

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

(19) World Intellectual Property Organization
International Bureau



(43) International Publication Date
17 December 2009 (17.12.2009)

(10) International Publication Number
WO 2009/151933 A2

(51) International Patent Classification:
GIIC 16/08 (2006.01) *GIIC 16/06* (2006.01)

(74) Agent: **ALTMAN, Daniel, E.**; Knobbe, Martens, Olson & Bear, LLP, 2040 Main Street, 14th Floor, Irvine, CA 92614 (US).

(21) International Application Number:
PCT/US2009/045081

(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.

(22) International Filing Date:
22 May 2009 (22.05.2009)

(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, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, TR),

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
12/138,137 12 June 2008 (12.06.2008) US

(71) Applicant (for all designated States except US): **MICRON TECHNOLOGY, INC.** [US/US]; 8000 South Federal Way, P.O. Box 6, Boise, ID 83707 (US).

(72) Inventor; and

(75) Inventor/Applicant (for US only): **HAN, Jin-Man** [US/US]; 1072 Wilmington Avenue, San Jose, CA 95129 (US).

[Continued on next page]

(54) Title: METHOD OF STORING DATA ON A FLASH MEMORY DEVICE

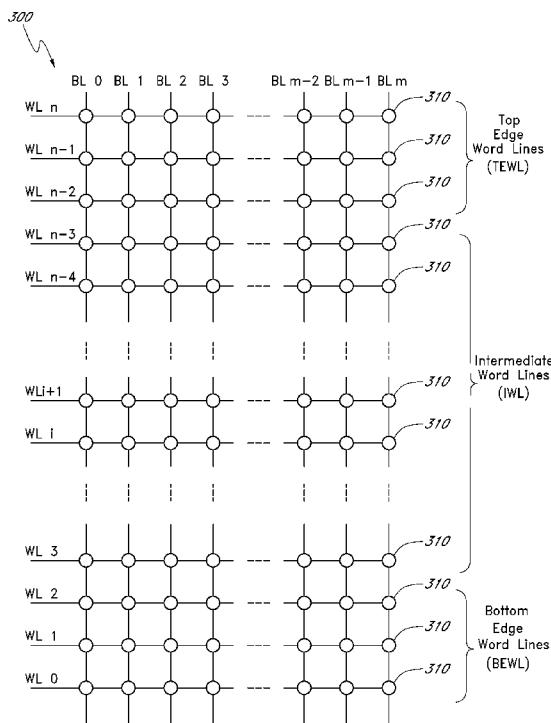


FIG. 3

(57) Abstract: Methods and apparatus are disclosed, such as those involving a flash memory device. One such method includes storing data on memory cells on a memory block (300) including a plurality of word lines (WL0- WL_n) and a plurality of memory cells (310) on the word lines (WL0- WL_n). The word lines (WL0- WL_n) comprising one or more bottom edge word lines (BEWL), one or more top edge word lines (TEWL), and intermediate word lines (IWL) between the bottom and top edge word lines. The data is stored first on memory cells on the intermediate word lines (IWL). Then, a remaining portion, if any, of the data is stored on memory cells on the bottom edge word lines (BEWL) and/or the top edge word lines (TEWL). This method enhances the life of the flash memory by preventing a premature failure of memory cells on the bottom or top edge word lines, which can be more prone to failure.



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 (Rule 48.2(g))*

METHOD OF STORING DATA ON A FLASH MEMORY DEVICE

BACKGROUND

Field of the Invention

[0001] Embodiments of the invention relate to memory devices, and more particularly, in one or more embodiments, to flash memory devices.

Description of the Related Art

[0002] Flash memory devices are non-volatile memory devices which store information on a semiconductor in a way that needs no power to maintain the information stored therein. Among flash memory devices, NAND flash memory devices have been widely used as mass-storage devices because of their high storage densities and low costs. In some applications, a NAND flash memory device serves as a solid state disk (SSD), replacing or supplementing a hard disk.

[0003] Referring to Figure 1, a conventional NAND flash memory device includes a plurality of memory blocks. The illustrated flash memory device 10 includes first to N -th memory blocks 100. Each of the memory blocks 100 includes a plurality of memory cells arranged in a matrix form.

[0004] Figure 2A illustrates one of the memory blocks 100 of the NAND flash memory device 10 of Figure 1. The illustrated memory block 100 includes first to m -th bit lines BL0-BL m and first to n -th word lines WL0-WL n . In some arrangements, m can be 32,767 or 65,535, and n can be 32 or 64. The bit lines BL0-BL m extend parallel to one another in a column direction. The word lines WL0-WL n extend parallel to one another in a row direction perpendicular to the column direction. The memory block 100 also includes upper and lower bit line select transistors 120a, 120b for selecting the one memory block 100 among the plurality of memory blocks of Figure 1 by coupling the one memory block 100 to bit lines extending outside the memory block 100.

[0005] Each bit line includes a string of memory cells 110. For example, the second bit line BL1 includes memory cells 110 connected in series. Each of the memory cells 100 includes a floating gate transistor. The floating gate transistors of the memory cells 100 are coupled to one another in series source to drain. The control gates of the floating gate transistors of memory cells 110 in the same row are coupled to the same word line. Each of the memory cells 110 stores a charge (or a lack of charge), wherein the amount of stored charge can be used to represent, for example, one or more states, and wherein the one or more states can represent one or more digits (e.g., bits) of data. The memory cell can be either a single-level cell (SLC) or a multi-level cell (MLC). In one arrangement, the amounts of charge stored in the memory cells 110 may be detected by sensing currents flowing through the floating gate transistors of the memory cells 110. In another arrangement, the amounts of charge stored in the memory cells 110 may be detected by sensing the threshold voltage values of the floating gate transistors of the memory cells 110.

[0006] Figure 2B illustrates a cross-section of the floating gate transistors of the memory cells 110 in the second bit line BL1. The floating gate transistors are formed on a substrate 201. Each of the floating gate transistors includes a source region 210 (which is a drain region for a neighboring transistor), a drain region 212 (which is a source region for a neighboring transistor), a doped channel region 214, a first dielectric (e.g., a tunnel oxide) 216, a floating gate 218, a second dielectric (e.g., a gate oxide, wherein the tunnel and gate oxide can be formed of the same or different material) 220, and a control gate 222. The tunnel oxide 216 is formed on the channel region 214 to insulate the floating gate 218 from the channel region 214. The gate dielectric 220 physically and electrically separates the floating gate 218 from the control gate 222. The control gate 222 is coupled to an appropriate word line, e.g., word line WL1. Electrons can be trapped on the floating gate 218 and be used to store data.

[0007] Referring now to Figures 1 and 2C, a conventional method of writing data on a memory block will be described. Figure 2C schematically illustrates the memory block 100 of Figure 2A, and only shows memory cells, bit lines, and word lines. However, it will be understood that the memory block 100 can include other components as described above with respect to Figures 2A and 2B.

[0008] During a write operation of the NAND flash memory device 10 (Figure 1), data is typically written on a set of memory cells on a single word line. Such a set of memory cells can be referred to as a “page.” In one arrangement, a page may include all memory cells on a word line. In other arrangements, a page may be formed by every other memory cells on a single word line. In certain arrangements, a page may be formed by every fourth memory cells on a single word line. It will be understood that a page may be formed by any suitable selected number of memory cells on a word line.

[0009] On the other hand, an erase operation of the NAND flash memory device 10 (Figure 1) is typically performed on a block-by-block basis. In other words, a page or memory cells cannot be selectively erased.

