(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
30 April 2009 (30.04.2009)

PCT

(10) International Publication Number
WO 2009/053963 A2

(51) International Patent Classification:
G06F 7/74 (2006.01)

(21) International Application Number:
PCT/IL2008/001242

(22) International Filing Date:
17 September 2008 (17.09.2008)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

60/960,943	22 October 2007 (22.10.2007)	US
60/996,782	5 December 2007 (05.12.2007)	US
61/006,805	31 January 2008 (31.01.2008)	US
61/064,853	31 March 2008 (31.03.2008)	US
61/071,465	30 April 2008 (30.04.2008)	US
61/071,469	30 April 2008 (30.04.2008)	US
61/129,414	25 June 2008 (25.06.2008)	US
61/129,608	8 July 2008 (08.07.2008)	US

(71) Applicant *(for all designated States except US)*: **DENSBITS TECHNOLOGIES LTD.** [IL/IL]; Building 30, Flr 3, Matam Industrial Park, P.O.Box 15111, 31015 Haifa (IL).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **WEINGARTEN, Hanan** [IL/IL]; Hayarden 2, 46377 Herzliya (IL). **SABBAG, Erez** [IL/IL]; 12 Haoranim street, 36043 Kiryat Tivon (IL). **KATZ, Michael** [IL/IL]; 75 Moriah Ave., 34616 Haifa (IL).

(74) Agent: **PEARL COHEN ZEDEK LATZER**; 5 Shenkar Street, P.O. Box 12704, Herzlia 46733 (IL).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

[Continued on next page]

(54) Title: METHODS FOR ADAPTIVELY PROGRAMMING FLASH MEMORY DEVICES AND FLASH MEMORY SYSTEMS INCORPORATING SAME

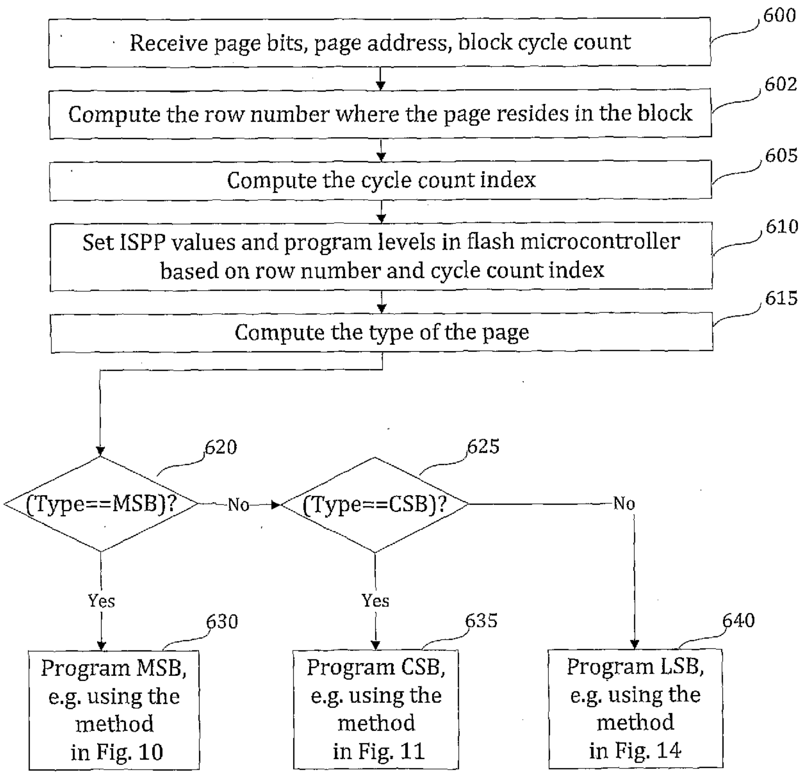
Receive page bits, page address, block cycle count
600
Compute the row number where the page resides in the block
602
Compute the cycle count index
605
Set ISPP values and program levels in flash microcontroller based on row number and cycle count index
610
Compute the type of the page
615
(Type==MSB)?
620
No
(Type==CSB)?
625
No
Yes
Yes
Program MSB, e.g. using the method in Fig. 10
630
Program CSB, e.g. using the method in Fig. 11
635
Program LSB, e.g. using the method in Fig. 14
640

Fig. 3

(57) **Abstract:** A method for programming a plurality of data sequences into a corresponding plurality of flash memory functional units using a programming process having at least one selectable programming duration-controlling parameter controlling the duration of the programming process for a given data sequence, the method comprising providing at least one indication of at least one varying situational characteristic and determining a value for said at least one selectable programming duration-controlling parameter controlling the duration of the programming process for a given data sequence, for each flash memory functional unit, depending at least partly on said indication of said varying characteristic; and, for each individual flash memory functional unit from among said plurality of flash memory functional units, programming a sequence of bits into said individual flash memory functional unit using a programming process having at least one selectable parameter, said at least one selectable parameter being set at said value determined for said individual flash memory functional unit.

(84) **Designated States** *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MT, NL, NO, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *without international search report and to be republished upon receipt of that report*

METHODS FOR ADAPTIVELY PROGRAMMING FLASH MEMORY DEVICES AND FLASH MEMORY SYSTEMS INCORPORATING SAME

REFERENCE TO CO-PENDING APPLICATIONS

Priority is claimed from the following co-pending applications: US Provisional Application No. 61/129,414, filed June 25, 2008 and entitled "Improved Programming Speed in Flash Devices Using Adaptive Programming", US Provisional Application No. 61/006,805, filed January 31, 2008 and entitled "A Method for Extending the Life of Flash Devices", US Provisional Application No. 61/071,465, filed April 30, 2008 and entitled "Systems and Methods for Temporarily Retiring Memory Portions", US Provisional Application No. 60/996,782, filed December 5, 2007 and entitled "Systems and Methods for Using a Training Sequence in Flash Memory", US Provisional Application No. 61/064,853, filed March 31, 2008 and entitled "Flash Memory Device with Physical Cell Value Deterioration Accommodation and Methods Useful in Conjunction Therewith", US Provisional Application No. 61/129,608, filed July 8, 2008 and entitled "A Method for Acquiring and Tracking Detection Thresholds in Flash Devices", US Provisional Application No. 60/960,943, filed October 22, 2007 and entitled "Systems and methods to reduce errors in Solid State Disks and Large Flash Devices" and US Provisional Application No. 61/071,469, filed April 30, 2008 and entitled "Systems and Methods for Averaging Error Rates in Non-Volatile Devices and Storage Systems".

Other co-pending applications include: US Provisional Application No. 60/960,207, filed September 20, 2007 and entitled "Systems and Methods for Coupling Detection in Flash Memory", US Provisional Application No. 61/071,467, filed April 30, 2008 and entitled "Improved Systems and Methods for Determining Logical Values of Coupled Flash Memory Cells", US Provisional Application No. 60/996,027, filed October 25, 2007 and entitled "Systems and Methods for Coping with Variable Bit Error Rates in Flash Devices", US Provisional Application No. 61/071,466, filed April 30, 2008 and entitled "Systems and Methods for Multiple Coding Rates in Flash Devices", US Provisional Application No. 61/006,120, filed December 19, 2007 and entitled "Systems and Methods for Coping with Multi Stage Decoding in Flash Devices", US Provisional Application No. 61/071,464, filed April

30, 2008 and entitled "A Decoder Operative to Effect A Plurality of Decoding Stages Upon Flash Memory Data and Methods Useful in Conjunction Therewith", US Provisional Application No. 61/006,385, filed January 10, 2008 and entitled "A System for Error Correction Encoder and Decoder Using the Lee Metric and Adapted to Work on Multi-Level Physical Media", US Provisional Application No. 61/064,995, filed April 8, 2008 and entitled "Systems and Methods for Error Correction and Decoding on Multi-Level Physical Media", US Provisional Application No. 60/996,948, filed December 12, 2007 and entitled "Low Power BCH/RS Decoding: a Low Power Chien-Search Implementation", US Provisional Application No. 61/071,487, filed May 1, 2008 and entitled "Chien-Search System Employing a Clock-Gating Scheme to Save Power for Error Correction Decoder and other Applications", US Provisional Application No. 61/071,468, filed April 30, 2008 and entitled "A Low Power Chien-Search Based BCH/RS Recoding System for Flash Memory, Mobile Communications Devices and Other Applications", US Provisional Application No. 61/006,806, filed January 31, 2008 and entitled "Systems and Methods for using a Erasure Coding in Flash memory", US Provisional Application No. 61/071,486, filed May 1, 2008 and entitled "Systems and Methods for Handling Immediate Data Errors in Flash Memory", US Provisional Application No. 61/006,078, filed December 18, 2007 and entitled "Systems and Methods for Multi Rate Coding in Multi Level Flash Devices", US Provisional Application No. 61/064,923, filed April 30, 2008 and entitled "Apparatus For Coding At A Plurality Of Rates In Multi-Level Flash Memory Systems, And Methods Useful In Conjunction Therewith", US Provisional Application No. 61/064,760, filed March 25, 2008 and entitled "Hardware efficient implementation of rounding in fixed-point arithmetic", US Provisional Application No. 61/071,404, filed April 28, 2008 and entitled "Apparatus and Methods for Hardware-Efficient Unbiased Rounding", US Provisional Application No. 61/136,234, filed August 20, 2008 and entitled "A Method Of Reprogramming A Non-Volatile Memory Device Without Performing An Erase Operation", and several other co-pending patent applications being filed concurrently (same day).

FIELD OF THE INVENTION

The present invention relates generally to flash memory devices and more particularly to programming thereof.

BACKGROUND OF THE INVENTION

Conventional flash memory technology is described in the following publications inter alia:

[1] Paulo Cappelletti, Clara Golla, Piero Olivo, Enrico Zanoni, "Flash Memories", Kluwer Academic Publishers, 1999

[2] G. Campardo, R. Micheloni, D. Novosel, "VLSI-Design of Non-Volatile Memories", Springer Berlin Heidelberg New York, 2005

[3] US Patents 6,3301,151 to Engh et al; 7,151,701 to Combe et al; 7,292,473 to Niset et al; and 7,355,896 to Li et al; as well as published PCT document WO 2006138333 and published European patent document EP 1833058.

[4] J. E. Brewer and M. Gill, "Nonvolatile memory technologies with emphasis on flash," IEEE press series on Microelectronic Systems, John Wiley & Sons, inc., 2008.

The disclosures of all publications and patent documents mentioned in the specification, and of the publications and patent documents cited therein directly or indirectly, are hereby incorporated by reference.

SUMMARY OF THE INVENTION

The following terms may be construed either in accordance with any definition thereof appearing in the prior art literature or in accordance with the specification, or as follows:

Bit error rate (BER) = a parameter that a flash memory device manufacturer commits to vis a vis its customers, expressing the maximum proportion of wrongly read bits (wrongly read bits/ total number of bits) that users of the flash memory device need to expect at any time during the stipulated lifetime of the flash memory device e.g. 10 years.

Block = a set of flash memory device cells which must, due to physical limitations of the flash memory device, be erased together. Also termed erase sector, erase block.

Cell: A component of flash memory that stores one bit of information (in single-level cell devices) or n bits of information (in a multi-level device having 2 exp n levels). Typically, each cell comprises a floating-gate transistor. n may or may not be an integer. "Multi-level" means that the physical levels in the cell are, to an acceptable level of certainty, statistically partitionable into multiple distinguishable regions, plus a region corresponding to zero, such that digital values each comprising multiple bits can be represented by the cell. In contrast, in single-level cells, the physical levels in the cell are assumed to be statistically partitionable into only two regions, one corresponding to zero and one other, non-zero region, such that only one bit can be represented by a single-level cell.

Charge level: the measured voltage of a cell which reflects its electric charge.

Cycling: Repeatedly writing new data into flash memory cells and repeatedly erasing the cells between each two writing operations.

Decision regions: Regions extending between adjacent decision levels, e.g. if decision levels are 0, 2 and 4 volts respectively, the decision regions are under 0 V, 0 V – 2 V, 2V – 4 V, and over 4 V.

Demapping: basic cell-level reading function in which a digital n-tuple originally received from an outside application is derived from a physical value representing a physical state in the cell having a predetermined correspondence to the digital n-tuple.

Digital value or "logical value": n-tuple of bits represented by a cell in flash memory capable of generating 2 exp n distinguishable levels of a typically continuous physical value such as charge , where n may or may not be an integer.

Erase cycle: The relatively slow process of erasing a block of cells (erase sector), each block typically comprising more than one page, or, in certain non-flash memory devices, of erasing a single cell or the duration of so doing. An advantage of erasing cells collectively in blocks as in flash memory, rather than individually, is enhanced programming speed: Many cells and typically even many pages of cells are erased in a single erase cycle.

Erase-write cycle: The process of erasing a block of cells (erase sector), each block typically comprising a plurality of pages, and subsequently writing new data into at least some of them. The terms "program" and "write" are used herein generally interchangeably.

Flash memory: Non-volatile computer memory including cells that are erased block by block, each block typically comprising more than one page, but are written into and read from, page by page. Includes NOR-type flash memory, NAND-type flash memory, and PRAM, e.g. Samsung PRAM, inter alia, and flash memory devices with any suitable number of levels per cell, such as but not limited to 2, 4, or 8.

Logical page: a portion of typically sequential data, whose amount is typically less than or equal to a predetermined amount of data defined to be a pageful of data, which has typically been defined by a host (data source/destination) or user thereof, as a page, and which is sent by the host to a flash memory device for storage and is subsequently read by the host from the flash memory device.

Mapping: basic cell-level writing function in which incoming digital n-tuple is mapped to a program level by inducing a program level in the cell, having a predetermined correspondence to the incoming logical value.

Physical Page = A portion, typically 512 or 2048 or 4096 bytes in size, of a flash memory e.g. a NAND or NOR flash memory device. Writing and reading is typically performed physical page by physical page, as opposed to erasing which can be performed only erase sector by erase sector. A few bytes, typically 16 – 32 for every 512 data bytes are associated with each page (typically 16, 64 or 128 per page), for storage of error correction information. A typical block may include 32 512-byte pages or 64 2048-byte pages. Alternatively, a physical page is an ordered set (e.g. sequence or array) of flash memory cells which are all written in simultaneously by each write operation, the set typically comprising a predetermined number of typically physically adjacent flash memory cells containing actual data written by and subsequently read by the host, as well as, typical error correction information and back pointers used for recognizing the true address of a page.

Precise read, soft read: Cell threshold voltages are read at a precision (number of bits) greater than the number of Mapping levels (2^n). The terms precise read or soft read are interchangeable. In contrast, in "hard read", cell threshold voltages are read at a precision (number of bits) smaller than, or equal to, the number of Mapping levels (2^n where n = number of bits per cell).

Present level, Charge level: The amount of charge in the cell. The amount of charge currently existing in a cell, at the present time, as opposed to "program level", the amount of charge originally induced in the cell (i.e. at the end of programming).

Program: same as "write".

Program level (programmed level, programming level): amount of charge originally induced in a cell to represent a given logical value, as opposed to "present level".

Reliability: Reliability of a flash memory device may be operationalized as the probability that a worst-case logical page written and stored in that device for a predetermined long time period such as 10 years will be successfully read i.e. that sufficiently few errors, if any, will be present in the physical page/s storing each logical page such that the error code appended to the logical page will suffice to overcome those few errors.

Reprogrammability (Np): An aspect of flash memory quality. This is typically operationalized by a reprogrammability parameter, also termed herein "Np", denoting the number of times that a flash memory can be re-programmed (number of erase-write cycles that the device can withstand) before the level of errors is so high as to make an unacceptably high proportion of those errors irrecoverable given a predetermined amount of memory devoted to redundancy. Typically, recoverability is investigated following a conventional aging simulation process which simulates or approximates the data degradation effect that a predetermined time period e.g. a 10 year period has on the flash memory device, in an attempt to accommodate for a period of up to 10 years between writing of data in flash memory and reading of the data therefrom.

Resolution: Number of levels in each cell, which in turn determines the number of bits the cell can store; typically a cell with 2^n levels stores n bits. Low resolution (partitioning the window, W, of physical values a cell can assume into a small rather than large number of levels per cell) provides high reliability.

Retention: Retention of original physical levels induced in the flash memory cells despite time which has elapsed and despite previous erase/write cycles; retention is typically below 100% resulting in deterioration of original physical levels into present levels.

Retention time: The amount of time that data has been stored in a flash device, typically without, or substantially without, voltage having been supplied to the flash device i.e. the time which elapses between programming of a page and reading of the same page.

Symbol: Logical value

Threshold level or "decision level": the voltage (e.g.) against which the charge level of a cell is measured. For example, a cell may be said to store a particular digital n-tuple D if the charge level or other physical level of the cell falls between two threshold values T.

