US 20250258617A1

(19) United States

(12) Patent Application Publication (10) Pub. No.: US 2025/0258617 A1

Yang et al. (43) Pub. Date: Aug. 14, 2025

(54) **APPARATUS AND METHODS FOR SUB-BLOCK READ REFRESH FOR NONVOLATILE MEMORY DEVICES**

(71) Applicant: **SanDisk Technologies LLC**, Austin, TX (US)

(72) Inventors: **Xiang Yang**, Santa Clara, CA (US); **Wei Cao**, Fremont, CA (US); **Deepanshu Dutta**, Fremont, CA (US)

(21) Appl. No.: **18/660,521**

(22) Filed: **May 10, 2024**

Related U.S. Application Data

(60) Provisional application No. 63/552,783, filed on Feb. 13, 2024.

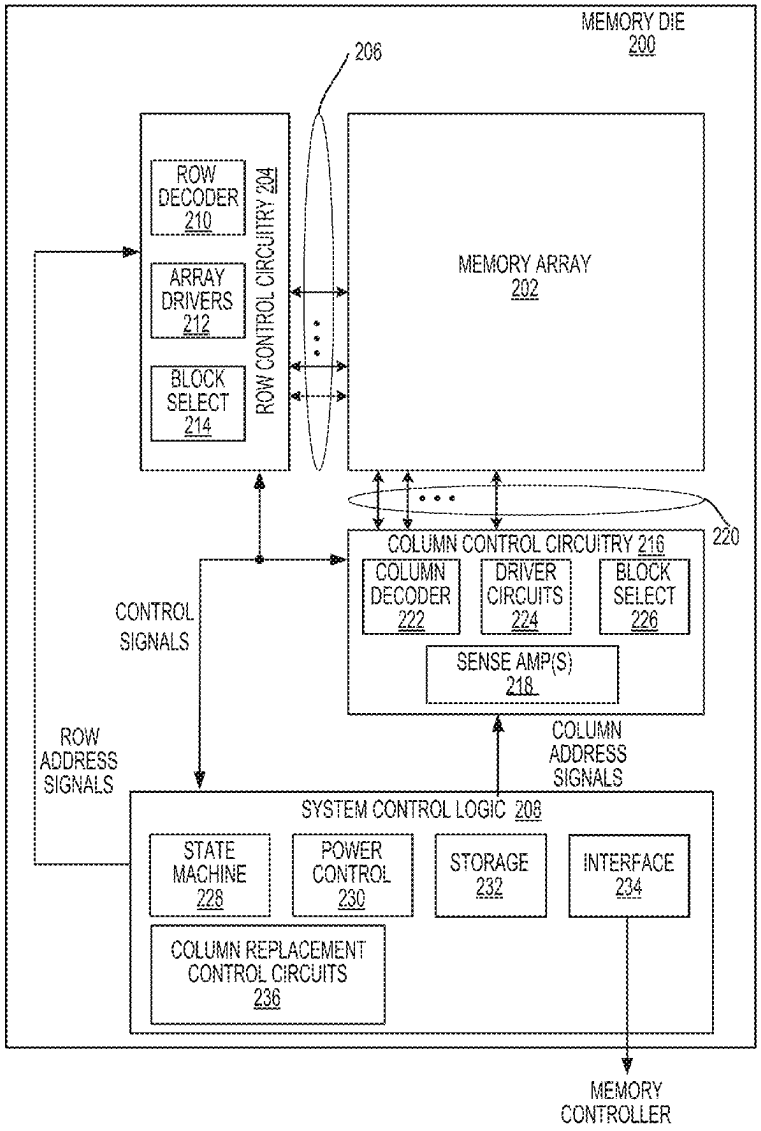
Publication Classification

(51) **Int. Cl.**
G06F 3/06 (2006.01)

(52) **U.S. Cl.**
CPC ***G06F 3/0625*** (2013.01); ***G06F 3/0653*** (2013.01); ***G06F 3/0679*** (2013.01)

(57) **ABSTRACT**

The memory device includes a memory block with an array of memory cells that are arranged in a plurality of word lines. The word lines are divided into a first sub-block and a second sub-block with the memory cells of the first sub-block containing data and with the memory cells of the second sub-block being erased. The memory device also includes circuitry that is configured to determine that the memory cells of the first sub-block have experienced significant read disturb. The circuitry is also configured to program the user data in the memory cells of the first sub-block into the memory cells of the second sub-block.



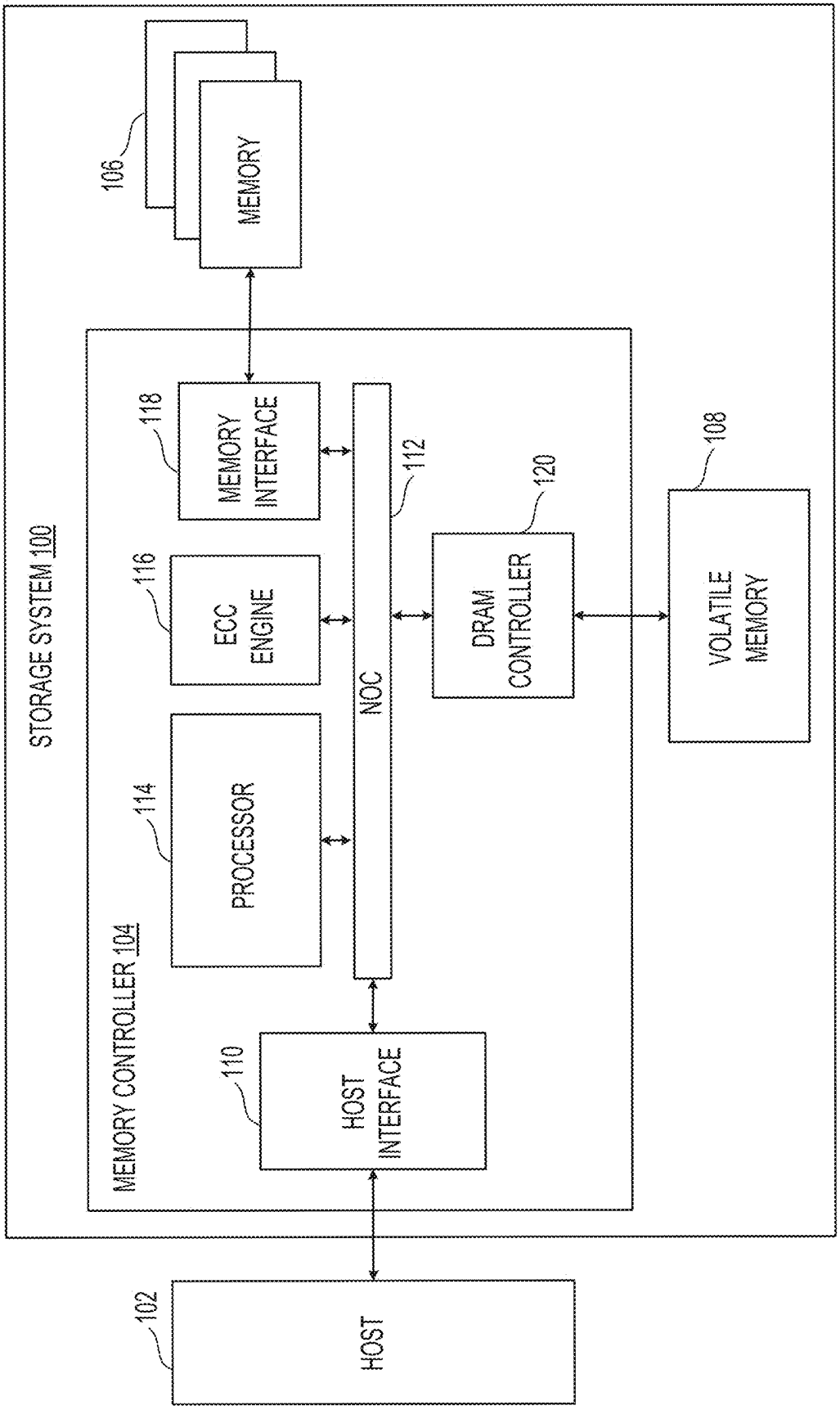

STORAGE SYSTEM 100
MEMORY CONTROLLER 104
106
MEMORY
118
MEMORY INTERFACE
112
NOC
116
ECC ENGINE
120
DRAM CONTROLLER
108
VOLATILE MEMORY
114
PROCESSOR
110
HOST INTERFACE
102
HOST

FIG. 1

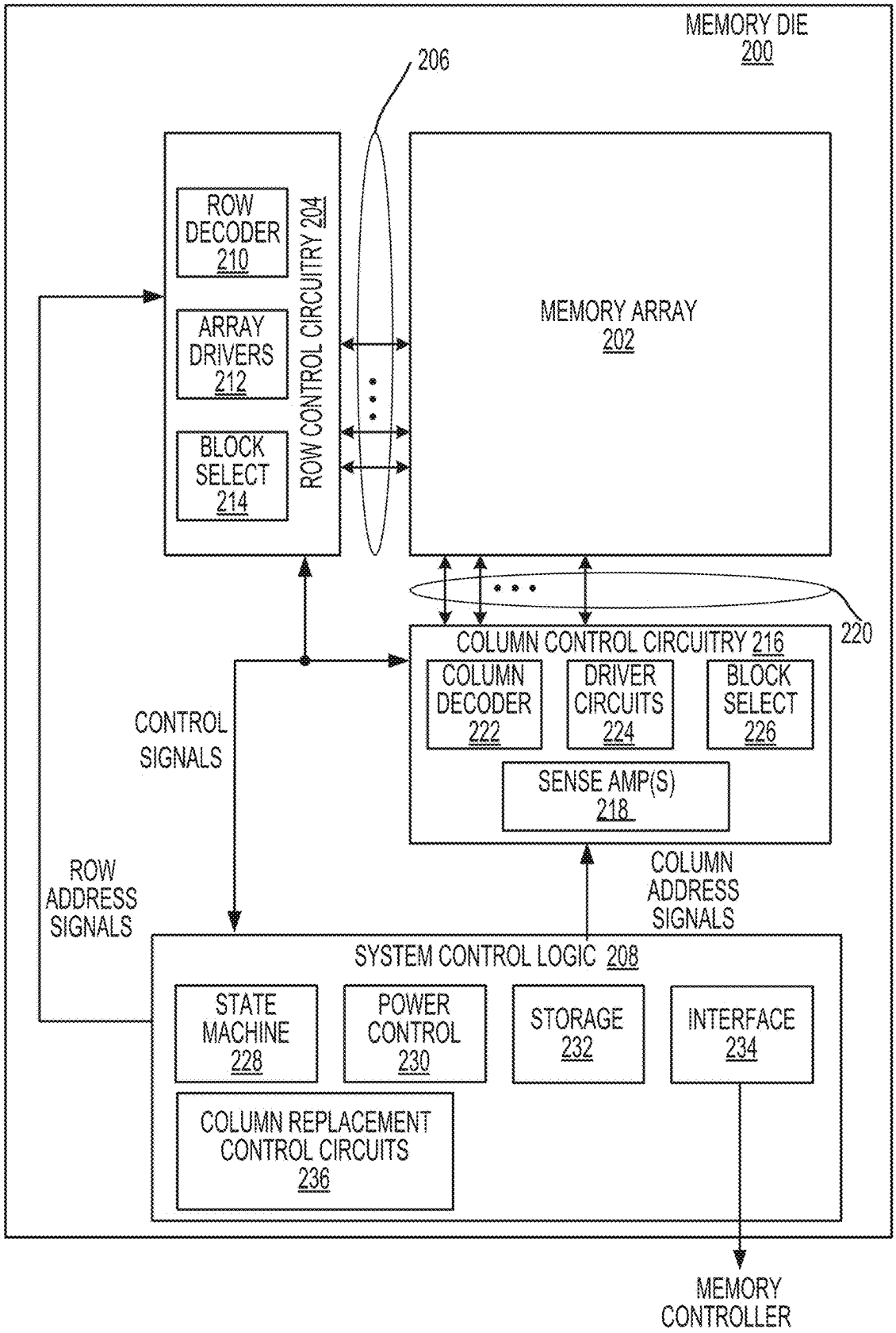

MEMORY DIE
200
206
ROW DECODER
210
ARRAY DRIVERS
212
BLOCK SELECT
214
ROW CONTROL CIRCUITRY 204
MEMORY ARRAY
202
220
COLUMN CONTROL CIRCUITRY 216
COLUMN DECODER
222
DRIVER CIRCUITS
224
BLOCK SELECT
226
SENSE AMP(S)
218
CONTROL SIGNALS
ROW ADDRESS SIGNALS
COLUMN ADDRESS SIGNALS
SYSTEM CONTROL LOGIC 208
STATE MACHINE
228
POWER CONTROL
230
STORAGE
232
INTERFACE
234
COLUMN REPLACEMENT CONTROL CIRCUITS
236
MEMORY CONTROLLER

FIG. 2A

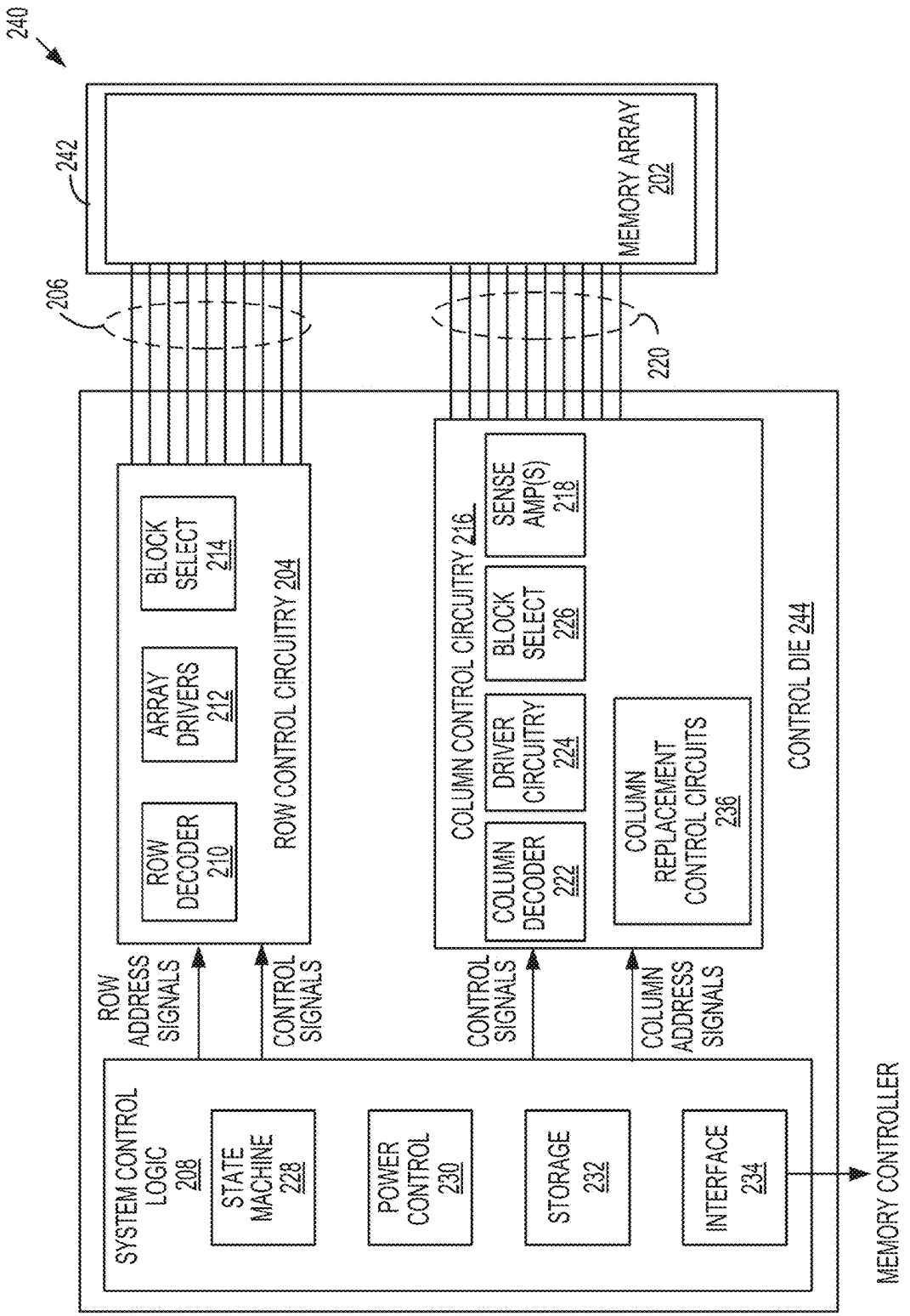

240
242
MEMORY ARRAY 202
206
220
ROW CONTROL CIRCUITRY 204
ROW DECODER 210
ARRAY DRIVERS 212
BLOCK SELECT 214
COLUMN CONTROL CIRCUITRY 216
COLUMN DECODER 222
DRIVER CIRCUITRY 224
BLOCK SELECT 226
SENSE AMP(S) 218
COLUMN REPLACEMENT CONTROL CIRCUITS 236
CONTROL DIE 244
ROW ADDRESS SIGNALS
CONTROL SIGNALS
CONTROL SIGNALS
COLUMN ADDRESS SIGNALS
SYSTEM CONTROL LOGIC 208
STATE MACHINE 228
POWER CONTROL 230
STORAGE 232
INTERFACE 234
MEMORY CONTROLLER

FIG. 2B

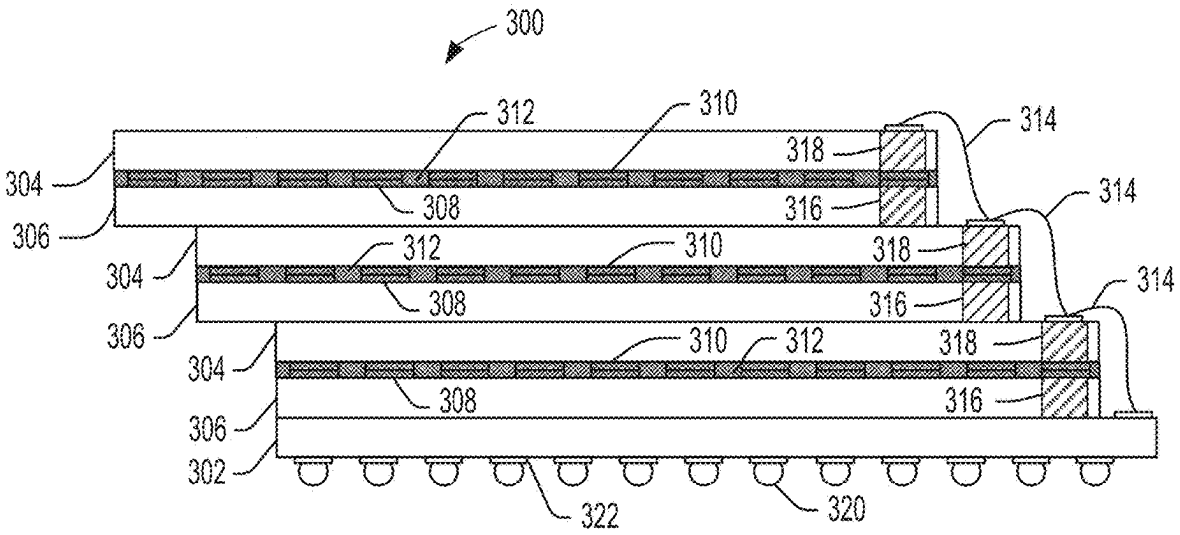

300
312
310
314
318
304
308
316
306
312
310
314
318
304
308
316
306
310
312
314
318
304
308
316
306
302
322
320

