OF THE INPUT DATA OR OF ERASE SUBGROUP, OR CONVOLVE INPUT DATA WITH A CHANGING MIXING PATTERN)
502
MANIPULATION CHANGES FROM CYCLE TO CYCLE
503
PERIODICALLY SCRAMBLE DATA DEPENDING ON A COUNTER OR PREDEFINED THRESHOLD PARAMETER
504
CHANGE MIXING PATTERN FROM CYCLE TO CYCLE OR ANY OTHER PERIOD - FIXED, CHANGING OR RANDOM
505
CONVOLVE INPUT PAGE DATA WITH RANDOM OR PSEUDO - RANDOM PATTERN DETERMINED BY A COUNTER AND BY THE PAGE LOCATION INSIDE THE ERASE SECTOR
506
AS COUNTER CHANGES FROM CYCLE TO CYCLE, ACTUAL DATA PROGRAMMED TO PAGE'S PHYSICAL ADDRESS CHANGES
507
ACTUAL PROGRAMMED DATA CHANGES FROM PAGE TO PAGE, EVEN IF INPUT DATA IS SAME FOR ALL PAGES
508
COUNTER USED TO SCRAMBLE DATA AND ERROR DETECTION PARAMETERS IS PHYSICALLY PROGRAMMED IN SAME FLOW
509