Code rate: ratio of redundancy bits to data bits in flash memory.

Data cells: cells storing data provided by host as opposed to "non-data cells" which do not store host-provided data, and may, for example, store instead error correction information, management information, redundancy information, spare bits or parity bits.

Logical page: a set of bits defined as a page typically having a unique page address, by a host external to a flash memory device.

In the present specification, the terms "row" and "column" refer to rows of cells and columns of cells, respectively and are not references to sub-divisions of a logical page.

The term "MSB" is used herein to denote the bit which is programmed into a multi-level cell, storing several bits, first. The term "LSB" is used herein to denote the bit which is programmed into the multi-level cell, last. The term "CSB" is used herein to denote the bit which is programmed into a 3-level cell, storing 3 bits, second, i.e. after the MSB and before the LSB. It is appreciated that more generally, e.g. if the multi-level cell stores 4 or more levels, there are more than one CSB and use of the term "CSB" herein, which implies that the cell is a 3-level cell, is merely by way of example and is not intended to be limiting.

A logical page is a set of bytes which is meaningful to an application. The location of a logical page in memory is termed herein a physical page. This location may comprise certain cells in their entirety, or, more commonly, may comprise only one or some bits within certain cells. The locations of each of a logical sequence of logical pages (page 0, page 1, page 2,....) within memory is pre-determined by a suitable mapping scheme mapping logical pages into the bits of the cells of a particular erase sector (block) in flash memory.

"Successfully reconstructed " means that using error correction code, the original logical page has been reconstructed generally satisfactorily, e.g., typically, that the logical page has been read, using reading thresholds, has undergone error correction as necessary and has successfully passed its CRC (cyclic redundancy check) criterion.

"Bit errors" are those errors found in the physical page corresponding to a logical page, which typically are corrected using ECC (error correction code) such that the page is successfully reconstructed despite these errors.

The term "reading threshold" and "detection threshold" are used generally interchangeably.

In the context of the present application, the term "programming" comprises the following operations: Take as input a sequence of bits to be stored in memory, transform respectively into "programmed values" which are physical values which are taken to represent these bits and induce the programmed values in cells of flash memory, resulting in physical values which cluster around the programmed values respectively. The term "program" in this application does not necessarily include the process of coding e.g. error correction coding in which redundancy bits are added. Typically, programming is a final procedure which transforms a sequence of binary logical values which have previously undergone processes such as scrambling, addition of CRC, and coding.

A programming process is a method for inducing given programmed values in flash memory cells. Typically, the programming process involves a sequence of voltage pulses applied to a flash memory cell, each pulse increasing the voltage level of the cell. After each such pulse, the process may determine whether or not to continue, depending on whether the programmed value has been achieved.

"Degradation state" is a changing characteristic of a flash memory device indicating the quality of storage provided by the device, e.g. the accuracy of data retrieved from the device.

A "duration controlling parameter" is a parameter which is monotonically related to the duration of programming (i.e. the duration controlling parameter is either a monotonically increasing function of the duration or a monotonically decreasing function of the duration) to the extent that the duration controlling parameter is increased, the duration either increases (if the parameter is an increasing function of the duration) or decreases (if the parameter is a decreasing function of the duration) such that a range of desired decreases of the duration is achievable by suitable modification of the duration controlling parameter.

The term "Even row-half" refers to cells within a physical flash memory row whose indices are even numbers e.g. cells 0, 2, 4, ... in a particular row. The term "odd row-half" refers to cells within a physical flash memory row whose indices are odd numbers e.g. cells 1, 3, 5, ... in a particular row. Bits of a logical page are usually mapped to a particular bit position (such as the MSB, CSB or LSB position) of all cells in a particular even row-half or odd row-half. The number of bits per page typically equals the number of cells in a row-half and therefore if each cell is an 8-level cell which stores 3 bits then there are 3 logical pages stored in each row-half. Given a logical page, the particular row-half in which it is stored, and the particular bit within the row-half in which it resides (MSB, CSB or LSB) are typically determined by what is available.

Certain embodiments of the present invention seek to provide improved methods and systems for programming flash memory.

Certain embodiments of the present invention seek to solve a problem which includes increasing efficiency of programming e.g. by increasing Incremental Step pulse Programming when possible.

Certain embodiments of the present invention seek to solve a problem which includes compensating for poor quality rows, e.g. by selecting row-appropriate program levels.

According to certain embodiments of the present invention, programming proceeds at least partly according to situational characteristics such as but not limited to the degradation state of the flash device. The manner in which a given digital sequence is programmed to a flash memory device changes in accordance with gathered situational information such as but not limited to a fluctuating state of the flash memory, whether at the cell level, page level, erase sector level or device level. The manner in which a sequence is programmed may or may not additionally depend on the data content of that sequence.

There is thus provided, in accordance with at least one embodiment of the present invention, a method for programming a plurality of data sequences into a corresponding plurality of flash memory functional units using a programming process having at least one selectable programming duration-controlling parameter controlling the duration of the programming process for a given data sequence, the method comprising providing at least one indication of at least one varying situational characteristic and determining a value for the at least one selectable programming duration-controlling parameter controlling the duration of the programming process for a given data sequence, for each flash memory functional unit, depending at least partly on the indication of the varying characteristic; and, for each individual flash memory functional unit from among the plurality of flash memory functional units, programming a sequence of bits into the individual flash memory functional unit using a programming process having at least one selectable parameter, the at least one selectable parameter being set at the value determined for the individual flash memory functional unit.

Further in accordance with at least one embodiment of the present invention, the varying situational characteristic comprises a varying characteristic of each flash memory functional unit from among the plurality of flash memory functional units.

Still further in accordance with at least one embodiment of the present invention, the varying situational characteristic comprises at least one characteristic of an application to which a flash memory functional unit has been assigned.

Additionally in accordance with at least one embodiment of the present invention, the characteristic of the application comprises the duration of time for which information in the flash memory functional unit is to be maintained.

Further in accordance with at least one embodiment of the present invention, the varying characteristic comprises a degradation state.

Still further in accordance with at least one embodiment of the present invention, the at least one indication comprises a cycle count of an individual flash memory functional unit.

Further in accordance with at least one embodiment of the present invention, the programming process comprises generating at least two pulses and wherein the selectable parameter comprises the difference between the voltages of the two pulses.

Still further in accordance with at least one embodiment of the present invention, the flash memory functional unit comprises an entire flash memory device.

Additionally in accordance with at least one embodiment of the present invention, the flash memory functional unit comprises an erase sector.

Also in accordance with at least one embodiment of the present invention, the flash memory functional unit comprises at least one row in a flash memory erase sector.

Also provided, in accordance with certain embodiments of the present invention, is a system for programming a plurality of data sequences using a programming process having at least one selectable programming duration-controlling parameter controlling the duration of the programming process for a given data sequence, the system comprising a plurality of flash memory functional units into which the plurality of data sequences are to be programmed, a situational analyzer operative to provide at least one indication of at least one varying situational characteristic and to determine a value for the at least one selectable programming duration-controlling parameter controlling the duration of the programming process for a given data sequence, for each flash memory functional unit, depending at least partly on the indication of the varying characteristic; and a bit sequence programmer operative, for each individual flash memory functional unit from among the plurality of flash memory functional units, to program a sequence of bits into the individual flash memory functional unit using a programming process having at least one selectable parameter which is set at the value determined for the individual flash memory functional unit.

Further provided, in accordance with certain embodiments of the present invention, is a method for rapidly programming at least two bits per cell, in a population of pages belonging to respective erase sectors in flash memory, the population of pages defining a multiplicity of cells, the method comprising, for at least one cell, initially programming at least one first bit into the cell thereby to induce one of a first plurality of program levels in the cell, the first plurality of program levels including an erase level, and subsequently programming at least one second bit into the cell, thereby to induce one of a second plurality of program levels in the cell, the second plurality exceeding the first plurality in number, wherein the second plurality of program levels includes at least one of the program levels in the first plurality of program levels other than the erase level.

Additionally in accordance with at least one embodiment of the present invention, the second plurality of program levels includes all of the program levels in the first plurality of program levels.

Further in accordance with at least one embodiment of the present invention, the at least one cell comprises only cells in pages belonging to erase sectors having a predeterminedly low cycle count.

Still further in accordance with at least one embodiment of the present invention, the at least one cell comprises all cells in pages belonging to erase sectors having a predeterminedly low cycle count.

Also provided, in accordance with certain embodiments of the present invention, is a system for rapidly programming at least two bits per cell, in a population of pages belonging to respective erase sectors in flash memory, the population of pages defining a multiplicity of cells, the system comprising a bit programmer operative to initially program at least one first bit into at least one cell thereby to induce one of a first plurality of program levels in the cell, the first plurality of program levels including an erase level, and for at least one cell, to subsequently program at least one second bit into the cell, thereby to induce one of a second plurality of program levels in the cell, the second plurality exceeding the first plurality in number, wherein the second plurality of program levels includes at least one of the program levels in the first plurality of program levels other than the erase level.

Additionally provided, in accordance with certain embodiments of the present invention, is a method for programming data into a first plurality of rows within a second plurality of erase sectors of a flash memory device using a programming process having at least one selectable parameter, the method comprising characterizing each of at least one row subsets, each row subset comprising at least one row from among the first plurality of rows, thereby to generate at least one row subset characteristic value and programming data into at least a portion of at least one individual row belonging to at least one row subset, using a programming process having at least one selectable parameter, the at least one selectable parameter being set at least partly in accordance with the row subset characteristic value characterizing a row subset to which the individual row belongs.

For example, there may be 3 row subsets per erase sector, the first subset having just one member: the first row from among the 32 (say) rows in the erase sector, the second subset also having just one member: the last row in the erase sector, and the third subset comprising the remaining rows in the erase sector (30 rows if the erase sector includes 32 rows).

When data is said to be "programmed into rows" and the like, it actually is often programmed into only a portion of a row, such as an even row-half or an odd row-half.

Further in accordance with at least one embodiment of the present invention, the at least one row subset comprises a third plurality of row subsets partitioning the first plurality of rows.

Still further in accordance with at least one embodiment of the present invention, each individual row subset within the third plurality of row subsets comprises a set of rows having an individual row position within erase sectors to which the set of rows respectively belong.

Additionally in accordance with at least one embodiment of the present invention, the at least one selectable parameter comprises at least one program level of at least one cell.

Further in accordance with at least one embodiment of the present invention, the at least one selectable parameter comprises an Incremental Step pulse Programming step voltage.

Still further in accordance with at least one embodiment of the present invention, the row subset characteristic value is indicative of quality of storage provided by at least one row in the subset.

Additionally in accordance with at least one embodiment of the present invention, if the row subset characteristic value indicates that a row's quality of storage is low, a low Incremental Step pulse Programming programming pulse is used and if the row characteristic value indicates that a row's quality of storage is high, a high Incremental Step pulse Programming programming pulse is used.

Also provided, in accordance with certain embodiments of the present invention, is a system for programming data into a first plurality of rows within a second plurality of erase sectors of a flash memory device using a programming process having at least one selectable parameter, the system comprising a row subset analyzer operative to characterize each of at least one row subsets, each row subset comprising at least one row from among the first plurality of rows, thereby to generate at least one row subset characteristic value; and a row subset-dependent programmer operative to program data into at least a portion of at least one individual row belonging to at least one row subset, using a programming process having at least one selectable parameter, the at least one selectable parameter being set at least partly in accordance with the row subset characteristic value characterizing a row subset to which the individual row belongs.

Any suitable processor, display and input means may be used to process, display, store and accept information, including computer programs, in accordance with some or all of the teachings of the present invention, such as but not limited to a conventional personal computer processor, workstation or other programmable device or computer or electronic computing device, either general-purpose or specifically constructed, for processing; a display screen and/or printer and/or speaker for displaying; machine-readable memory such as optical disks, CDROMs, magnetic-optical discs or other discs; RAMs, ROMs, EPROMs, EEPROMs, magnetic or optical or other cards, for storing, and keyboard or mouse for accepting. The term "process" as used above is intended to include any type of computation or manipulation or transformation of data represented as physical, e.g. electronic, phenomena which may occur or reside e.g. within registers and /or memories of a computer.

The above devices may communicate via any conventional wired or wireless digital communication means, e.g. via a wired or cellular telephone network or a computer network such as the Internet.

The apparatus of the present invention may include, according to certain embodiments of the invention, machine readable memory containing or otherwise storing a program of instructions which, when executed by the machine, implements some or all of the apparatus, methods, features and functionalities of the invention shown and described herein. Alternatively or in addition, the apparatus of the present invention may include, according to certain embodiments of the invention, a program as above which may be written in any conventional programming language, and optionally a machine for executing the program such as but not limited to a general purpose computer which may optionally be configured or activated in accordance with the teachings of the present invention.

The embodiments referred to above, and other embodiments, are described in detail in the next section.

Any trademark occurring in the text or drawings is the property of its owner and occurs herein merely to explain or illustrate one example of how an embodiment of the invention may be implemented.

Unless specifically stated otherwise, as apparent from the following discussions, it is appreciated that throughout the specification discussions, utilizing terms such as, "processing", "computing", "estimating", "selecting", "ranking", "grading", "calculating", "determining", "generating", "reassessing", "classifying", "generating", "producing", "stereo-matching", "registering", "detecting", "associating", "superimposing", "obtaining" or the like, refer to the action and/or processes of a computer or computing system, or processor or similar electronic computing device, that manipulate and/or transform data represented as physical, such as electronic, quantities within the computing system's registers and/or memories, into other data similarly represented as physical quantities within the computing system's memories, registers or other such information storage, transmission or display devices.

BRIEF DESCRIPTION OF THE DRAWINGS

Certain embodiments of the present invention are illustrated in the following drawings:

Fig. 1 is a prior art graph of a theoretical probability density function of program levels as a function of a flash memory cell's voltage;

Fig. 2 is a prior art diagram of a procedure for writing program levels in stages, where in each stage only a single bit is programmed, the procedure being suitable for a flash memory device storing 3 bits per cell;

Fig. 3 is a simplified flowchart illustration of an adaptive programming method operative in accordance with certain embodiments of the present invention;

Fig. 4 is a simplified flowchart illustration of a method for performing the page-holding row number computation step of Fig. 3, the method being operative in accordance with certain embodiments of the present invention;

Fig. 5 is a page-to-row table used by the method of Fig. 4 in accordance with certain embodiments of the present invention;

Fig. 6 is a simplified flowchart illustration of a method for performing the cycle count index computing step of Fig. 3, the method being operative in accordance with certain embodiments of the present invention;

Fig. 7 is a simplified flowchart illustration of a method for performing the program level and Incremental Step pulse Programming value setting step of Fig. 3, the method being operative in accordance with certain embodiments of the present invention;

Fig. 8 is a simplified flowchart illustration of a method for performing the page type determination step of Fig. 3, the method being operative in accordance with certain embodiments of the present invention;

Fig. 9 is a page-to-type table useful in accordance with certain embodiments of the present invention;

Fig. 10 is a simplified flowchart illustration of an MSB programming method useful in implementing certain embodiments of the present invention;

Fig. 11 is a simplified flowchart illustration of a method for performing the CSB programming step of Fig. 3, the method being operative in accordance with certain embodiments of the present invention;

Fig. 12 is a simplified flowchart illustration of a cycle-count dependent CSB programming method suitable for cells with a high cycle count, the method being operative in accordance with certain embodiments of the present invention;

Fig. 13 is a simplified flowchart illustration of a cycle-count depending CSB programming method suitable for cells with a low cycle count, the method being operative in accordance with certain embodiments of the present invention;

Fig. 14 is a simplified flowchart illustration of a method for performing the LSB programming step of Fig. 3, the method being operative in accordance with certain embodiments of the present invention.

Fig. 15 is a simplified flowchart illustration of a cycle-count dependent LSB programming method suitable for cells with a low cycle count, the method being operative in accordance with certain embodiments of the present invention;

Fig. 16 is a simplified flowchart illustration of a cycle-count dependent LSB programming method suitable for cells with a high cycle count, the method being operative in accordance with certain embodiments of the present invention;

Fig. 17 is a simplified flowchart illustration of a cell programming method useful in implementing each of the cell programming operations illustrated herein e.g. in Figs. 10, 12 – 13, 15 - 16, all in accordance with certain embodiments of the present invention;

Fig. 18 is a table holding adaptive programming parameters for even pages which is useful in accordance with certain embodiments of the present invention;

Fig. 19 is a table holding adaptive programming parameters for odd pages which is useful in accordance with certain embodiments of the present invention;

Fig. 20 is a simplified functional block diagram of an adaptively programming controller which programs Incremental Step pulse Programming step voltage and/or performs skipped and/or row-dependent programming and which is constructed and operative in accordance with certain embodiments of the present invention; and

Fig. 21 is a simplified functional block diagram of flash memory apparatus incorporating the controller of Fig. 20 which is constructed and operative in accordance with certain embodiments of the present invention.