FIG. 3A

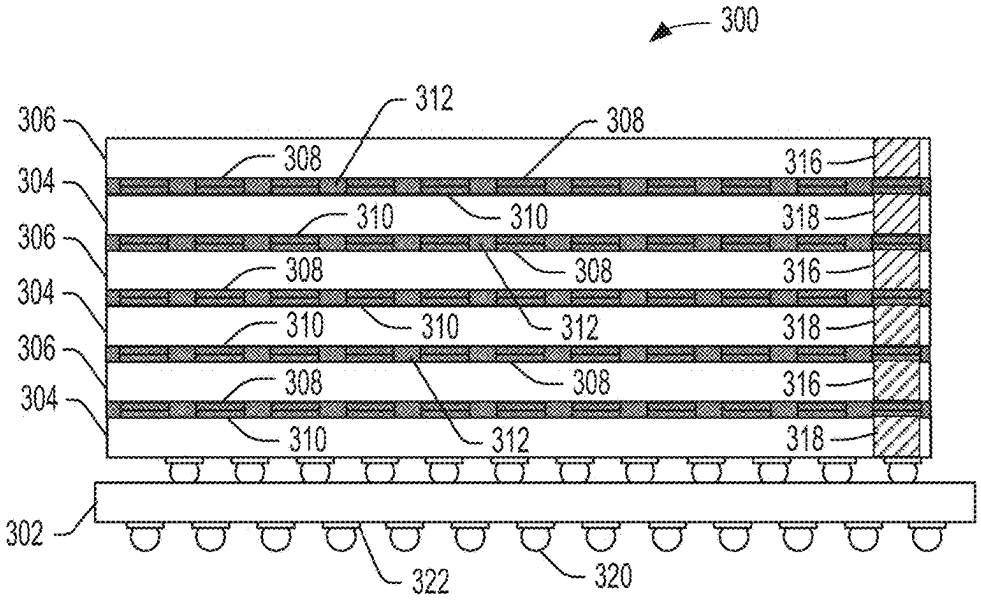

300
312
308
306
308
316
304
310
310
318
306
308
308
316
304
310
310
312
318
306
308
308
316
304
310
312
318
302
322
320

FIG. 3B

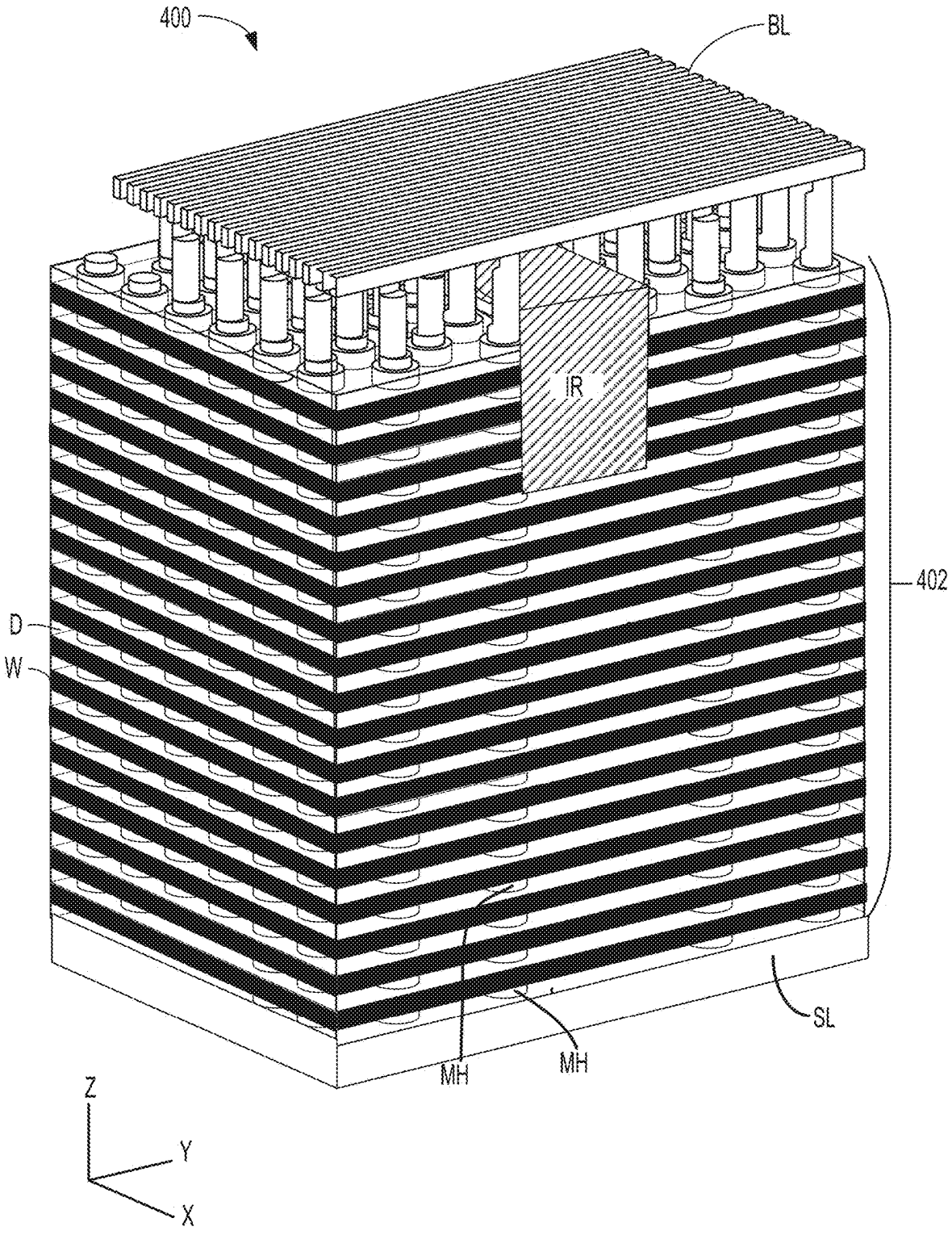


FIG. 4A

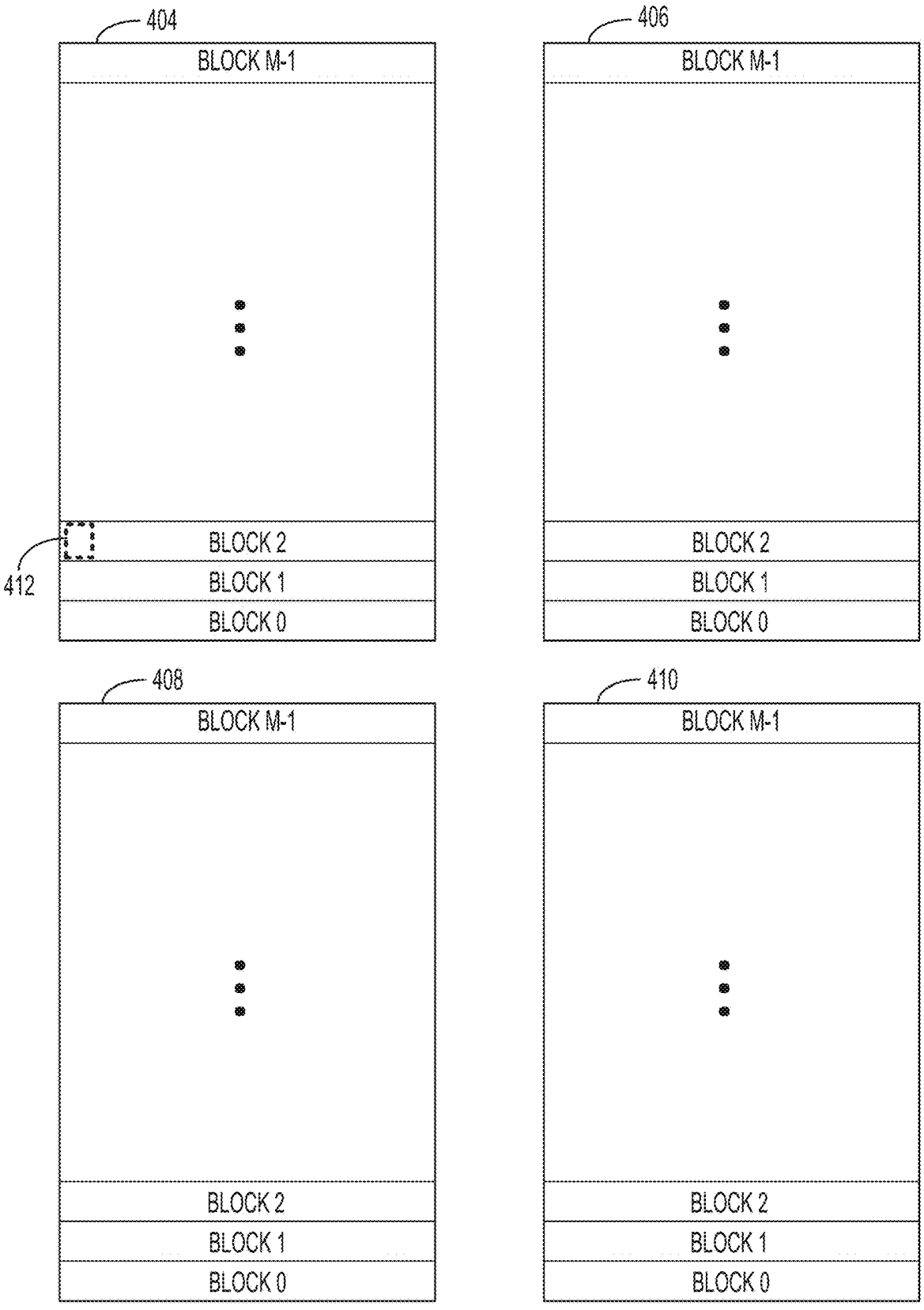

404
BLOCK M-1
BLOCK 2
BLOCK 1
BLOCK 0
412
406
BLOCK M-1
BLOCK 2
BLOCK 1
BLOCK 0
408
BLOCK M-1
BLOCK 2
BLOCK 1
BLOCK 0
410
BLOCK M-1
BLOCK 2
BLOCK 1
BLOCK 0

FIG. 4B

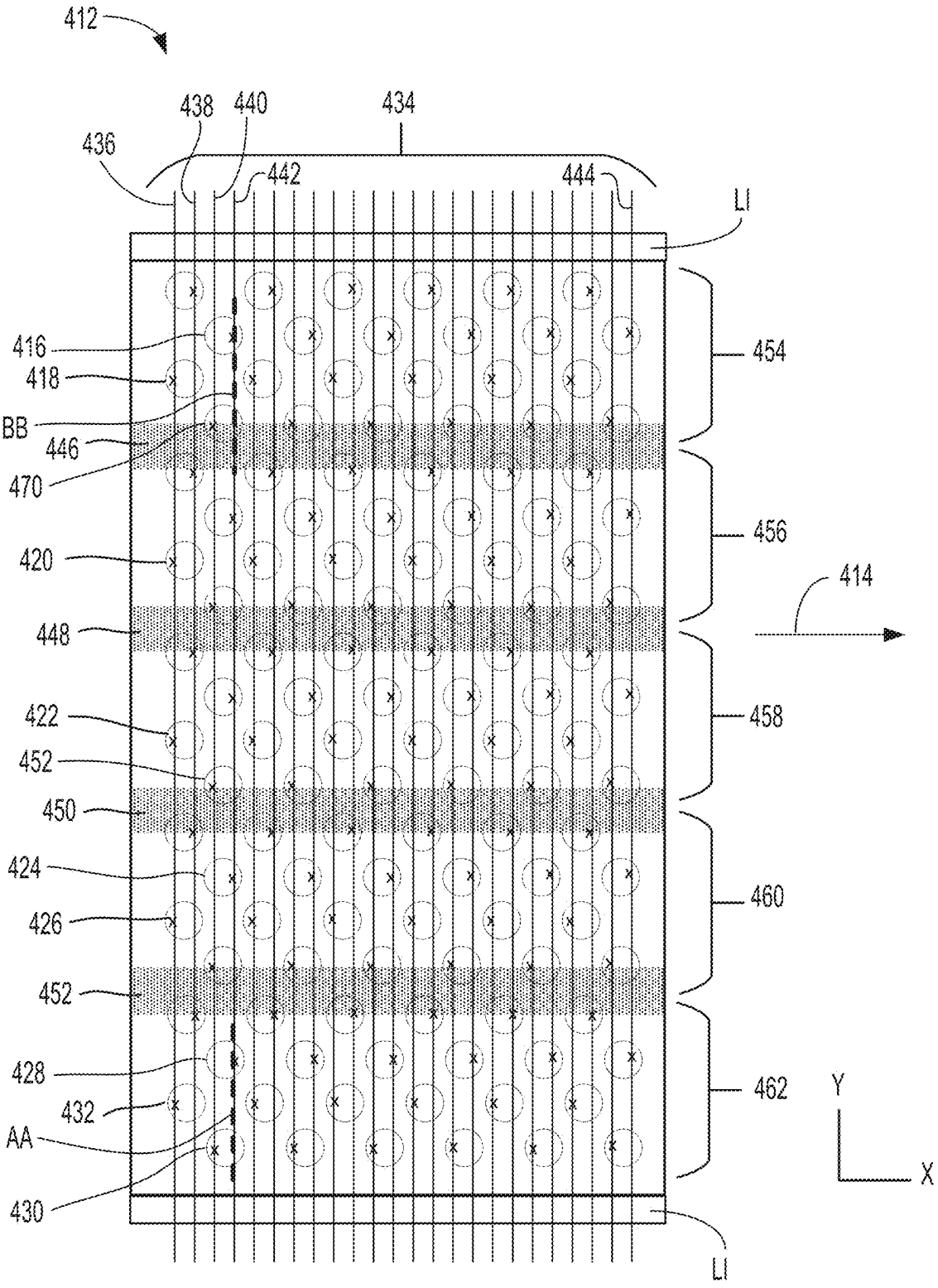
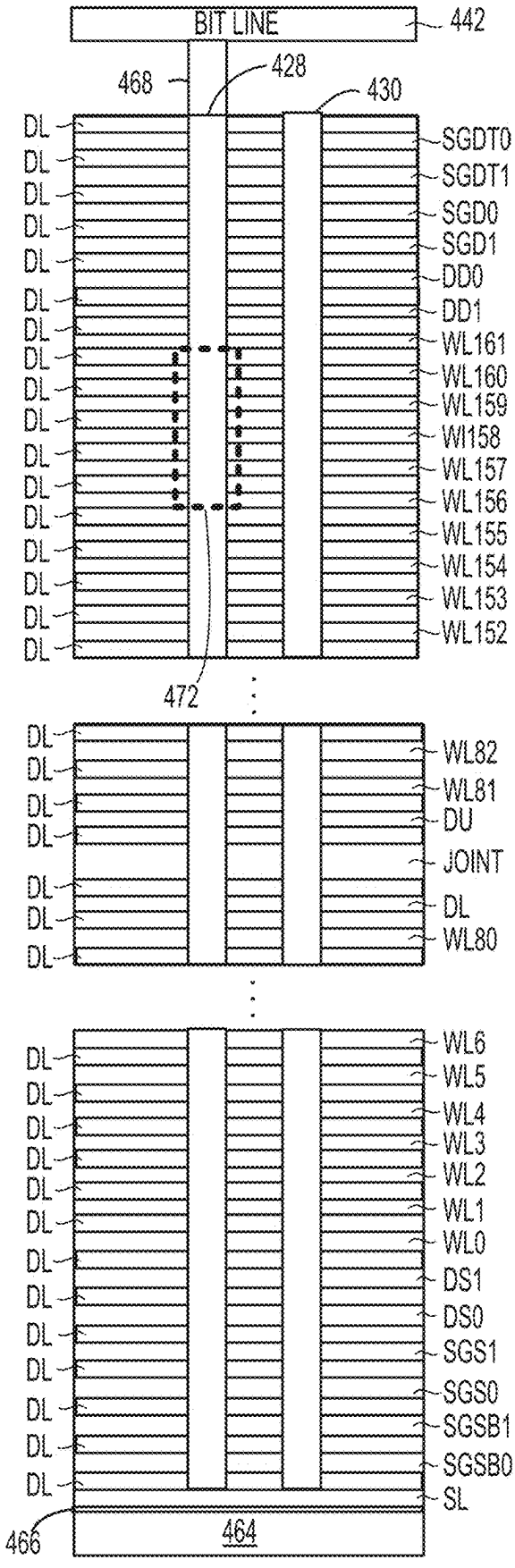

412
438
440
434
436
442
444
LI
416
418
BB
446
470
454
456
420
414
448
422
458
452
450
424
460
426
452
428
462
Y
432
AA
X
430
LI

FIG. 4C


BIT LINE
442
468
428
430
DL
SGDT0
SGDT1
SGD0
SGD1
DD0
DD1
WL161
WL160
WL159
WI158
WL157
WL156
WL155
WL154
WL153
WL152
472
WL82
WL81
DU
JOINT
DL
WL80
WL6
WL5
WL4
WL3
WL2
WL1
WL0
DS1
DS0
SGS1
SGS0
SGSB1
SGSB0
SL
464
466

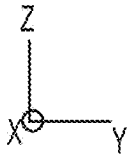

Z
X
Y

FIG. 4D

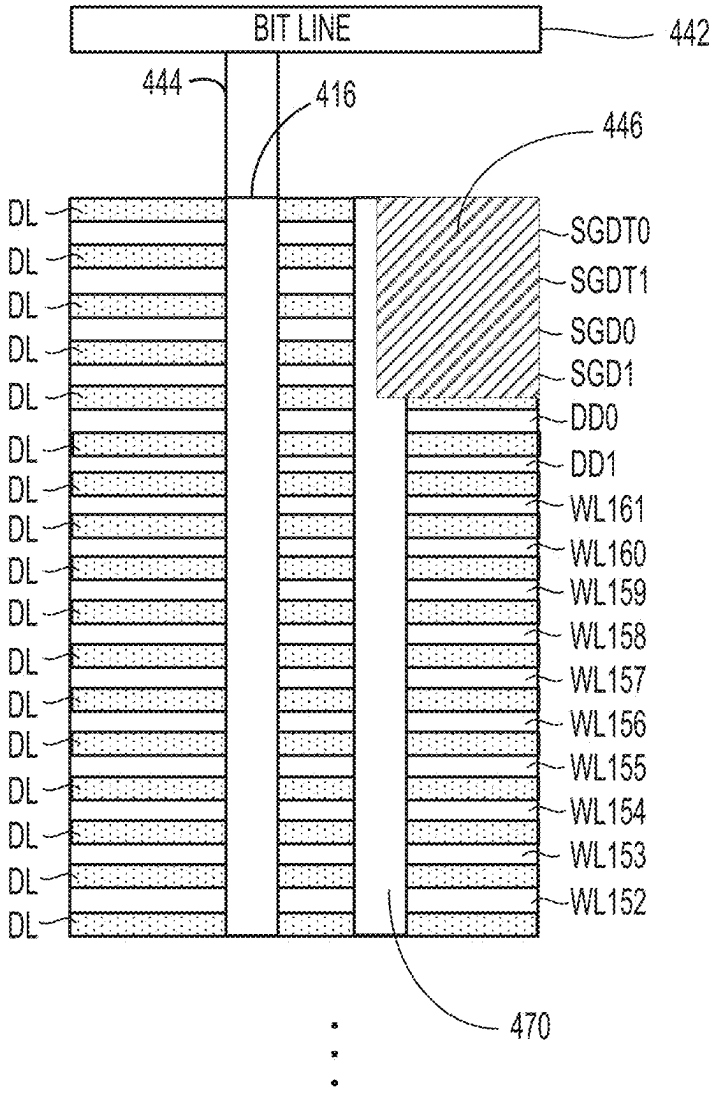

BIT LINE
442
444
416
446
DL
DL
DL
DL
DL
DL
DL
DL
DL
DL
DL
DL
DL
DL
DL
DL
DL
SGDT0
SGDT1
SGD0
SGD1
DD0
DD1
WL161
WL160
WL159
WL158
WL157
WL156
WL155
WL154
WL153
WL152
470