[0010] Likewise, when changing data values in some of memory cells in a memory block, the data values in the memory cells cannot be selectively changed. Instead, the entire memory block is erased and re-written (or programmed) with changed data values. For this process, data values stored in the entire memory block are copied to another memory block. For example, when some of data values in the I -th memory block are to be modified, the data values in the entire I -th memory block are copied to an unused memory block, for example, the J -th memory block. Subsequently, the data values in the entire I -th memory block are erased, and then updated data including unmodified and modified data values is written onto the erased I -th memory block.

BRIEF DESCRIPTION OF THE DRAWINGS

[0011] The embodiments will be better understood from the Detailed Description of Embodiments and from the appended drawings, which are meant to illustrate and not to limit the embodiments, and wherein:

[0012] Figure 1 is a schematic diagram of a conventional NAND flash memory device including a plurality of memory blocks;

[0013] Figure 2A is a schematic diagram of a memory block of a conventional NAND flash memory device;

[0014] Figure 2B is a schematic cross-section of the memory block of Figure 2A;

[0015] Figure 2C is a schematic diagram illustrating a conventional method of storing data on a block of a NAND flash memory device;

[0016] Figure 3 is a schematic diagram illustrating one embodiment of a method of storing data on a block of a NAND flash memory device;

[0017] Figure 4A is a schematic block diagram of a NAND flash memory device including an address decoder for changing word line addresses according to one embodiment;

[0018] Figure 4B is a schematic block diagram illustrating a method of mapping word lines addresses by the address decoder of Figure 4A;

[0019] Figure 4C is a schematic block diagram illustrating the function of the address decoder of Figure 4A; and

[0020] Figure 5 is a schematic block diagram of a computer system including a controller employing the method of Figure 3.

DETAILED DESCRIPTION OF EMBODIMENTS

[0021] In certain applications (for example, when a NAND flash memory serves as a solid state disk), data is frequently written or modified on a NAND flash memory. Thus, at least some memory blocks in the NAND flash memory are subjected to a great number of programming cycles.

[0022] In a conventional write method, data is written on a memory block in a bottom-up fashion, starting from memory cells on or near the lowermost (or bottom) word line. Then, data is sequentially written page-by-page on memory cells on word lines above the lowermost word line until memory cells on the uppermost (or top) word line are filled up. For example, data is written first on memory cells 110 on the lowermost word line WL0, and last on memory cells 110 on the uppermost word line WL_n in Figure 2C. In other instances, data may be written first on memory cells 110 on the second lowermost word line WL1, then memory cells 110 on the lowermost word line WL0, and may be written on memory cells 110 on word lines above the second lowermost word line WL1 in a bottom-up fashion. In the context of this document, the phrase “memory cells on a word line” indicates that the memory cells are electrically connected to the word line for address selection, not necessarily indicating that the memory cells are physically on the word line.

[0023] In certain cases employing the conventional write method described above, at least one of the memory blocks in a NAND flash memory may not be completely filled up during write operations. In such cases, some of the memory cells may remain unprogrammed. For example, memory cells on or near the uppermost word line WL_n of Figure 2C may not be reprogrammed to store data. Even in such cases, memory cells on several lowermost word lines of the memory block are always subjected to a programming cycle whenever data is written onto the memory block. Thus, memory cells on or near the lowermost word line WL₀ of the memory block experience the most number of programming cycles during the lifespan of the NAND flash memory device.

[0024] In addition, memory cells on or near the lowermost word line WL₀ or the uppermost word line WL_n are most vulnerable to damage because they are adjacent to bit line select transistors. For example, in Figure 2A, memory cells on or near the lowermost word line WL₀ and uppermost word lines WL_n are adjacent to the lower bit line select transistors 120b and the upper bit line select transistors 120a, respectively. It is well known in the state-of-the-art NAND flash memory industry that these memory cells experience more GIDL (Gate Induced Drain Leakage) and more charge trap-up than other memory cells in the memory block. Because of this inherent vulnerability in addition to more frequent exposure to programming cycles, these memory cells tend to fail before the other memory cells fail, rendering the entire block useless.

[0025] In one embodiment, during a write operation, data is written first on memory cells on word lines that are not adjacent to the uppermost word line or the lowermost word line. Then, if there is a remaining portion of the data, the remaining portion is written on memory cells on or near the uppermost word line and/or the lowermost word line. Because the entire memory block is not always filled up, this write scheme exposes the memory cells on or near the uppermost word line or the lowermost word line to less frequent programming cycles than the conventional method described above with respect to Figure 2C. Thus, the write scheme may prevent the memory cells on or near the uppermost word line WL_n or the lowermost word line WL₀ from prematurely failing, thereby lengthening the life of the memory block, and ultimately the life of the NAND flash memory device.

[0026] On the other hand, memory cells that are not on or near the uppermost word line or the lowermost word line are less vulnerable to damage than those on or near the uppermost word line or the lowermost word line. Thus, more frequent exposure of these memory cells to programming cycles does not necessarily shorten the lives of these memory cells.

[0027] Referring to Figure 3, a method of storing data on a memory block of a NAND flash memory device according to one embodiment will be described below. The illustrated memory block 300 includes word lines WL0-WLn, bit lines BL0-BLm, and memory cells 310 at the intersections of the word lines WL0-WLn and the bit lines BL0-BLm. The detailed configuration of the memory block 300 can be as described earlier in connection with the memory block 100 of Figure 2A.

[0028] In the context of this document, the lowermost word line WL0 and optionally a number of word lines immediately adjacent to the lowermost word line WL0 are collectively referred to as bottom edge word lines (BEWL). In one embodiment, the bottom edge word lines can be the word lines WL0 to WL4. In another embodiment, the bottom edge word lines can be the word lines WL0 to WL3. In yet another embodiment, the bottom edge word lines can be the word lines WL0 to WL2. In yet another embodiment, the bottom edge word lines can be the word lines WL0 and WL1. In yet another embodiment, the bottom edge word line can be the word line WL0 only. In other embodiments, the bottom edge word lines can include other word lines adjacent to the lowermost word line WL0.

[0029] In the context of this document, the uppermost word line WL_n and optionally a number of word lines immediately adjacent to the uppermost word line WL_n are collectively referred to as top edge word lines (TEWL). In one embodiment, the top edge word lines can be the word lines WL_n to WL_{n-4}. In another embodiment, the top edge word lines can be the word lines WL_n to WL_{n-3}. In yet another embodiment, the top edge word lines can be the word lines WL_n to WL_{n-2}. In yet another embodiment, the top edge word lines can be the word lines WL_n to WL_{n-1}. In yet another embodiment, the top edge word line can be the word lines WL_n only. In other embodiments, the top edge word lines can include other word lines adjacent to the uppermost word line WL_n.

[0030] The number of the top edge word lines may be the same as the number of the bottom edge word lines. In another embodiment, the number of the top edge word lines may be different from the number of the bottom edge word lines. The numbers of the top and bottom edge word lines can be determined during a manufacturing process by, for example, testing sample NAND flash memory devices.

[0031] Further, word lines that are neither a top edge word line nor a bottom edge word line are collectively referred to as intermediate word lines (IWL) in the context of this document. The intermediate word lines are positioned between the top edge word lines and the bottom edge word lines. The number of the intermediate word lines is equal to the total number of all the word lines less the total number of the top and bottom edge word lines in the memory block.

[0032] In one embodiment, during a write operation, data is written first onto memory cells on the intermediate word lines IWL, and if there is remaining data, then onto memory cells on the bottom edge word lines BEWL, and finally onto memory cells on the top edge word lines TEWL. In another embodiment, during a write operation, data is written first onto memory cells on the intermediate word lines IWL, and if there is remaining data, then onto memory cells on the top edge word lines TEWL, and finally onto memory cells on the bottom edge word lines BEWL.