DETAILED DESCRIPTION OF CERTAIN EMBODIMENTS

Conventional Flash memory devices store information as charge in "cells", each made of either a floating gate transistor or an NROM transistor. In single-level cell (SLC) devices, each cell stores only one bit of information. Multi-level cell (MLC) devices can store more than one bit per cell by choosing between multiple levels of electrical charge to apply to the floating gates of their cells. The amount of charge (also known as charge level) is then measured by a detector, by comparing the voltage of the transistor gate (also known as charge level and denoted V_T) to a decision threshold voltage (also known as charge level boundary point and denoted V_D). The amount of charge is then used to determine the programmed level (logical value) of the cell. Due to inaccuracies during the programming procedure and charge loss due to time and temperature (also known as retention), the measured levels suffer from a random distortion.

Fig. 1 illustrates an example of the eight (8) separate probability distributions of a cell which can be programmed with one of eight (8) corresponding program levels (111 , 110 , 100 , 101 , 001 , 000 , 010 , and 011, respectively). For each distribution curve, the Y-axis represents the probability that the cell is programmed to the corresponding level, given the value of the charge level V_T (represented by the x-axis).

The cell's programmed level may be determined using several methods. One method is to apply a voltage to the cell's gate and measure if the cell conducts current. The cell has a certain threshold voltage such that if voltage above that threshold is applied to the gate, the gate will conduct. Below that threshold voltage the gate does not conduct current, or conducts a small amount of current, below a certain demarcation level. As the amount of charge in the cell changes this threshold voltage, the charge may be inferred by determining at which voltage the cell starts to conduct current. Thus, the programmed level is determined by iteratively applying different voltages to the gate and measuring whether the cells conduct or not. Another method is based on the fact that when applying a voltage above the threshold voltage, the cell conducts current and the amount of current depends on the difference between the applied voltage and the threshold voltage. As the threshold voltage changes as a function of the amount of charge in the cell, the programmed level may be inferred by measuring the current going through the cell.

A programmed level may therefore be obtained by simultaneously comparing the conducted current with a given set of fixed currents distinguishing between all programmed levels. In other words, each cell's programmed level may be determined by simultaneously comparing the V_T level against several decision threshold levels (detection thresholds). For example, if there are eight (8) possible programmed levels, the cell's V_T is simultaneously compared against seven decision threshold levels which divide the voltage axis into eight (8) regions, as demonstrated in Fig. 1.

As aforesaid, in MLC flash devices, each cell can store more than one bit per cell. The program level of each cell is determined by an n-tuple of bits (e.g., n=1, 2, 3). However, this does not mean that the cell is directly programmed to one of its possible program levels in a single step. In state-of-the-art flash devices, the program levels may be written in stages, where in each stage only a single bit is programmed. This procedure is depicted in prior art Fig. 2 for a flash memory device storing 3 bits per cell.

Programming of the most significant bit (MSB) to a cell is effected using a single program level (in addition to the erase level). If the MSB of the cell equals 1, no programming is done and the cell remains in the erase state. Otherwise, if the MSB of the cell equals 0, the cell is programmed to a positive program level, namely $P_{1,1}$. The programming procedure for the MSB is depicted in Fig. 10.

When the bit of centered significance (CSB) is programmed, then if the MSB of the same cell equals 1, the cell is left in the erase state if the value of the CSB equals 1, and programmed to program level $P_{2,1}$, if the value of the CSB equals 0. If on the other hand, the value of the MSB of the same cell equals 0, then if the value of the CSB equals 0, the cell is programmed to program level $P_{2,2}$, and if the value of the CSB equals 1, the cell is programmed to program level $P_{2,3}$. It follows that a cell whose MSB and CSB were programmed has one of four (4) program levels (including the erase state). The programming procedure for the CSB is depicted in Fig. 13.

When the least significant bit (LSB) is programmed, then if the cell is in the erased state (corresponding to the MSB and CSB being equal to 1 and 1, respectively) then if the value of the LSB equals 1, the cell is left in the erase state, and if the value of the LSB equals 0, the cell is programmed to program level $P_{3,1}$. If the cell is in program level $P_{2,1}$ (corresponding to the MSB and CSB being equal to 1 and 0, respectively) then if the value of the LSB equals 0, the cell is programmed to program level $P_{3,2}$, and if the value of the LSB equals 1, the cell is programmed to program level $P_{3,3}$. If the cell is in program level $P_{2,2}$ (corresponding to the MSB and CSB being equal to 0 and 0, respectively) then if the value of the LSB equals 1, the cell is programmed to program level $P_{3,4}$, and if the value of the LSB equals 0, the cell is programmed to program level $P_{3,5}$. Finally, If the cell is in program level $P_{2,3}$ (corresponding to the MSB and CSB being equal to 0 and 1, respectively) then if the value of the LSB equals 0, the cell is programmed to program level $P_{3,6}$, and if the value of the LSB equals 1, the cell is programmed to program level $P_{3,7}$. The programming procedure for the LSB is depicted in Fig. 16.

One advantage to programming the cells in such a way is that this reduces the effects of coupling between adjacent cells in the memory array. If, at each time, only one bit is programmed to a cell, then the changes in threshold voltage occur only within a small voltage window. This reduces the interference which adjacent cells experience due to the programming of their neighboring cells. The procedure outlined above is typically referred to as decoupling.

Typically, programming of a cell to a given threshold voltage (or charge level) involves applying a sequence of program pulses with increasing levels of voltage on the cell's gate - causing the insertion of electric charge into the floating gate. After each pulse is applied, the threshold level of the cell is measured. If the target level has been reached, no further program pulses are applied to this cell. If, on the other hand, the target threshold voltage has not been reached, the program pulses are continued.

Such a scheme is known as Incremental Step pulse Programming (ISPP) and is described e.g. in the above-referenced Brewer and Gill publication. In this scheme, after an initial (e.g., 15 V) program pulse, each subsequent pulse, if any, is incremented in (e.g., ~0.5 V) steps up to a higher voltage (e.g., 20 V), Since sufficiently programmed cells are automatically disconnected from the voltage sequence in the verification step, easily programmed cells are not affected by the higher program level. The Incremental Step pulse Programming scheme overcomes the problem of different programming times being employed for different cells without causing easy-to-program cells to be over-programmed. A simplified illustration of an Incremental Step pulse Programming method appears in Fig. 17.

The Incremental Step pulse Programming step parameter influences both the programming speed and the narrowness of the program level distribution. A higher value of the Incremental Step pulse Programming step enables the cells to reach their target threshold voltage or charge level after a smaller number of programming pulses, hence the programming speed increases. However, the larger step will also cause some cells to be programmed further away from the target program level, thus resulting in a wider distribution of the cell's threshold voltages. Conversely, a lower Incremental Step pulse Programming step voltage achieves a tighter Vt distribution while lengthening the program time.

In state-of-the-art flash devices, different Incremental Step pulse Programming steps may be used to program the MSB, CSB, or LSB. When programming the MSB, since only two program levels are present, a larger standard deviation (STD) can be tolerated, and so a larger Incremental Step pulse Programming step voltage may be used. When programming the CSB, slightly smaller Incremental Step pulse Programming voltages may be employed to keep the standard deviation at a level which is acceptable for the error correcting code. Finally, programming of the LSB typically employs the smallest Incremental Step pulse Programming step. The relative sizes of the standard deviations of the program level distributions corresponding to MSB, CSB, and LSB are represented in Fig. 2 by their relative thicknesses.

In some state-of-the-art flash devices, some periodic irregularities are noticed with respect to the program level statistical behavior in different rows in an erase sector. Specifically, the mean of the program level distributions in the first and last rows in the erase sector are significantly shifted with respect to the corresponding means of the program level distributions in other rows in the erase sector. The implications of such a phenomenon are that any set of read thresholds which is suited for, say, the first row in an erase sector, might not be suited, e.g. cause too many errors, when employed in the second or third row. Such a situation might put an extra burden on the controller which may have to obtain a new set of read threshold whenever it reaches a row which possesses the above irregularities.

Certain embodiments of the present invention seek to expedite the programming of a flash memory device by employing cycle based adaptive programming.

Certain embodiments of the present invention seek to expedite programming of a flash memory device by employing skipped programming.

Certain embodiments of the present invention seek to compensate for poor quality rows, e.g. by selecting row-appropriate program levels.

Certain embodiments of the present invention seek to expedite programming of MLC flash memory devices. One such embodiment includes adaptively changing the programming parameters as a function of the cycle count of the device. Specifically, devices, or sectors within a device, which have undergone a small number of program/erase cycles and hence are less susceptible to future retention effects, are programmed using high Incremental Step pulse Programming step voltage. As the number of program/erase cycles increases, such that the expected deterioration due to retention increases as well, programming is done more accurately using lower Incremental Step pulse Programming step voltages. Finally, when the cycle count of a device reaches, say, several hundred, the Incremental Step pulse Programming step with smallest voltage is typically used, if it is desired to restrict the expected impact of retention to the minimum required for successful reading. The usage of high-voltage Incremental Step pulse Programming steps for lower cycle count sectors enables the overall average programming time to be decreased.

Another MLC flash programming expediting embodiment includes skipping the programming of certain program levels and using instead the program levels of the previous stage. Specifically, the positive program level of the MSB is chosen as a subset of the program levels of the CSB, and the positive program levels of the CSB are chosen as a subset of the program levels of the LSB. This conserves time associated with the programming whenever the desired program level is just the one in which the cell is already programmed.

Certain embodiments of the present invention seek to improve the read efficiency of the flash device by employing row-based adaptive programming. These embodiments may include adaptively changing the programming parameters as a function of the row number within the erase sector in which the data is written. Specifically, rows within an erase sector which exhibit a shift of the means of their program level distributions with respect to other rows within the erase sector are programmed with different program levels than the other rows in the erase sector. The row based adaptive programming causes the statistical behavior of the program levels of all the rows in the erase sector to appear the same when read is carried out on this erase sector. The applicability of row-based adaptive programming is not limited to situations in which certain rows have inherently poor quality and instead, is more generally applicable to any situation in certain rows within one erase sector, which behave very non-uniformly when they are read.

It is appreciated that conflicting considerations may optionally cause Incremental Step pulse Programming steps with larger voltages to be used, at times, even when the cycle count is large. Conversely, conflicting considerations may optionally cause Incremental Step pulse Programming steps with smaller voltages to be used, at times, even when the cycle count is still small. The invention shown and described herein is intended to include any applications in which the cycle count is one of the considerations, although perhaps not the sole consideration, governing the voltage level of programming pulses.

Consider, as an example, a baseline system of a flash memory device where each cell has 8 possible charge levels (or program levels), thus storing 3 bits per cell, and where each erase sector contains 32 rows. That said, this invention is also applicable to flash devices with less or more charge levels and / or less or more rows per erase sector. Each of the 3 bits may be programmed separately using different programming pulses and different Incremental Step pulse Programming step voltages.

Reference is now made to Figs. 3 – 21 which illustrate certain embodiments of the present invention. Specifically, Fig. 3 is a simplified flowchart illustration of a adaptive programming method operative in accordance with certain embodiments of the present invention. Fig. 4 is a simplified flowchart illustration of a method for performing the page-holding row number computation step 602 of Fig. 3, the method being operative in accordance with certain embodiments of the present invention.

Fig. 5 is a page-to-row table used by the method of Fig. 4 in accordance with certain embodiments of the present invention.

Fig. 6 is a simplified flowchart illustration of a method for performing the cycle count index computing step 605 of Fig. 3, the method being operative in accordance with certain embodiments of the present invention. Fig. 7 is a simplified flowchart illustration of a method for performing the program level and Incremental Step pulse Programming value setting step 610 of Fig. 3, the method being operative in accordance with certain embodiments of the present invention. Fig. 8 is a simplified flowchart illustration of a method for performing the page type determination step 615 of Fig. 3, the method being operative in accordance with certain embodiments of the present invention. An example of the page-to-type lookup table used by step 1210 in Fig. 8 is shown in Fig. 9. Fig. 10 is a simplified flowchart illustration of a method for performing the MSB programming step 630 of Fig. 3.

Reference is now made to Fig. 11 which is a simplified flowchart illustration of a method for performing the CSB programming step 635 of Fig. 3. Fig. 12 is a simplified flowchart illustration of a cycle-count dependent CSB programming method suitable for cells with a high cycle count, the method being operative in accordance with certain embodiments of the present invention to implement step 1380 of Fig. 11. Fig. 13 is a simplified flowchart illustration of a cycle-count depending CSB programming method suitable for cells with a low cycle count, the method being operative in accordance with certain embodiments of the present invention to implement step 1370 of Fig. 11.

Fig. 14 is a simplified flowchart illustration of a method for performing the LSB programming step 640 of Fig. 3.

Fig. 15 is a simplified flowchart illustration of a cycle-count dependent LSB programming method suitable for cells with a low cycle count, the method being operative in accordance with certain embodiments of the present invention to implement step 1784 of Fig. 14.

Fig. 16 is a simplified flowchart illustration of a cycle-count dependent LSB programming method suitable for cells with a high cycle count, the method being operative in accordance with certain embodiments of the present invention to implement step 1786 of Fig. 14.

Fig. 17 is a simplified flowchart illustration of a cell programming method useful in implementing each of the cell programming operations illustrated herein e.g. operations 1320, 1450, 1460, 1470, 1750, 1760, 1850, 1855, 1860, 1865, 2535, 2540, 2545, 2565, 2570, 2575 and 2580 in Figs. 10, 12 – 13, 15 - 16, which differ as to the program level to which the cell is to be programmed, all in accordance with certain embodiments of the present invention.

Fig. 3 is now described in detail. In certain embodiments of this invention, the controller receives a page to be written to the flash, and a page address (or number). The address also indicates to which erase sector the page belongs. The controller is operative to keep track of the number of program/erase cycles each erase sector has undergone. In step 602 the controller computes the row number in which the page resides, e.g. using the method in Fig. 4. In step 605 the controller computes the cycle count index of the erase block to which the page will be written. In step 610, the controller sets the values for the programming parameters based on the row number or the cycle count or both. In step 615, the controller selects either an MSB programming process, or a CSB programming process, or an LSB programming process. Programming is then performed using the parameters which were selected as detailed above.

A method for cycle count dependent adaptive programming operative in accordance with certain embodiments of the present invention is now described. It is known, e.g. as described in the above-referenced Brewer and Gill publication, that the degradation of a flash device, in terms of the increase in the standard deviation level of the program level distribution due to retention, is more severe when the device is heavily cycled prior to programming. Devices which are only lightly cycled suffer less from retention. It follows that Incremental Step pulse Programming step voltages which are generally used for programming MSBs, can be used to program CSBs providing the cycle count of the flash device is relatively low. This is because, although the usage of a larger Incremental Step pulse Programming step will increase the standard deviation, the expected degradation due to retention is expected to be limited, and the error correcting code will be able to correct all the bit errors. Similarly, Incremental Step pulse Programming step voltages which were used to program CSBs can be used to program LSBs when the flash devices have undergone few, if any, program/erase cycles.

Fig. 18 and Fig. 19 contain programming parameters which are used, according to certain embodiments of the present invention, to program even pages and odd pages. Each row in the tables corresponds to a cycle count range. For instance, the first row whose cycle count index is 0 is used for programming flash devices which have undergone at most 200 program/erase cycles. The second row in each table is used for programming flash devices which have undergone between 201 and 500 program/erase cycles. Finally, the last row in each table whose cycle count index is 4 is used for programming flash devices which have undergone more than 2000 program/erase cycles.

In the tables in Fig. 18 and Fig. 19, the column Incremental Step pulse Programming corresponds to the Incremental Step pulse Programming step voltage used for programming the MSB, CSB, and LSB, respectively. In a certain embodiment of this invention, the tables in Fig. 18 and Fig. 19 are used to determine which Incremental Step pulse Programming step voltages are used. For instance, it can be seen that for devices with low cycle count, the Incremental Step pulse Programming step used to program LSB is larger for devices with low cycle counts, and conversely, the Incremental Step pulse Programming step used to program LSB is smaller for devices with large cycle count. For example, even LSB pages in row 0 are programmed with Incremental Step pulse Programming step of 0.4 V for flash devices with less than 200 program/erase cycles, while the same pages are programmed using an Incremental Step pulse Programming step of 0.2 V for flash devices which have undergone more than 2000 program/erase cycles.

In certain embodiments of this invention, the program levels are chosen as a function of the cycle count. One advantage of this embodiment is that it enables to save time in programming flash devices which were not cycled heavily, while not affecting the read performance of flash devices which were heavily cycled.