FIG. 4E

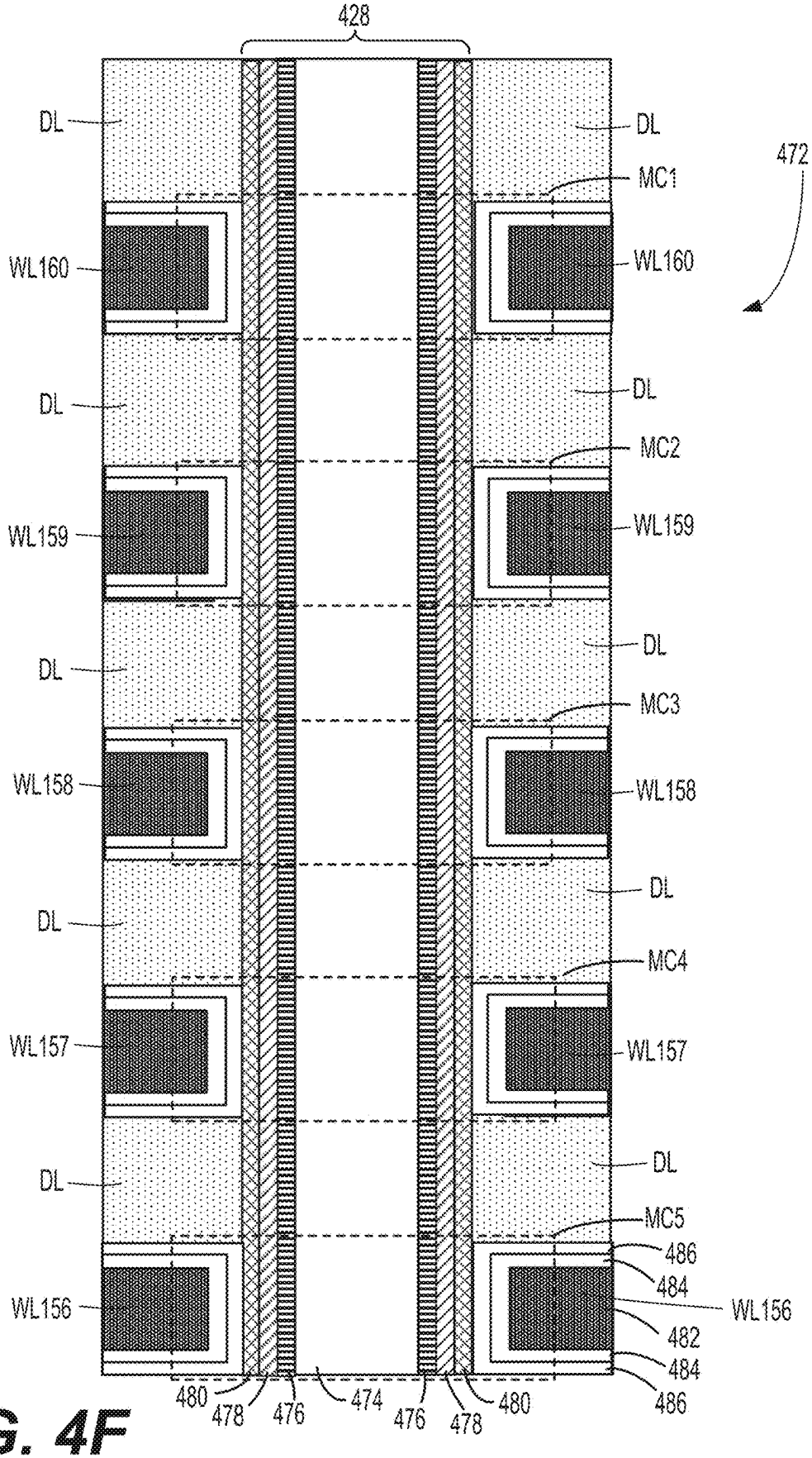

428
472
DL
MC1
WL160
MC2
WL159
MC3
WL158
MC4
WL157
MC5
WL156
486
484
482
480
478
476
474

FIG. 4F

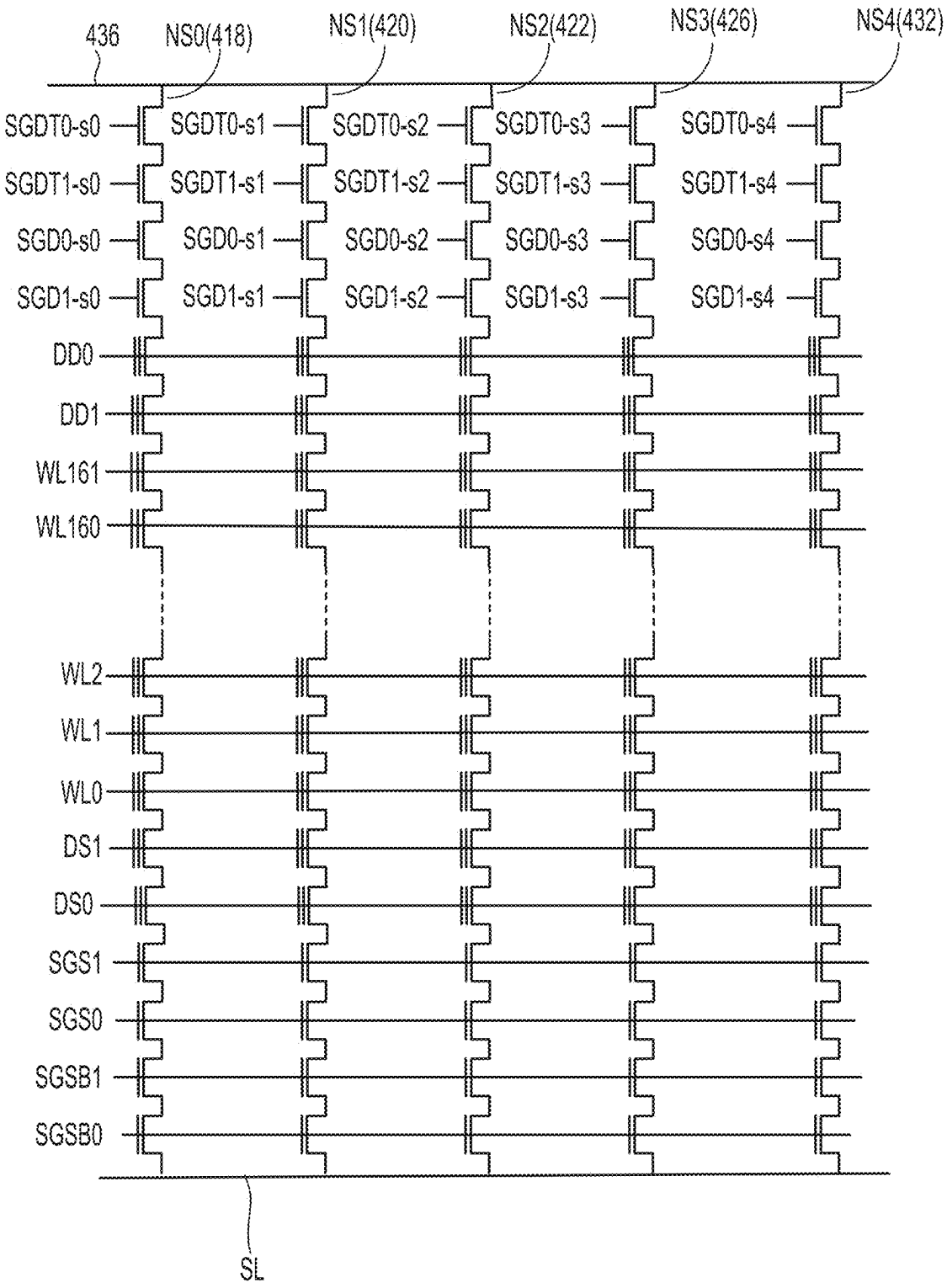

436
NS0(418)
NS1(420)
NS2(422)
NS3(426)
NS4(432)
SGDT0-s0
SGDT0-s1
SGDT0-s2
SGDT0-s3
SGDT0-s4
SGDT1-s0
SGDT1-s1
SGDT1-s2
SGDT1-s3
SGDT1-s4
SGD0-s0
SGD0-s1
SGD0-s2
SGD0-s3
SGD0-s4
SGD1-s0
SGD1-s1
SGD1-s2
SGD1-s3
SGD1-s4
DD0
DD1
WL161
WL160
WL2
WL1
WL0
DS1
DS0
SGS1
SGS0
SGSB1
SGSB0
SL

FIG. 4G

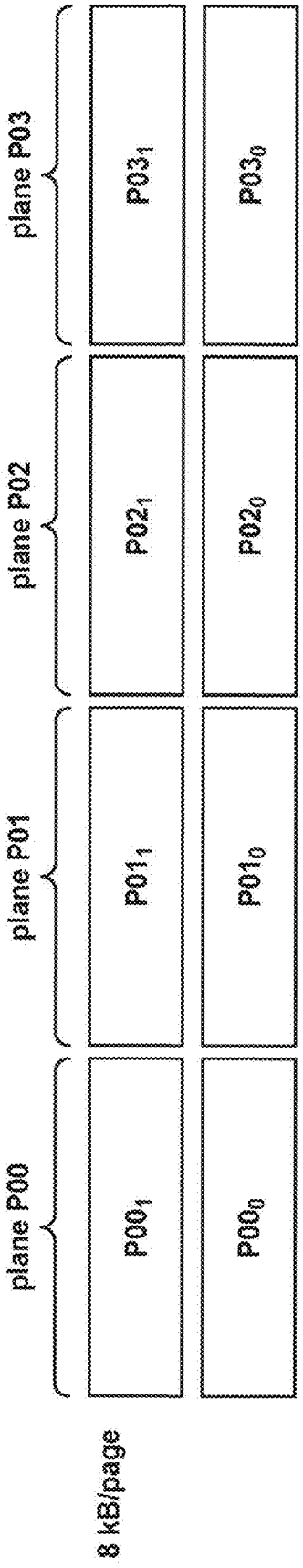

plane P00
plane P01
plane P02
plane P03
P00$_1$
P00$_0$
P01$_1$
P01$_0$
P02$_1$
P02$_0$
P03$_1$
P03$_0$
8 kB/page

FIG. 5A

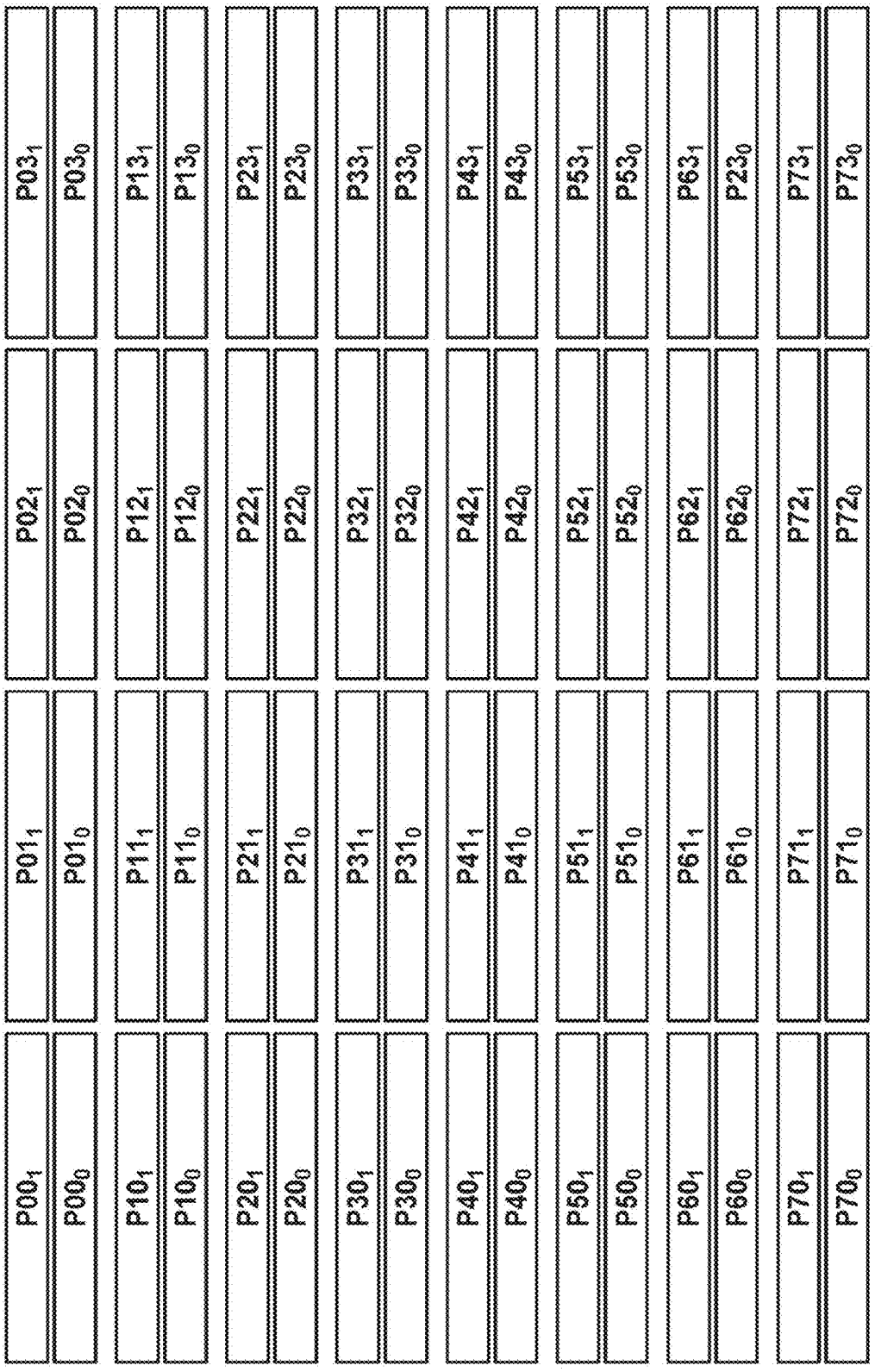

8 kB/page
P00$_1$
P00$_0$
P01$_1$
P01$_0$
P02$_1$
P02$_0$
P03$_1$
P03$_0$
P10$_1$
P10$_0$
P11$_1$
P11$_0$
P12$_1$
P12$_0$
P13$_1$
P13$_0$
P20$_1$
P20$_0$
P21$_1$
P21$_0$
P22$_1$
P22$_0$
P23$_1$
P23$_0$
P30$_1$
P30$_0$
P31$_1$
P31$_0$
P32$_1$
P32$_0$
P33$_1$
P33$_0$
P40$_1$
P40$_0$
P41$_1$
P41$_0$
P42$_1$
P42$_0$
P43$_1$
P43$_0$
P50$_1$
P50$_0$
P51$_1$
P51$_0$
P52$_1$
P52$_0$
P53$_1$
P53$_0$
P60$_1$
P60$_0$
P61$_1$
P61$_0$
P62$_1$
P62$_0$
P63$_1$
P63$_0$
P70$_1$
P70$_0$
P71$_1$
P71$_0$
P72$_1$
P72$_0$
P73$_1$
P73$_0$

FIG. 5B

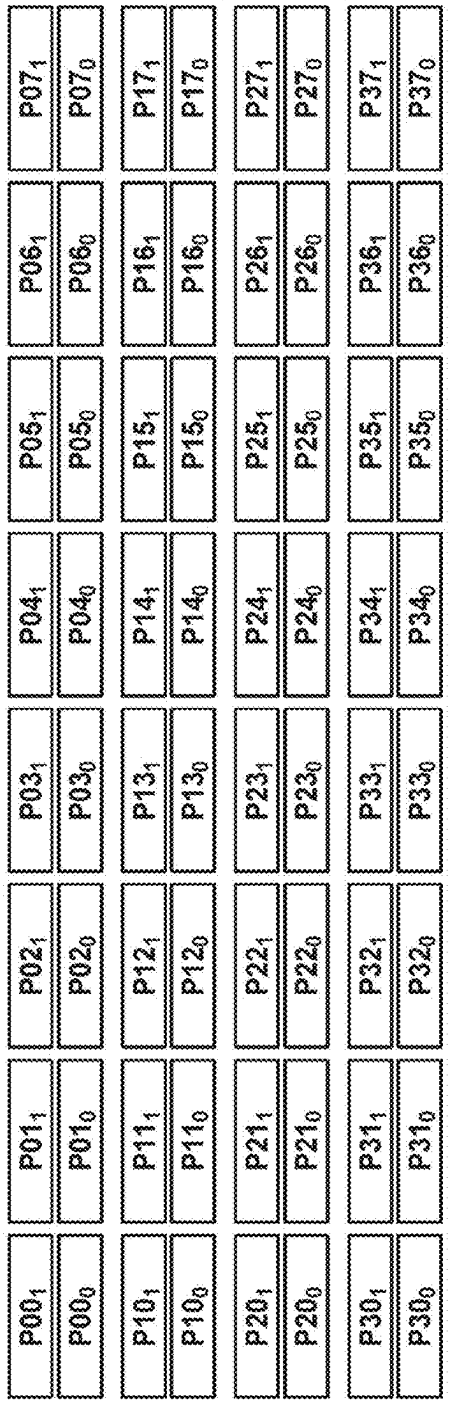

4 kB/page
P00$_1$ P01$_1$ P02$_1$ P03$_1$ P04$_1$ P05$_1$ P06$_1$ P07$_1$
P00$_0$ P01$_0$ P02$_0$ P03$_0$ P04$_0$ P05$_0$ P06$_0$ P07$_0$
P10$_1$ P11$_1$ P12$_1$ P13$_1$ P14$_1$ P15$_1$ P16$_1$ P17$_1$
P10$_0$ P11$_0$ P12$_0$ P13$_0$ P14$_0$ P15$_0$ P16$_0$ P17$_0$
P20$_1$ P21$_1$ P22$_1$ P23$_1$ P24$_1$ P25$_1$ P26$_1$ P27$_1$
P20$_0$ P21$_0$ P22$_0$ P23$_0$ P24$_0$ P25$_0$ P26$_0$ P27$_0$
P30$_1$ P31$_1$ P32$_1$ P33$_1$ P34$_1$ P35$_1$ P36$_1$ P37$_1$
P30$_0$ P31$_0$ P32$_0$ P33$_0$ P34$_0$ P35$_0$ P36$_0$ P37$_0$

FIG. 5C

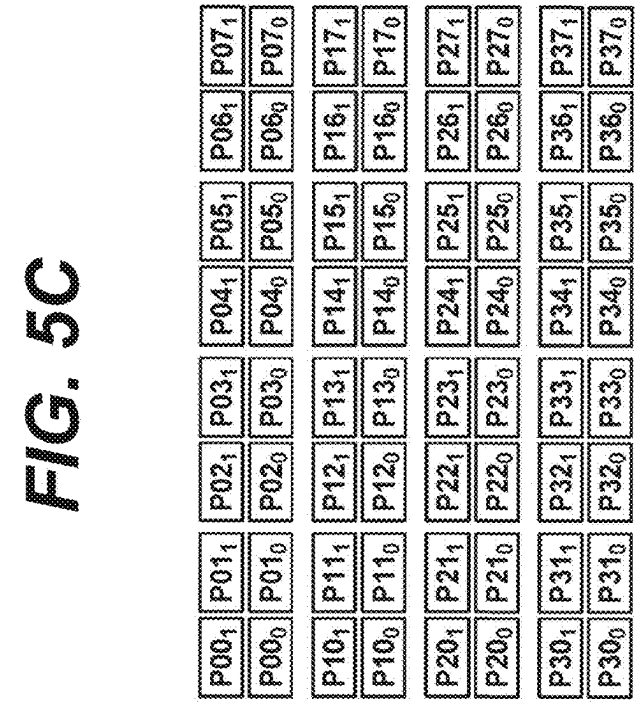

2 kB/page
P00$_1$ P01$_1$ P02$_1$ P03$_1$ P04$_1$ P05$_1$ P06$_1$ P07$_1$
P00$_0$ P01$_0$ P02$_0$ P03$_0$ P04$_0$ P05$_0$ P06$_0$ P07$_0$
P10$_1$ P11$_1$ P12$_1$ P13$_1$ P14$_1$ P15$_1$ P16$_1$ P17$_1$
P10$_0$ P11$_0$ P12$_0$ P13$_0$ P14$_0$ P15$_0$ P16$_0$ P17$_0$
P20$_1$ P21$_1$ P22$_1$ P23$_1$ P24$_1$ P25$_1$ P26$_1$ P27$_1$
P20$_0$ P21$_0$ P22$_0$ P23$_0$ P24$_0$ P25$_0$ P26$_0$ P27$_0$
P30$_1$ P31$_1$ P32$_1$ P33$_1$ P34$_1$ P35$_1$ P36$_1$ P37$_1$
P30$_0$ P31$_0$ P32$_0$ P33$_0$ P34$_0$ P35$_0$ P36$_0$ P37$_0$