[0033] The memory cells on the intermediate word lines IWL can be programmed in a bottom-up fashion, starting from memory cells on the lowest word line among the intermediate word lines IWL. For example, memory cells on the lowest of the intermediate word lines IWL can be programmed first, and memory cells on the highest of the intermediate word lines IWL can be programmed last while memory cells on the other intermediate word lines therebetween can be sequentially programmed line-by-line from bottom to top. In an example shown in Figure 3 where the bottom and top edge word lines BEWL and TEWL are three (3) word lines at the bottom and top edges, respectively, (i.e., BEWL are WL0, WL1, and WL2; and TEWL are WL_n, WL_{n-1}, and WL_{n-2}), the fourth word line WL3 may be programmed first, and then intermediate word lines above the fourth word line WL3 may be sequentially programmed line-by-line until the (n-3)-th word line WL_{n-3} is programmed. In another embodiment, the memory cells on the intermediate word

lines IWL can be programmed in a top-down fashion. A skilled artisan will appreciate that the sequence of programming among the intermediate word lines can vary widely.

[0034] The memory cells on the bottom edge word lines BEWL can be programmed in a top-down fashion, starting from memory cells on the highest word line among the bottom edge word lines BEWL. For example, memory cells on the highest of the bottom edge word lines BEWL can be programmed first, and memory cells on the lowest of the bottom edge word lines BEWL can be programmed last while memory cells on word lines therebetween can be sequentially programmed from top to bottom. In the example shown in Figure 3, the third word line WL2 may be programmed first, and then the second and first word lines WL1, WL0 below the third word line WL2 may be sequentially selected for being programmed. In another embodiment, the memory cells on the bottom edge word lines BEWL can be programmed in a bottom-up fashion. A skilled artisan will appreciate that the sequence of programming among the bottom edge word lines can vary widely.

[0035] The memory cells on the top edge memory cells TEWL can be programmed in a bottom-up fashion, starting from memory cells on the lowest of the top edge word lines. For example, memory cells on the lowest of the top edge word lines TEWL can be programmed first, and memory cells on the highest of the top edge word lines TEWL can be programmed last while memory cells on word lines therebetween can be sequentially programmed from bottom to top. In the example shown in Figure 3, the $(n-2)$ -th word line WL $n-2$ may be programmed first, and then the $(n-1)$ -th and n -th word lines WL $n-1$, WL n may be sequentially selected for being programmed. In another embodiment, the memory cells on the top edge memory cells TEWL can be programmed in a top-down fashion. A skilled artisan will appreciate that the sequence of programming among the top edge word lines can vary widely.

[0036] In yet another embodiment where the edge word lines BEWL, TEWL include two or more word lines, writing data on the memory cells on the bottom edge word lines BEWL and the top edge word lines TEWL can be alternated. In the example shown in Figure 3, writing data on the memory cells on these word lines can be performed in the following sequence: WL2, WL $n-2$, WL1, WL $n-1$, WL0, and WL n ; or WL $n-2$, WL2, WL $n-1$, WL1, WL n , and WL0. A skilled artisan will appreciate that these sequences can vary widely.

[0037] In this manner, the memory cells on the bottom and top edge word lines TEWL, BEWL are less frequently subjected to programming cycles when the memory block is not always completely filled up during write operations. On the other hand, the memory cells on the intermediate word lines IWL can be subjected to more programming cycles than memory cells on either the bottom or top edge word lines TEWL or BEWL. However, because the memory cells on the intermediate word lines IWL are less vulnerable to damage than those on the bottom and top edge word lines TEWL, BEWL, more frequent exposure of these memory cells to programming cycles does not necessarily shorten the lives of these memory cells. Thus, the overall life of the memory block, and ultimately the life of the NAND flash memory can be lengthened.

[0038] Referring to Figures 4A-4C, one embodiment of a NAND flash memory device employing the method described above in connection with Figure 3 will be described below. The NAND flash memory device 400 includes a memory block 401 and a decoder 450.

[0039] The memory block 401 includes word lines WL0-WLn-1, bit lines BL0-BLm, and a plurality of memory cells 410 in a matrix form. The detailed configuration of these components can be as described above in connection with Figure 2A.

[0040] The address decoder 450 receives addresses from an external memory controller (not shown). The addresses are generated by the memory controller, and designate the addresses of memory cells where values of the data are to be stored. The controller may be of any suitable type (for example, a microprocessor or CPU) that can be used for providing data read and/or write instructions to a NAND flash memory device. Thus, the controller generates the addresses in a conventional manner, in other words, in a bottom-up fashion to program memory cells first on the lowermost word lines and sequentially upward, as shown in a memory block 421 in Figure 4B. The numbers in the block 421 indicate the sequence of addresses for a write operation. In certain embodiments, the controller may generate the addresses in a reverse manner, in other words, in a top-down fashion to program memory cells first on the uppermost word lines and sequentially downward, as shown in a memory block 422 in Figure 4B. The numbers in the block 422 indicate the sequence of addresses for a write operation.

[0041] In the illustrated embodiment, the address decoder 450 maps the incoming addresses from the external memory controller to implement the method described above in connection with Figure 3. The address decoder 450 may map the received addresses such that a write operation is performed in the order of the intermediate word lines, the top edge word lines, and the bottom edge word lines, as shown in a memory block 423 in Figure 4B. The numbers in the block 423 indicate the sequence of addresses for a write operation. In the illustrated embodiment, the number of the bottom edge word lines is the same as the number of the top edge word lines.

[0042] In the illustrated embodiment, a word line address i is mapped into $i +$ the number of bottom edge word lines if i does not indicate a top edge word line (TEWL). If a word line address i indicates a top edge word line (TEWL), i is mapped into $i - (n + 1 -$ the number of the bottom edge word lines), as shown in Figure 4C. The address decoder 450 includes a plurality of logic gates to map incoming addresses as described above. The logic gates may be hard-wired to map incoming addresses in the manner described above. A skilled artisan will appreciate that various schemes can be used at the address decoder 450 or any other component in the NAND flash memory device 400 for implementing any of the embodiments described above in connection with Figure 3.

[0043] Referring to Figure 5, one embodiment of a computer system employing the method described above in connection with Figure 3 for storing data in a NAND flash memory-based solid state disk will be described below. The illustrated computer system 500 includes a central processing unit (CPU) (or a controller) 510, a volatile memory 520, and a solid state disk 530. Although not illustrated, the computer system 500 may also include other components such as a user interface, peripheral devices, and a hard disk.

[0044] The CPU 510 can be of any suitable type, and have any suitable processing speed. The volatile memory 520 can also be of any suitable type and have any suitable data transfer rate. The solid state disk 530 can include a NAND flash memory device which may include a plurality of memory blocks as described above with reference to Figure 1. The NAND flash memory device can also include an address decoder and other components for the operation of the NAND flash memory device. The NAND flash memory device can be any suitable commercially available NAND flash memory.

[0045] In the illustrated embodiment, the CPU 510 is configured to generate word line addresses such that data is stored in the solid state disk 530 in any of the write sequences described earlier in connection with Figure 3. The CPU 510 then sends the volatile memory 520 the word line addresses along with bit line addresses and data. The volatile memory 520 temporarily stores the word line and bit line addresses and data, and then relays them to the solid state disk 530.

[0046] The solid state disk 530 stores the data in one or more memory blocks therein according to the addresses provided by the CPU 510 via the volatile memory 520. Because the word lines are mapped such that memory cells on the intermediate word lines are written or read first, the solid state disk 530 can store the data in the manner described earlier in connection with Figure 3 without modifying or changing the addresses at the input of address decoder of the solid state disk 530.