ERASURE IS PERFORMED BY TUNNEL ENHANCED HOT HOLE INJECTION

FIG.1

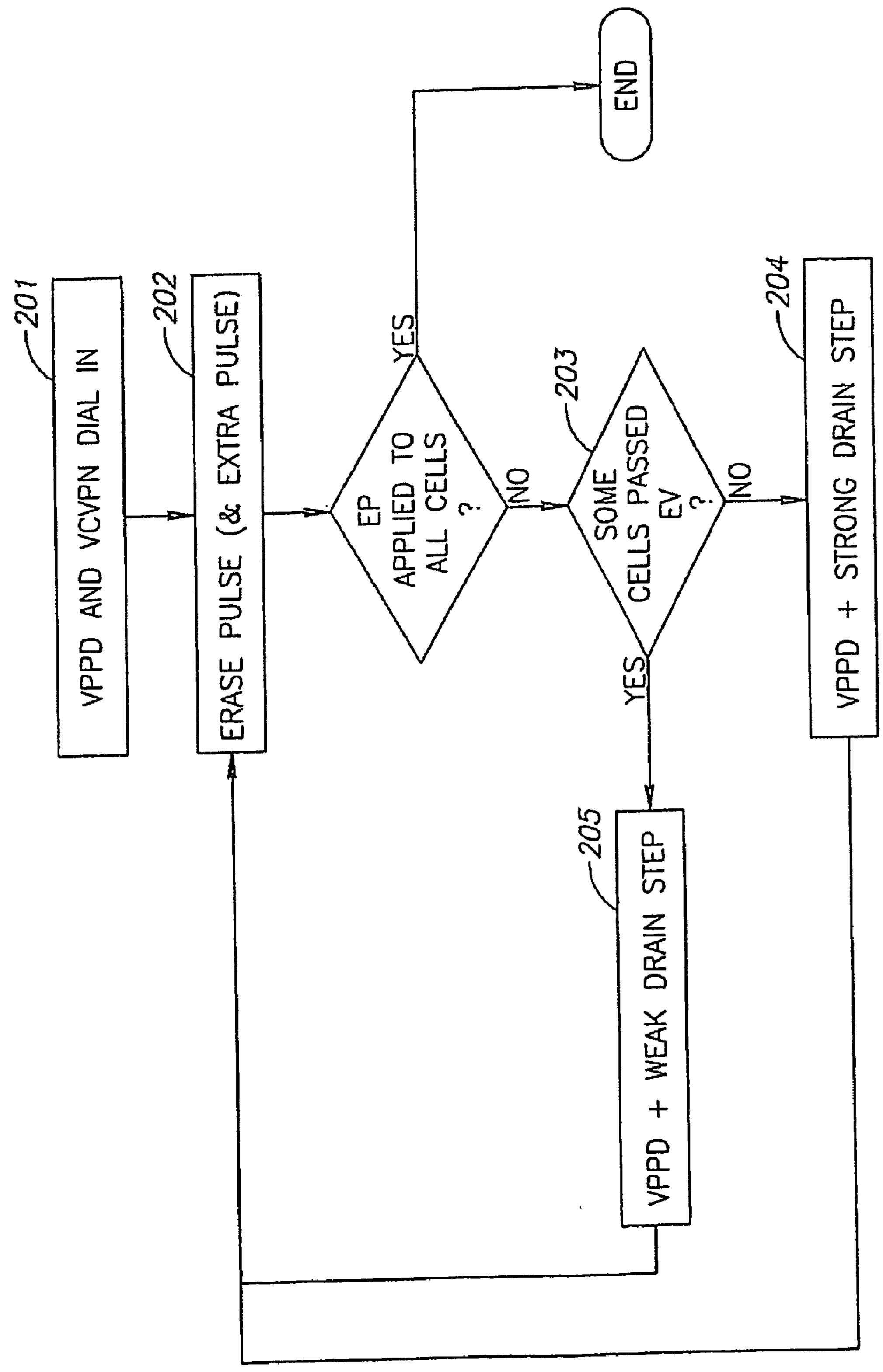

201
VPPD AND VCVPN DIAL IN
202
ERASE PULSE (& EXTRA PULSE)
EP APPLIED TO ALL CELLS ?
YES
END
NO
203
SOME CELLS PASSED EV ?
YES
NO
204
VPPD + STRONG DRAIN STEP
205
VPPD + WEAK DRAIN STEP