A method for row dependent adaptive programming operative in accordance with certain embodiments of the present invention is now described.

According to certain embodiments of this invention, the programming parameters used to program a page into the flash memory are selected as a function of the row number. The tables in Fig. 18 and Fig. 19 have separate columns for row 0, rows 1 – 30, and row 31. In certain embodiments of this invention, prior to programming a page into the flash memory, the controller checks to which row the page is to be programmed. The controller then uses the appropriate values from the table according to the row number. In certain embodiments of this invention, if the row number is 0, the programming parameters are taken from the column designated as “Row 0”. If the row number is between 1 – 30, the programming parameters are taken from the column designated as “Row 1 – 30”. Finally, for pages written to rows 31, the programming parameters are taken from column “Row 31”. One advantage of this embodiment is in its ability to handle rows with different statistical behavior within a single erase sector in the programming stage, such that during the read operation, the rows appear statistically the same, obviating any need to calibrate read threshold for each row individually.

A method for skipped programming operative in accordance with certain embodiments of the present invention is now described.

In certain embodiments of this invention, the procedure outlined in Fig. 12 for programming the CSB is altered. Specifically, the positive program level of the MSB is chosen as a subset of the program levels of the CSB. Fig. 13 is a simplified flowchart illustration of a method for programming the CSB with skipped programming. As opposed to the implementation of Fig. 12, no programming is required whenever the MSB equals 0 and the CSB equals 0.

The method of Fig. 16 for programming the LSB may be modified: the positive program level of the CSB may be selected as a subset of the program levels of the LSB. Fig. 15 is a simplified flowchart illustration of a method for programming the LSB with skipped programming. As opposed to the implementation depicted in Fig. 16, no programming is required whenever the MSB, CSB, and LSB equal either (1, 0, 0), (0, 0, 1), or (0, 1, 0), respectively.

An advantage of using the skipped programming, according to certain embodiments, is that the total number of programming operations is reduced with respect to the standard implementation,

In certain embodiments of the present invention, the skipped programming is used only if the cycle count of the flash device is lower than some number.

Fig. 11 is a simplified flowchart illustration of a method for performing programming of a CSB. In certain embodiments of this invention, the controller compares the cycle count of the erase sector to which the page is to be programmed. If this cycle count is lower than a threshold value L (e.g. ~100), the controller uses the skipped programming method of Fig. 13. Otherwise, the controller uses the method of Fig. 12.

Fig. 14 is a simplified flowchart illustration of a method for performing programming of an LSB. In certain embodiments of this invention, the controller compares the cycle count of the erase sector to which the page is to be programmed. If this cycle count is lower than a threshold value L (e.g. ~100), the controller uses the skipped programming method of Fig. 15. Otherwise, the controller uses the method of Fig. 16.

One advantage of using the skipped programming when the cycle count of the flash device is low may be that the total number of programming operations may be reduced, relative to conventional implementations, whenever the cycle count of the device is relatively low. When the cycle count is low, some of the program levels distribution lobes of Fig. 1 can be allowed to be "thick", since the expected widening of the distribution due to retention is expected to be limited.

In certain embodiments of the present invention, the thresholds governing the skipped programming usage can be set differently for either the CSB programming or the LSB programming depending on the particular behavior of the NAND flash device as revealed by offline experiments.

Fig. 20 is a simplified functional block diagram of an adaptively programming controller which programs Incremental Step pulse Programming step voltage and/or performs skipped and/or row-dependent programming. Specifically, the controller of Fig. 20 is operative to adaptively program a flash memory device e.g. by programming its Incremental Step pulse Programming step voltage as described above and/or by performing skipped programming as described above and/or by performing row-dependent programming as described above.

Fig. 21 is a simplified functional block diagram of flash memory apparatus incorporating the controller of Fig. 20. As shown, the flash memory system of Fig. 21 includes a host or outside application 3100 which interfaces, via an interface controller 3102, with a flash memory device 3105. An internal microcontroller 3110 typically manages the functional units of the flash memory device 3105. The storage portion of the flash memory device includes one or more typically many erase sectors 3120 each storing one or more typically many physical pages 3130 each including one or more typically many cells 3140 having more than one possible state such that logical values may be stored therein. Erasing circuitry 3150 is provided to erase data from cells, writing circuitry 3160 writes data into cells, and reading circuitry 3170 reads data from cells. Typically, the controller of Fig. 20 is incorporated in the external I/F controller 3102 although alternatively, some or all of the functionality of the controller of Fig. 20 may be incorporated in an internal microcontroller 3110 which is disposed internally of a flash memory device 3105.

When the Incremental Step pulse Programming step voltage programming embodiment shown and described herein is used, the flash memory device 3105 of Fig. 21 may be similar to conventional Incremental Step pulse Programming flash memory devices except that the Incremental Step pulse Programming step voltage is controllable.

When the skipped programming embodiment shown and described herein is used, the flash memory device 3105 may be similar to conventional flash memory devices except that a user can elect to de-activate some of the programming levels provided by the flash memory device.

When the row dependent embodiment shown and described herein is used, the flash memory device 3105 may be similar to conventional flash memory devices except that the programming levels provided by the flash memory device are programmable rather than being pre-set.

It is appreciated that the various embodiments shown and described herein, e.g. the Incremental Step pulse Programming step voltage programming embodiment, the skipped programming embodiment and the row dependent embodiment, can be practiced either in isolation or in any suitable combination.

The adaptively programming controller of Fig. 20 typically comprises a programming parameter computation unit 3020 which receives information which is used to determine programming parameters, such as page numbers and associated block cycle counts and computes programming parameter values accordingly. Programming parameters may be stored in a suitable database 3200. A parameter setting unit 3010 sets internal parameters residing in internal registers of flash memory device 3105, to values generated by the parameter computation unit 3020. A conventional control unit 3230 controls all units.

It is appreciated that software components of the present invention including programs and data may, if desired, be implemented in ROM (read only memory) form including CD-ROMs, EPROMs and EEPROMs, or may be stored in any other suitable computer-readable medium such as but not limited to disks of various kinds, cards of various kinds and RAMs. Components described herein as software may, alternatively, be implemented wholly or partly in hardware, if desired, using conventional techniques.

Included in the scope of the present invention, inter alia, are electromagnetic signals carrying computer-readable instructions for performing any or all of the steps of any of the methods shown and described herein, in any suitable order; machine-readable instructions for performing any or all of the steps of any of the methods shown and described herein, in any suitable order; program storage devices readable by machine, tangibly embodying a program of instructions executable by the machine to perform any or all of the steps of any of the methods shown and described herein, in any suitable order; a computer program product comprising a computer useable medium having computer readable program code having embodied therein, and/or including computer readable program code for performing, any or all of the steps of any of the methods shown and described herein, in any suitable order; any technical effects brought about by any or all of the steps of any of the methods shown and described herein, when performed in any suitable order; any suitable apparatus or device or combination of such, programmed to perform, alone or in combination, any or all of the steps of any of the methods shown and described herein, in any suitable order; information storage devices or physical records, such as disks or hard drives, causing a computer or other device to be configured so as to carry out any or all of the steps of any of the methods shown and described herein, in any suitable order; a program pre-stored e.g. in memory or on an information network such as the Internet, before or after being downloaded, which embodies any or all of the steps of any of the methods shown and described herein, in any suitable order, and the method of uploading or downloading such, and a system including server/s and/or client/s for using such; and hardware which performs any or all of the steps of any of the methods shown and described herein, in any suitable order, either alone or in conjunction with software.

Certain operations are described herein as occurring in the microcontroller internal to a flash memory device. Such description is intended to include operations which may be performed by hardware which may be associated with the microcontroller such as peripheral hardware on a chip on which the microcontroller may reside. It is also appreciated that some or all of these operations, in any embodiment, may alternatively be performed by the external, host-flash memory device interface controller including operations which may be performed by hardware which may be associated with the interface controller such as peripheral hardware on a chip on which the interface controller may reside. Finally it is appreciated that the internal and external controllers may each physically reside on a single hardware device, or alternatively on several operatively associated hardware devices.

Any data described as being stored at a specific location in memory may alternatively be stored elsewhere, in conjunction with an indication of the location in memory with which the data is associated. For example, instead of storing page- or erase-sector-specific information within a specific page or erase sector, the same may be stored within the flash memory device's internal microcontroller or within a microcontroller interfacing between the flash memory device and the host, and an indication may be stored of the specific page or erase sector associated with the cells.

It is appreciated that the teachings of the present invention can, for example, be implemented by suitably modifying, or interfacing externally with, flash controlling apparatus. The flash controlling apparatus controls a flash memory array and may comprise either a controller external to the flash array or a microcontroller on-board the flash array or otherwise incorporated therewithin. Examples of flash memory arrays include Samsung's K9XXG08UXM series, Hynix' HY27UK08BGFM Series, Micron's MT29F64G08TAAWP or other arrays such as but not limited to NOR or phase change memory. Examples of controllers which are external to the flash array they control include STMicroelectrocincs's ST7265x microcontroller family, STMicroelectrocincs's ST72681 microcontroller, and SMSC's USB97C242, Traspan Technologies' TS-4811, Chipsbank CBM2090/CBM1190. Example of commercial IP software for Flash file systems are: Denali's Spectra™ NAND Flash File System, Aarsan's NAND Flash Controller IP Core and Arasan's NAND Flash File System. It is appreciated that the flash controller apparatus need not be NAND-type and can alternatively, for example, be NOR-type or phase change memory-type.

Flash controlling apparatus, whether external or internal to the controlled flash array, typically includes the following components: a Memory Management/File system, a NAND interface (or other flash memory array interface), a Host Interface (USB, SD or other), error correction circuitry (ECC) typically comprising an Encoder and matching decoder, and a control system managing all of the above. The present invention may for example interface with or modify, as per any of the embodiments described herein, one, some or all of the above components.

Features of the present invention which are described in the context of separate embodiments may also be provided in combination in a single embodiment. Conversely, features of the invention, including method steps, which are described for brevity in the context of a single embodiment or in a certain order may be provided separately or in any suitable subcombination or in a different order. "e.g." is used herein in the sense of a specific example which is not intended to be limiting.

CLAIMS

1. A method for programming a plurality of data sequences into a corresponding plurality of flash memory functional units using a programming process having at least one selectable programming duration-controlling parameter controlling the duration of the programming process for a given data sequence, the method comprising:

providing at least one indication of at least one varying situational characteristic and determining a value for said at least one selectable programming duration-controlling parameter controlling the duration of the programming process for a given data sequence, for each flash memory functional unit, depending at least partly on said indication of said varying characteristic; and

for each individual flash memory functional unit from among said plurality of flash memory functional units, programming a sequence of bits into said individual flash memory functional unit using a programming process having at least one selectable parameter, said at least one selectable parameter being set at said value determined for said individual flash memory functional unit.

2. A method according to claim 1 wherein said varying situational characteristic comprises a varying characteristic of each flash memory functional unit from among said plurality of flash memory functional units.

3. A method according to claim 1 wherein said varying situational characteristic comprises at least one characteristic of an application to which a flash memory functional unit has been assigned.

4. A method according to claim 3 wherein said characteristic of said application comprises the duration of time for which information in said flash memory functional unit is to be maintained.

5. A method according to claim 1 wherein said varying characteristic comprises a degradation state.

6. A method according to claim 1 wherein said at least one indication comprises a cycle count of an individual flash memory functional unit.

7. A method according to claim 1 wherein said programming process comprises generating at least two pulses and wherein said selectable parameter comprises the difference between the said two pulses.

8. A method according to claim 1 wherein said flash memory functional unit comprises an entire flash memory device.

9. A method according to claim 1 wherein said flash memory functional unit comprises an erase sector.

10. A method according to claim 1 wherein said flash memory functional unit comprises at least one row in a flash memory erase sector.

11. A system for programming a plurality of data sequences using a programming process having at least one selectable programming duration-controlling parameter controlling the duration of the programming process for a given data sequence, the system comprising:

a plurality of flash memory functional units into which the plurality of data sequences are to be programmed;

a situational analyzer operative to provide at least one indication of at least one varying situational characteristic and to determine a value for said at least one selectable programming duration-controlling parameter controlling the duration of the programming process for a given data sequence, for each flash memory functional unit, depending at least partly on said indication of said varying characteristic; and

a bit sequence programmer operative, for each individual flash memory functional unit from among said plurality of flash memory functional units, to program a sequence of bits into said individual flash memory functional unit using a programming process having at least one selectable parameter which is set at said value determined for said individual flash memory functional unit.

12. A method for rapidly programming at least two bits per cell, in a population of pages belonging to respective erase sectors in flash memory, the population of pages defining a multiplicity of cells, the method comprising:

for at least one cell,

initially programming at least one first bit into the cell thereby to induce one of a first plurality of program levels in the cell, said first plurality of program levels including an erase level, and

subsequently programming at least one second bit into the cell, thereby to induce one of a second plurality of program levels in the cell, said second plurality exceeding said first plurality in number, wherein said second plurality of program levels includes at least one of the program levels in said first plurality of program levels other than the erase level.

13. A method according to claim 12 wherein said second plurality of program levels includes all of the program levels in said first plurality of program levels.

14. A method according to claim 12 wherein said at least one cell comprises only cells in pages belonging to erase sectors having a predeterminedly low cycle count.

15. A method according to claim 12 wherein said at least one cell comprises all cells in pages belonging to erase sectors having a predeterminedly low cycle count.

16. A system for rapidly programming at least two bits per cell, in a population of pages belonging to respective erase sectors in flash memory, the population of pages defining a multiplicity of cells, the system comprising:

a bit programmer operative to initially program at least one first bit into at least one cell thereby to induce one of a first plurality of program levels in the cell, said first plurality of program levels including an erase level, and for at least one cell, to subsequently program at least one second bit into the cell, thereby to induce one of a second plurality of program levels in the cell, said second plurality exceeding said first plurality in number, wherein said second plurality of program levels includes at least one of the program levels in said first plurality of program levels other than the erase level.

17. A method for programming data into a first plurality of rows within a second plurality of erase sectors of a flash memory device using a programming process having at least one selectable parameter, the method comprising:

characterizing each of at least one row subsets, each row subset comprising at least one row from among said first plurality of rows, thereby to generate at least one row subset characteristic value; and

programming data into at least a portion of at least one individual row belonging to at least one row subset, using a programming process having at least one selectable parameter, said at least one selectable parameter being set at least partly in accordance with the row subset characteristic value characterizing a row subset to which said individual row belongs.

18. A method according to claim 17 wherein said at least one row subset comprises a third plurality of row subsets partitioning said first plurality of rows.

19. A method according to claim 18 wherein each individual row subset within said third plurality of row subsets comprises a set of rows having an individual row position within erase sectors to which said set of rows respectively belong.

20. A method according to claim 17, wherein said at least one selectable parameter comprises at least one program level of at least one cell.

21. A method according to claim 17 wherein said at least one selectable parameter comprises an Incremental Step pulse Programming step voltage.

22. A method according to claim 17 wherein said row subset characteristic value is indicative of quality of storage provided by at least one row in said subset.

23. A method according to claim 22 wherein if said row subset characteristic value indicates that a row's quality of storage is low, a low ISPP programming pulse is used and if said row characteristic value indicates that a row's quality of storage is high, a high ISPP programming pulse is used.

24. A system for programming data into a first plurality of rows within a second plurality of erase sectors of a flash memory device using a programming process having at least one selectable parameter, the system comprising:

a row subset analyzer operative to characterize each of at least one row subsets, each row subset comprising at least one row from among said first plurality of rows, thereby to generate at least one row subset characteristic value; and

a row subset-dependent programmer operative to program data into at least a portion of at least one individual row belonging to at least one row subset, using a programming process having at least one selectable parameter, said at least one selectable parameter being set at least partly in accordance with the row subset characteristic value characterizing a row subset to which said individual row belongs.