[0047] In some embodiments, the solid state disk 530 may request the CPU 510 to provide word line addresses in the fashion described above at the start of operation. In certain embodiments, the solid state disk 530 can contain information on desired numbers of the top and bottom edge word lines of its memory blocks. The desired numbers of the top and bottom edge word lines can vary, depending on the flash memory device in the solid state disk 530. The solid state disk 530 may send the information to the CPU 510 via the volatile memory 520. Upon receiving the information, the CPU 510 can generate word line addresses based at least partly on the information. A skilled artisan will appreciate that various schemes can be used to instruct the CPU 510 to generate word line addresses as described above.

[0048] In the methods of the embodiments described above, memory cells on intermediate word lines are programmed more frequently than those on bottom and top edge word lines. Because the memory cells on intermediate word lines are more durable than those on the bottom or top edge word lines, the methods can prevent a premature failure of memory blocks that may occur in the conventional write method described above with reference to Figure 2C.

[0049] The embodiments above are described in the context of NAND flash memory devices. The embodiments can also be adapted for other types of memory devices.

In addition, the embodiments above can be adapted for any other types of solid state memory devices that have similar problems described above due to their write methods and inherent vulnerability of certain memory cells to damages.

[0050] A flash memory device according to the embodiments described above can be incorporated in various electronic devices. Examples of the electronic devices can include, but are not limited to, consumer electronic products, electronic circuits, electronic circuit components, parts of the consumer electronic products, electronic test equipments, etc. Examples of the consumer electronic products include, but are not limited to, a mobile phone, a telephone, a television, a computer monitor, a computer, a portable memory device (e.g., USB drive), a solid state disk, a hand-held computer, a personal digital assistant (PDA), a microwave, a refrigerator, a stereo system, a cassette recorder or player, a DVD player, a CD player, a VCR, an MP3 player, a radio, a camcorder, an optical camera, a digital camera, a washer, a dryer, a washer/dryer, a copier, a facsimile machine, a scanner, a multi functional peripheral device, a wrist watch, a clock, a game device, etc. Further, the electronic device can include unfinished products.

[0051] One embodiment is a method of arranging data within a block of a flash memory device. The block includes a plurality of word lines and a plurality of memory cells of the word lines. The word lines include one or more bottom edge word lines, one or more top edge word lines, and intermediate word lines between the bottom and top edge word lines. The method includes mapping data to be stored such that initial data of a block is stored on at least one of memory cells on the intermediate word lines of the block before storing to one or more bottom edge word lines or one or more top edge word lines of the block; and mapping at least a portion of subsequent data, if any, to be stored on memory cells on the one or more bottom edge word lines and/or the one or more top edge word lines.

[0052] Another embodiment is a flash memory device including one or more memory blocks. Each of the memory blocks includes a plurality of word lines having a plurality of memory cells. The word lines extend parallel to one another in a direction. The word lines include one or more bottom edge word lines, one or more top edge word lines, and intermediate word lines interposed between the bottom and top edge word lines. The memory device also includes an address decoder configured to map data to be stored such

that initial data of a block is stored on at least one of memory cells on the intermediate word lines of the block before storing to the one or more bottom edge word lines or the one or more top edge word lines of the block. The address decoder is further configured to map at least a portion of subsequent data, if any, to be stored on the one or more bottom edge word lines and/or the one or more top edge word lines of the block.

[0053] Yet another embodiment is a computer system including: a controller; and a flash memory device configured to receive addresses for storing data therein. The flash memory device includes one or more memory blocks. Each of the memory blocks includes a plurality of word lines and a plurality of memory cells of the word lines. The word lines extend parallel to one another in a direction. The word lines include one or more bottom edge word lines, one or more top edge word lines, and intermediate word lines interposed between the bottom and top edge word lines. The controller is configured to generate the addresses such that the addresses are indicative of a write sequence to store an initial portion of the data on memory cells on the intermediate word lines of one of the blocks, and to store a subsequent portion, if any, of the data on memory cells on the bottom and/or top edge word lines of the one of the blocks.

[0054] Yet another embodiment is an apparatus including: a controller configured to generate a set of word line addresses for storing data in a flash memory device. The flash memory device includes one or more memory blocks. Each of the memory blocks includes a plurality of word lines and a plurality of memory cells of the word lines. The word lines extend parallel to one another in a direction. The word lines include one or more bottom edge word lines, one or more top edge word lines, and intermediate word lines interposed between the bottom and top edge word lines. The set of word line addresses is indicative of a write sequence to store an initial portion of the data on memory cells on the intermediate word lines of one of the blocks, and to store a subsequent portion, if any, of the data on memory cells on the bottom and/or top edge word lines of the one of the blocks.

[0055] Although this invention has been described in terms of certain embodiments, other embodiments that are apparent to those of ordinary skill in the art, including embodiments that do not provide all of the features and advantages set forth herein, are also within the scope of this invention. Moreover, the various embodiments described

above can be combined to provide further embodiments. In addition, certain features shown in the context of one embodiment can be incorporated into other embodiments as well. Accordingly, the scope of the present invention is defined only by reference to the appended claims.

I CLAIM:

1. A method of arranging data within a block of a flash memory device, wherein the block comprises a plurality of word lines and a plurality of memory cells of the word lines, the word lines comprising one or more bottom edge word lines, one or more top edge word lines, and intermediate word lines between the bottom and top edge word lines, the method comprising:

mapping data to be stored such that initial data of a block is stored on at least one of memory cells on the intermediate word lines of the block before storing to one or more bottom edge word lines or one or more top edge word lines of the block; and

mapping at least a portion of subsequent data, if any, to be stored on memory cells on the one or more bottom edge word lines and/or the one or more top edge word lines.

2. The method of Claim 1, wherein the one or more top edge word lines comprise 1 to 5 word lines, and wherein the one or more bottom edge word lines comprise 1 to 5 word lines.

3. The method of Claim 2, wherein the number of the one or more top edge word lines is the same as the number of the one or more bottom edge word lines.

4. The method of Claim 2, wherein the number of the one or more top edge word lines is different from the number of the one or more bottom edge word lines.

5. The method of Claim 1, wherein mapping the at least a portion of subsequent data comprises:

mapping a part of the at least a portion of subsequent data to be stored on the memory cells on the bottom edge word lines; and

mapping another part of the at least a portion of subsequent data to be stored on the memory cells on the top edge word lines.

6. The method of Claim 5, wherein mapping the part comprises mapping the part to be stored first on memory cells on a bottom edge word line closest to the intermediate word lines.

7. The method of Claim 5, wherein mapping the other part comprises mapping the other part to be stored first on memory cells on a top edge word line closest to the intermediate word lines.

8. The method of Claim 5, wherein mapping the at least a portion of subsequent data comprises alternating:

mapping a part of the at least a portion to be stored on the memory cells on the bottom edge word lines; and

mapping another part of the at least a portion to be stored on the memory cells on the top edge word lines.

9. The method of Claim 1, further comprising:

receiving, by the flash memory device, a first set of word line addresses indicative of a first write sequence to store the data first on the memory cells on the bottom or top edge word lines; and

converting, by the flash memory device, the first set of word line addresses into a second set of word line addresses indicative of a second write sequence to store the data first on the memory cells on the intermediate word lines.

10. The method of Claim 1, further comprising: receiving, by the flash memory device, word lines addresses indicative of a write sequence to store the data first on the memory cells on the intermediate word lines.

11. A flash memory device comprising:

one or more memory blocks, each of the memory blocks including a plurality of word lines having a plurality of memory cells, the word lines extending parallel to one another in a direction, the word lines comprising one or more bottom edge word lines, one or more top edge word lines, and intermediate word lines interposed between the bottom and top edge word lines; and

an address decoder configured to map data to be stored such that initial data of a block is stored on at least one of memory cells on the intermediate word lines of the block before storing to the one or more bottom edge word lines or the one or more top edge word lines of the block,

wherein the address decoder is further configured to map at least a portion of subsequent data, if any, to be stored on the one or more bottom edge word lines and/or the one or more top edge word lines of the block.