FIG.2

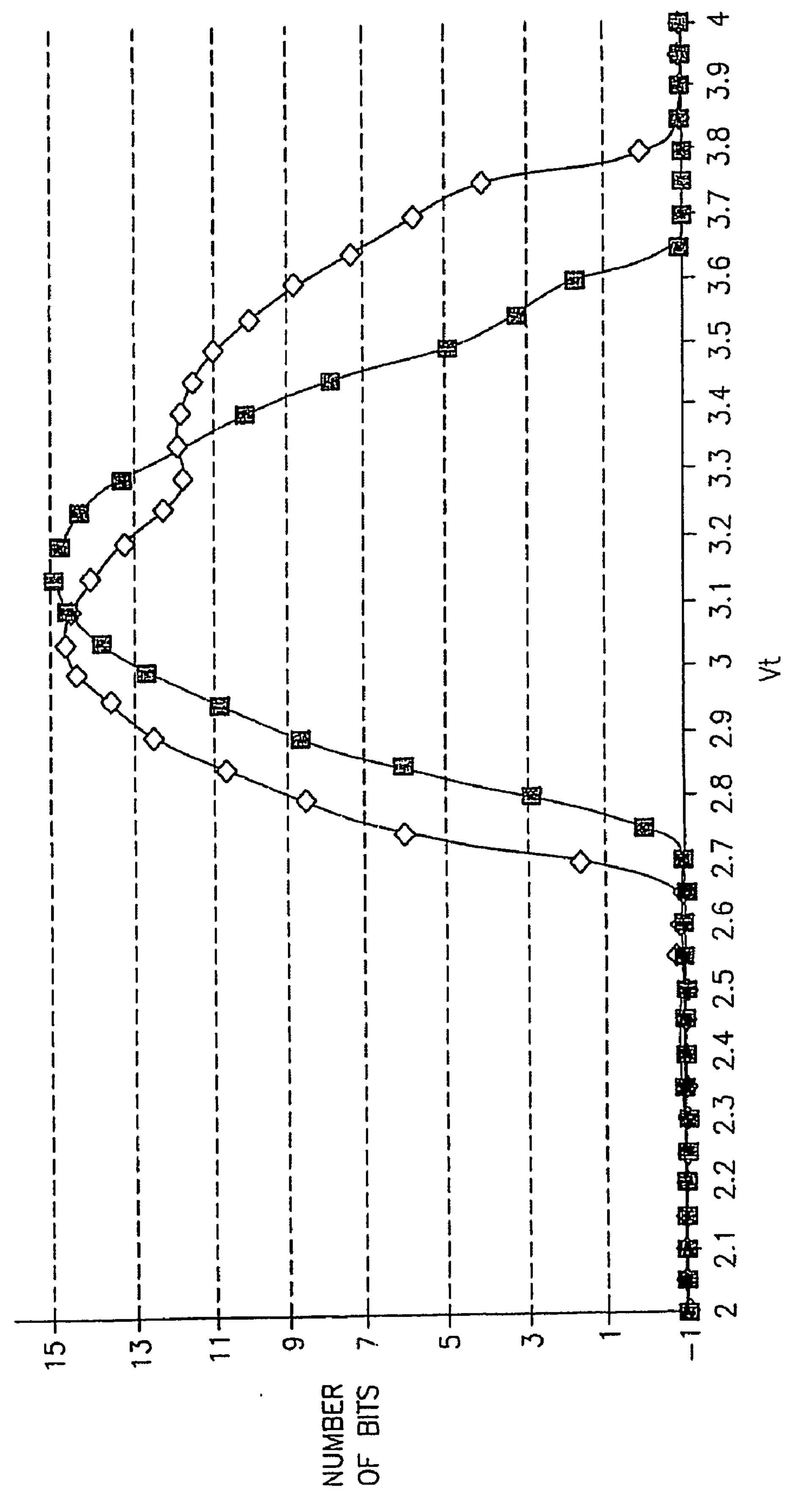

NUMBER OF BITS
15
13
11
9
7
5
3
1
-1
2
2.1
2.2
2.3
2.4
2.5
2.6
2.7
2.8
2.9
3
3.1
3.2
3.3
3.4
3.5
3.6
3.7
3.8
3.9
4
Vt