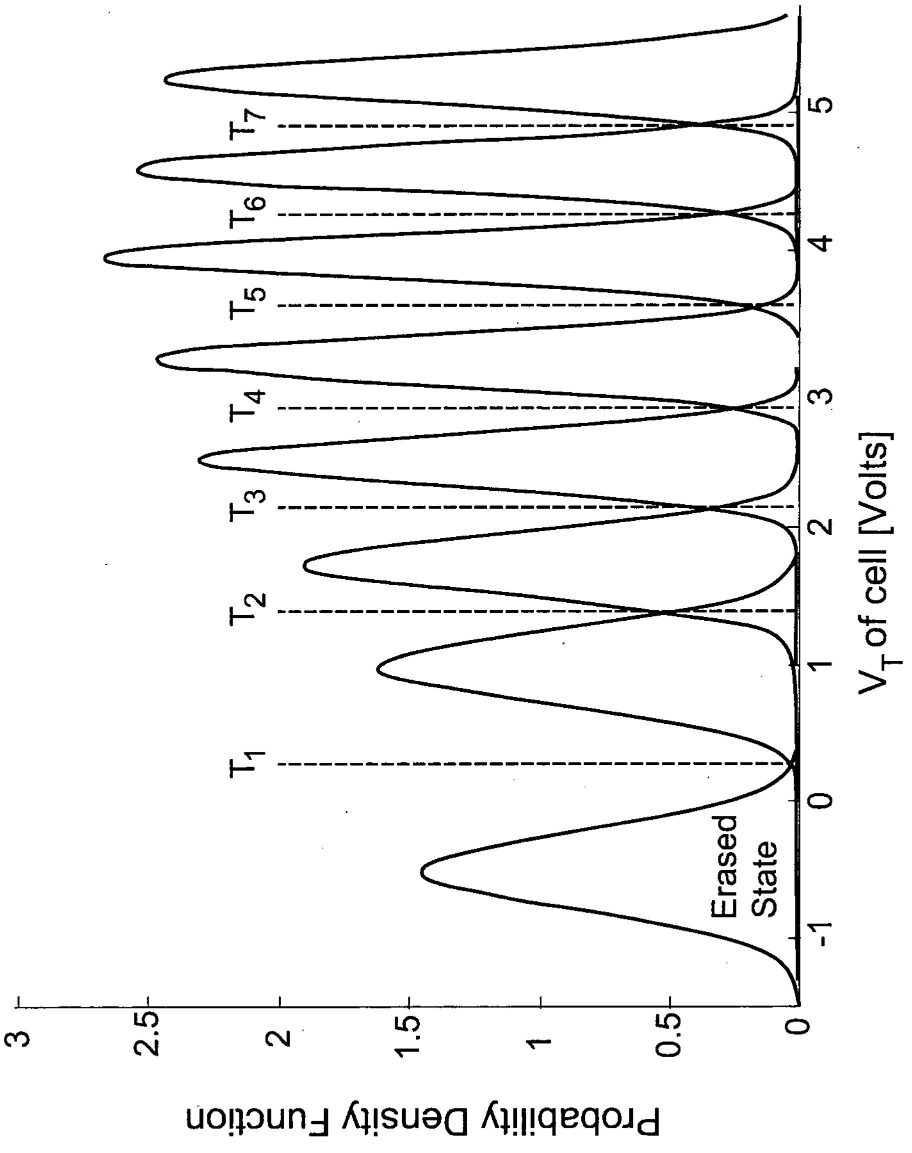
Probability Density Function
0
0.5
1
1.5
2
2.5
3
V_T of cell [Volts]
-1
0
1
2
3
4
5
Erased State
T_1
T_2
T_3
T_4
T_5
T_6
T_7

Fig. 1

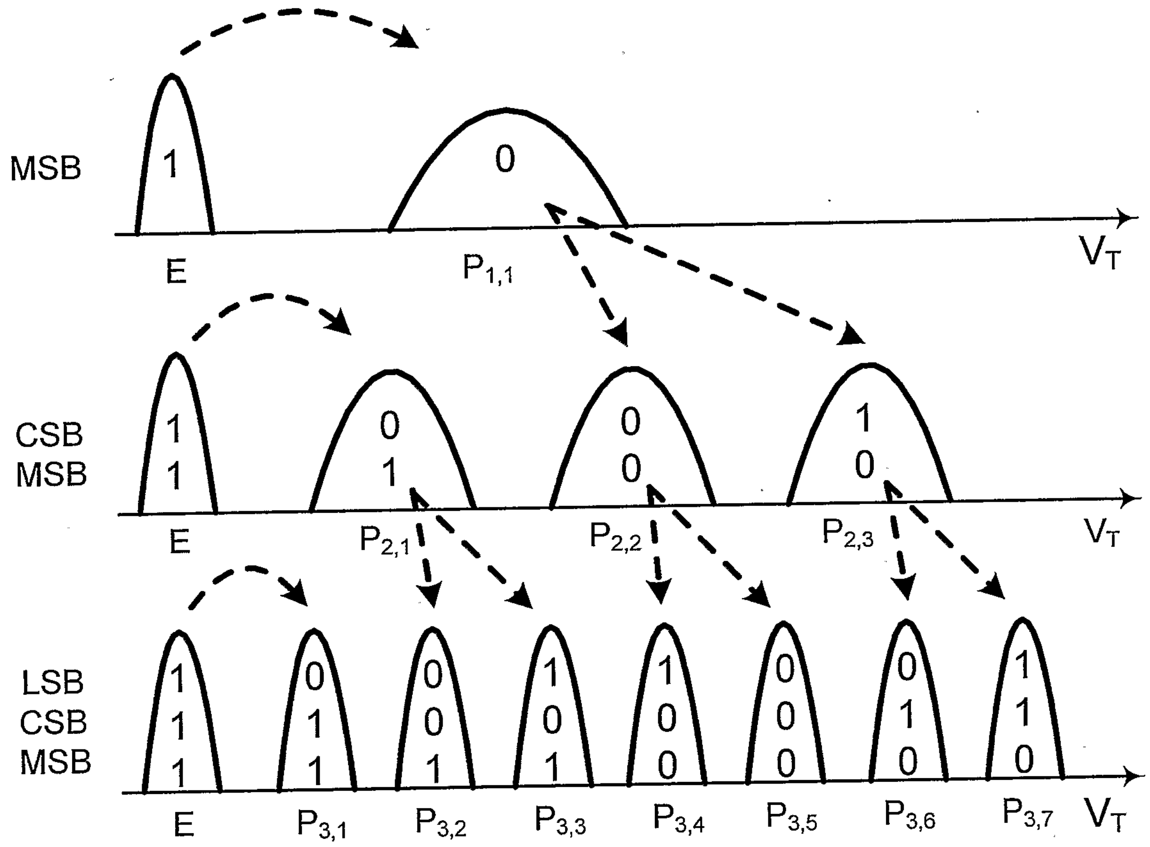
MSB
1
0
E
P_{1,1}
V_T
CSB
MSB
1
1
0
1
0
0
1
0
E
P_{2,1}
P_{2,2}
P_{2,3}
V_T
LSB
CSB
MSB
1 1 1
0 1 1
0 0 1
1 0 1
1 0 0
0 0 0
0 1 0
1 1 0
E
P_{3,1}
P_{3,2}
P_{3,3}
P_{3,4}
P_{3,5}
P_{3,6}
P_{3,7}
V_T

Fig. 2

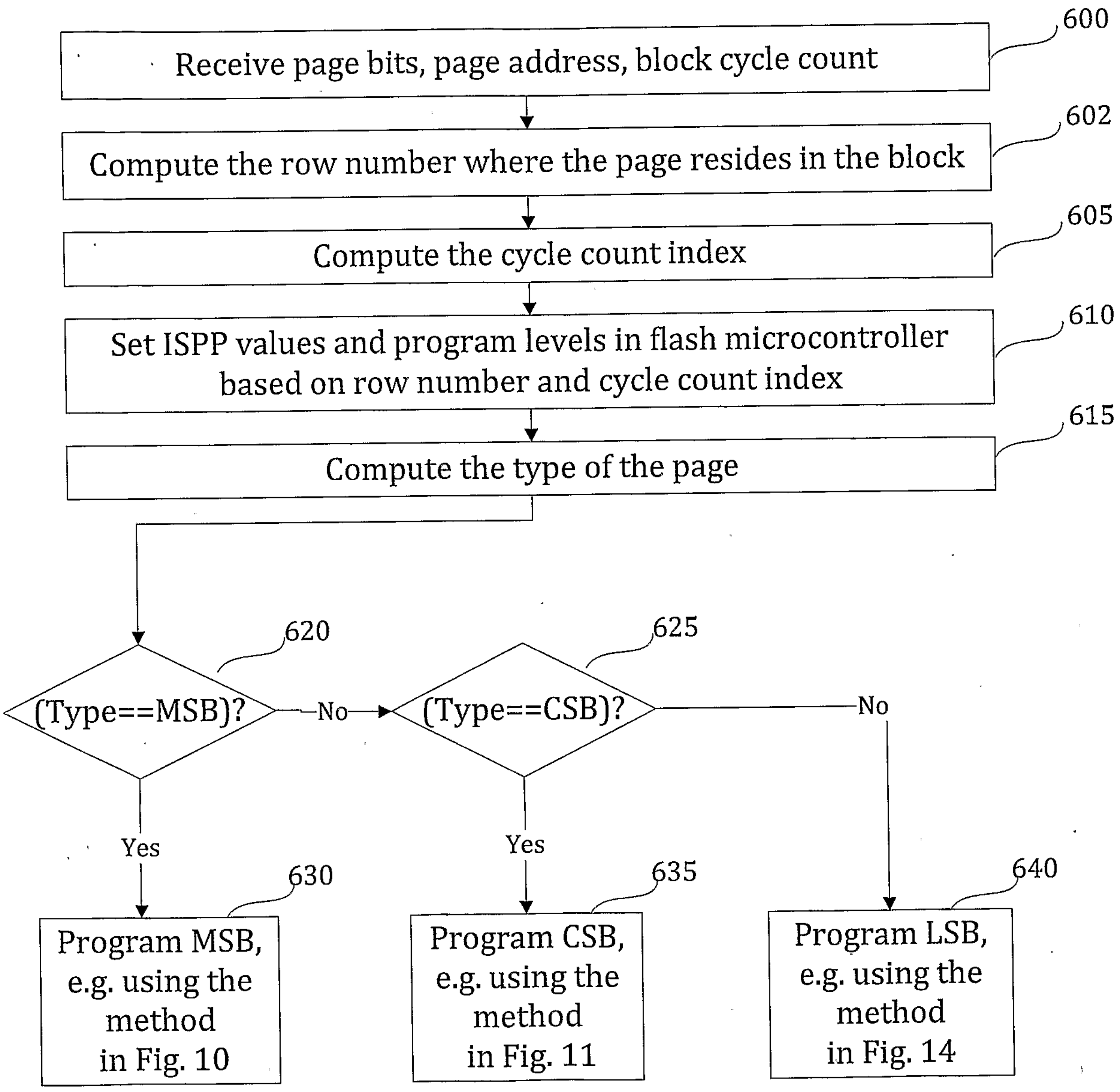
Receive page bits, page address, block cycle count
600
Compute the row number where the page resides in the block
602
Compute the cycle count index
605
Set ISPP values and program levels in flash microcontroller based on row number and cycle count index
610
Compute the type of the page
615
(Type==MSB)?
620
No
(Type==CSB)?
625
No
Yes
Yes
Program MSB, e.g. using the method in Fig. 10
630
Program CSB, e.g. using the method in Fig. 11
635
Program LSB, e.g. using the method in Fig. 14
640

Fig. 3

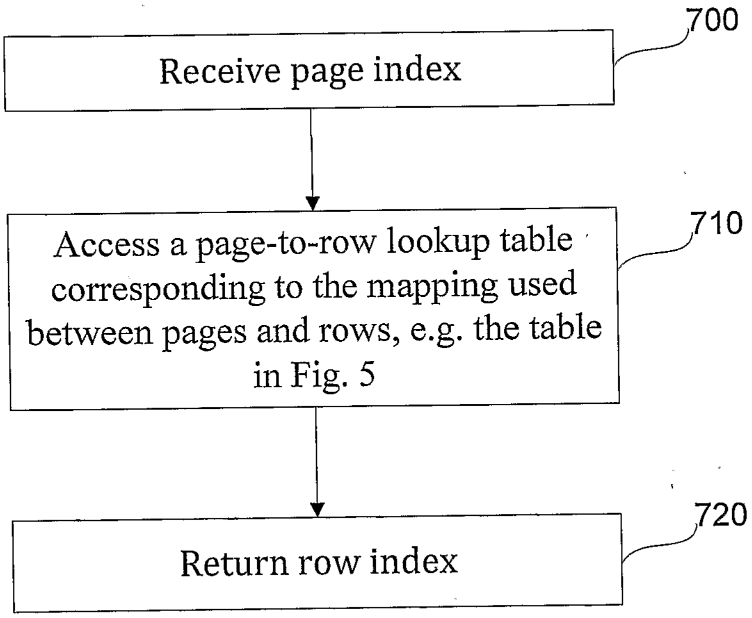
700
Receive page index
710
Access a page-to-row lookup table corresponding to the mapping used between pages and rows, e.g. the table in Fig. 5
720
Return row index

Fig. 4

Page	Row	Page	Row	Page	Row	Page	Row	Page	Row	Page	Row
0	0	32	5	64	10	96	16	128	21	160	26
1	0	33	5	65	10	97	16	129	21	161	26
2	0	34	5	66	11	98	16	130	21	162	27
3	0	35	5	67	11	99	16	131	21	163	27
4	0	36	6	68	11	100	16	132	22	164	27
5	0	37	6	69	11	101	16	133	22	165	27
6	1	38	6	70	11	102	17	134	22	166	27
7	1	39	6	71	11	103	17	135	22	167	27
8	1	40	6	72	12	104	17	136	22	168	28
9	1	41	6	73	12	105	17	137	22	169	28
10	1	42	7	74	12	106	17	138	23	170	28
11	1	43	7	75	12	107	17	139	23	171	28
12	2	44	7	76	12	108	18	140	23	172	28
13	2	45	7	77	12	109	18	141	23	173	28
14	2	46	7	78	13	110	18	142	23	174	29
15	2	47	7	79	13	111	18	143	23	175	29
16	2	48	8	80	13	112	18	144	24	176	29
17	2	49	8	81	13	113	18	145	24	177	29
18	3	50	8	82	13	114	19	146	24	178	29
19	3	51	8	83	13	115	19	147	24	179	29
20	3	52	8	84	14	116	19	148	24	180	30
21	3	53	8	85	14	117	19	149	24	181	30
22	3	54	9	86	14	118	19	150	25	182	30
23	3	55	9	87	14	119	19	151	25	183	30
24	4	56	9	88	14	120	20	152	25	184	30
25	4	57	9	89	14	121	20	153	25	185	30
26	4	58	9	90	15	122	20	154	25	186	31
27	4	59	9	91	15	123	20	155	25	187	31
28	4	60	10	92	15	124	20	156	26	188	31
29	4	61	10	93	15	125	20	157	26	189	31
30	5	62	10	94	15	126	21	158	26	190	31
31	5	63	10	95	15	127	21	159	26	191	31

Fig. 5

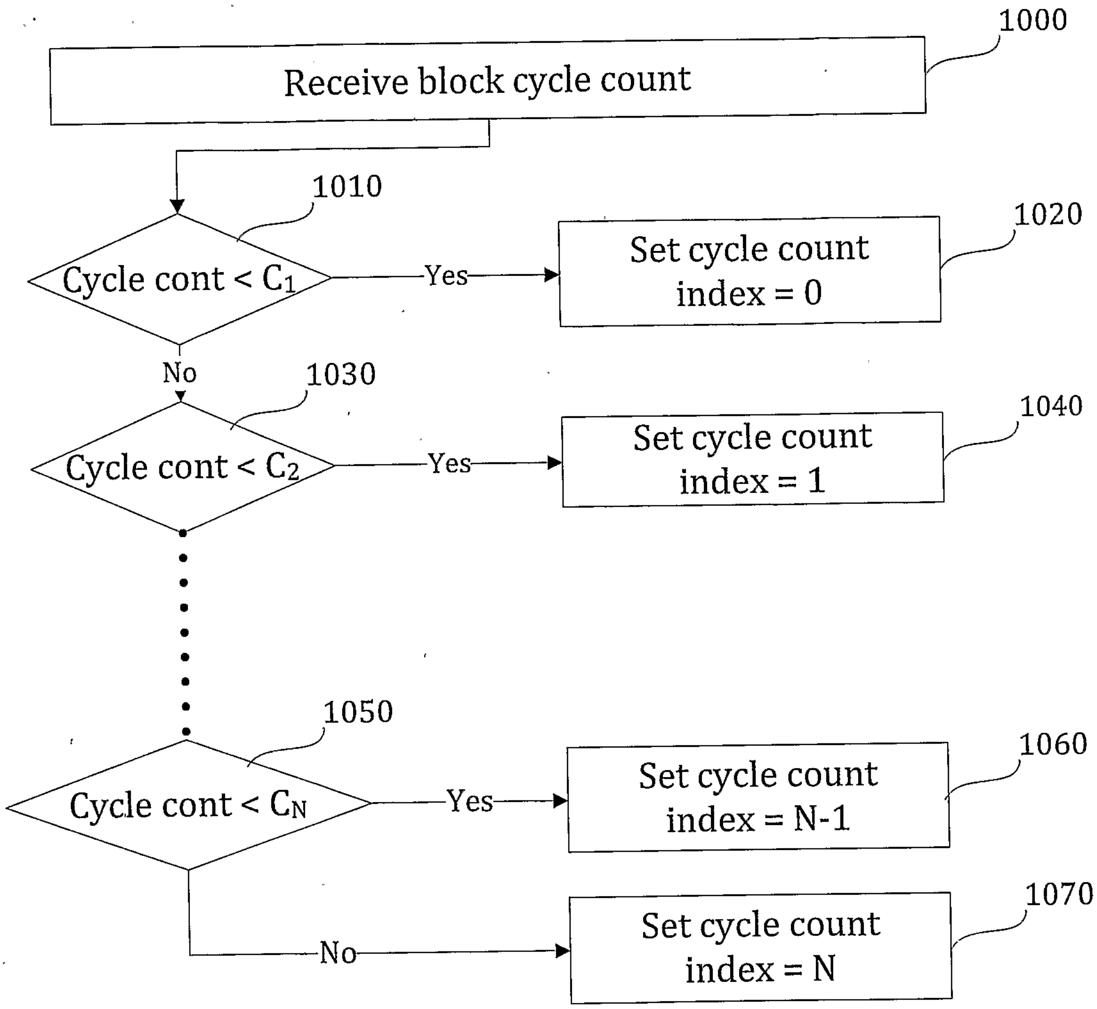
1000
Receive block cycle count
1010
Cycle cont < C1
Yes
1020
Set cycle count index = 0
No
1030
Cycle cont < C2
Yes
1040
Set cycle count index = 1
1050
Cycle cont < CN
Yes
1060
Set cycle count index = N-1
No
1070
Set cycle count index = N