12. The device of Claim 11, wherein the one or more top edge word lines comprise 1 to 5 word lines, and wherein the one or more bottom edge word lines comprise 1 to 5 word lines..

13. The device of Claim 11, wherein the address decoder is configured to receive a first set of word line addresses indicative of a first write sequence to store data first on memory cells on the bottom or top edge word lines, and to convert the first set of word line addresses into a second set of word line addresses indicative of a second write sequence to store the data first on memory cells on the intermediate word lines.

14. The device of Claim 11, wherein the flash memory comprises a NAND flash memory.

15. A computer system comprising:

a controller; and

a flash memory device configured to receive addresses for storing data therein, the flash memory device comprising one or more memory blocks, each of the memory blocks including a plurality of word lines and a plurality of memory cells of the word lines, the word lines extending parallel to one another in a direction, the word lines comprising one or more bottom edge word lines, one or more top edge word lines, and intermediate word lines interposed between the bottom and top edge word lines,

wherein the controller is configured to generate the addresses such that the addresses are indicative of a write sequence to store an initial portion of the data on memory cells on the intermediate word lines of one of the blocks, and to store a subsequent portion, if any, of the data on memory cells on the bottom and/or top edge word lines of the one of the blocks.

16. The system of Claim 15, wherein the one or more top edge word lines comprise 1 to 5 word lines, and wherein the one or more bottom edge word lines comprise 1 to 5 word lines.

17. The system of Claim 16, wherein the flash memory device is further configured to provide the controller with information on at least one of the numbers of the top and bottom edge word lines.

18. The system of Claim 15, wherein the controller comprises a central processing unit (CPU).

19. The system of Claim 15, wherein the flash memory device serves as a solid state disk.

20. The system of Claim 15, wherein the flash memory device comprises a NAND flash memory device.

21. An apparatus comprising:

a controller configured to generate a set of word line addresses for storing data in a flash memory device, the flash memory device comprising one or more memory blocks, each of the memory blocks including a plurality of word lines and a plurality of memory cells of the word lines, the word lines extending parallel to one another in a direction, the word lines comprising one or more bottom edge word lines, one or more top edge word lines, and intermediate word lines interposed between the bottom and top edge word lines,

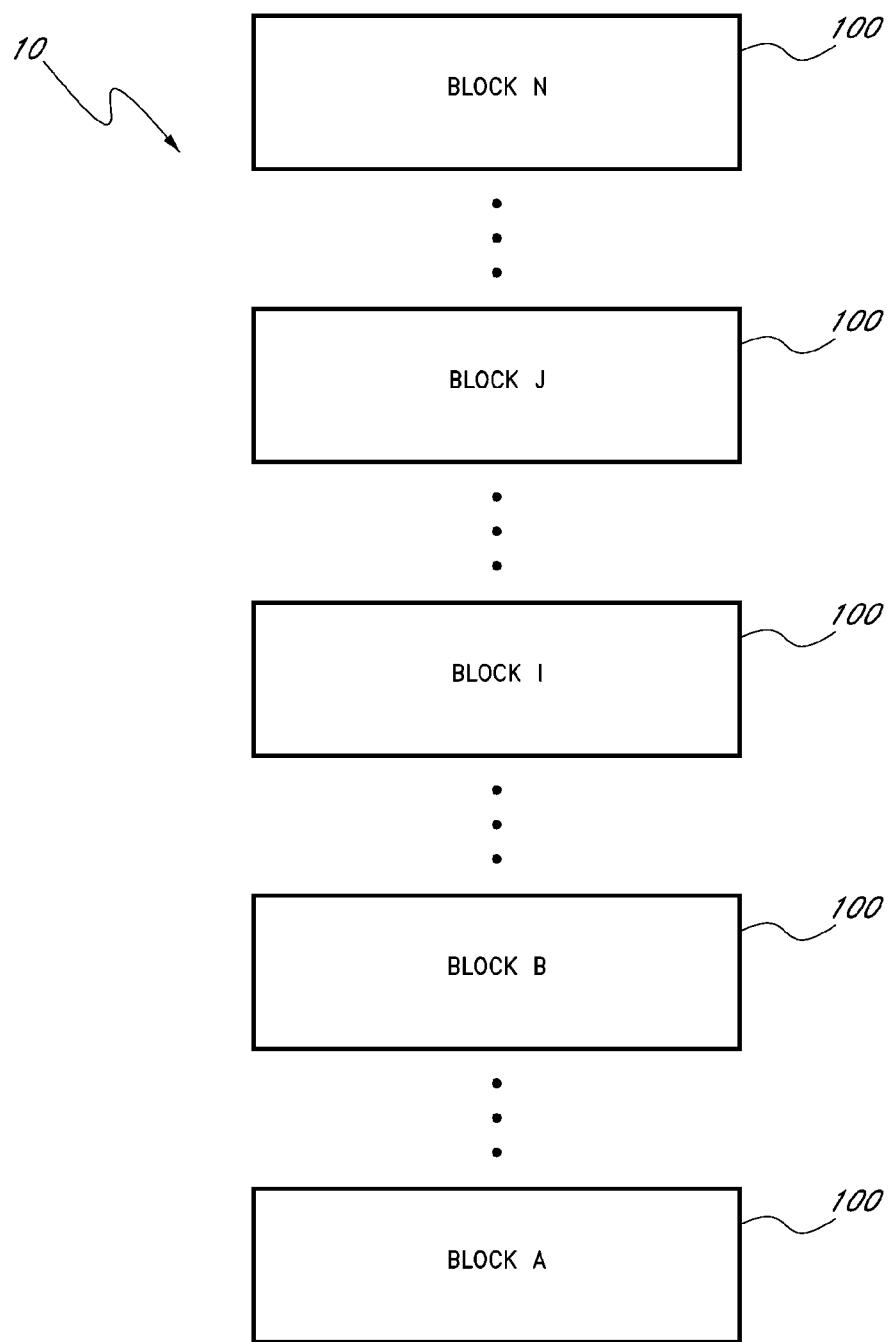
wherein the set of word line addresses is indicative of a write sequence to store an initial portion of the data on memory cells on the intermediate word lines of one of the blocks, and to store a subsequent portion, if any, of the data on memory cells on the bottom and/or top edge word lines of the one of the blocks.

22. The apparatus of Claim 21, wherein the one or more top edge word lines comprise 1 to 5 word lines, and wherein the one or more bottom edge word lines comprise 1 to 5 word lines.

23. The apparatus of Claim 22, wherein the controller is further configured to receive information on at least one of the numbers of the top and bottom edge word lines, and to generate the set of word line addresses based at least partly on the information.