FIG. 5D

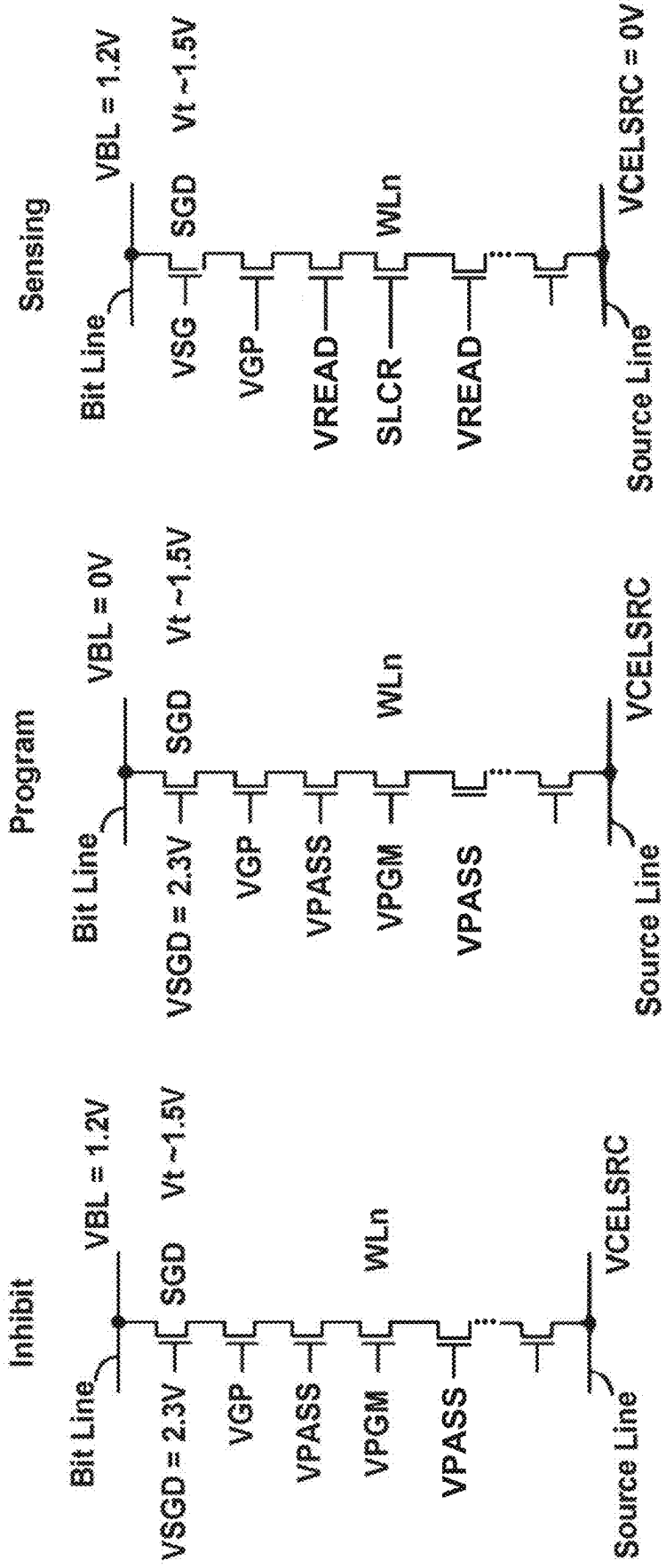


FIG. 6A

FIG. 6B

FIG. 6C

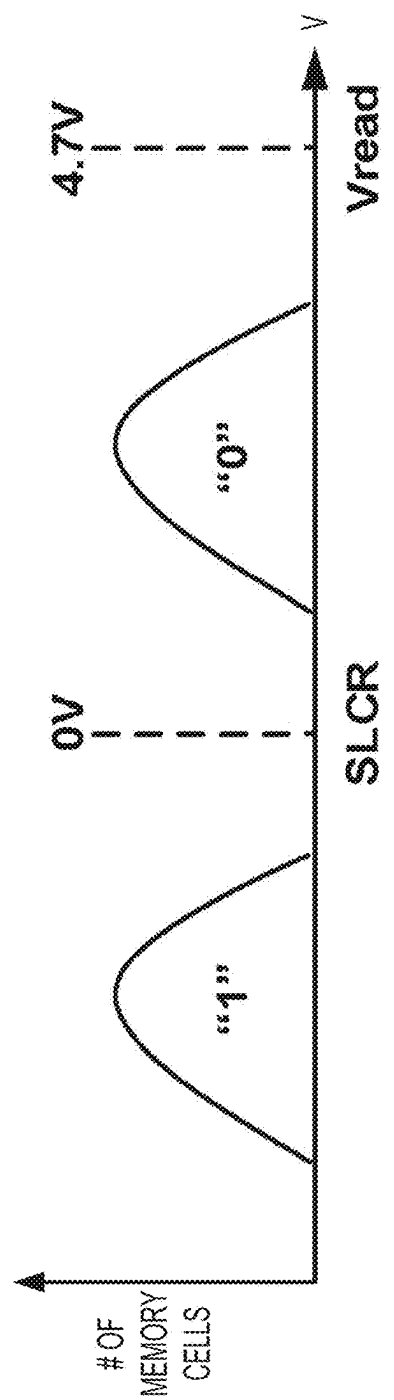

V
4.7V
Vread
0V
SLCR
"0"
"1"
OF MEMORY CELLS

FIG. 7A

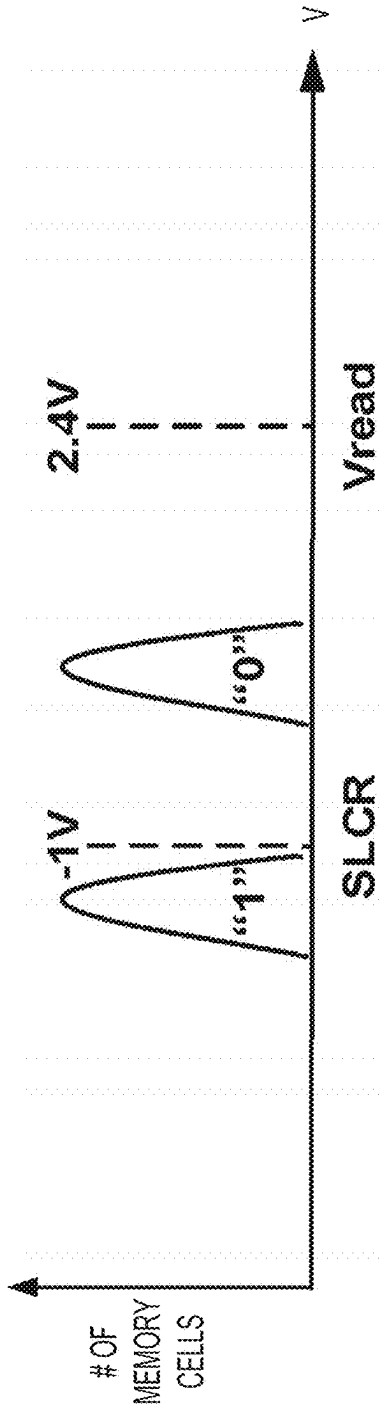

V
2.4V
Vread
-1V
SLCR
"0"
"1"
OF MEMORY CELLS

FIG. 7B

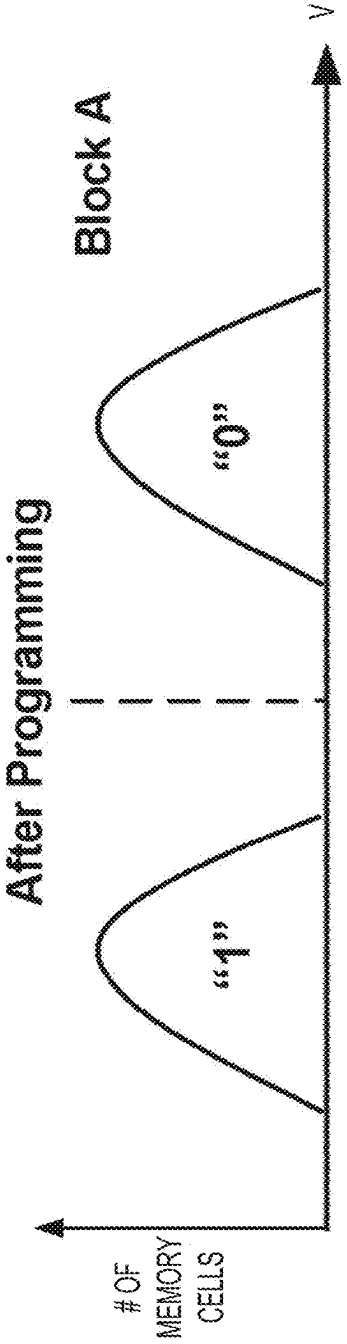

After Programming
Block A
"1"
"0"
SLCR
OF MEMORY CELLS
V

FIG. 8A

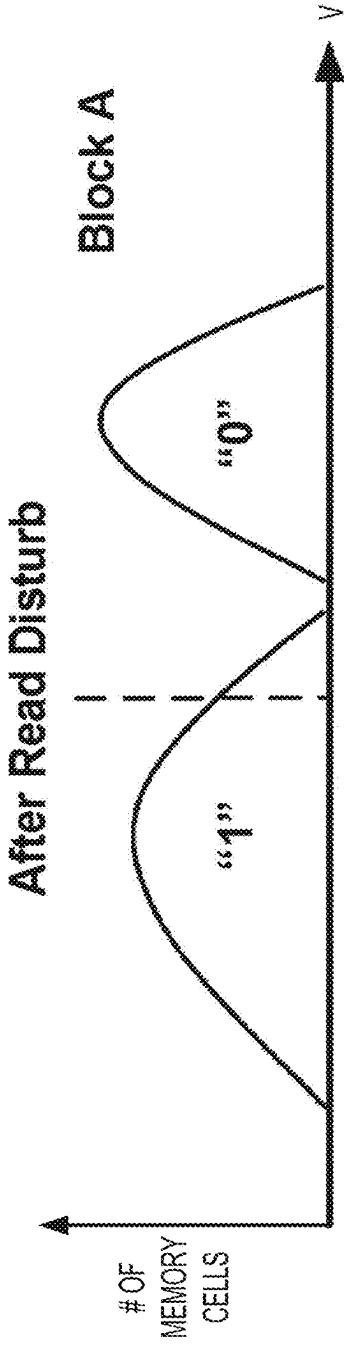

After Read Disturb
Block A
"1"
"0"
SLCR
OF MEMORY CELLS
V

FIG. 8B

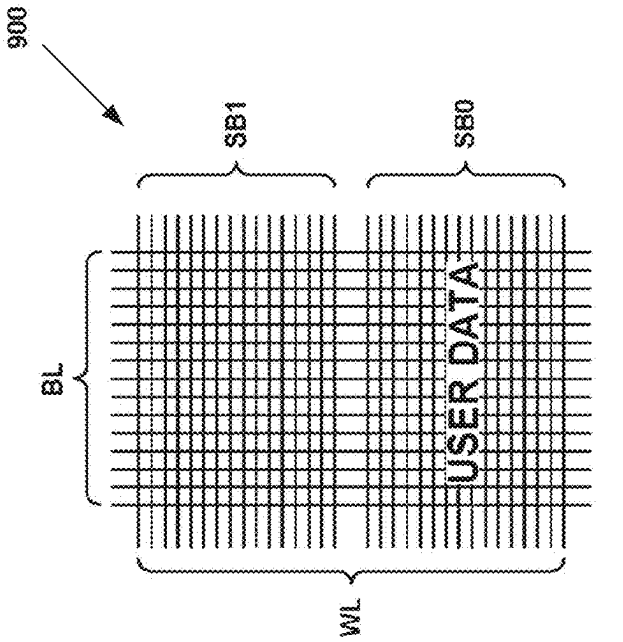


FIG. 9A

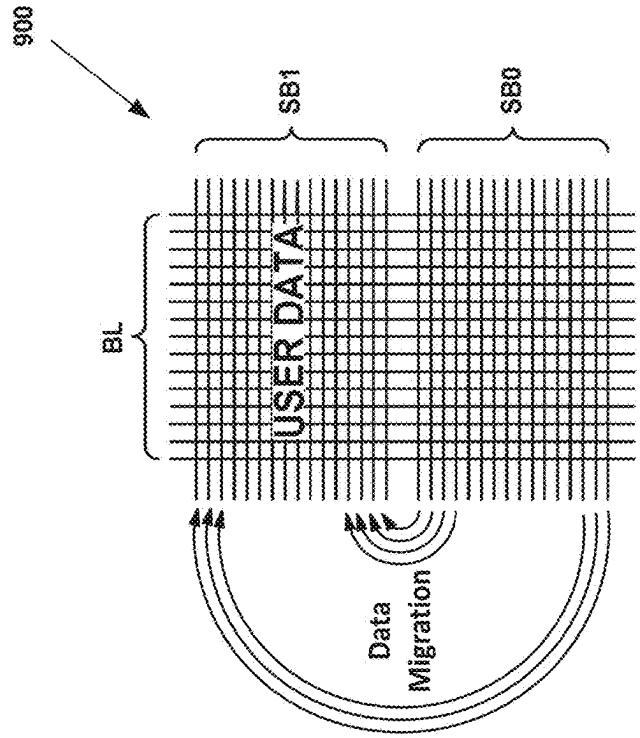


FIG. 9B

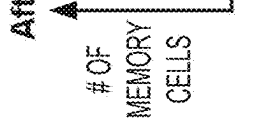


FIG. 9C

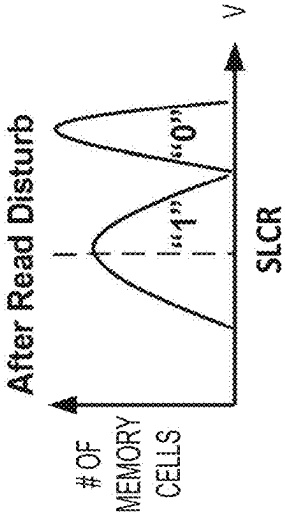


FIG. 9D

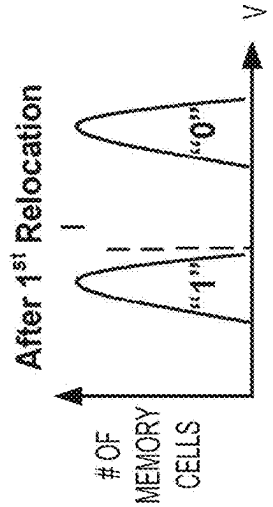


FIG. 9E

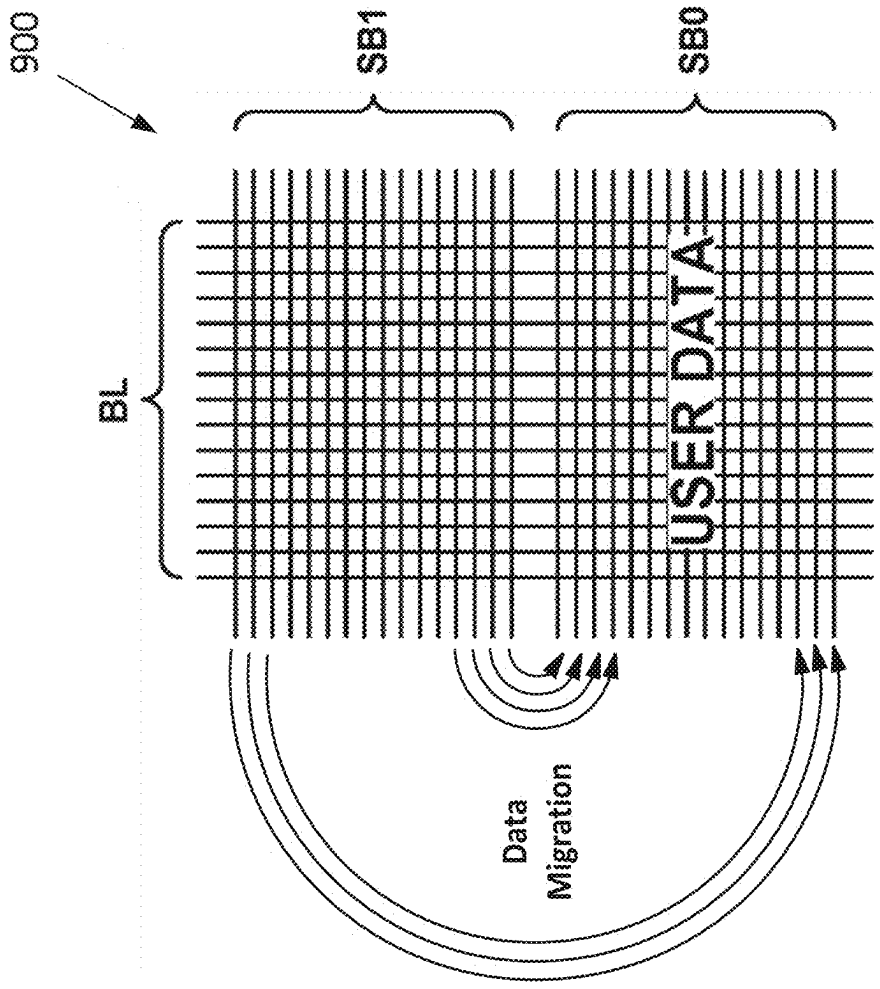


FIG. 9F

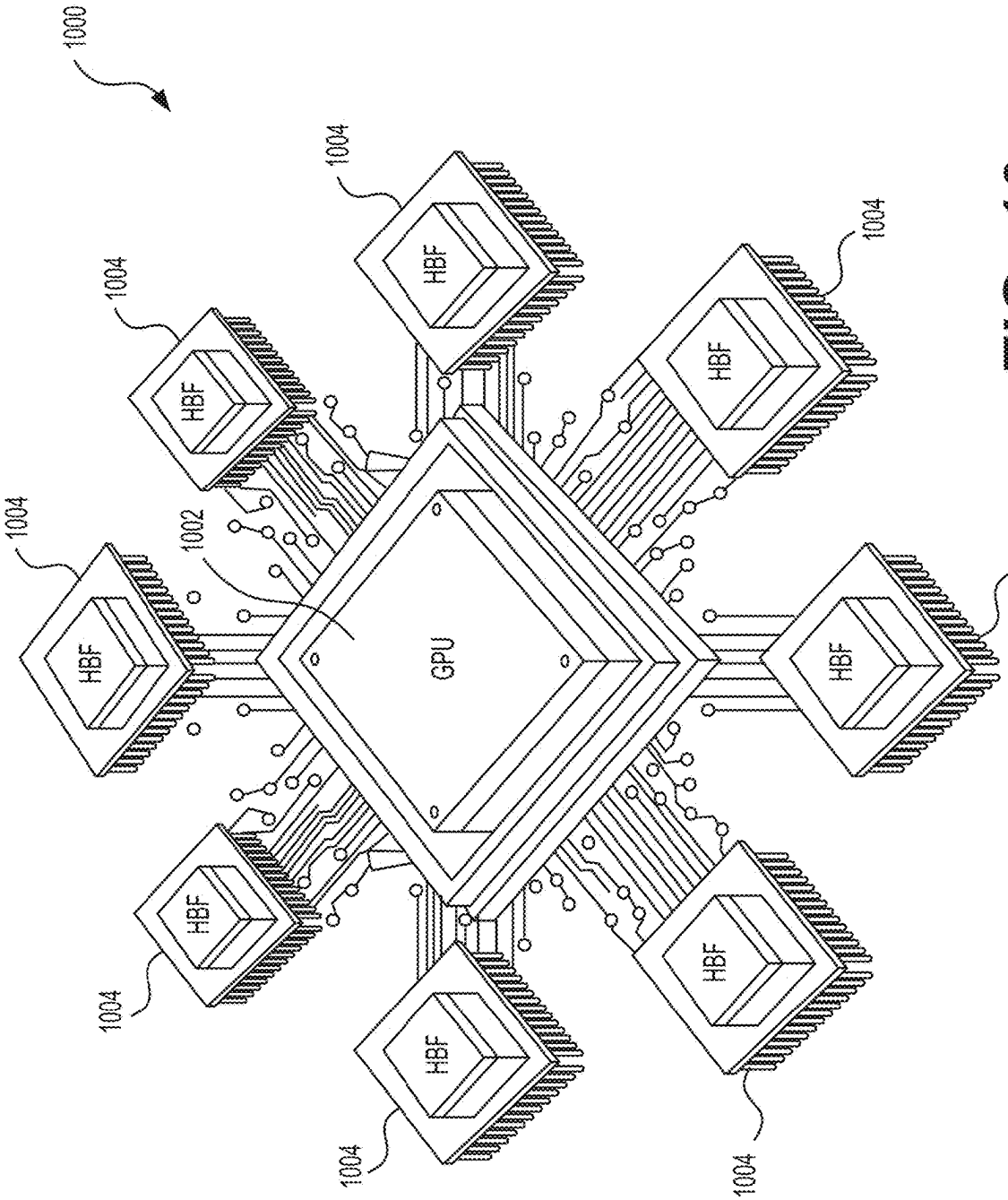


FIG. 10

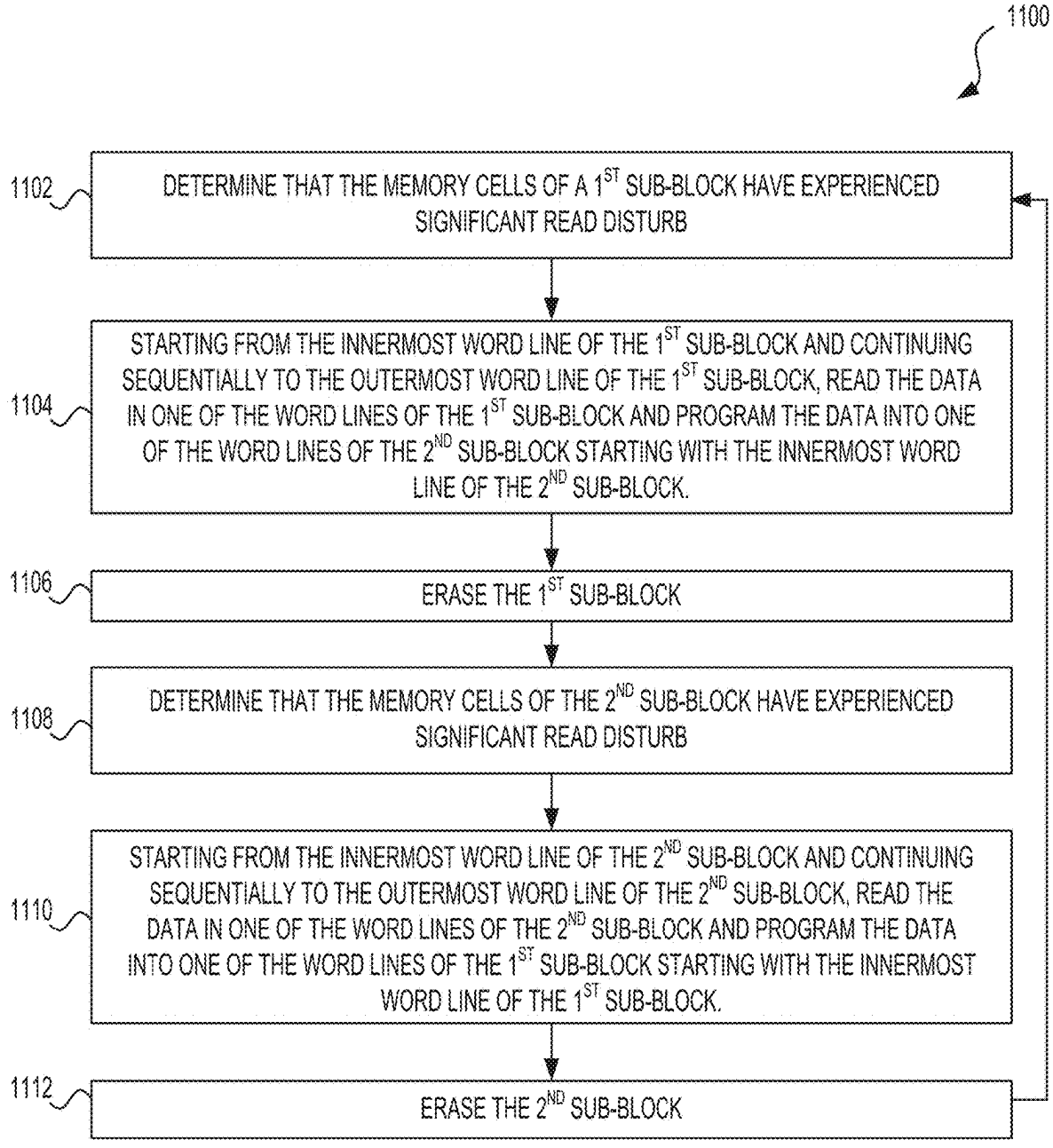

1100
1102
DETERMINE THAT THE MEMORY CELLS OF A 1ST SUB-BLOCK HAVE EXPERIENCED SIGNIFICANT READ DISTURB
1104
STARTING FROM THE INNERMOST WORD LINE OF THE 1ST SUB-BLOCK AND CONTINUING SEQUENTIALLY TO THE OUTERMOST WORD LINE OF THE 1ST SUB-BLOCK, READ THE DATA IN ONE OF THE WORD LINES OF THE 1ST SUB-BLOCK AND PROGRAM THE DATA INTO ONE OF THE WORD LINES OF THE 2ND SUB-BLOCK STARTING WITH THE INNERMOST WORD LINE OF THE 2ND SUB-BLOCK.
1106
ERASE THE 1ST SUB-BLOCK
1108
DETERMINE THAT THE MEMORY CELLS OF THE 2ND SUB-BLOCK HAVE EXPERIENCED SIGNIFICANT READ DISTURB
1110
STARTING FROM THE INNERMOST WORD LINE OF THE 2ND SUB-BLOCK AND CONTINUING SEQUENTIALLY TO THE OUTERMOST WORD LINE OF THE 2ND SUB-BLOCK, READ THE DATA IN ONE OF THE WORD LINES OF THE 2ND SUB-BLOCK AND PROGRAM THE DATA INTO ONE OF THE WORD LINES OF THE 1ST SUB-BLOCK STARTING WITH THE INNERMOST WORD LINE OF THE 1ST SUB-BLOCK.
1112
ERASE THE 2ND SUB-BLOCK

FIG. 11

APPARATUS AND METHODS FOR SUB-BLOCK READ REFRESH FOR NONVOLATILE MEMORY DEVICES

CROSS-REFERENCE TO RELATED APPLICATION

[0001] This application claims priority to and the benefit of U.S. Provisional App. No. 63/552,783, filed Feb. 13, 2024, entitled "APPARATUS AND METHODS FOR SUB-BLOCK READ REFRESH FOR NONVOLATILE MEMORY DEVICES," the entire contents of which is herein incorporated by reference.

BACKGROUND

1. Field

[0002] The present disclosure is related generally to non-volatile memory and, more particularly to improved memory devices that are optimized to operate at very high read performance and with a very low power consumption.

2. Related Art

[0003] Semiconductor memory is widely used in various electronic devices such as cellular telephones, digital cameras, personal digital assistants, medical electronics, mobile computing devices, servers, solid state drives, non-mobile computing devices and other devices. Semiconductor memory may be non-volatile memory or volatile memory. A non-volatile memory allows information to be stored and retained even when the non-volatile memory is not connected to a source of power (e.g., a battery).

[0004] Non-volatile memory devices include one or more memory chips having multiple arrays of memory cells. The memory arrays may have associated decoders and circuits for performing read, write, and erase operations. Memory cells within the arrays may be arranged in horizontal rows and vertical columns. Each row may be addressed by a word line, and each column may be addressed by a bit line. Data may be loaded into columns of the array using a series of data busses. Each column may hold a predefined unit of data, for instance, a word encompassing two bytes of information.

[0005] In some applications, semiconductor memory is used to store very large amounts of data that are repeatedly accessed (e.g., read) very rapidly. For example, in some machine learning applications, large language models that include a terabyte (or more) of data must be stored in memory and retrieved at a very high data rate. Accordingly, such applications require very high bandwidth and low power.

[0006] Currently, high bandwidth volatile memory devices (e.g., DRAM memory devices called "high bandwidth memory" or "HBM") are used for such applications. Non-volatile memory (e.g., NAND) is significantly less expensive than DRAM, but the bandwidth of conventional NAND memory devices is too low, and the power consumption of conventional NAND memory devices is too high to provide a viable alternative to HBM devices. Therefore, there is a need to provide high bandwidth, low power non-volatile memory.

SUMMARY

[0007] One aspect of the present disclosure is related to a method of operating a memory device. The method includes the step of preparing a memory device that has a memory block with an array of memory cells. The memory cells are arranged in a plurality of word lines, and the plurality of word lines are divided into a first sub-block and a second sub-block. The memory cells of the first sub-block contain data, and the memory cells of the second sub-block are erased (do not contain data). The method continues with the step of determining that the memory cells of the first sub-block have experienced significant read disturb. The method proceeds with the step of programming the data in the memory cells of the first sub-block into the memory cells of the second sub-block.

[0008] According to another aspect of the present disclosure, the method further includes the step of counting a number of read cycles to establish a read cycle count. The method proceeds with the step of comparing the read count cycle to a predetermined threshold. The step of determining that the memory cells of the first sub-block have experienced significant read disturb occurs in response to the read cycle count exceeding the predetermined threshold.

[0009] According to yet another aspect of the present disclosure, the memory block has a source side and a drain side. Prior to the step of programming the data in the memory cells of the first sub-block to the memory cells of the second sub-block, the data has a first order within the first sub-block. After step of programming the data in the memory cells of the first sub-block to the memory cells of the second sub-block, the data has a second order within the second sub-block. The second order is opposite of the first order.

[0010] According to still another aspect of the present disclosure, the method further includes the step of erasing the memory cells of the first sub-block. The method proceeds with the step of determining that the memory cells of the second sub-block have experienced significant read disturb. The method continues with the step of programming the user data in the memory cells of the second sub-block into the memory cells of the first sub-block.

[0011] According to a further aspect of the present disclosure, after the step of programming the user data in the memory cells of the second sub-block into the memory cells of the first sub-block, the data has the first order.

[0012] According to yet a further aspect of the present disclosure, the first sub-block is a lower sub-block and wherein the second sub-block is an upper sub-block. The step of programming the user data in the memory cells of the lower sub-block into the memory cells of the upper sub-block includes programming according to a normal order programming direction.

[0013] According to still a further aspect of the present disclosure, the step of programming the user data in the memory cells of the upper sub-block into the memory cells of the lower sub-block includes programming according to a reverse order programming direction.

[0014] Another aspect of the present disclosure is related to a memory device that includes a memory block with an array of memory cells that are arranged in a plurality of word lines. The word lines are divided into a first sub-block and a second sub-block with the memory cells of the first sub-block containing data and with the memory cells of the second sub-block being erased. The memory device also includes circuitry that is configured to determine that the memory cells of the first sub-block have experienced significant read disturb. The circuitry is also configured to

program the user data in the memory cells of the first sub-block into the memory cells of the second sub-block.

[0015] According to another aspect of the present disclosure, the circuitry is further configured to count a number of read cycles to establish a read cycle count and to compare the read count cycle to a predetermined threshold. The circuitry determines that the memory cells of the first sub-block have experienced significant read disturb in response to the read cycle count exceeding the predetermined threshold.

[0016] According to yet another aspect of the present disclosure, the memory block has a source side and a drain side. Prior to programming the user data in the memory cells of the first sub-block to the memory cells of the second sub-block, the data has a first order within the first sub-block. After programming the user data in the memory cells of the first sub-block to the memory cells of the second sub-block, the data has a second order within the second sub-block. The second order is opposite of the first order.

[0017] According to still another aspect of the present disclosure, the circuitry is further configured to erase the memory cells of the first sub-block and determine that the memory cells of the second sub-block have experienced significant read disturb. The circuitry is configured to program the data in the memory cells of the second sub-block into the memory cells of the first sub-block.

[0018] According to a further aspect of the present disclosure, after programming the user data in the memory cells of the second sub-block into the memory cells of the first sub-block, the data has the first order.

[0019] According to yet a further aspect of the present disclosure, the first sub-block is a lower sub-block and the second sub-block is an upper sub-block. The circuitry is configured to program the user data in the memory cells of the lower sub-block into the memory cells of the upper sub-block according to a normal order programming direction.

[0020] According to still a further aspect of the present disclosure, the circuitry is configured to program the user data in the memory cells of the upper sub-block into the memory cells of the lower sub-block according to a reverse order programming direction.