FIG.3

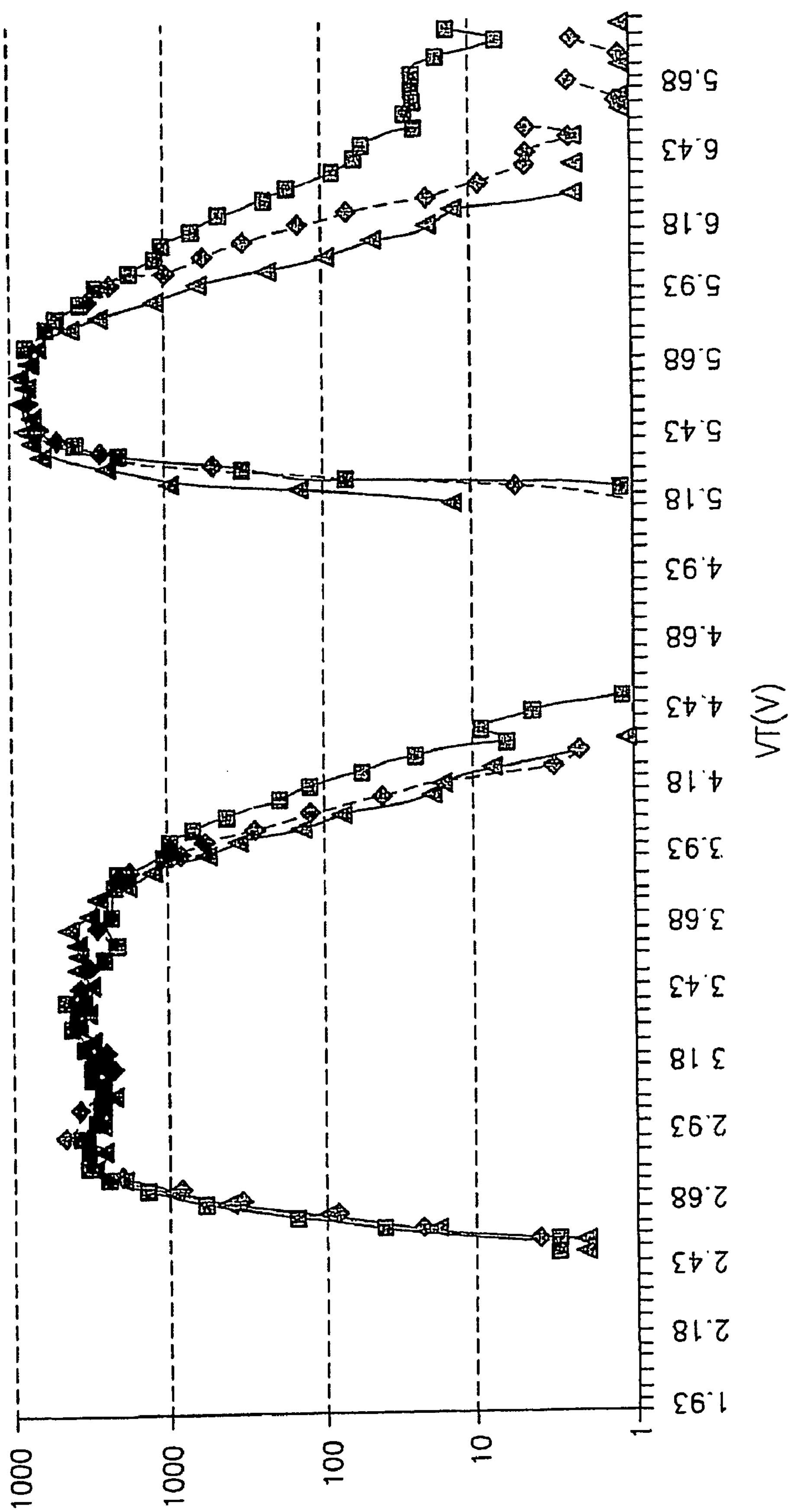

1000
1000
100
10
1
1.93
2.18
2.43
2.68
2.93
3 18
3.43
3.68
3.93
4.18
4.43
4.68
4.93
5.18
5.43
5.68
5.93
6.18
6.43
5.68
VT(V)