Fig. 6

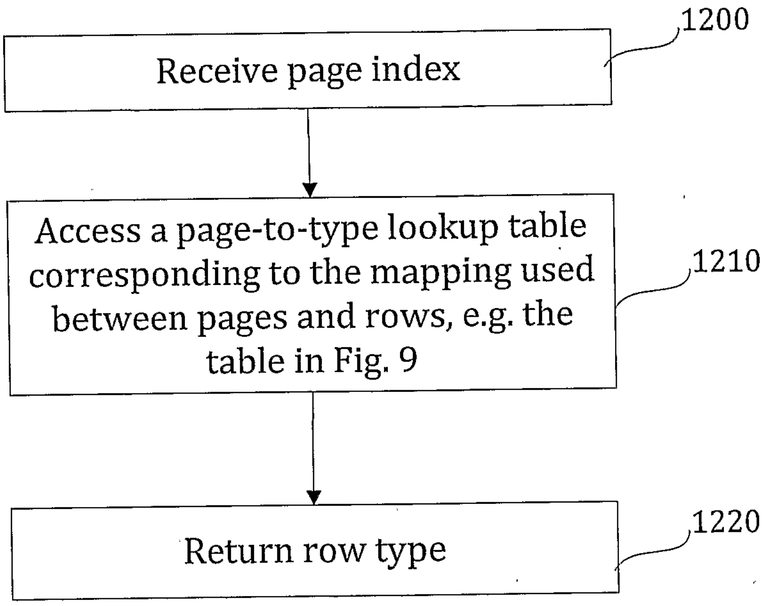
Receive page index
1200
Access a page-to-type lookup table corresponding to the mapping used between pages and rows, e.g. the table in Fig. 9
1210
Return row type
1220

Fig. 8

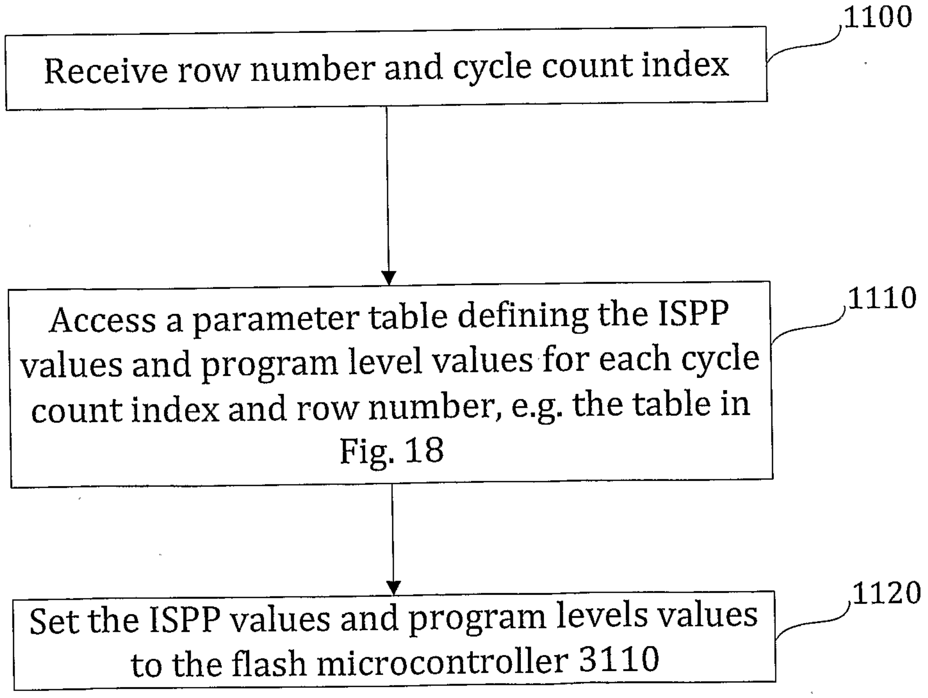
Receive row number and cycle count index
1100
Access a parameter table defining the ISPP values and program level values for each cycle count index and row number, e.g. the table in Fig. 18
1110
Set the ISPP values and program levels values to the flash microcontroller 3110
1120

Fig. 7

	Even Pages			Odd Pages		
Row	**MSB**	**CSB**	**LSB**	**MSB**	**CSB**	**LSB**
0	0	2	4	1	3	5
1	6	8	10	7	9	11
2	12	14	16	13	15	17
3	18	20	22	19	21	23
4	24	26	28	25	27	29
5	30	32	34	31	33	35
6	36	38	40	37	39	41
7	42	44	46	43	45	47
8	48	50	52	49	51	53
9	54	56	58	55	57	59
10	60	62	64	61	63	65
11	66	68	70	67	69	71
12	72	74	76	73	75	77
13	78	80	82	79	81	83
14	84	86	88	85	87	89
15	90	92	94	91	93	95
16	96	98	100	97	99	101
17	102	104	106	103	105	107
18	108	110	112	109	111	113
19	114	116	118	115	117	119
20	120	122	124	121	123	125
21	126	128	130	127	129	131
22	132	134	136	133	135	137
23	138	140	142	139	141	143
24	144	146	148	145	147	149
25	150	152	154	151	153	155
26	156	158	160	157	159	161
27	162	164	166	163	165	167
28	168	170	172	169	171	173
29	174	176	178	175	177	179
30	180	182	184	181	183	185
31	186	188	190	187	189	191

Fig. 9

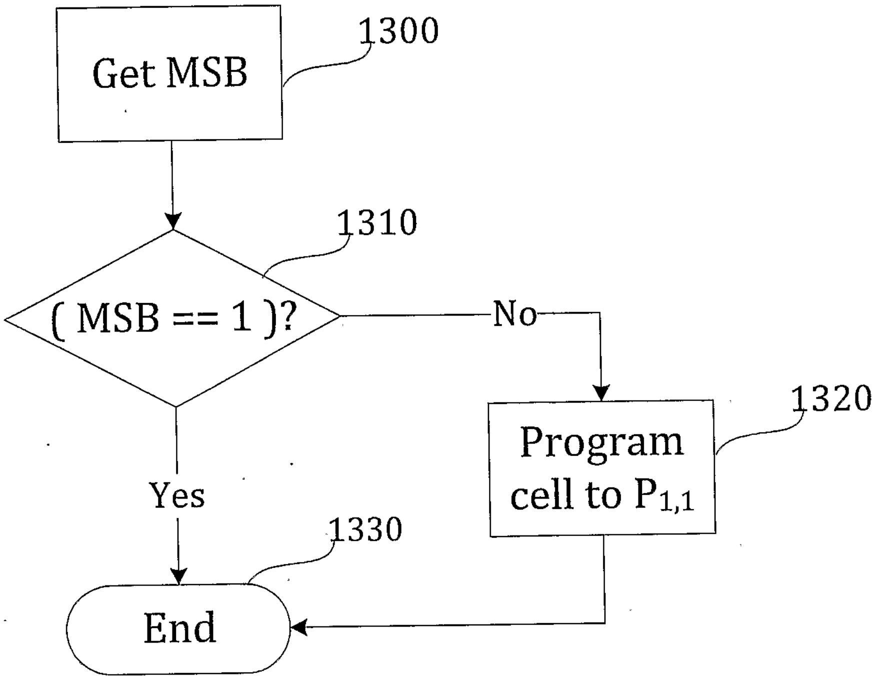
Get MSB
1300
(MSB == 1)?
1310
No
Yes
Program cell to $P_{1,1}$
1320
End
1330

Fig. 10

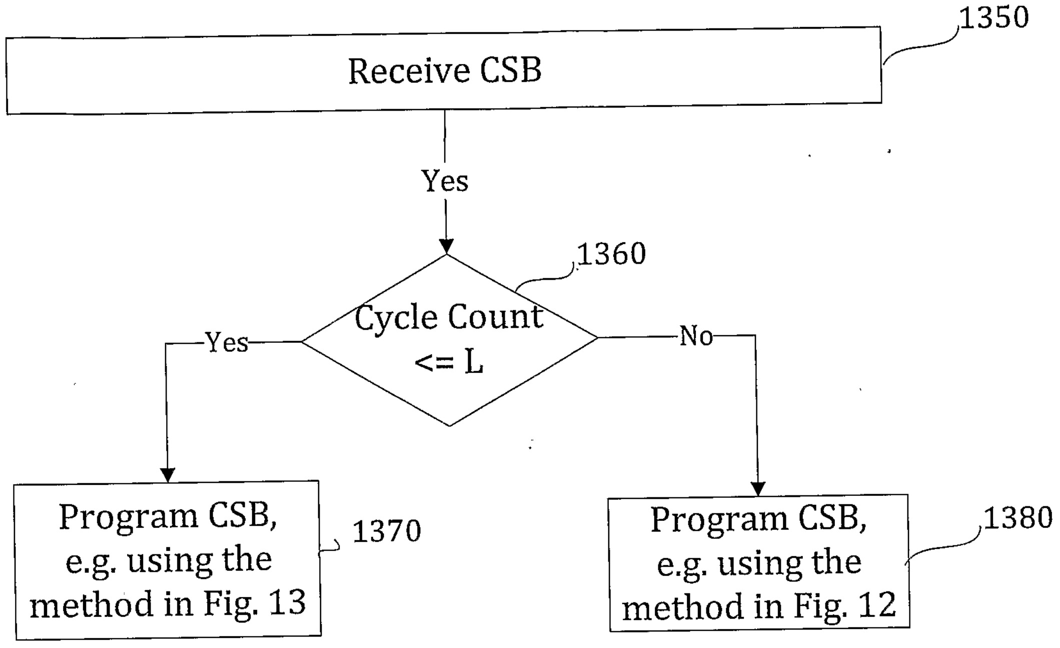
1350
Receive CSB
Yes
1360
Cycle Count <= L
Yes
No
Program CSB, e.g. using the method in Fig. 13
1370
Program CSB, e.g. using the method in Fig. 12
1380

Fig. 11

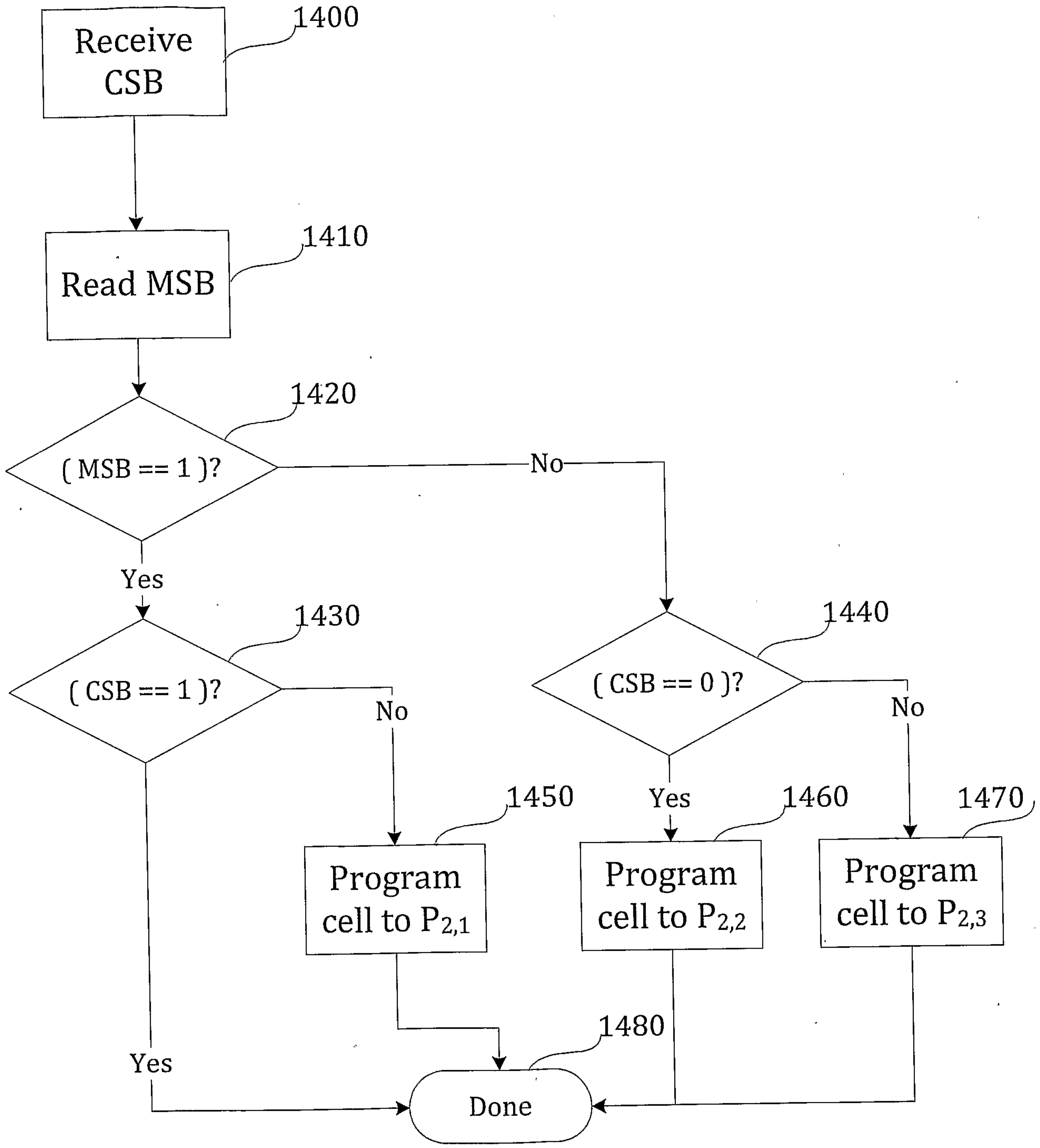
Receive CSB
1400
Read MSB
1410
(MSB == 1)?
1420
No
Yes
(CSB == 1)?
1430
No
(CSB == 0)?
1440
Yes
No
Program cell to $P_{2,1}$
1450
Program cell to $P_{2,2}$
1460
Program cell to $P_{2,3}$
1470
Yes
Done
1480