[0021] Yet another aspect of the present disclosure is related to a computing system. The computing system includes a processor unit. The computing system also includes a plurality of high bandwidth flash (HBF) packages that are in electrical communication with the processor unit. At least one of the HBF packages includes a memory block with an array of memory cells that are arranged in a plurality of word lines. The word lines are divided into a first sub-block and a second sub-block. The memory cells of the first sub-block contain data, and the memory cells of the second sub-block are erased (do not contain data). The at least one of the HBF packages further includes circuitry that is configured to determine that the memory cells of the first sub-block have experienced significant read disturb. The circuitry is also configured to program the data in the memory cells of the first sub-block into the memory cells of the second sub-block.

[0022] According to another aspect of the present disclosure, the circuitry is further configured to count a number of read cycles to establish a read cycle count. The circuitry is also configured to compare the read count cycle to a predetermined threshold, and the circuitry determines that the memory cells of the first sub-block have experienced significant read disturb in response to the read cycle count exceeding the predetermined threshold.

[0023] According to yet another aspect of the present disclosure, the memory block has a source side and a drain side. Prior to programming the user data in the memory cells of the first sub-block to the memory cells of the second sub-block, the data has a first order within the first sub-block. After programming the user data in the memory cells of the first sub-block to the memory cells of the second sub-block, the data has a second order within the second sub-block. The second order is opposite of the first order.

[0024] According to still another aspect of the present disclosure, the circuitry is further configured to erase the memory cells of the first sub-block and determine that the memory cells of the second sub-block have experienced significant read disturb. The circuity is then configured to program the user data in the memory cells of the second sub-block into the memory cells of the first sub-block.

[0025] According to a further aspect of the present disclosure, after programming the user data in the memory cells of the second sub-block into the memory cells of the first sub-block, the data has the first order.

[0026] According to still a further aspect of the present disclosure, the first sub-block is a lower sub-block and wherein the second sub-block is an upper sub-block. The circuitry is configured to program the user data in the memory cells of the lower sub-block into the memory cells of the upper sub-block according to a normal order programming direction.

BRIEF DESCRIPTION OF THE DRAWINGS

[0027] These and other features and advantages of the subject disclosure will become more readily appreciated when considered in connection with the following description of the presently preferred embodiments, appended claims and accompanying drawings, in which:

[0028] FIG. **1** is a block diagram depicting one embodiment of a storage system;

[0029] FIG. **2**A is a block diagram of one embodiment of a memory die;

[0030] FIG. **2**B is a block diagram of one embodiment of an integrated memory assembly;

[0031] FIGS. **3**A and **3**B depict different embodiments of integrated memory assemblies;

[0032] FIG. **4**A is a perspective view of a portion of one embodiment of a monolithic three dimensional memory structure;

[0033] FIG. **4**B is a block diagram of one embodiment of a memory structure having four planes;

[0034] FIG. **4**C depicts a top view of a portion of one embodiment of a block of memory cells;

[0035] FIG. **4**D depicts a cross sectional view of a portion of one embodiment of a block of memory cells;

[0036] FIG. **4**E depicts a cross sectional view of a portion of one embodiment of a block of memory cells;

[0037] FIG. **4**F is a cross sectional view of one embodiment of a vertical column of memory cells;

[0038] FIG. **4**G is a schematic of a plurality of NAND strings in multiple regions of a same block;

[0039] FIG. **5**A is a block diagram of one embodiment of a memory structure having four planes;

[0040] FIG. **5**B is a block diagram of one embodiment of a memory structure having thirty-two planes;

[0041] FIG. **5**C is a block diagram of another embodiment of a memory structure having thirty-two planes;

[0042] FIG. **5**D is a block diagram of still another embodiment of a memory structure having thirty-two planes;

[0043] FIGS. **6**A-**6**C depict an example NAND string during inhibit, program and sensing, respectively;

[0044] FIG. **7**A depicts an example threshold voltage distribution of a page of NAND memory cells;

[0045] FIG. **7**B depicts another example threshold voltage distribution of a page of NAND memory cells;

[0046] FIG. **8**A depicts a threshold voltage distribution of a page of memory cells initially after programming;

[0047] FIG. **8**B is depicts a threshold voltage distribution of the page of memory cells after experiencing significant read disturb;

[0048] FIG. **9**A is a schematic view of an example memory block that is been divided into a pair of sub-blocks;

[0049] FIG. **9**B is a schematic view showing data in a first sub-block being re-located into a second sub-block;

[0050] FIG. **9**C depicts a threshold voltage distribution of a page of memory cells in the first sub-block after programming;

[0051] FIG. **9**D depicts a threshold voltage distribution of the page of memory cells in the first sub-block after experiencing significant read disturb;

[0052] FIG. **9**E depicts a threshold voltage distribution of a page of memory cells containing the same data but now in the second sub-block;

[0053] FIG. **9**F is a schematic view showing data in the second sub-block being re-located back to the first sub-block;

[0054] FIG. **10** is a schematic view depicting an example computing system that includes a single processing unit and a plurality of high bandwidth flash (HBF) units; and

[0055] FIG. **11** is a flow chart depicting the steps of re-locating data between sub-blocks in a memory block according to an example embodiment.

DETAILED DESCRIPTION OF THE ENABLING EMBODIMENT

[0056] Technology is described for increasing the bandwidth and improving the power efficiency of NAND memory to provide a viable alternative to HBM devices. More specifically, as discussed in further detail below, a high bandwidth flash (HBF) package is provided that is specifically configured to operate at both a very high bandwidth (specifically, very high read performance) and with very low power consumption. The HBF package is thus optimized for use in large language model operations. Additionally, as described in further detail below, the HBF packages are configured to automatically refresh the data according stored therein according to a sub-block read refresh technique to protect against read disturb.

[0057] FIG. **1** is a block diagram of one embodiment of a storage system **100** that implements the proposed technology described herein. In one embodiment, the storage system **100** is a solid state drive ("SSD"). The storage system **100** also can be a memory card, a USB drive, or any other type of storage system. In other words, the proposed technology is not limited to any one type of memory system.

[0058] The storage system **100** is connected to a host **102**, which can be a computer; server; electronic device (e.g., smart phone, tablet or other mobile device); appliance; or another apparatus that uses memory and has data processing capabilities. In some embodiments, the host **102** is separate from, but connected to, the storage system **100**. In other embodiments, the storage system **100** is embedded within the host **102**.

[0059] The components of the storage system **100** depicted in FIG. **1** are electrical circuits. The storage system **100** includes a memory controller **104** connected to non-volatile memory **106** and local high speed volatile memory **108** (e.g., DRAM). A local high speed volatile memory **108** is used by memory controller **104** to perform certain functions. For example, the local high speed volatile memory **108** stores logical to physical address translation tables ("L2P tables").

[0060] The memory controller **104** includes a host interface **110** that is connected to and in communication with the host **102**. In one embodiment, a host interface **110** implements an NVM Express (NVMe) over PCI Express (PCIe). Other interfaces can also be used, such as SCSI, SATA, etc. The host interface **110** also is connected to a network-on-chip (NOC) **112**.

[0061] An NOC is a communication subsystem on an integrated circuit. The NOC's can span synchronous and asynchronous clock domains or use un-clocked asynchronous logic. NOC technology applies networking theory and methods to on-chip communications and brings notable improvements over conventional bus and crossbar interconnections. The NOC improves the scalability of systems on a chip (SoC) and the power efficiency of complex SoCs compared to other designs.

[0062] The wires and the links of the NOC are shared by many signals. A high level of parallelism is achieved because all links in the NOC can operate simultaneously on different data packets. Therefore, as the complexity of integrated subsystems keep growing, a NOC provides enhanced performance (such as throughput) and scalability in comparison with previous communication architectures (e.g., dedicated point-to-point signal wires, shared buses, or segmented buses with bridges). In other embodiments, the NOC **112** can be replaced by a bus.

[0063] Connected to and in communication with NOC **112** is a processor **114**, an ECC engine **116**, a memory interface **118**, and a DRAM controller **120**. The DRAM controller **120** is used to operate and communicate with local high speed volatile memory **108** (e.g., DRAM). In other embodiments, the local high speed volatile memory **108** can be SRAM or another type of volatile memory.

[0064] In operation, the processor **114** performs the various controller memory operations, such as programming, erasing, reading, and memory management processes. In one embodiment, the processor **114** is programmed by firmware. In other embodiments, the processor **114** is a custom and dedicated hardware circuit without any software. The processor **114** also implements a translation module, as a software/firmware process or as a dedicated hardware circuit.