FIG.4

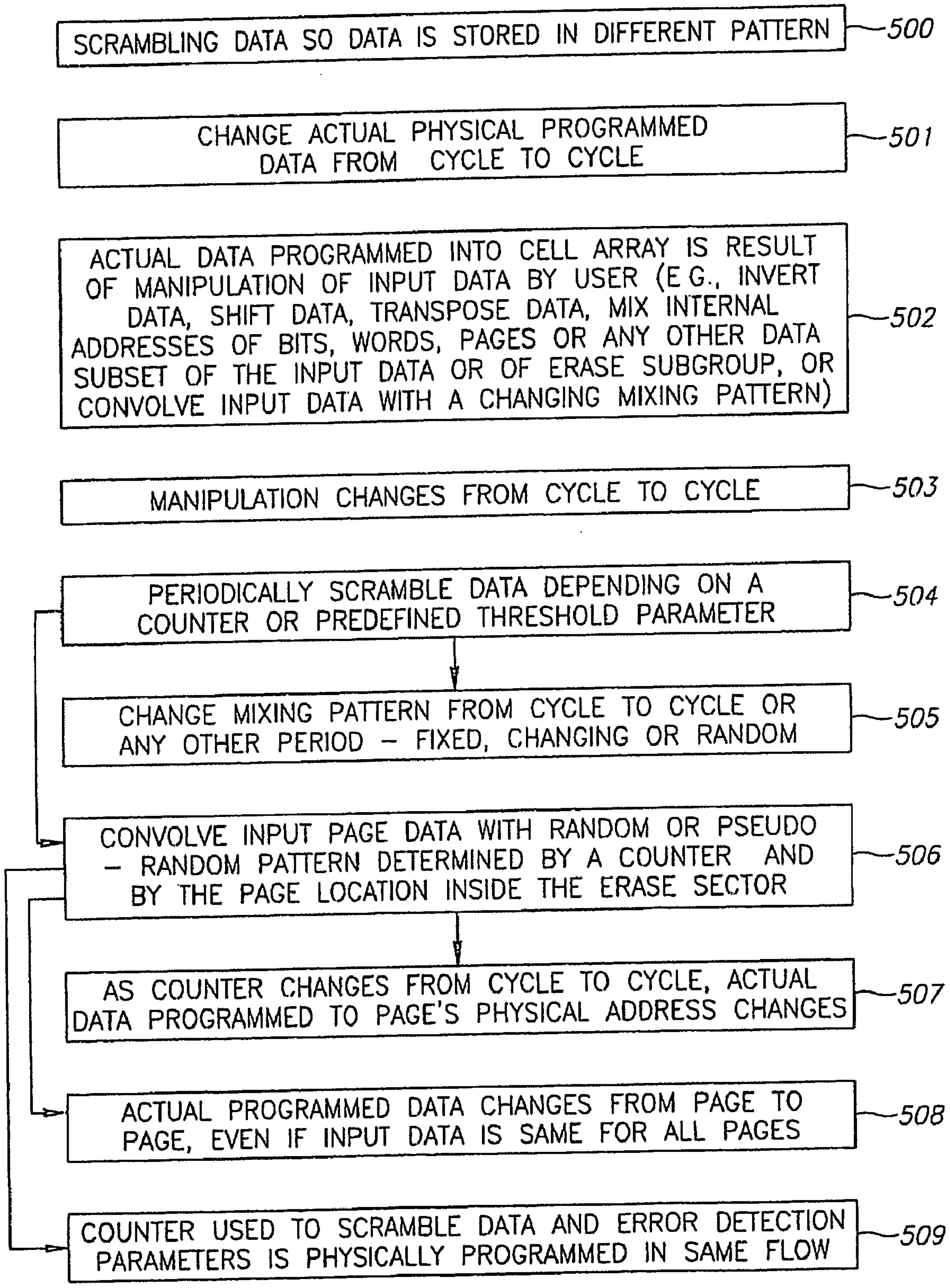

SCRAMBLING DATA SO DATA IS STORED IN DIFFERENT PATTERN
500
CHANGE ACTUAL PHYSICAL PROGRAMMED DATA FROM CYCLE TO CYCLE
501
ACTUAL DATA PROGRAMMED INTO CELL ARRAY IS RESULT OF MANIPULATION OF INPUT DATA BY USER (E.G., INVERT DATA, SHIFT DATA, TRANSPOSE DATA, MIX INTERNAL ADDRESSES OF BITS, WORDS, PAGES OR ANY OTHER DATA SUBSET OF THE INPUT DATA OR OF ERASE SUBGROUP, OR CONVOLVE INPUT DATA WITH A CHANGING MIXING PATTERN)
502
MANIPULATION CHANGES FROM CYCLE TO CYCLE
503
PERIODICALLY SCRAMBLE DATA DEPENDING ON A COUNTER OR PREDEFINED THRESHOLD PARAMETER
504
CHANGE MIXING PATTERN FROM CYCLE TO CYCLE OR ANY OTHER PERIOD – FIXED, CHANGING OR RANDOM
505
CONVOLVE INPUT PAGE DATA WITH RANDOM OR PSEUDO – RANDOM PATTERN DETERMINED BY A COUNTER AND BY THE PAGE LOCATION INSIDE THE ERASE SECTOR
506
AS COUNTER CHANGES FROM CYCLE TO CYCLE, ACTUAL DATA PROGRAMMED TO PAGE'S PHYSICAL ADDRESS CHANGES
507
ACTUAL PROGRAMMED DATA CHANGES FROM PAGE TO PAGE, EVEN IF INPUT DATA IS SAME FOR ALL PAGES
508
COUNTER USED TO SCRAMBLE DATA AND ERROR DETECTION PARAMETERS IS PHYSICALLY PROGRAMMED IN SAME FLOW
509