Fig. 12

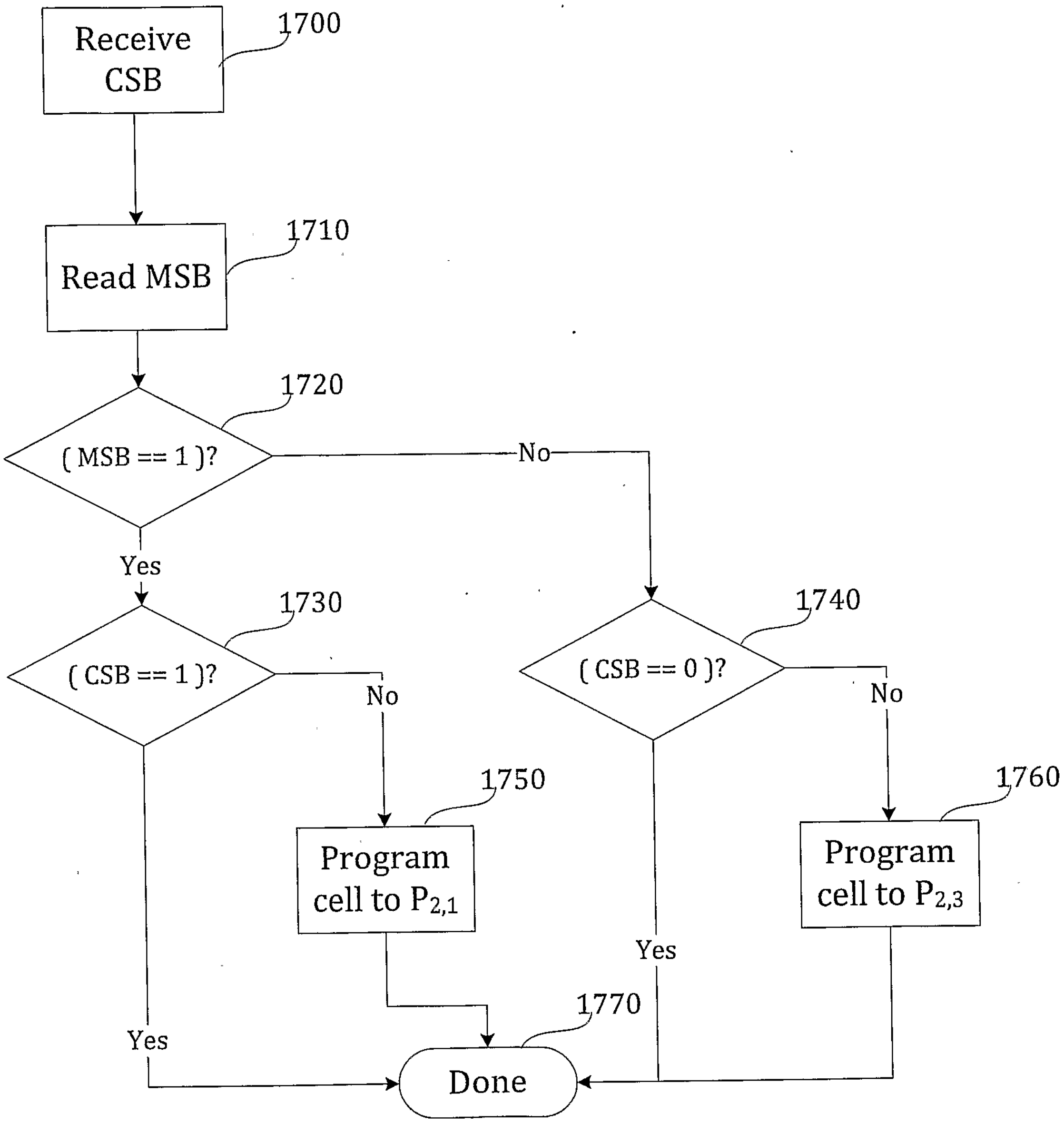
Receive CSB
1700
Read MSB
1710
(MSB == 1)?
1720
No
Yes
(CSB == 1)?
1730
No
(CSB == 0)?
1740
No
Program cell to $P_{2,1}$
1750
Program cell to $P_{2,3}$
1760
Yes
1770
Yes
Done

Fig. 13

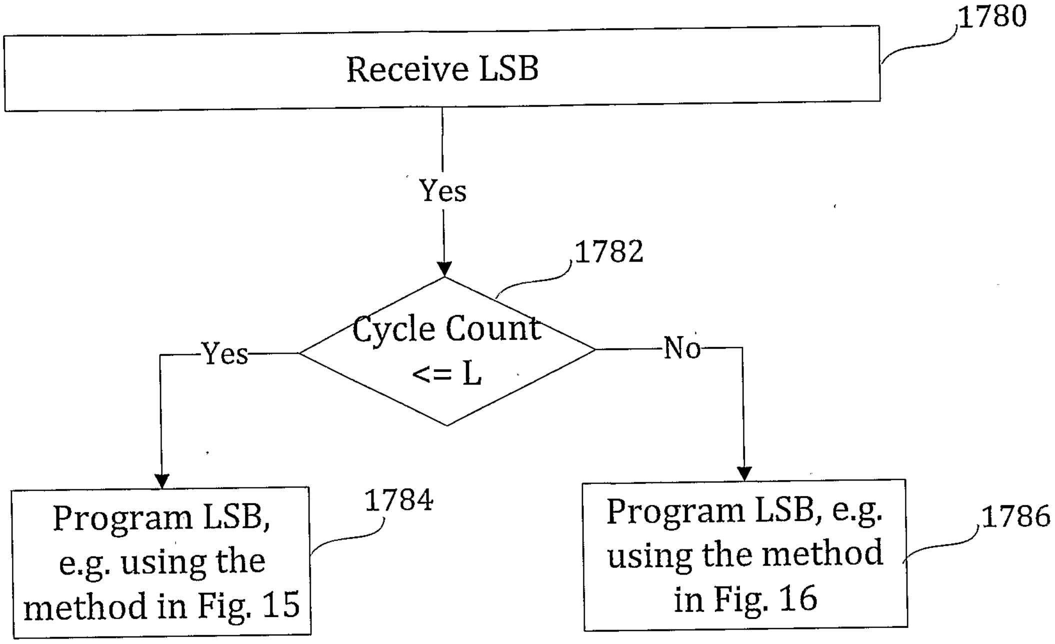
Receive LSB
1780
Yes
1782
Cycle Count <= L
Yes
No
Program LSB, e.g. using the method in Fig. 15
1784
Program LSB, e.g. using the method in Fig. 16
1786

Fig. 14

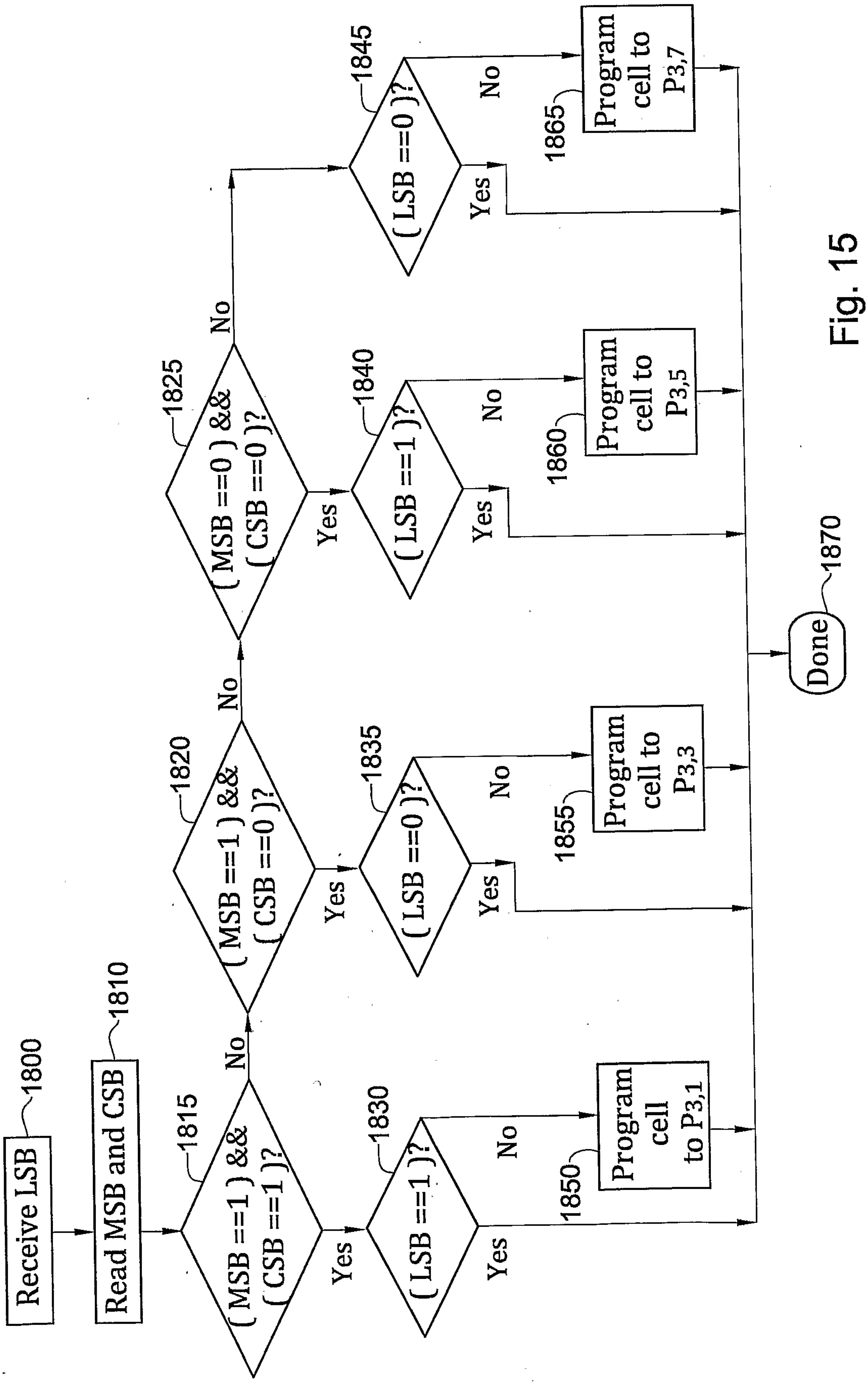
Receive LSB
1800
Read MSB and CSB
1810
(MSB ==1) &&
(CSB ==1)?
1815
Yes
No
(LSB ==1)?
1830
Yes
No
Program
cell
to P3,1
1850
(MSB ==1) &&
(CSB ==0)?
1820
Yes
No
(LSB ==0)?
1835
Yes
No
Program
cell to
P3,3
1855
(MSB ==0) &&
(CSB ==0)?
1825
Yes
No
(LSB ==1)?
1840
Yes
No
Program
cell to
P3,5
1860
(LSB ==0)?
1845
Yes
No
Program
cell to
P3,7
1865
Done
1870

Fig. 15

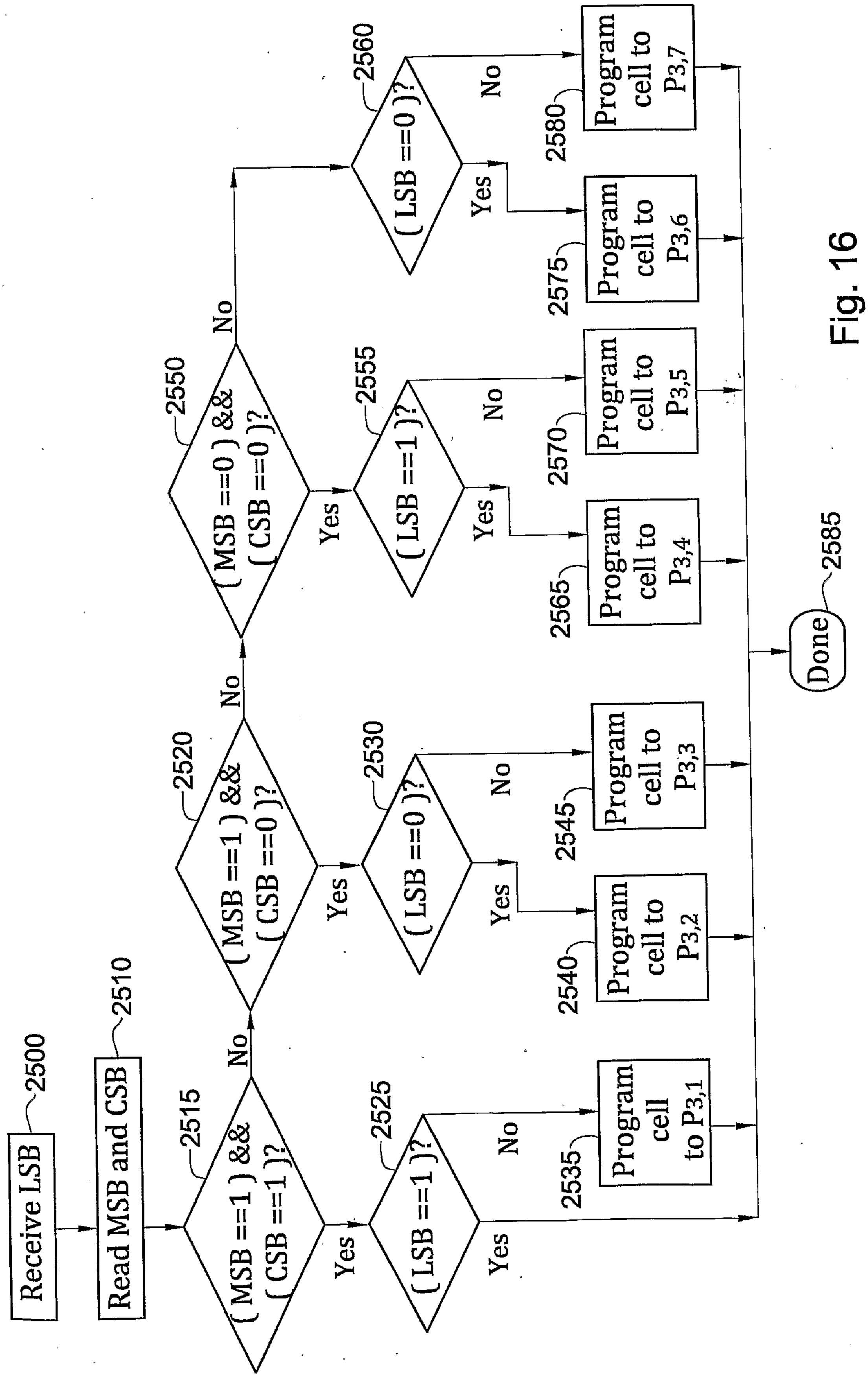
Receive LSB
2500
Read MSB and CSB
2510
(MSB ==1) && (CSB ==1)?
2515
Yes
No
(LSB ==1)?
2525
Yes
No
Program cell to P3,1
2535
(MSB ==1) && (CSB ==0)?
2520
Yes
No
(LSB ==0)?
2530
Yes
No
Program cell to P3,2
2540
Program cell to P3,3
2545
(MSB ==0) && (CSB ==0)?
2550
Yes
No
(LSB ==1)?
2555
Yes
No
Program cell to P3,4
2565
Program cell to P3,5
2570
(LSB ==0)?
2560
Yes
No
Program cell to P3,6
2575
Program cell to P3,7
2580
Done
2585

Fig. 16

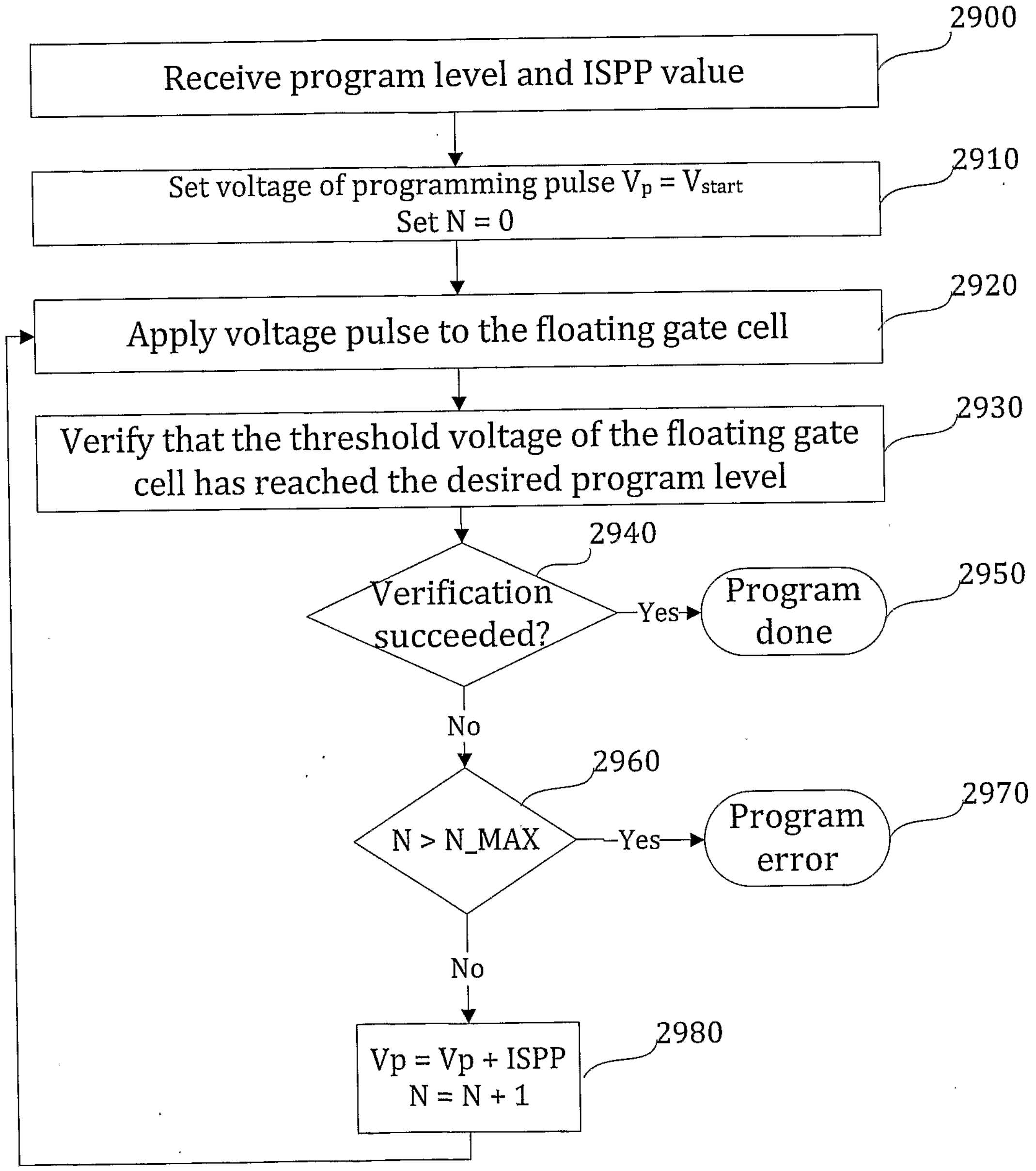
2900
Receive program level and ISPP value
2910
Set voltage of programming pulse $V_p = V_{start}$
Set N = 0
2920
Apply voltage pulse to the floating gate cell
2930
Verify that the threshold voltage of the floating gate cell has reached the desired program level
2940
Verification succeeded?
Yes
Program done
2950
No
2960
N > N_MAX
Yes
Program error
2970
No
Vp = Vp + ISPP
N = N + 1
2980