[0065] In many systems, the non-volatile memory is addressed internally to the storage system using physical addresses associated with one or more memory dies. However, the host system will use logical addresses to address the various memory locations. This enables the host to assign data to consecutive logical addresses, while the storage system is free to store the data as it wishes among the locations of the one or more memory dies. To implement this system, the memory controller **104** (e.g., the translation

module) performs address translation between the logical addresses used by the host and the physical addresses used by the memory dies.

[0066] One example implementation is to maintain tables (i.e., the L2P tables referenced above) that identify the current translation between logical addresses and physical addresses. An entry in the L2P table may include an identification of a logical address and corresponding physical address. Although logical address to physical address tables (or L2P tables) include the word "tables" they need not literally be tables. Rather, the logical address to physical address tables (or L2P tables) can be any type of data structure. In some examples, the memory space of a storage system is so large that the local memory **108** cannot hold all of the L2P tables. In such a case, the entire set of L2P tables are stored in non-volatile memory **106** and a subset of the L2P tables are cached (L2P cache) in the local high speed volatile memory **108**.

[0067] The ECC engine **116** performs error correction services. For example, the ECC engine **116** performs data encoding and decoding, as per an implemented ECC technique. In one embodiment, the ECC engine **116** is an electrical circuit programmed by software. For example, the ECC engine **116** can be a processor that can be programmed. In other embodiments, the ECC engine **116** is a custom and dedicated hardware circuit without any software. In another embodiment, the function of ECC engine **116** is implemented by the processor **114**.

[0068] The memory interface **118** communicates with the non-volatile memory **106**. In one embodiment, the memory interface provides a Toggle Mode interface. However, other interfaces also can be used. In some example implementations, the memory interface **118** (or another portion of the controller **104**) implements a scheduler and buffer for transmitting data to and receiving data from one or more memory die.

[0069] In one embodiment, the non-volatile memory **106** includes one or more memory die. FIG. **2**A is a functional block diagrams of one embodiment of a memory die **200** that includes the non-volatile memory **106**. Each of the one or more memory dies of non-volatile memory **106** can be implemented as the memory die **200** of FIG. **2**A. The components depicted in FIG. **2**A are electrical circuits.

[0070] The memory die **200** includes a memory array **202** that can include non-volatile memory cells, as described in further detail below. The memory array **202** includes a plurality of layers of word lines that are organized as rows, and a plurality of layers of bit lines that are organized as columns. However, other orientations can also be implemented.

[0071] The memory die **200** also includes row control circuitry **204**, whose outputs **206** are connected to respective word lines of the memory array **202**. In operation, the row control circuitry **204** receives a group of M row address signals and one or more various control signals from a system control logic circuit **208** and may include such circuits as row decoders **210**, array terminal drivers **212**, and block select circuitry **214** for both reading and writing (programming) operations.

[0072] The row control circuitry **204** also may include read/write circuitry. The memory die **200** also includes column control circuitry **216** including sense amplifier(s) **218** whose input/outputs **220** are connected to respective bit lines of the memory array **202**. Although only a single block is shown for memory array **202**, the memory die **200** can include multiple arrays that can be individually accessed.

[0073] The column control circuitry **216** receives a group of N column address signals and one or more various control signals from system control logic **208**. The column control circuitry **216** may also include such circuits as column decoders **222**; array terminal receivers or driver circuits **224**; block select circuitry **226**; read/write circuitry; and I/O multiplexers.

[0074] The system control logic **208** receives data and commands from memory controller **104** (FIG. **1**) and provides output data and status to host **102**. In some embodiments, the system control logic **208**, which includes one or more electrical circuits, includes a state machine **228** that provides die-level control of memory operations. In one embodiment, the state machine **228** is programmable by software. In other embodiments, the state machine **228** does not use software and is completely implemented in hardware (e.g., electrical circuits). In another embodiment, the state machine **228** is replaced by a micro-controller or microprocessor, either on or off the memory chip.

[0075] The system control logic **208** also can include a power control module **230** that controls the power and voltages supplied to the rows and columns of memory structure **202** during memory operations and may include charge pumps and regulator circuits for creating regulating voltages. The system control logic **208** also includes storage **232** (e.g., RAM, registers, latches, etc.), which may be used to store parameters for operating memory array **202**.

[0076] In operation, commands and data are transferred between the memory controller **104** and the memory die **200** via a memory controller interface **234** (also referred to as a "communication interface"). The memory controller interface **234** is an electrical interface for communicating with memory controller **104**. Examples of the memory controller interface **234** include a Toggle Mode Interface and an Open NAND Flash Interface (ONFI). Other I/O interfaces can also be used in other embodiments.

[0077] In an embodiment, the system control logic **208** also includes column replacement control circuits **236**, described in more detail below.

[0078] In some embodiments, all elements of the memory die **200**, including the system control logic **208**, can be formed as part of a single die. In other embodiments, some or all of the system control logic **208** can be formed on a different die.

[0079] In one embodiment, the memory structure **202** comprises a three-dimensional memory array of non-volatile memory cells in which multiple memory levels are formed above a single substrate, such as a wafer. The memory structure **202** may include any type of non-volatile memory that are monolithically formed in one or more physical levels of memory cells having an active area disposed above a silicon (or other type of) substrate. In one example, the non-volatile memory cells include charge-trapping layers and are arranged in a plurality of vertical NAND strings.

[0080] In another embodiment, the memory structure **202** includes a two-dimensional memory array of non-volatile memory cells. In one example, the non-volatile memory cells are NAND flash memory cells utilizing floating gates. Other types of memory cells (e.g., NOR-type flash memory) can also be used.

[0081] The exact type of memory array architecture or memory cell included in memory structure **202** is not limited

to the examples above. Many different types of memory array architectures or memory technologies can be used to form memory structure **202**. No particular non-volatile memory technology is required for purposes of the new claimed embodiments proposed herein. For example, suitable technologies for the memory cells of the memory structure **202** include ReRAM memories (resistive random access memories), magnetoresistive memory (e.g., MRAM, Spin Transfer Torque MRAM, Spin Orbit Torque MRAM), FeRAM, phase change memory (e.g., PCM), and the like. Examples of suitable technologies for memory cell architectures of memory structure **202** include two dimensional arrays, three dimensional arrays, cross-point arrays, stacked two dimensional arrays, vertical bit line arrays, and the like. One example of a ReRAM cross-point memory includes reversible resistance-switching elements arranged in cross-point arrays accessed by X lines and Y lines (e.g., word lines and bit lines).

[0082] In another embodiment, the memory cells may include conductive bridge memory elements. A conductive bridge memory element may also be referred to as a programmable metallization cell. A conductive bridge memory element may be used as a state change element based on the physical relocation of ions within a solid electrolyte. In some cases, a conductive bridge memory element may include two solid metal electrodes, one relatively inert (e.g., tungsten) and the other electrochemically active (e.g., silver or copper), with a thin film of the solid electrolyte between the two electrodes. As temperature increases, the mobility of the ions also increases causing the programming threshold for the conductive bridge memory cell to decrease. Thus, the conductive bridge memory element may have a wide range of programming thresholds over temperature.

[0083] Another example is magnetoresistive random access memory (MRAM) that stores data by magnetic storage elements. The elements are formed from two ferromagnetic layers, each of which can hold a magnetization, and the ferromagnetic layers are separated by a thin insulating layer. One of the two ferromagnetic layers is a permanent magnet that is set to a particular polarity, and the other ferromagnetic layer's magnetization can be changed to match that of an external field to store memory. The memory array may be built from a grid of such memory cells. In one embodiment, for programming, each memory cell lies between a pair of write lines that are arranged at right angles to each other, parallel to the cell, one above and one below the cell. When current is passed through the write lines, an induced magnetic field is created. MRAM based memory embodiments will be discussed in more detail below.

[0084] Phase change memory (PCM) exploits the unique behavior of chalcogenide glass. One embodiment uses a $GeTe\text{-}Sb_2Te_3$ super lattice to achieve non-thermal phase changes by simply changing the co-ordination state of the Germanium atoms with a laser pulse (or light pulse from another source). Therefore, the doses of programming are laser pulses. The memory cells can be inhibited by blocking the memory cells from receiving the light. In other PCM embodiments, the memory cells are programmed by current pulses. Note that the use of "pulse" in this document does not require a square pulse but includes a (continuous or non-continuous) vibration or burst of sound, current, voltage light, or another wave. These memory elements within the individual selectable memory cells, or bits, may include a further series element that is a selector, such as an ovonic threshold switch or metal insulator substrate.

[0085] The technology described herein is not limited to a single specific memory structure, memory construction or material composition, but covers many relevant memory structures within the spirit and scope of the technology as described herein and as understood by one of ordinary skill in the art.

[0086] The elements of FIG. **2**A can be grouped into two parts: (1) the memory structure **202** and (2) peripheral circuitry, which includes all of the other components depicted in FIG. **2**A. An important characteristic of a memory circuit is its capacity, which can be increased by increasing the area of the memory die of storage system **100** that is given over to the memory structure **202**. However, this reduces the area of the memory die available for the peripheral circuitry. This can place quite severe restrictions on these elements of the peripheral circuitry. For example, the need to fit sense amplifier circuits within the available area can be a significant restriction on sense amplifier design architectures. With respect to system control logic **208**, reduced availability of area can limit the available functions that can be implemented on-chip. Consequently, a basic trade-off in the design of a memory die for the storage system **100** may be the amount of area to devote to the memory structure **202** and the amount of area to devote to the peripheral circuitry.

[0087] Another area in which the memory structure **202** and the peripheral circuitry are often at odds is in the processing involved in forming these regions, since these regions often involve differing processing technologies and the trade-off in having differing technologies on a single die. For example, when the memory structure **202** is NAND flash, this is an NMOS structure, while the peripheral circuitry is often CMOS based.

[0088] Elements such as the sense amplifier circuits, charge pumps, logic elements in a state machine, and other peripheral circuitry in the system control logic **208** often employ PMOS devices. Processing operations for manufacturing a CMOS die will differ in many aspects from the processing operations optimized for an NMOS flash NAND memory or other memory cell technologies.

[0089] To improve upon these limitations, embodiments described below can separate the elements of FIG. **2**A onto a separately formed die that is then bonded together with another die. More specifically, the memory structure **202** can be formed on one die (referred to as the memory die) and some or all of the peripheral circuitry elements, including one or more control circuits, can be formed on a separate die (referred to as the control die). A memory die can be formed of just the memory elements, such as the array of memory cells of flash NAND memory, MRAM memory, PCM memory, ReRAM memory, or other memory type. Some or all of the peripheral circuitry, even including elements such as decoders and sense amplifiers, can then be moved on to a separate control die. This allows each of the memory die to be optimized individually according to its technology.

[0090] For example, a NAND memory die can be optimized for an NMOS based memory array structure, without worrying about the CMOS elements that have now been moved onto a control die that can be optimized for CMOS processing. This allows more space for the peripheral elements, which can now incorporate additional capabilities

that could not be readily incorporated were they restricted to the margins of the same die holding the memory cell array.

[0091] The two die can then be bonded together in a bonded multi-die memory circuit, with the array on the one die connected to the periphery elements on the other die. Although the following will focus on a bonded memory circuit of one memory die and one control die, other embodiments can use more die, such as two memory die and one control die, for example.

[0092] FIG. **2**B shows an alternative arrangement to that of FIG. **2**A which may be implemented using wafer-to-wafer bonding to provide a bonded die pair. FIG. **2**B depicts a functional block diagram of one embodiment of an integrated memory assembly **240**. One or more integrated memory assemblies **240** may be used to implement the non-volatile memory **106** of storage system **100**.

[0093] The integrated memory assembly **240** includes two types of semiconductor die (or more succinctly, "die"). The memory die **242** includes the memory structure **202** with the non-volatile memory cells. A control die **244** includes control circuitry **208**, **216**, and **204** (as described above). In some embodiments, the control die **244** is configured to connect to the memory structure **202** in the memory die **242**. In some embodiments, the memory die **242** and control die **244** are bonded together.

[0094] FIG. **2**B shows an example of the peripheral circuitry, including control circuits, formed in a peripheral circuit or control die **244** coupled to memory structure **202** formed in memory die **242**. Common components are labelled similarly to FIG. **2**A. The system control logic **208**, the row control circuitry **204**, and the column control circuitry **216** are located in the control die **244**. In some embodiments, all or a portion of column control circuitry **216** and all or a portion of the row control circuitry **204** are located on memory die **242**. In some embodiments, some of the circuitry in the system control logic **208** is located on the memory die **242**.

[0095] The system control logic **208**, the row control circuitry **204**, and the column control circuitry **216** may be formed by a common process (e.g., CMOS process), so that adding elements and functions, such as the ECC controller, more typically found on a memory controller **104** may require few or no additional process steps, i.e., the same process steps used to fabricate controller **104** may also be used to fabricate the system control logic **208**, the row control circuitry **204**, and the column control circuitry **216**.

[0096] Thus, while moving such circuits from a die such as the memory die **242** may reduce the number of steps needed to fabricate such a die, adding such circuits to a die such as control die **244** may not require many additional process steps. The control die **244** also could be referred to as a CMOS die, due to the use of CMOS technology to implement some or all of the control circuitry **204**, **208**, **216**.

[0097] FIG. **2**B shows column control circuitry **216**, including the sense amplifier(s) **218**, on control die **244** coupled to memory structure **202** on memory die **242** through electrical paths **220**. The electrical paths **220** may provide an electrical connection between the column decoder **222**, the driver circuitry **224**, the block select **226**, and the bit lines of the memory structure **202**. In an embodiment, the column control circuitry **216** also includes column replacement control circuits **236**, which are described in more detail below.

[0098] Electrical paths may extend from the column control circuitry **216** in the control die **244** through pads on the control die **244** that are bonded to corresponding pads of the memory die **242**, which are connected to the bit lines of the memory structure **202**. Each bit line of the memory structure **202** may have a corresponding one of the electrical paths **220**, including a pair of bond pads, which connects to the column control circuitry **216**.

[0099] Similarly, the row control circuitry **204**, including the row decoder **210**, the array drivers **212**, and the block select **214** are coupled to the memory structure **202** through electrical paths **206**. Each of the electrical paths **206** may correspond to a data containing word line, a dummy word line, or a select gate line. Additional electrical paths may also be provided between control die **244** and memory die **242**.

[0100] For purposes of this document, the phrases "a control circuit," "control circuitry," or "one or more control circuits" can include any one of or any combination of the memory controller **104**; the state machine **228**; all or a portion of the system control logic **208**; all or a portion of row control circuitry **204**; all or a portion of column control circuitry **216**; a microcontroller; a microprocessor; and/or other similar functioned circuits.

[0101] The control circuit can include hardware only or a combination of hardware and software (including firmware). For example, one or more controllers programmed by firmware to perform the functions described herein is one example of a control circuit. A control circuit can include a processor, FGA, ASIC, integrated circuit, or other type of circuit.

[0102] In some embodiments, there is more than one control die **244** and more than one memory die **242** in an integrated memory assembly **240**. In some embodiments, the integrated memory assembly **240** includes a stack of multiple control dies **244** and multiple memory dies **242**.

[0103] FIG. **3**A depicts a side view of an embodiment of an integrated memory assembly **300** stacked on a substrate **302** (e.g., a stack including control die **304** and memory die **306**). In this embodiment, the integrated memory assembly **300** has three control die **304** and three memory die **306**. In some embodiments, there are more than three memory die **306** and more than three control die **304**.

[0104] Each control die **304** is affixed (e.g., bonded) to at least one memory die **306**. Some of the bond pads **308**/**310** are depicted, although there may be many more bond pads. A space between two die **306**, **304** that are bonded together is filled with a solid layer **312**, which may be formed from epoxy or other resin or polymer. This solid layer **312** protects the electrical connections between the die **306**, **304** and further secures the die together. Various materials may be used as solid layer **312**, but in some embodiments, it may be Hysol epoxy resin from Henkel Corp., having offices in California, USA.

[0105] Integrated memory assembly **300** may for example be stacked with a stepped offset, leaving the bond pads at each level uncovered and accessible from above. Wire bonds **314** connected to the bond pads connect control die **304** to substrate **302**. A number of such wire bonds may be formed across the width of each control die **304** (i.e., into the page of FIG. **3**A).

[0106] A memory die through silicon via (TSV) **316** may be used to route signals through each memory die **306**. A control die TSV **318** may be used to route signals through

each control die **304**. The TSVs **316**, **318** may be formed before, during or after formation of the integrated circuits in semiconductor die **306**, **304**. The TSVs may be formed by etching holes through the wafers. The holes may then be lined with a barrier against metal diffusion. The barrier layer may in turn be lined with a seed layer, and the seed layer may be plated with an electrical conductor such as copper, although other suitable materials such as aluminum, tin, nickel, gold, Solder balls **320** optionally may be affixed to contact pads **322** on a lower surface of substrate **302**. Solder balls **320** may be used to couple integrated memory assembly **300** electrically and mechanically to a host device such as a printed circuit board. Solder balls **320** may be omitted where the integrated memory assembly **300** is to be used as an LGA package. Solder balls **320** may form a part of an interface between integrated memory assembly **300** and memory controller **104** (FIG. **1**).

[0107] FIG. **3**B depicts a side view of another embodiment of an integrated memory assembly **300** stacked on a substrate **302**. The integrated memory assembly **300** of FIG. **3**B has three control die **304** and three memory die **306**. In some embodiments, there are many more than three memory die **306** and many more than three control die **304**. In this example, each control die **304** is bonded to at least one memory die **306**. Optionally, a control die **304** may be bonded to two or more memory die **306**.

[0108] Some of the bond pads **308**, **310** are depicted, but there may be many more bond pads than are illustrated. A space between two die **306**, **304** that are bonded together is filled with a solid layer **312**, which may be formed from epoxy or other resin or polymer. In contrast to the example in FIG. **3**A, the integrated memory assembly **300** of FIG. **3**B does not have a stepped offset. A memory die TSV **316** may be used to route signals through each memory die **306**. A control die TSV **318** may be used to route signals through each control die **304**.

[0109] As has been briefly discussed above, control die **304** and memory die **306** may be bonded together. Bond pads on each control die **304** and each memory die **306** may be used to bond the two die together. In some embodiments, the bond pads are bonded directly to each other, without solder or other added material, in a so-called Cu-to-Cu bonding process.

[0110] In a Cu-to-Cu bonding process, the bond pads are controlled to be highly planar and formed in a highly controlled environment largely devoid of ambient particulates that might otherwise settle on a bond pad and prevent a close bond. Under such properly controlled conditions, the bond pads are aligned and pressed against each other to form a mutual bond based on surface tension.

[0111] As has been briefly discussed above, the control die **304** and the memory die **306** may be bonded together. Bond pads on each control die **304** and each memory die **306** may be used to bond the two die together. In some embodiments, the bond pads are bonded directly to each other, without solder or other added material, in a so-called Cu-to-Cu bonding process. In a Cu-to-Cu bonding process, the bond pads are controlled to be highly planar and formed in a highly controlled environment largely devoid of ambient particulates that might otherwise settle on a bond pad and prevent a close bond. Under such properly controlled conditions, the bond pads are aligned and pressed against each other to form a mutual bond based on surface tension. Such bonds may be formed at room temperature, though heat also may be applied. In embodiments using cu-to-cu bonding, the bond pads may be about 5 μm square and spaced from each other with a pitch of 5 μm to 5 μm. Although this process is referred to herein as cu-to-cu bonding, this term also may apply even where the bond pads are formed of materials other than copper. When the area of bond pads is small, it may be difficult to bond the semiconductor die together. The size of and pitch between bond pads may be further reduced by providing a film layer on the surfaces of the semiconductor die including the bond pads. The film layer is provided around the bond pads. When the die are brought together, the bond pads may bond to each other, and the film layers on the respective die may bond to each other. Such a bonding technique may be referred to as hybrid bonding. In embodiments using hybrid bonding, the bond pads may be about 5 μm square and spaced from each other with a pitch of 1 μm to 5 μm. Bonding techniques may be used providing bond pads with even smaller (or greater) sizes and pitches.

[0112] Some embodiments may include a film on a surface of the control die **304** and the memory die **306**. Where no such film is initially provided, a space between the die may be under filled with an epoxy or other resin or polymer. The under-fill material may be applied as a liquid which then hardens into a solid layer. This under-fill step protects the electrical connections between control die **304** and memory die **306**, and further secures the die together. Various materials may be used as under-fill material, such as Hysol epoxy resin from Henkel Corp., having offices in California, U.S.A.