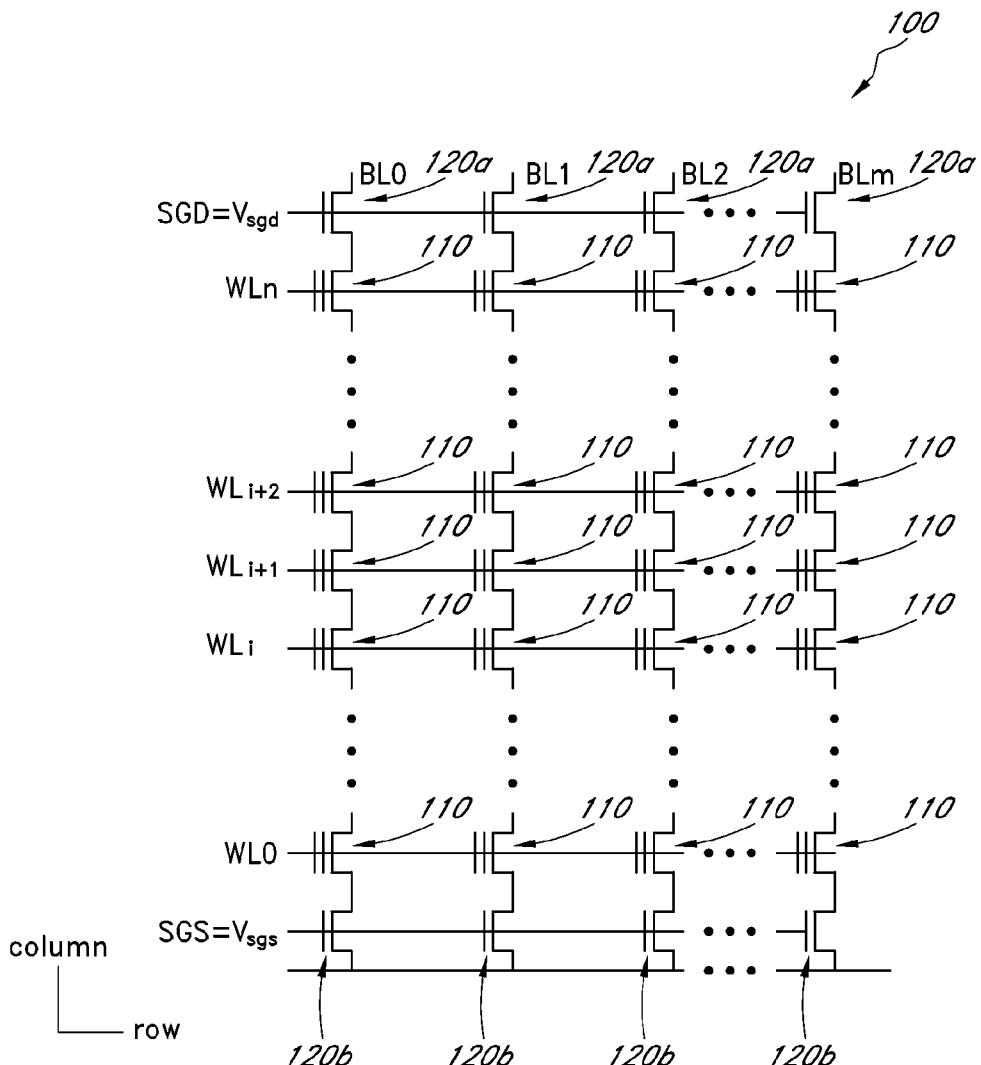
24. The apparatus of Claim 21, wherein the controller comprises a central processing unit (CPU).