Fig. 17

Fig. 18

	Cycle count range	Row 0		Rows 1 – 30		Row 31	
Cycle count index	Cycle range	**ISPP { MSB , CSB , LSB**	**Program levels { 3 BPC }**	**ISPP { MSB , CSB , LSB**	**Program levels { 3 BPC }**	**ISPP { MSB , CSB , LSB }**	**Program levels { 3 BPC }**
0	0 – 200	{ 1 , 0.5 , 0.4 }	{ 0.3900 , 1.5210 , 2.3520 , 3.2960 , 4.1890 , 5.1030 6.0920 }	{ 1 , 0.5 , 0.45 }	{ 0.4000 , 1.5530 , 2.3570 , 3.3200 , 4.1730 , 5.1070 , 6.1000 }	{ 1 , 0.5 , 0.4 }	{ 0.4830 , 1.5850 , 2.3990 , 3.3540 , 4.2340 , 5.1570 , 6.1340 }
1	201 – 500	{ 1 , 0.5 , 0.35 }	{ 0.4260 , 1.4420 , 2.3000 , 3.2750 , 4.1380 , 5.0910 , 6.0540 }	{ 1 , 0.5 , 0.35 }	{ 0.4800 , 1.5430 , 2.3470 , 3.2500 , 4.1530 , 5.1070 , 6.1000 }	{ 1 , 0.5 , 0.35 }	{ 0.4940 , 1.5270 , 2.3750 , 3.3280 , 4.1980 , 5.1470 , 6.1040 }
2	501 – 1250	{ 1 , 0.5 , 0.2 }	{ 0.5230 , 1.5640 , 2.4080 , 3.3830 , 4.1800 , 5.1640 , 6.0630 }	{ 1 , 0.5 , 0.2 }	{ 0.5600 , 1.5930 , 2.4470 , 3.4000 , 4.1930 , 5.1570 , 6.1000 }	{ 1 , 0.5 , 0.2 }	{ 0.6010 , 1.6100 , 2.4660 , 3.4330 , 4.2490 , 5.2040 , 6.1250 }
3	1251 – 2000	{ 1 , 0.5 , 0.2 }	{ 0.5110 , 1.5520 , 2.3980 , 3.3880 , 4.1940 , 5.1480 , 6.0730 }	{ 1 , 0.5 , 0.2 }	{ 0.5400 , 1.5930 , 2.4470 , 3.4100 , 4.2130 , 5.1670 , 6.1000 }	{ 1 , 0.5 , 0.2 }	{ 0.5720 , 1.6160 , 2.4730 , 3.4470 , 4.2590 , 5.1950 , 6.1180 }
4	> 2000	{ 1 , 0.5 , 0.2 }	{ 0.6800 , 1.8030 , 2.6670 , 3.6300 , 4.4330 , 5.4670 , 6.3950 }	{ 1 , 0.5 , 0.2 }	{ 0.5700 , 1.6930 , 2.5570 , 3.5200 , 4.3230 , 5.3570 , 6.4400 }	{ 1 , 0.5 , 0.2 }	{ 0.6800 , 1.8030 , 2.6670 , 3.6300 , 4.4330 , 5.4670 , 6.4171 }

	Cycle count range	Row 0		Rows 1 – 30		Row 31	
Cycle count index	Cycle range	**ISPP { MSB , CSB , LSB }**	**Program levels { 3 BPC }**	**ISPP { MSB , CSB , LSB }**	**Program levels { 3 BPC }**	**ISPP { MSB , CSB , LSB }**	**Program levels { 3 BPC }**
0	0 – 200	{ 1 , 0.5 , 0.4 }	{ 0.4600 , 1.5000 , 2.2810 , 3.1730 , 4.0090 , 5.1450 , 6.1220 }	{ 1 , 0.5 , 0.45 }	{ 0.4400 , 1.5330 , 2.2770 , 3.2100 , 3.9830 , 5.0970 , 6.1000 }	{ 1 , 0.5 , 0.4 }	{ 0.5880 , 1.5620 , 2.3340 , 3.2240 , 4.0510 , 5.1950 , 6.1750
1	201 – 500	{ 1 , 0.5 , 0.35 }	{ 0.5330 , 1.4360 , 2.2480 , 3.1840 , 3.9670 , 5.0690 , 6.1100 }	{ 1 , 0.5 , 0.45 }	{ 0.5200 , 1.4930 , 2.2670 , 3.2100 , 3.9530 , 5.0370 , 6.1000 }	{ 1 , 0.5 , 0.35 }	{ 0.6010 , 1.5390 , 2.3190 , 3.2570 , 4.0220 , 5.1120 , 6.1680 }
2	501 – 1250	{ 1 , 0.5 , 0.3 }	{ 0.6070 , 1.5520 , 2.4020 , 3.3610 , 4.1760 , 5.1170 , 6.0690 }	{ 1 , 0.5 , 0.3 }	{ 0.6100 , 1.5830 , 2.4170 , 3.3700 , 4.1830 , 5.1370 , 6.1000 }	{ 1 , 0.5 , 0.3 }	{ 0.6770 , 1.6360 , 2.4780 , 3.4300 , 4.2430 , 5.1870 , 6.1390 }
3	1251 – 2000	{ 1 , 0.5 , 0.2 }	{ 0.6350 , 1.5770 , 2.4060 , 3.3600 , 4.1720 , 5.1480 , 6.0800 }	{ 1 , 0.5 , 0.2 }	{ 0.6400 , 1.5930 , 2.4270 , 3.3700 , 4.1630 , 5.1370 , 6.1000 }	{ 1 , 0.5 , 0.2 }	{ 0.7220 , 1.6600 , 2.4770 , 3.4060 , 4.2390 , 5.1760 , 6.1060 }
4	> 2000	{ 1 , 0.5 , 0.2 }	{ 0.6800 , 1.8030 , 2.6670 , 3.6300 , 4.4330 , 5.4670 , 6.3770 }	{ 1 , 0.5 , 0.2 }	{ 0.6800 , 1.8030 , 2.6670 , 3.6300 , 4.4330 , 5.4670 , 6.4400 }	{ 1 , 0.5 , 0.2 }	{ 0.6800 , 1.8030 , 2.6670 , 3.6300 , 4.4330 , 5.4670 , 6.4380 }

Fig. 19

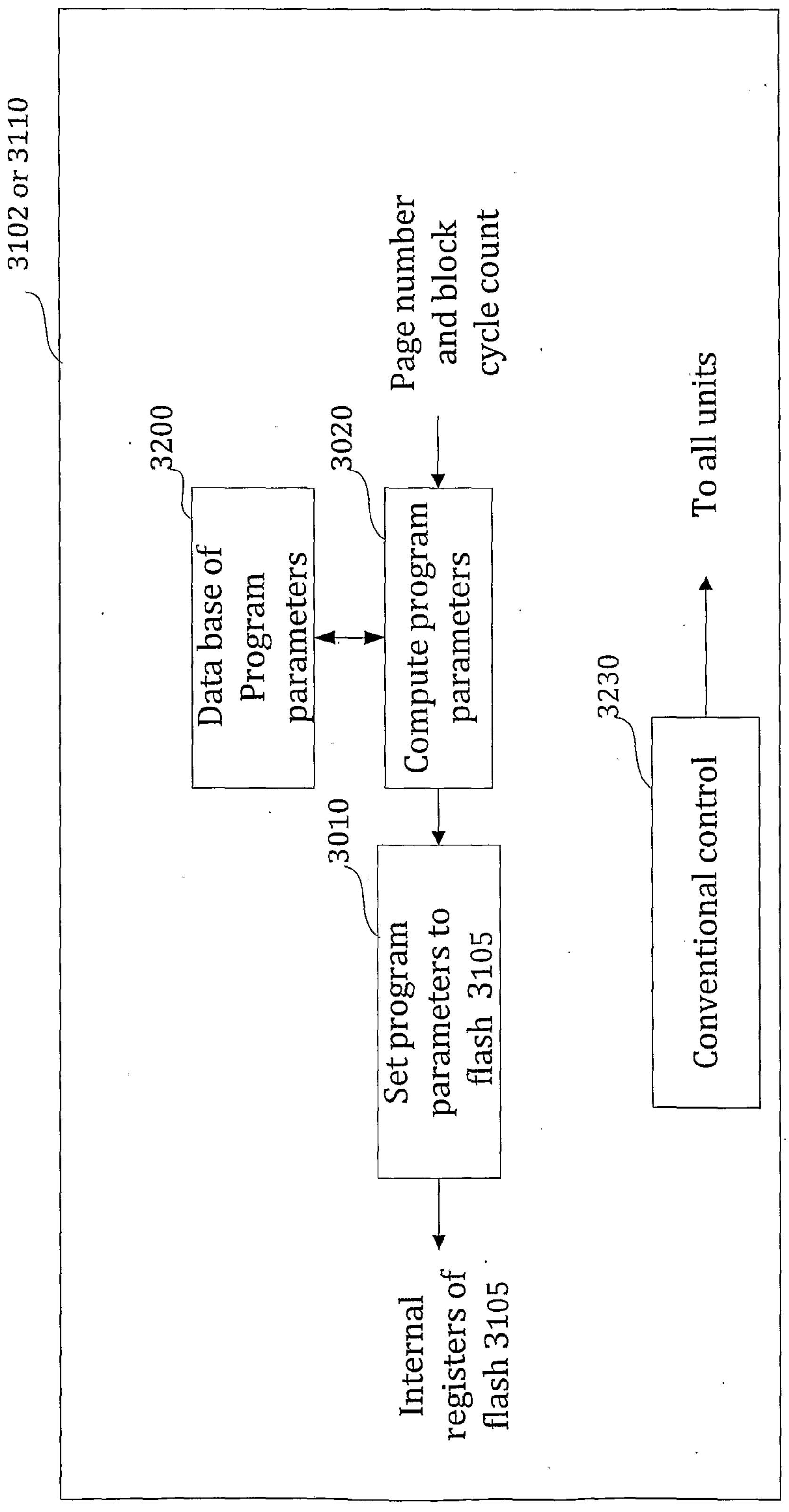
3102 or 3110
3200
Data base of Program parameters
3020
Compute program parameters
Page number and block cycle count
3010
Set program parameters to flash 3105
Internal registers of flash 3105
3230
Conventional control
To all units

Fig. 20

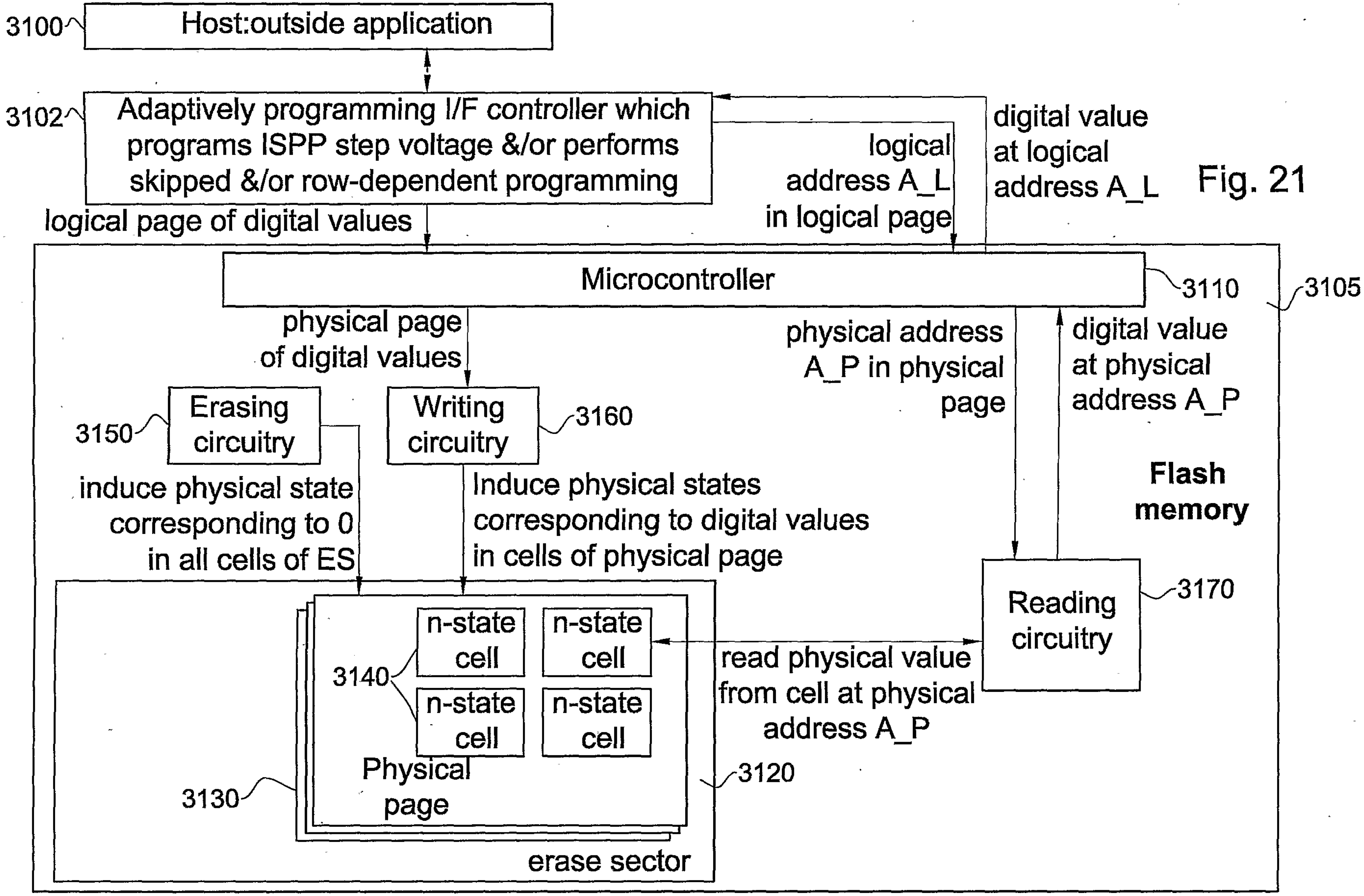
3100
Host:outside application
3102
Adaptively programming I/F controller which programs ISPP step voltage &/or performs skipped &/or row-dependent programming
logical page of digital values
logical address A_L in logical page
digital value at logical address A_L
Microcontroller
3110
3105
physical page of digital values
physical address A_P in physical page
digital value at physical address A_P
Flash memory
3150
Erasing circuitry
Writing circuitry
3160
induce physical state corresponding to 0 in all cells of ES
Induce physical states corresponding to digital values in cells of physical page
Reading circuitry
3170
n-state cell
n-state cell
n-state cell
n-state cell
3140
read physical value from cell at physical address A_P
Physical page
3130
3120
erase sector

Fig. 21