[0113] FIG. **4**A is a perspective view of a portion of one example embodiment of a monolithic three dimensional memory array/structure included in memory structure **202**, which includes a plurality non-volatile memory cells arranged as vertical NAND strings. For example, FIG. **4**A shows a portion **400** of one block of memory. The structure depicted includes a set of bit lines BL positioned above a stack **402** of alternating dielectric layers and conductive layers. For example purposes, one of the dielectric layers is marked as D and one of the conductive layers (also called word line layers) is marked as W. The number of alternating dielectric layers and conductive layers can vary based on specific implementation requirements.

[0114] As will be explained below, in one embodiment the alternating dielectric layers and conductive layers are divided into, for example, four or five (or a different number of) regions by isolation regions IR. FIG. **4**A shows one isolation region IR separating two regions. Below the alternating dielectric layers and word line layers is a common source line layer SL. Memory holes are formed in the stack of alternating dielectric layers and conductive layers. For example, one of the memory holes is marked as MH. Note that in FIG. **4**A, the dielectric layers are depicted as see-through so that the reader can see the memory holes positioned in the stack of alternating dielectric layers and conductive layers. In one embodiment, NAND strings are formed by filling the memory hole with materials including a charge-trapping material to create a vertical column of memory cells.

[0115] The non-volatile memory cells are arranged in memory holes, and each memory cell can store one or more bits of data, e.g., up to five bits of data per memory cell. More details of the three dimensional monolithic memory array that comprises memory structure **202** is provided below.

[0116] FIG. **4**B is a block diagram explaining one example organization of memory structure **202**, which is divided into four planes **404**, **406**, **408** and **410**. Each plane is then divided into M blocks. In one example, each plane has about 2,000 blocks ("Block **0**" to "Block M−1" with M being 2,000). However, different numbers of blocks and planes can also be used.

[0117] In one embodiment, a block of memory cells is a unit of erase. That is, all memory cells of a block are erased together. In other embodiments, the blocks can be divided into sub-blocks, each of which includes a plurality of word lines, and the sub-blocks can be the unit of erase. Memory cells also can be grouped into blocks for other reasons, such as to organize the memory structure to enable the signaling and selection circuits.

[0118] In some embodiments, a block represents groups of connected memory cells as the memory cells of a block share a common set of word lines. For example, the word lines for a block are all connected to all of the vertical NAND strings for that respective block. Although FIG. **4**B shows four planes, each of which includes a plurality of blocks, more or fewer than four planes can be implemented in the memory structure **202**. In some embodiments, the memory structure includes eight planes.

[0119] Each block typically is divided into one or more pages, with each page being a unit of programming/writing and a unit of reading. Other units of programming also can be used. In an embodiment, one or more pages of data are typically stored in one row of memory cells. For example, one or more pages of data may be stored in memory cells connected to a common word line. In an embodiment, a page includes data stored in all memory cells connected to a common word line within the block.

[0120] FIGS. **4**C-**4**G depict an example three dimensional ("3D") NAND structure that corresponds to the structure of FIG. **4**A and can be used to implement the memory structure **202** of FIGS. **2**A and **2**B. FIG. **4**C is a block diagram that depicts a top view of a portion **412** of Block **2** of plane **404**. As can be seen from FIG. **4**C, the block depicted in FIG. **4**C extends in the direction of **414**. In one embodiment, the memory array has many such layers with only the top layer being illustrated in FIG. **4**C.

[0121] FIG. **4**C depicts a plurality of circles that represent the memory holes, which are also referred to as vertical columns. Each of the memory holes/vertical columns includes multiple select transistors (also referred to as a select gate or selection gate) and multiple memory cells. In one embodiment, each memory hole/vertical column implements a NAND string. For example, FIG. **4**C labels a subset of the memory holes/vertical columns/NAND strings **416**, **418**, **420**, **422**, **424**, **426**, **428**, **430**, and **432**.

[0122] FIG. **4**C also depicts a set of bit lines **434**, including bit lines **436**, **438**, **440**, **442**, . . . **444**. FIG. **4**C shows twenty four bit lines because only a portion of the block is depicted. It is contemplated that more than twenty four bit lines connected to memory holes/vertical columns of the block. Each of the circles representing memory holes/vertical columns has an "x" to indicate its connection to one of the bit lines. For example, bit line **436** is connected to the memory holes/vertical columns **418**, **420**, **422**, **426**, and **432**. The bit lines **436**, **438**, **440**, **442** also are in electrical communication with all other blocks in a given plane.

[0123] The block depicted in FIG. **4**C includes a set of isolation regions **446**, **448**, **450** and **452**, which are formed of SiO_2. However, other dielectric materials also can be used. Isolation regions **446**, **448**, **450**, and **452** serve to divide the top layers of the block into five regions. For example, the top layer depicted in FIG. **4**C is divided into regions **454**, **456**, **458**, **460**, and **462**.

[0124] In one embodiment, the isolation regions only divide the layers used to implement select gates so that NAND strings in different regions can be independently selected. In one example implementation, a bit line connects to one memory hole/vertical column/NAND string in each of regions **454**, **456**, **458**, **460**, and **462**. In that implementation, each block has twenty-four rows of active columns and each bit line connects to five rows in each block.

[0125] In one embodiment, all of the five memory holes/vertical columns/NAND strings connected to a common bit line are connected to the same set of word lines; therefore, the system uses the drain side selection lines to choose one (or another subset) of the five to be subjected to a memory operation (program, verify, read, and/or erase).

[0126] FIG. **4**C also shows Line Interconnects LI, which are metal connections to the source line SL from above the memory array. Line Interconnects LI are positioned adjacent regions **454** and **462**.

[0127] Although FIG. **4**C shows each region **454**, **456**, **458**, **460**, and **462** as having four rows of memory holes/vertical columns, five regions and twenty four rows of memory holes/vertical columns in a block, those exact numbers are an example implementation. Other embodiments may include more or fewer regions per block; more or fewer rows of memory holes/vertical columns per region; and more or fewer rows of vertical columns per block.

[0128] FIG. **4**C also shows the memory holes/vertical columns being staggered. In other embodiments, different patterns of staggering can be used. In some embodiments, the memory holes/vertical columns are not staggered.

[0129] FIG. **4**D depicts a portion of one embodiment of a three dimensional memory structure **202** showing a cross-sectional view along line AA of FIG. **4**C. This cross sectional view cuts through memory holes/vertical columns (NAND strings) **428** and **430** of region **462** (see FIG. **4**C).

[0130] The structure of FIG. **4**D includes two drain side select layers SGD**0** and SGD**1**; two source side select layers SGS**0** and SGS**1**; two drain side GIDL generation transistor layers SGDT**0** and SGDT**1**; two source side GIDL generation transistor layers SGSB**0** and SGSB**1**; two drain side dummy word line layers DD**0** and DD**1**; two source side dummy word line layers DS**0** and DS**1**; dummy word line layers DU and DL that are separated by a joint; one hundred and sixty two word line layers WL**0**-WL**161** for connecting to data memory cells; and dielectric layers DL. Other embodiments can implement more or fewer than the numbers described above for FIG. **4**D. In one embodiment, SGD**0** and SGD**1** are connected together and SGS**0** and SGS**1** are connected together. In other embodiments, more or fewer SGDs (greater or lesser than two) are connected together and more or fewer SGS devices (greater or lesser than two) are connected together.

[0131] In one embodiment, erasing the memory cells is performed using gate induced drain leakage (GIDL), which includes generating charge carriers at the GIDL generation transistors such that the carriers get injected into the charge trapping layers of the NAND strings to change (reduce) respective threshold voltages Vt of the memory cells. In the embodiment of FIG. **4**D, there are two GIDL generation

transistors at each end of the NAND string; however, in other embodiments there are more or fewer than two GIDL generation transistors.

[0132] Embodiments that use GIDL at both sides of the NAND string may have GIDL generation transistors at both sides. Embodiments that use GIDL at only the drain side of the NAND string may have GIDL generation transistors only at the drain side. Embodiments that use GIDL at only the source side of the NAND string may have GIDL generation transistors only at the source side.

[0133] The GIDL generation transistors have an abrupt PN junction to generate the charge carriers for GIDL and, during fabrication, a phosphorous diffusion is performed at the polysilicon channel of the GIDL generation transistors. In some cases, the GIDL generation transistor with the shallowest phosphorous diffusion is the GIDL generation transistor that generates the charge carriers during erase. However, in some embodiments charge carriers can be generated by GIDL at multiple GIDL generation transistors at a particular side of the NAND string.

[0134] The memory holes/vertical columns **428**, **430** are depicted protruding through the drain side select layers, source side select layers, dummy word line layers, GIDL generation transistor layers and word line layers. In one embodiment, each memory hole/vertical column comprises a vertical NAND string. Below the memory holes/vertical columns and the layers listed below is substrate **464**, an insulating film **466** on the substrate, and source line SL. The NAND string of memory hole/vertical column **428** has a source end at a bottom of the stack and a drain end at a top of the stack. As in agreement with FIG. **4**C, FIG. **4**D show vertical memory hole/column **428** connected to bit line **442** via connector **468**.

[0135] For ease of reference, drain side select layers, source side select layers, dummy word line layers, GIDL generation transistor layers and data word line layers collectively are referred to as conductive layers.

[0136] In one embodiment, the conductive layers are made from a combination of TiN and Tungsten. In other embodiments, other materials can be used to form the conductive layers, such as doped polysilicon, metal such as Tungsten, metal silicide, such as nickel silicide, tungsten silicide, aluminum silicide or the combination thereof.

[0137] In some embodiments, different conductive layers can be formed from different materials. Between conductive layers are dielectric layers DL. In one embodiment, the dielectric layers are made from SiO_2. In other embodiments, other dielectric materials can be used to form the dielectric layers.

[0138] The non-volatile memory cells are formed along memory holes/vertical columns which extend through alternating conductive and dielectric layers in the stack. In one embodiment, the memory cells are arranged in NAND strings. The word line layers WL**0**-W**161** connect to memory cells (also called data memory cells). The dummy word line layers connect to a plurality of dummy memory cells, which do not store data. In some embodiments, the data memory cells and the dummy memory cells may have a same structure. The drain side select layers SGD**0** and SGD**1** are used to electrically connect and disconnect the NAND strings to and from the bit lines. The source side select layers SGS**0** and SGS**1** are used to electrically connect and disconnect the NAND strings to and from the source line SL.

[0139] FIG. **4**D shows that the memory array is implemented as a two tier architecture, with the tiers separated by a joint area. In one embodiment, it is expensive and/or challenging to etch so many word line layers intermixed with dielectric layers. To ease this burden, a first stack of word line layers (e.g., WL**0**-WL**80**) are laid down with alternating dielectric layers, then the Joint area is laid down, and next, a second stack of word line layers (e.g., WL**81**-WL**161**) are laid down with alternating dielectric layers. The joint area is thus positioned between the first stack of word line layers and the second stack of word line layers. In one embodiment, the joint areas are made from the same materials as the word line layers. In other embodiments, there can no joint area or there can be multiple joint areas.

[0140] FIG. **4**E depicts a portion of one embodiment of a three dimensional memory structure **202** showing a cross-sectional view along line BB of FIG. **4**C. This cross sectional view cuts through memory holes/vertical columns (NAND strings) **416** and **470** of region **454** (see FIG. **4**C). FIG. **4**E shows the same alternating conductive and dielectric layers as FIG. **4**D.

[0141] FIG. **4**E also shows isolation region **446**, which occupies a space that would have been used for a portion of the memory holes/vertical columns/NAND stings, including a space that would have been used for a portion of memory hole/vertical column **470**. More specifically, a portion (e.g., half the diameter) of vertical column **470** has been removed in layers SGDT**0**, SGDT**1**, SGD**0**, and SGD**1** to accommodate isolation region **446**. Thus, while most of the vertical column **470** is cylindrical (has a circular cross section), the portion of vertical column **470** in layers SGDT**0**, SGDT**1**, SGD**0**, and SGD**1** has a semi-circular cross section. In one embodiment, after the stack of alternating conductive and dielectric layers is formed, the stack is etched to create space for the isolation region and that space is then filled in with SiO_2. This structure allows for separate control of SGDT**0**, SGDT**1**, SGD**0**, and SGD**1** for regions **454**, **456**, **458**, **460**, and **462** (illustrated in FIG. **4**C).

[0142] FIG. **4**F depicts a cross sectional view of region **472** of FIG. **4**D that includes a portion of memory hole/vertical column **428**. In one embodiment, the memory holes/vertical columns are round. However, in other embodiments other shapes can be used. In one embodiment, memory hole/vertical column **428** includes an inner core layer **474** that is made of a dielectric, such as SiO_2. Surrounding the inner core **474** is a polysilicon channel **476** (materials other than polysilicon can alternately be used). The channel **476** extends between and is connected with the bit line and the source line. Surrounding the channel **476** is a tunneling dielectric **478** layer, which may have an ONO structure. Surrounding the tunneling dielectric **478** layer is charge trapping layer **480**, which may be formed of, for example, Silicon Nitride. It should be appreciated that the technology described herein is not limited to any particular material or structure.

[0143] FIG. **4**F depicts the dielectric layers DL as well as the word line layers WL**160**, WL**159**, WL**158**, WL**157**, and WL**156**. Each of these word line layers includes a word line region **482** surrounded by an aluminum oxide layer **484**, which is surrounded by a blocking oxide layer **486**. In other embodiments, the blocking oxide layer **486** can be a vertical layer that is parallel with and adjacent to the charge trapping layer **480**. The physical interaction of the word line layers with the vertical column forms the memory cells of the

NAND string. Thus, in one embodiment a memory cell includes the channel **476**, the tunneling dielectric **478**, the charge trapping layer **480**, the blocking oxide layer **486**, the aluminum oxide layer **484**, and the word line region **482**. For example, word line layer WL**160** and a portion of memory hole/vertical column **428** comprise a memory cell MC**1**. Word line layer WL**159** and a portion of memory hole/vertical column **428** comprise a memory cell MC**2**. Word line layer WL**158** and a portion of memory hole/vertical column **428** comprise a memory cell MC**3**. Word line layer WL**157** and a portion of memory hole/vertical column **428** comprise a memory cell MC**4**. Word line layer WL**156** and a portion of memory hole/vertical column **428** comprise a memory cell MC**5**. In other architectures, a memory cell may have a different structure; however, the memory cell would still be the storage unit.

[0144] When a memory cell is programmed, electrons are stored in a portion of the charge trapping layer **480** which is associated with (e.g., in) the memory cell. These electrons are drawn into the charge trapping layer **480** from the channel **476**, through the tunneling dielectric **478**, in response to an appropriate voltage on word line region **482**. The threshold voltage (Vth) of a memory cell is increased in proportion to the amount of stored charge.

[0145] In one embodiment, the programming is achieved through Fowler-Nordheim tunneling of the electrons into the charge trapping layer **480**. During an erase operation, the electrons return to the channel **476** or holes are injected into the charge trapping layer **480** to recombine with electrons. In one embodiment, erasing is achieved using hole injection into the charge trapping layer **480** via a physical mechanism such as GIDL, as described above.

[0146] FIG. **4**G is a schematic diagram of a portion of the three dimensional memory array depicted in in FIGS. **4**B-**4**F. FIG. **4**G shows physical data word lines WL**0**-WL**161** running across the entire block. The structure of FIG. **4**G corresponds to a portion **412** in Block **2** of FIG. **4**B, including bit line **436**. Within the block, in one embodiment, each bit line is connected to five NAND strings, one in each region of regions **454**, **456**, **458**, **460**, **462** (illustrated in FIG. **4**C).

[0147] In one embodiment, the programming is achieved through Fowler-Nordheim tunneling of the electrons into the charge trapping layer **480**. During an erase operation, the electrons return to the channel **476** or holes are injected into the charge trapping layer **480** to recombine with electrons. In one embodiment, erasing is achieved using hole injection into the charge trapping layer **480** via a physical mechanism such as GIDL, as described above.

[0148] FIG. **4**G is a schematic diagram of a portion of the three dimensional memory array depicted in in FIGS. **4**B-**4**F. FIG. **4**G shows physical data word lines WL**0**-WL**161** running across the entire block. The structure of FIG. **4**G corresponds to a portion **412** in Block **2** of FIG. **4**B, including bit line **436**. Within the block, in one embodiment, each bit line is connected to five NAND strings, one in each region of regions **454**, **456**, **458**, **460**, **462** (illustrated in FIG. **4**C).

[0149] Similarly, the drain side select line/layer SGD**1** is separated by isolation regions **446**, **448**, **450**, and **452** (illustrated in FIG. **4**C) to form SGD**1**-s**0**, SGD**1**-s**1**, SGD**1**-s**2**, SGD**1**-s**3** and SGD**1**-s**4** in order to separately connect to and independently control regions **454**, **456**, **458**, **460**, **462** (illustrated in FIG. **4**C). The drain side GIDL generation transistor control line/layer SGDT**0** is also separated by isolation regions **446**, **448**, **450** and **452** to form SGDT**0**-s**0**, SGDT**0**-s**1**, SGDT**0**-s**2**, SGDT**0**-s**3** and SGDT**0**-s**4** in order to separately connect to and independently control regions **454**, **456**, **458**, **460**, **462**. Further, the drain side GIDL generation transistor control line/layer SGDT**1** is separated by isolation regions **446**, **448**, **450** and **452** to form SGDT**1**-s**0**, SGDT**1**-s**1**, SGDT**1**-s**2**, SGDT**1**-s**3** and SGDT**1**-s**4** in order to separately connect to and independently control regions **454**, **456**, **458**, **460**, **462**.

[0150] FIG. **4**G only shows NAND strings connected to bit line **436**. However, a full schematic of the block would show every bit line and five vertical NAND strings, which are in separate regions, connected to each bit line.

[0151] Although the example memories of FIGS. **4**B-**4**G are three dimensional memory structures that include vertical NAND strings with charge-trapping material, other (2D and 3D) memory structures can also be used with the technology described herein.

[0152] The present disclosure is related to operating techniques that increase the bandwidth of a non-volatile memory device for the purpose of producing a high bandwidth flash (HBF) memory device that may be used as a replacement for HBM, particularly for machine learning inferencing operations. The HBF memory devices of the present disclosure are configured to be operated according to a 1 bit per memory cell (sometimes referred to as "single level cell" or "SLC" memory) storage scheme. The SLC storage scheme (as opposed to other storage schemes that allow multiple bits to be stored in each memory cell), is chosen because it is the fastest solution for reducing read latency (and increasing bandwidth).

[0153] FIG. **10** illustrates an example embodiment of a computing system **1000** constructed according to aspects of the present disclosure. The computing system **1000** includes a single graphics processor unit (GPU) **1002** (or a similar processor unit) and eight HBF packages **1004**, which are all in electrical communication with the single GPU **1002**. Such a computing system **1000** may be particularly adapted for use in storing data pertaining to a large language mode because once the model data have been stored in the HBF packages **1004**, the model data are not updated or changed very often. Thus, for a machine learning inferencing application, the HBF packages **1004** can be considered write once, read many times memory. In some embodiments, the computing system **1000** can include more or fewer than eight HBF packages **1004**. For example, in another embodiment, the computing system includes five HBF packages that are in electrical communication with a single GPU.

[0154] In an exemplary embodiment, each of the HBF packages **1004** includes eight memory dies, each of which includes thirty-two planes that can be independently and simultaneously be operated on. In some embodiments, the number of dies in each HBF packages and the number of planes per die can vary from these figures.

[0155] As discussed in further detail below, an aspect of the present disclosure is related to a method of operating the computing system **1000** that includes transmitting data between the single GBU **1002** and the HBF packages **1004** at rates of greater than 2.7 TB/s and preferably greater than 3 TB/s.

[0156] As described above, in some machine learning applications, large language models that include a terabyte (or more) of data that must be stored in memory and

retrieved at a very high data rate. Such applications, which require very high bandwidth and low power, typically store data in HBM DRAM. For example, in an existing machine learning system application, a processor (e.g., a CPU, GPU or other processor) is coupled to six (6) HBM DRAM devices and has a system bandwidth of 3 TB/s. However, DRAM is very expensive and each DRAM device has limited capacity. Thus, the number of HBM chips needed to store an entire large language model is very costly.

[0157] Non-volatile memory (e.g., NAND) is significantly less expensive than DRAM, but the bandwidth of conventional NAND memory devices is much lower than that of HBM DRAM. For example, an HBM DRAM die has a bandwidth of about 75 GB/sec. In contrast, a conventional NAND memory die has a bandwidth of about 4.4 GB/sec.