1/7

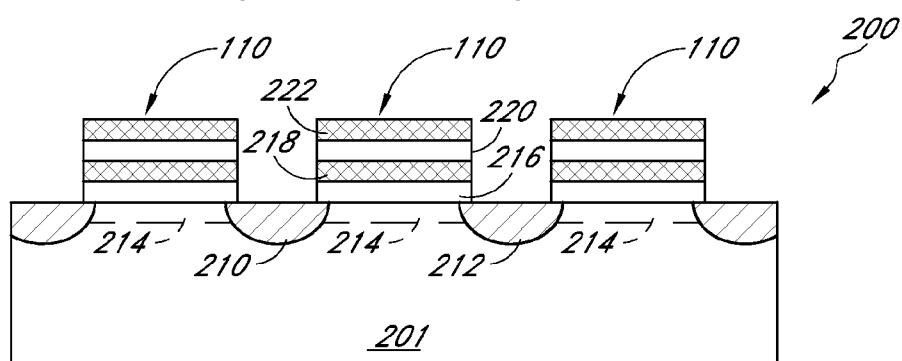


*FIG. 1
(PRIOR ART)*

2/7



*FIG. 2A
(PRIOR ART)*



*FIG. 2B
(PRIOR ART)*

3/7

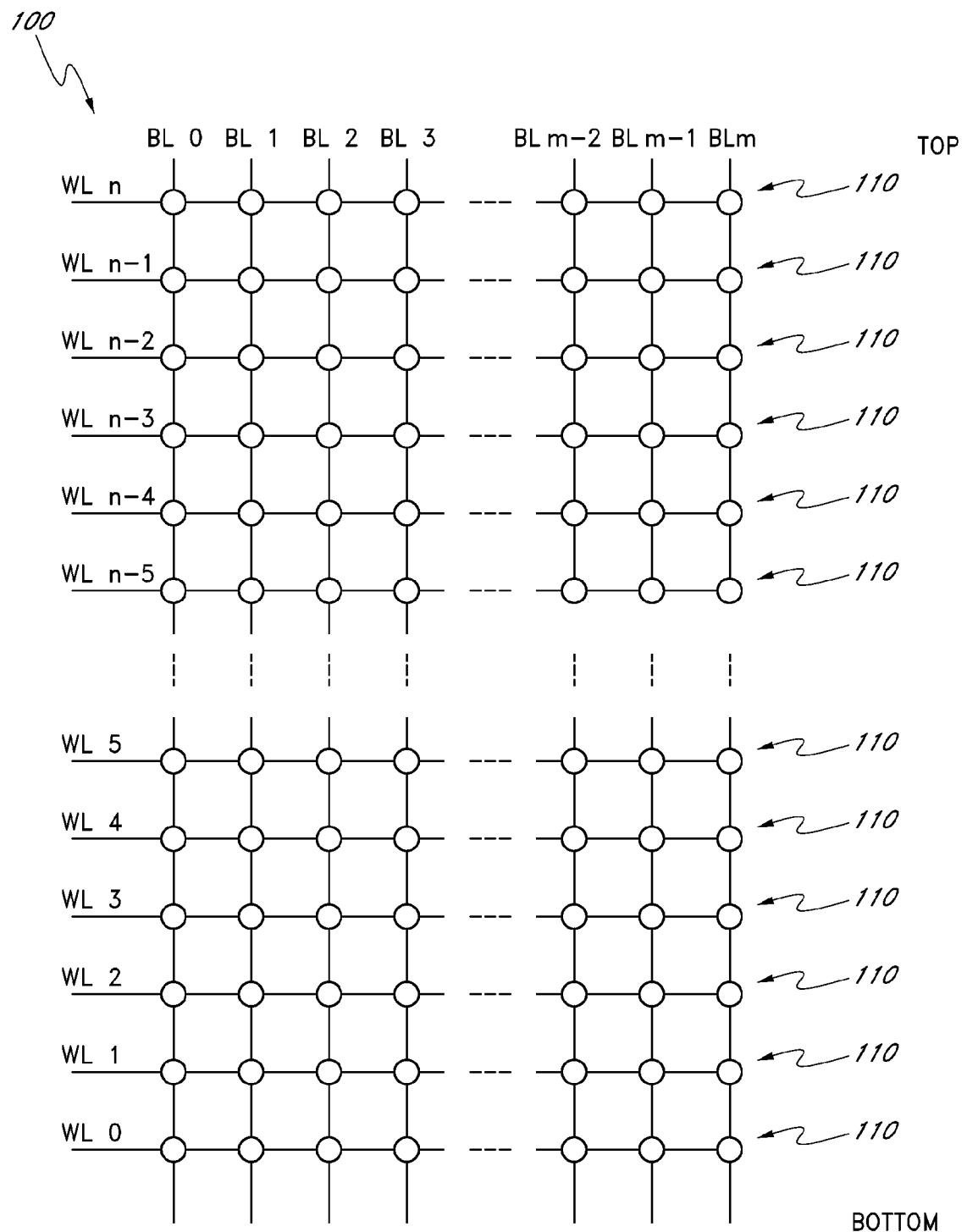


FIG.2C

4/7

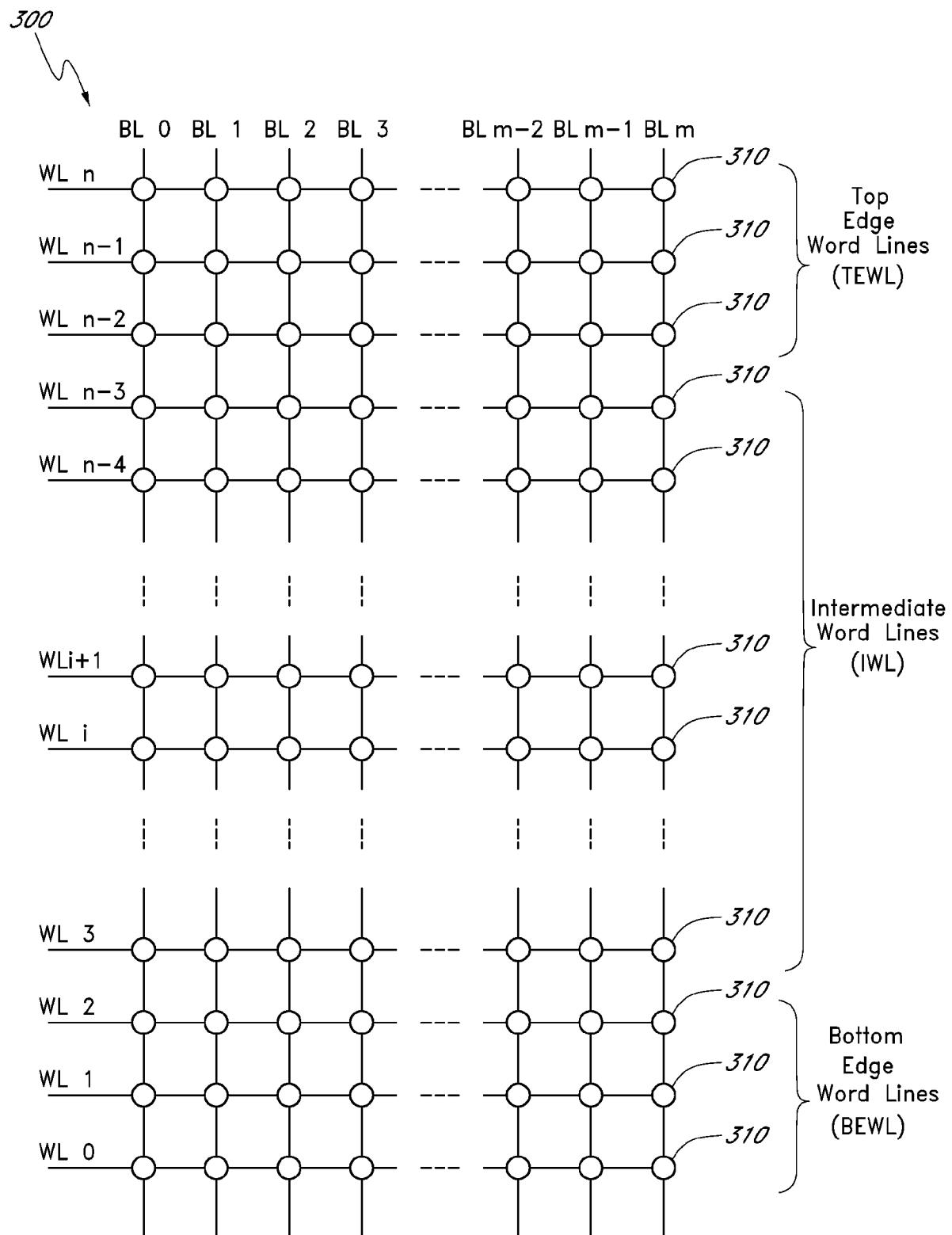