FIG.5

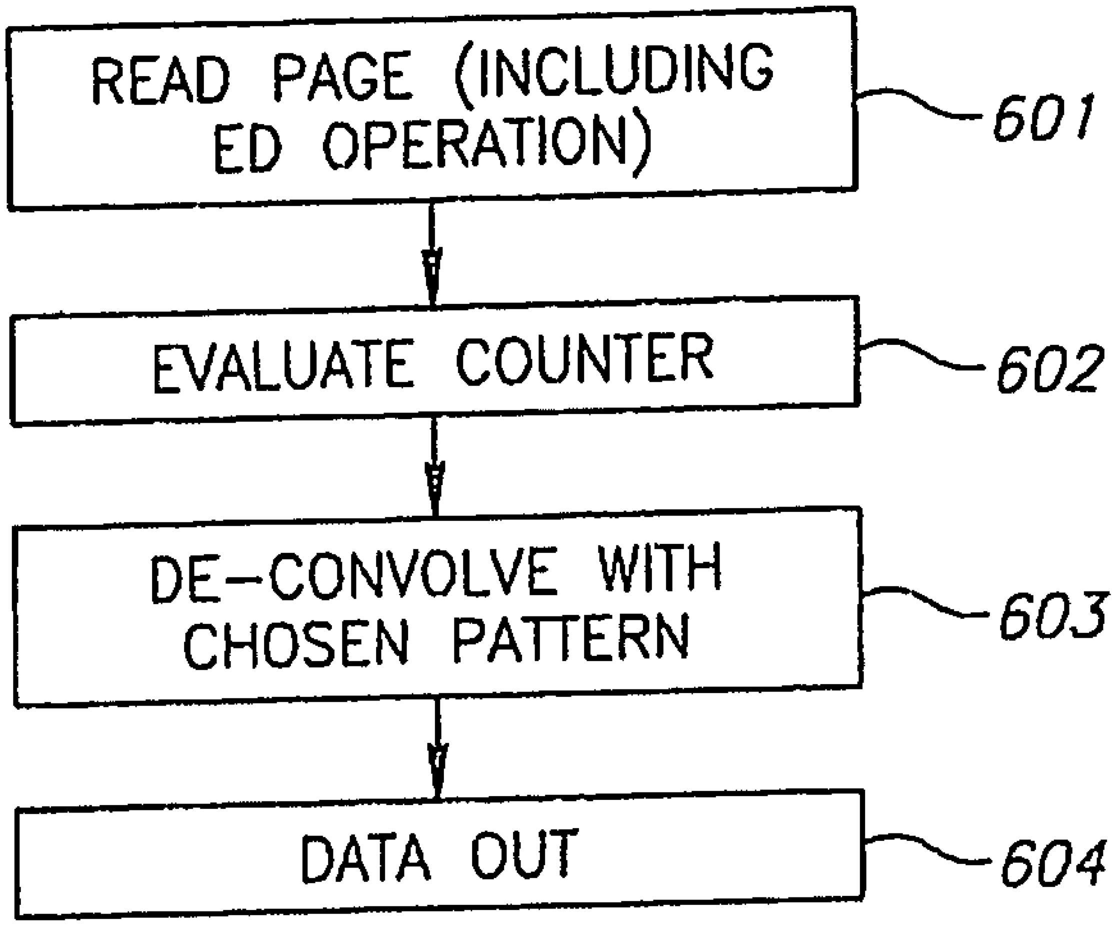

READ PAGE (INCLUDING ED OPERATION)
601
EVALUATE COUNTER
602
DE-CONVOLVE WITH CHOSEN PATTERN
603
DATA OUT
604

FIG.6

METHODS FOR PREVENTING FIXED PATTERN PROGRAMMING

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] The present application claims priority from U.S. Provisional Application Ser. No. 60/644,569, filed Jan. 19, 2005, which is incorporated herein by reference

FIELD OF THE INVENTION

[0002] The present invention relates generally to operating memory cells of non-volatile memory (NVM) arrays, such as programming and erasing, and particularly to methods for preventing large differences in the program and erase history of cells, such as by data scrambling.

BACKGROUND OF THE INVENTION

[0003] Modern day non-volatile memory products incorporate the ability to electrically program and erase the memory cells. In most cases, the erase operation is performed on a subset of cells and not individually cell-by-cell, as normally performed during the programming operation. This means that erasure conditions are applied to the subset until the last (slowest) cell finishes erasure, including verification that the cell has passed a predetermined level (erase verify).

[0004] Memory products incorporating tunneling enhanced hot hole injection during erasure, as in NROM (nitride read-only memory) technology, require high biasing of the transistor junction to create the injected holes, through band-to-band tunneling, as may be seen in **FIG. 1**. Charge injection must be controlled to insure proper device operation, and accordingly, step and verify algorithms are typically implemented. In a typical algorithm, charge is injected at a certain bias following by a verify operation to ascertain whether the cell has reached its destination. If the destination has not been achieved, stronger charge injection is initiated via a higher bias and vice versa.

[0005] Reference is now made to **FIG. 2**, which illustrate a typical flow diagram of an erase algorithm of the prior art for NROM devices.

[0006] An erase pulse may be selected for erasing bits of the cells, comprising selecting ("dialing in") a negative gate voltage (Vg or Vcvpn—voltage from a charge pump) and a positive drain voltage (Vppd) (step **201**) The erase pulse may then be applied to the bits in a cell ensemble (step **202**). The threshold voltage Vt of the cells may be read with an erase verify step (step **203**) that checks if the memory cell threshold voltage has been lowered to an erase verify (EV) voltage level or not.

[0007] If no cells have passed EV, then a new Vppd level may be set (dialed in) with a strong (i.e., large) increment (step **204**). If some cells have passed EV, then a new Vppd level may be set with a weak (i.e., relatively smaller) increment (step **205**). The process continues until the erase pulse has been applied to all cells. Each subgroup may receive an extra erase pulse at a higher level than the last pulse used to reach full erasure for improving reliability. Application of additional pulses is taught in various patent documents, such as U.S. Pat. No. 6,700,818 and US Patent Applications 20050117395 and 20050058005, all assigned to the present assignee of the present application, the disclosures of which are incorporated herein by reference.

[0008] For tunneling enhanced hot hole injection, the process shown in **FIG. 2** usually has to be performed on both sides of the memory cell separately, resulting in longer erase time and lower performance

[0009] As the data stored in a subset of cells will most likely be random in nature, some of the cells will be in the programmed state and some will be in the erased state, prior to the erasure operation. Thus, if no special actions are taken, it is possible that a previously programmed cell upon which the erase operation is performed may reach a threshold level near or at the erase verify level, while cells which were not programmed may become over-erased, wherein the threshold voltage is substantially below the erase verify level. This may be seen in **FIG. 3**, which shows a prior art threshold voltage distribution of a subset of cells in their native state, before any operation was performed, and after erasure, where only a subgroup was previously programmed. The erased cell distribution has two peaks. The higher peak is the threshold voltage distribution of the previously programmed cells, while the lower peak is that of the non-programmed cells, i.e., the cells that were over-erased.

[0010] Several concerns arise from the over-erased situation. Over-erased cells may become leaky, i.e., conduct current without being biased to the "on" state (positive gate voltage for an n-MOSFET based memory cell) Over-erased cells may become hard to program, i.e., require excessive voltages and time to bring them to the programmed state (above a predefined level, program verify level) A substantial difference in the operating conditions (program and erase) may develop between cells which have been over-erased and cells which have not been over-erased. The sum of these effects may result in failure of the memory device, i.e., loss of data integrity.

[0011] Reference is now made to **FIG. 4**, which illustrates prior art threshold voltage distributions in a subgroup of cells in a memory array following extensive cycling of a fixed pattern (10^5 cycles) and a subsequent programming of a checkerboard pattern. A programming tail is shown to have formed, due to the lack of sufficient over-erasure prevention measures

[0012] Prior art methods to prevent over-erasure of cell include partitioning of the erase subgroup, which has the advantage that smaller erase subgroups insure better erase uniformity. However, disadvantages include additional overhead (reduced performance) and design complexity. Another prior art method involves programming before erasure, which ensures that cells will not constantly go through erasure without ever being programmed. However, this carries a substantial performance penalty in time and power.

[0013] Another prior art method involves programming after erasure, which ensures that cells will not be over-erased beyond a set level. However, this method also carries a substantial performance penalty in time and power.

SUMMARY OF THE INVENTION

[0014] The present invention seeks to provide methods for preventing fixed pattern programming, which may prevent large differences in the program and erase history of cells of memory arrays, as is described hereinbelow. The invention is described in detail hereinbelow with reference to memory cells of NVM arrays, and particularly to single bit, dual bit, multi-bit and multi-level NROM cells. However, it should be emphasized that the invention is not limited to NROM arrays.

[0015] There is thus provided in accordance with an embodiment of the invention a method for preventing fixed pattern programming, the method including programming data into a pattern of memory cells in a memory array, and preventing fixed pattern programming by periodically scrambling the data so that the data is stored in a different pattern of memory cells in the memory array.

[0016] The data may be scrambled by periodically inverting the data. Additionally or alternatively, the data may be scrambled by periodically rearranging a physical address of the data. Additionally or alternatively, the data may be scrambled by convolving the data with a mixing pattern during programming and de-convolving the convolved data during a read operation. Additionally or alternatively, the data may be scrambled by periodically scrambling the data in accordance with a program and erase cycle counter. The data may be periodically scrambled randomly or pseudo-randomly.

[0017] In accordance with a non-limiting embodiment of the invention, the data may be scrambled as a function of a stored parameter inside the memory array. The stored parameter may be programmed in conjunction with the data.

[0018] Additionally or alternatively, the data may be scrambled as a function of a logical and/or physical address of the data.

BRIEF DESCRIPTION OF THE DRAWINGS

[0019] The present invention will be understood and appreciated more fully from the following detailed description taken in conjunction with the appended drawings in which:

[0020] **FIG. 1** is a simplified graph of erasing NROM cells by tunneling enhanced hot hole injection in the prior art;

[0021] **FIG. 2** is a simplified flow diagram of an erase algorithm of the prior art for NROM devices;

[0022] **FIG. 3** is a simplified graph of a prior art threshold voltage distribution of a subgroup of cells in a memory array in their native state and after erasure, wherein only a subset of the cells have been previously programmed;

[0023] **FIG. 4** is a simplified graph of prior art threshold voltage distributions in a subgroup of cells in a memory array following extensive cycling of a fixed pattern (10^5 cycles) and a subsequent programming of a checkerboard pattern;

[0024] **FIG. 5** is a simplified flow chart of the programming flow in a data scrambling implementation used to prevent fixed pattern programming, in accordance with an embodiment of the invention; and

[0025] **FIG. 6** is a simplified flow diagram of the read flow in the data scrambling implementation, in accordance with an embodiment of the invention.