FIG. 3

5/7

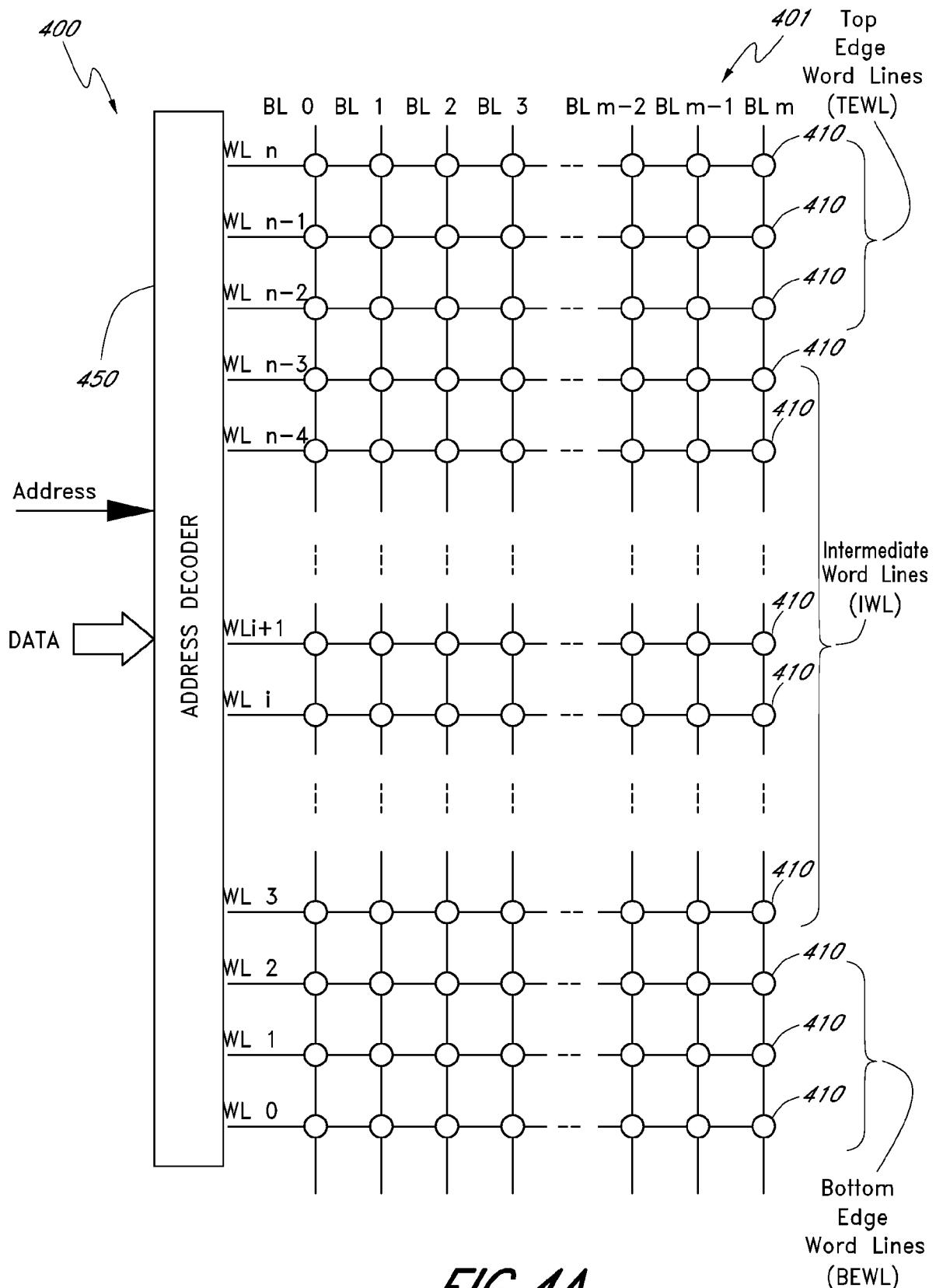


FIG. 4A

6/7

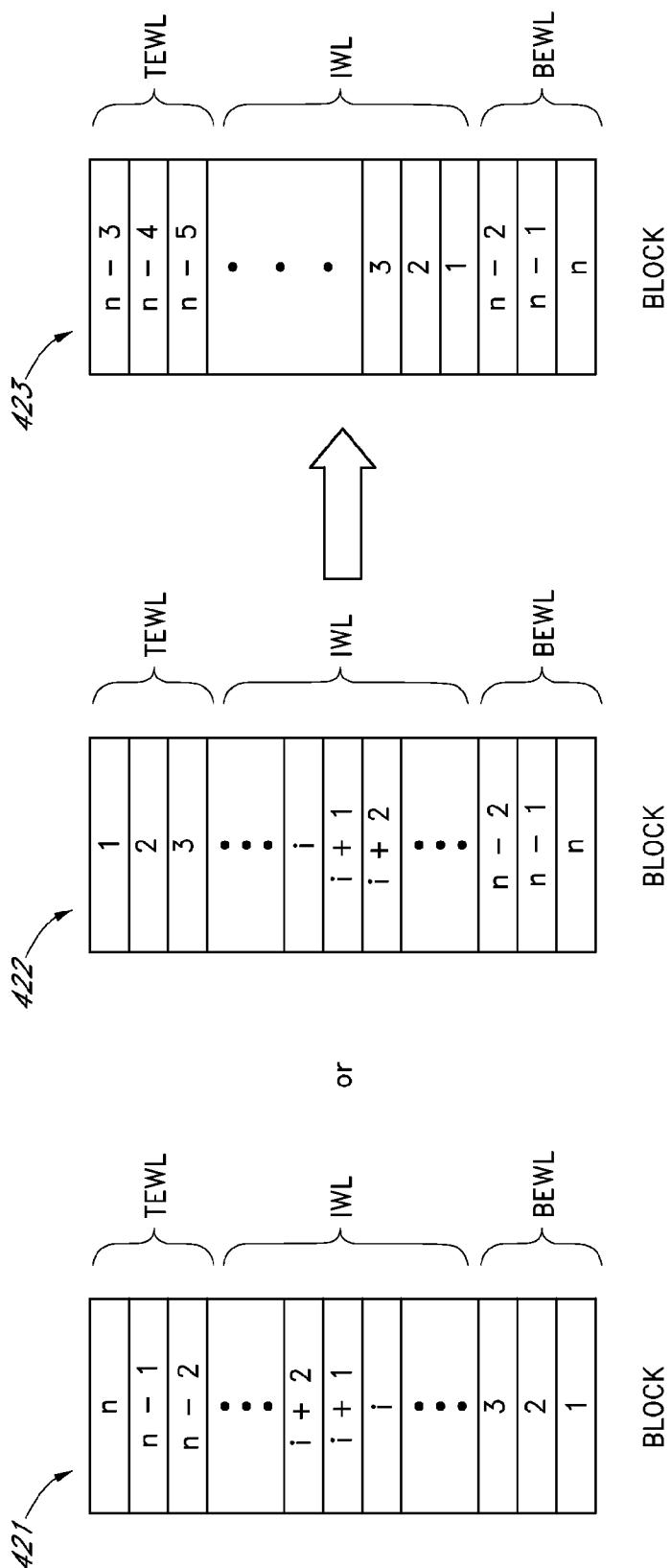


FIG. 4B

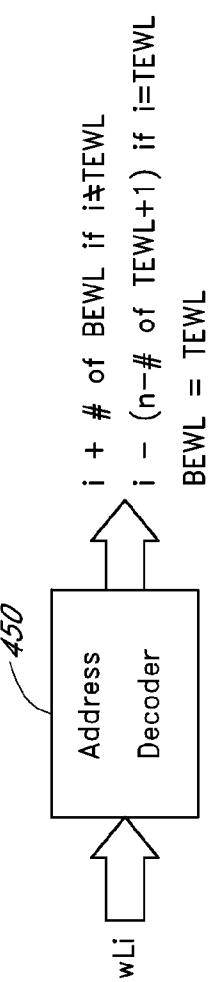
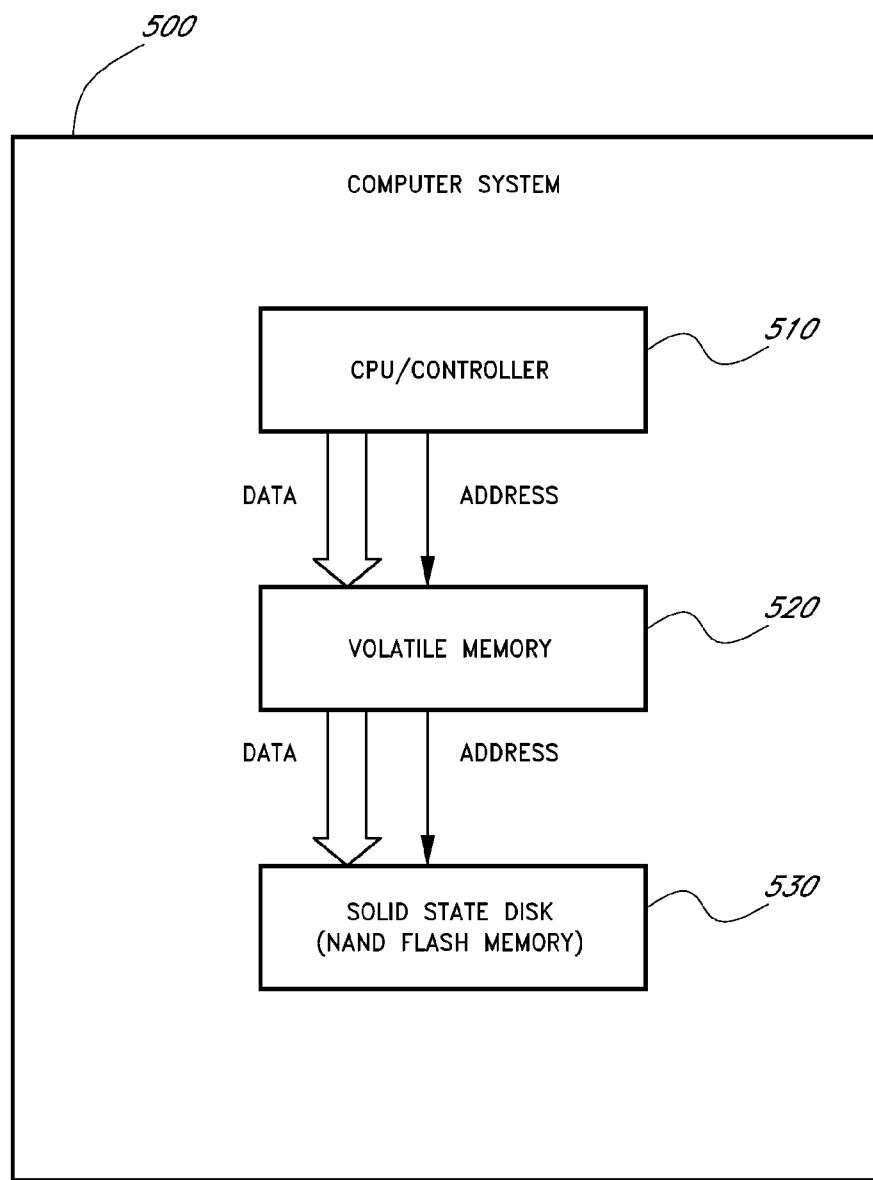


FIG. 4C

7/7

*FIG.5*