DETAILED DESCRIPTION OF EMBODIMENTS

[0026] Reference is now made to **FIG. 5**, which illustrates a method for preventing fixed pattern programming in a non-volatile memory cell array, in accordance with an embodiment of the present invention. Specifically, **FIG. 5** is a simplified flow chart of the programming flow in a data scrambling implementation, as is now explained.

[0027] In accordance with an embodiment of the present invention, after programming data into a pattern of memory cells in a memory array, fixed pattern programming is prevented by periodically scrambling the data so that the data is stored in a different pattern of memory cells in the memory array (step **500**). In a non-limiting embodiment of the present invention, the actual physical programmed data is forced to change from cycle to cycle (step **501**). Thus the rate at which systematic differences can develop, due to fix pattern programming, may be substantially reduced. The actual data programmed into the cell array may be a result of a manipulation of the input data by the user (step **502**). This manipulation may change from cycle to cycle (step **503**), or periodically change depending on a counter or a predefined threshold parameter (step **504**). Such a manipulation includes, but is not limited to, inverting the input data, shifting it, transposing it, mixing the internal addresses of the bits, words, pages or any other data subset of the input data or of the erase subgroup, or convolving the input data with a changing mixing pattern. The changing mixing pattern may be from cycle to cycle or any other period, fixed, changing or random (step **505**).

[0028] In accordance with a non-limiting embodiment of the present invention, wherein an erase sector consists of a group of pages (each programmed individually), the input page data is convolved with a random or pseudo-random pattern determined by a counter and by the page location inside the erase sector (step **506**). As the counter changes from cycle to cycle, the actual data programmed to the page's physical address also changes (step **507**). Moreover, since the random or pseudo-random pattern is also determined by the page's location within the erase sector, the actual programmed data changes from page to page, even if the input data is the same for all pages (step **508**). A counter used to scramble the data and error detection parameters may be physically programmed in the same flow (step **509**). Error detection parameters are discussed, for example, in U.S. patent application Ser. No. 10/695,457 (publication number US 20040136236), the disclosure of which is incorporated herein by reference, and corresponding PCT Application WO 2005/041108 "A Method Circuit And System For Read Error Detection In A Non-Volatile Memory Array", both assigned to the present assignee of the present invention. The read error rate may be determined using a variety of error rate sampling and/or error detection techniques, for example, parity bit, checksum, CRC and various other techniques. Any error detection coding and/or evaluation technique, presently known or to be devised in the future, may be applicable to present invention.

[0029] Reference is now made to **FIG. 6**, which illustrates the read operation for the data scrambling implementation of **FIG. 5**, in accordance with an embodiment of the invention. The data reliability is not affected by the scrambling scheme.

[0030] In one non-limiting embodiment, the page may be read, including the error detection parameters (step **601**). If a counter has been used, the algorithm then evaluates the counter (step **602**). The particular pattern used may then be deconvolved (step **603**) and the data extracted (step **604**).

[0031] Although the invention has been described in conjunction with specific embodiments thereof, it is evident that many alternatives, modifications and variations will be apparent to those skilled in the art. Accordingly, it is intended to embrace all such alternatives, modifications and variations.

What is claimed is:

1. A method for preventing fixed pattern programming, the method comprising:

programming data into a pattern of memory cells in a memory array; and

preventing fixed pattern programming by periodically scrambling the data so that the data is stored in a different pattern of memory cells in the memory array.

2. The method according to claim 1, wherein periodically scrambling the data comprises periodically inverting the data.

3. The method according to claim 1, wherein periodically scrambling the data comprises periodically rearranging a physical address of said data.

4. The method according to claim 1, wherein periodically scrambling the data comprises convolving said data with a mixing pattern during programming and de-convolving the convolved data during a read operation.

5. The method according to claim 1, wherein periodically scrambling the data comprises periodically scrambling the data in accordance with a program and erase cycle counter.

6. The method according to claim 1, wherein the data is periodically scrambled randomly.

7. The method according to claim 1, wherein the data is periodically scrambled pseudo-randomly.

8. The method according to claim 1, wherein periodically scrambling the data comprises scrambling as a function of a stored parameter inside the memory array.

9. The method according to claim 8, wherein said stored parameter is programmed in conjunction with said data

10. The method according to claim 1, wherein periodically scrambling the data comprises scrambling as a function of a logical address of said data.

11. The method according to claim 1, wherein periodically scrambling the data comprises scrambling as a function of a physical address of said data.

* * * * *