[0158] Achieving a bandwidth of greater than 2.7 TB/s (preferably at least 3 TB/s) using conventional NAND memory devices would require a prohibitively large number of memory packages. For example, with 16 memory die per memory package, each memory package has a bandwidth of 16×4.4 GB/s=70.4 GB/s. To provide a bandwidth of 3 TB/sec, forty-four (44) memory packages would be required, which is not practical.

[0159] In addition, the power consumption of conventional NAND memory devices is too high to provide a viable alternative to HBM devices. For example, in conventional NAND technology, the memory array itself has a power efficiency of approximately 4.5 pJ/bit, e.g., greater than 4.1 pJ/bit. However, in the exemplary embodiment, the HBF packages **1004** have a power efficiency of no greater than approximately 1 pJ/bit, i.e., less than 1.1 pJ/bit.

[0160] The present disclosure is related generally to technology that increases the bandwidth of and reduces the power consumption of non-volatile memory, such as flash memory. In particular, the aforementioned HBF packages utilize this technology to provide the HBF packages with sufficient bandwidth and acceptable power efficiency for use in large language model processing.

[0161] In embodiments, the bandwidth of HBF memory is increased by: (1) increasing the number of memory planes per memory die, (2) increasing the number of input/output (I/O) channels per memory die to accommodate the increased bandwidth of the memory die, and (3) reducing the physical page size to decrease read latency. These are discussed below.

[0162] As described above, a memory structure (such as memory structure **202** of FIG. **2**A) may include multiple planes, and each plane may operate in parallel. For example, FIG. **5**A is a diagram depicting a memory structure that includes four planes: P**00**, P**01**, P**02** and P**03**.

[0163] In an embodiment, each of planes P**00**, P**01**, P**02** and P**03** is divided into two sub-planes. For example, plane P**00** includes a first sub-plane $\mathbf{P00}_0$ and a second sub-plane $\mathbf{P00}_1$, second sub-plane P**01** includes a first sub-plane $\mathbf{P01}_0$ and a second sub-plane $\mathbf{P01}_1$, third sub-plane P**02** includes a first sub-plane $\mathbf{P02}_0$ and a second sub-plane $\mathbf{P02}_1$, and fourth sub-plane P**03** includes a first sub-plane $\mathbf{P03}_0$ and a second sub-plane $\mathbf{P03}_1$.

[0164] In an embodiment, each of planes P**00**, P**01**, P**02** and P**03** includes a logical page and a physical page size. In the embodiment of FIG. **5**A, the physical page size is 8 kB/page, and each logical page includes two physical pages (i.e., 16 kB per logical page).

[0165] In an embodiment, a memory die that includes planes P**00**, P**01**, P**02** and P**03** has a capacity of 32 GB. In an embodiment, sixteen (16) memory die are included in a memory package, and the memory package has a capacity of 16×32 GB=512 GB. Other capacities per memory die, and other numbers of memory die per memory package may be used.

[0166] In an embodiment, a memory die that includes planes P**00**, P**01**, P**02** and P**03** has a read latency ("tR") of approximately 15 μsec. The per-die bandwidth can be determined from read latency and the logical page size as follows:

$$BW = \frac{\left(16kB \times \frac{1024B}{kB}\right)}{15 \times 10^{-6}\ \text{sec}} \times 4\ \text{planes} = 4.4\ GB/\text{s}$$

[0167] In an embodiment, a memory die that includes planes P**00**, P**01**, P**02** and P**03** has an 8 bit I/O, and has an I/O speed of 4.8 G-transfers/s. Because the I/O speed (4.8 G-transfers/s) is faster that the memory die data rate (4.4 GB/sec), the I/O speed is not a factor limiting the data rate of the memory die.

[0168] In an embodiment, the number of planes per memory die is increased to increase the memory die bandwidth. For example, FIG. **5**B is a diagram depicting a memory structure that includes thirty-two (32) planes: four planes in the x-direction (e.g., planes P**00**, P**01**, P**02** and P**03**) and eight planes in the y-direction (e.g., planes P**03**, P**13**, P**23**, P**33**, P**43**, P**53**, P**63** and P**73**).

[0169] In an embodiment, each of planes P**00**, P**01**, P**02**, . . . , P**72** and P**73** is divided into two sub-planes. For example, plane P**00** includes a first sub-plane $\mathbf{P00}_0$ and a second sub-plane $\mathbf{P00}_1$, a second sub-plane P**01** includes a first sub-plane $\mathbf{P01}_0$ and a second sub-plane $\mathbf{P01}_1$, . . . , and a thirty-secondth sub-plane P**73** includes a first sub-plane $\mathbf{P73}_0$ and a second sub-plane $\mathbf{P73}_1$.

[0170] In an embodiment, each of planes P**00**, P**01**, P**02**, . . . , P**72** and P**73** includes a logical page and a physical page size. In the embodiment of FIG. **5**B, the physical page size is 8 kB/page, and each logical page includes two physical pages (i.e., 16 kB per logical page).

[0171] The structure of FIG. **5**B can be derived from the structure of FIG. **5**A by keeping the same number of planes in the x-direction (4 planes in x-direction), splitting the planes in half in the y-direction, and repeating four copies (for a total of 8 planes in y-direction), for a total of 4×8=32 planes per memory die. This is eight (8) times the number of planes of FIG. **5**A embodiment. Thus, the total capacity of each die is 0.5×8×32 GB=128 GB and the bandwidth of each die is 8×4.4 GB/s=35.2 GB/s.

[0172] In an embodiment, sixteen (16) memory die are included in a memory package (e.g., each HBF packages **1004** illustrated in FIG. **10**), and thus the memory package has a capacity of 16×128 GB=2 TB. Each memory package has a bandwidth of 16×35.2 GB/s=563.2 GB/s. To provide a bandwidth of greater than 2.7 TB/s (preferably, approximately 3 TB/sec), five (5) memory packages would be required. Eight HBF packages **1004** are depicted in the exemplary embodiment of FIG. **10**.

[0173] To provide this bandwidth, the number of I/O channels per die must be increased. As described above, an 8 bit I/O has a speed of 4.8 G-transfers/s. If the number of I/O are increased by a factor of 8 to 64 I/O, the I/O speed

increases to 8×4.8 G-transfers/s=38.4 G-transfers/s, which is faster that the memory die data rate (35.2 GB/sec).

[0174] In additional embodiments, the memory die bandwidth can be increased by reducing the physical page size to reduce read latency tR. For example, FIG. **5**C is a diagram depicting a memory structure that includes thirty-two (32) planes: eight (8) planes in the x-direction (e.g., planes **P00**, **P01**, . . . , and **P07**) and four (4) planes in the y-direction (e.g., planes **P07**, **P17**, **P27** and **P37**).

[0175] In an embodiment, each of planes **P00**, **P01**, **P02**, . . . , **P36** and **P37** is divided into two sub-planes. For example, plane **P00** includes a first sub-plane $\mathbf{P00}_0$ and a second sub-plane $\mathbf{P00}_1$, a second sub-plane **P01** includes a first sub-plane $\mathbf{P01}_0$ and a second sub-plane $\mathbf{P01}_1$, . . . , and a thirty-secondth sub-plane **P37** includes a first sub-plane $\mathbf{P37}_0$ and a second sub-plane $\mathbf{P37}_1$.

[0176] In an embodiment, each of planes **P00**, **P01**, **P02**, . . . , **P36** and **P37** includes a logical page and a physical page size. In the embodiment of FIG. **5**C, the physical page size is 4 kB/page, and each logical page includes two physical pages (i.e., 8 kB per logical page).

[0177] The structure of FIG. **5**C can be derived from the structure of FIG. **5**B by dividing each 8 kB plane into two (2) 4 kB planes in the x-direction (eight planes in x-direction), and reducing the number of planes by half in the y-direction (four planes in y-direction), for a total of thirty-two (32) planes. This is the same number of planes as the FIG. **5**B embodiment. The total capacity of each die is 64 GB.

[0178] By cutting each word line in half (from 8 kB to 4 kB), the read latency tR decreases. In an embodiment, the read latency tR of the embodiment of FIG. **5**C is 4 μsec (compared with a read latency tR of 15 μsec for the FIG. **5**B embodiment). The per-die bandwidth can be determined from read latency and the logical page size as follows:

$$BW = \frac{\left(8kB \times \frac{1024B}{kB}\right)}{4 \times 10^{-6}\ \text{sec}} \times 32\ \text{planes} = 66\ GB/\text{s}$$

[0179] In an embodiment, sixteen (16) memory die are included in a memory package (e.g., the HBF packages **1004** depicted in FIG. **10**), and thus the memory package has a capacity of 16×64 GB=1 TB. Each memory package has a bandwidth of 16×66 GB/s=1.1 TB/s. To provide a bandwidth of approximately 3 TB/sec, three (3) memory packages would be required, which is practical.

[0180] To provide this bandwidth, the number of I/O per die is also increased. As described above, an 8 bit I/O has a speed of 4.8 G-transfers/s. If the number of I/O are increased by a factor of sixteen (16) to one hundred and twenty-eight (128) I/O, the I/O speed increases to 16×4.8 G-transfers/s=77 G-transfers/s, which is faster than the memory die data rate (66 GB/sec). In other words, by significantly increasing the number of I/O, the I/O does not limit the bandwidth.

[0181] The memory die bandwidth can be further increased by further reducing the physical page size to reduce read latency tR. For example, FIG. **5**D is a diagram depicting a memory structure that includes thirty-two (32) planes: eight (8) planes in the x-direction (e.g., planes **P00**, **P01**, . . . , and **P07**) and four (4) planes in the y-direction (e.g., planes **P07**, **P17**, **P27** and **P37**).

[0182] In an embodiment, each of planes **P00**, **P01**, **P02**, . . . , **P36** and **P37** is divided into two sub-planes. For example, plane **P00** includes a first sub-plane $\mathbf{P00}_0$ and a second sub-plane $\mathbf{P00}_1$, a second sub-plane **P01** includes a first sub-plane $\mathbf{P01}_0$ and a second sub-plane $\mathbf{P01}_1$, . . . , and a thirty-secondth sub-plane **P37** includes a first sub-plane $\mathbf{P37}_0$ and a second sub-plane $\mathbf{P37}_1$.

[0183] In an embodiment, each of planes **P00**, **P01**, **P02**, . . . , **P36** and **P37** includes a logical page and a physical page size. In the embodiment of FIG. **5**D, the physical page size is 2 kB/page, and each logical page includes two physical pages (i.e., 4 kB per logical page).

[0184] The structure of FIG. **5**D can be derived from the structure of FIG. **5**C by dividing each 4 kB plane into two 2 kB planes in the x-direction (8 planes in x-direction), and keeping the same number of planes by half in the y-direction (4 planes in y-direction), for a total of 32 planes. This is the same number of planes as the FIG. **5**C embodiment. The total capacity of each die is 32 GB.

[0185] By cutting each word line in half (from 4 kB to 2 kB), the read latency tR decreases. In an embodiment, the read latency tR of the embodiment of FIG. **5**D is 1.7 μsec (compared with a read latency tR of 15 μsec for the FIG. **5**B embodiment and a read latency of 4 μsec for the FIG. **5**C embodiment). The per-die bandwidth can be determined from read latency and the logical page size as follows:

$$BW = \frac{\left(4kB \times \frac{1024B}{kB}\right)}{1.7 \times 10^{-6}\ \text{sec}} \times 32\ \text{planes} = 77\ GB/\text{s}$$

[0186] In an embodiment, eight (8) memory die are included in a memory package, and thus the memory package has a capacity of 8×32 GB=256 GB. Each memory package has a bandwidth of 8×77 GB/s=616 GB/s. To provide a bandwidth of approximately 3 TB/sec, at least five (5) memory packages are provided.

[0187] To provide this bandwidth, the number of I/O per die must be increased. As described above, an 8 bit I/O has a speed of 4.8 G-transfers/s. If the number of I/O are increased by a factor of sixteen (16) to one hundred and twenty-eight (128) I/O, the I/O speed increases to 16×4.8 G-transfers/s=77 G-transfers/s, which is about the same as the memory die data rate (77 GB/sec).

[0188] A first technique to improve the power efficiency of the HBF package **1004** (depicted in FIG. **10**) is to reduce the supply voltage. The power consumed by a NAND array is approximately equal to VCC×ICC, where VCC is the supply voltage and ICC is the supply current. For current NAND memory, a supply voltage of VCC=2.5 V is typically used. One technique for reducing power consumption (and thereby improve the power efficiency) is to reduce supply voltage VCC, such as VCC=1.2 V.

[0189] With the reduced supply voltage VCC, the memory device must still perform various functions, such as programming, inhibiting, and sensing. FIGS. **6**A-**6**C depict an example NAND string during these three different memory operations. In particular FIGS. **6**A-**6**C depict an example NAND string during inhibit, program and sensing, respectively.

[0190] For inhibit, depicted in FIG. **6**A, the 1.2V VCC is provided to the inhibit bit line. This voltage needs to be high

enough to cut off the unselected SGD transistors to provide the channel boosting to provide the inhibit operation. That is, (VSGD−VDD)<Vt.

[0191] For programming, depicted in FIG. **8**B, a bit line voltage VBL=0 V is delivered to the channel. This requires that the bias VSGD on the SGD cells is greater than the threshold voltage of the SGD cells, VSGD>Vt. In an embodiment, the threshold voltage of the SGD cells is approximately 1.5 V.

[0192] Thus, for inhibit it is a requirement that (VSGD−VDD)<Vt, and for programming, it is a requirement that VSGD>Vt. From these two requirements the following can be derived:

[0193] VSGD>Vt upper tail

[0194] VSGD<VDDSA+Vt lower tail

[0195] In embodiments, the Vt lower tail is about 1.5 V and the Vt upper tail is about 1.9 V, and VDDSA is about 1.1 V. Thus, from the two conditions above:

1.9 V<VSGD<2.6 V

[0196] For sensing, depicted in FIG. **6**C, the bit line voltage VBL needs to provide enough drain-to-source voltage for the NAND string. In a conventional NAND device a source line voltage VCELSRC is set to 1 V. In such a scenario, the bit line voltage needs to provide VBLC (~0.2 V)+VCELSRC+a few transistor threshold voltages+some temperature compensation (TCO) voltage, which leads to approximately 1.5 V, which cannot be derived from VCC if VCC=1.2 V. Therefore, the source line voltage VCELSRC is instead set to 0 V in a scheme that is referred to herein as a "positive sensing" scheme. Because the source line voltage VCELSRC is set at the lower voltage, the bit line voltage needs to provide VBLC (~0.2 V)+0 V+a few transistor threshold voltages+some temperature compensation (TCO) voltage, which leads to approximately 0.5 V, which can be provided with the lower VCC=1.2 V. Thus, the VCC 1.2 V external power supply can offer the proper voltage for the HBF packages **1004**.

[0197] A second technique to improve the power efficiency of the HBF packages **1004** is to reduce all internal voltages inside the NAND device. In an example embodiment the Vread, VDDSA, and VBL voltages are set as follows:

	Conventional NAND	HBF
Vread (V)	4.7	2.4
VDDSA (V)	2.2	1.1
VBL (V)	0.3	0.15
Isense (nA)	20	10

[0198] FIG. **7**A depicts an example threshold voltage distribution of a NAND memory cell. In particular, the example threshold voltage distribution includes an erased state (e.g., "1") distribution and a programmed state (e.g., "0") distribution for a conventional NAND memory cell.

[0199] In the illustrated example, both threshold distributions are broad (e.g., such as may be achieved using a one program, zero verify "1P0V" program method) and there is a wide separation between the two threshold distributions. In the depicted example, a read verify level SLCR of 0 V is used and unselected word lines are biased at a voltage of Vread=4.7 V.

[0200] FIG. **7**B depicts an example threshold voltage distribution of a page of memory cells in an HBF package **1004** (shown in FIG. **10**). In particular, the example threshold voltage distribution includes an erased state (e.g., "1") distribution and a programmed state (e.g., "0"). Because programming operations in the HBF package **1004** are performed infrequently, during programming the threshold voltage distributions can be made very narrow, much narrower than the threshold voltage distribution of a conventional NAND memory device depicted in FIG. **7**A.

[0201] In addition, the gap between the two distributions can be made very close together. By reducing the gap and the threshold voltage distribution width, the unselected word line voltage of Vread can be reduced from a conventional voltage, e.g., 4.7 V. In an exemplary embodiment, Vread is set to 2.4 V to further reduce power consumption.

[0202] In addition, in an exemplary embodiment of an HBF package **1004**, the read verify level SLCR can be reduced from 0 V (e.g., in the middle of the two distributions in FIG. **7**A). In an embodiment, read verify level SLCR is set at the upper tail of the erase distribution. For example, read verify level SLCR=−1 V or some other level. By setting read verify level SLCR at the upper tail of the erase distribution, the amount of overdrive between the erase state distribution and the read verify level SLCR is reduced, which in turn reduces power consumption during sensing.

[0203] As described above, the large language model processing is read intensive for the HBF packages **1004**. As a result, the HBF packages **1004** will experience a lot of read disturb. When a selected memory cell is read, pass voltages VREAD are applied to the unselected word lines of a memory block, and a reference voltage SLCR is applied to the selected word line containing the selected memory cell (see FIG. **6**C). A sensing operation is then performed to determine if a threshold voltage Vt of the selected memory cell is above or below the reference voltage VCG. This process can inadvertently induce a weak programming effect in certain memory cells of the memory block. Over time, if this process is repeated frequently without any intervening erase or program cycles, the accumulated charges in the memory cells can alter their threshold voltages Vt, potentially leading to bit errors.

[0204] For example, FIG. **8**A depicts example threshold voltage Vt distributions of a plurality of memory cells (e.g., a page) in an example memory block immediately after programming. After a large number of reads in the memory block (e.g., one hundred thousand reads), the erase distribution will widen through repeated, unintentional weak programming. Thus, as illustrated in FIG. **8**B, an upper tail of the erased data state "1" has encroached on a lower tail of the programmed data state "0." As a result, due to read disturb, a margin (the voltage gap between the upper tail of the erased data state and the lower tail of the programmed data state) has shrunk considerably.

[0205] Conventionally, after severe read disturb is experienced in one block, the data is relocated to a new block (called "relocation"), and then the host can continue reading from the new block. Although this results in well-defined threshold voltage Vt distributions in the relocated block, this requires consuming a program-erase cycle to program the data to new block B, which hurts endurance. In addition, the block management function becomes more complicated because a logical-to-physical block table must keep track of the relocated data.

[0206] In one embodiment, read disturb can be detected by monitoring error rates that are detected and corrected by the ECC engine. Corrective action is then triggered in response to the error rate exceeding a predetermined threshold error rate. In another embodiment, the read cycles in a memory block since the data was programmed are counted. After the count reaches a predetermined threshold (e.g., one hundred thousand read cycles), then corrective action is triggered. In yet another embodiment, read is performed at two voltages that are between the upper tail of the erased data state and the lower tail of the programmed data state when the data is initially programmed (e.g., see FIG. **8**A). A number of memory cells that have threshold voltages Vt between these two voltages is counted. If the number of memory cells in this voltage range is below a predetermined threshold, then the memory cells have not experienced significant read disturb yet. On the other hand, if the number of memory cells in this voltage range is above the predetermined threshold, then it is determined that the memory cells have experienced significant read disturb.

[0207] An aspect of the present disclosure is a data refresh technique for addressing read disturb that is hereinafter referred to as "sub-block read refresh." In embodiments, the described sub-block read refresh technique may be implemented by any one of or any combination of memory controller **104**, state machine **228**, all or a portion of system control logic **208**, all or a portion of row control circuitry **204**, all or a portion of column control circuitry **216**, a microcontroller, a microprocessor, and/or other similar functioned circuits.

[0208] Turning now to FIG. **9**A, in an embodiment, an exemplary memory block **900** includes two logical sub-blocks: a lower sub-block SB**0** and an upper sub-block SB**1**. More specifically, the data word lines of the memory block are divided into these two sub-blocks SB**0**, SB**1**, which can be erased independently of one another. In an exemplary embodiment, one or more dummy word lines, which do not contain data, can be positioned between the data word lines of the lower and upper sub-blocks SB**0**, SB**1**.

[0209] In the condition depicted in FIG. **9**A, the lower sub-block SB**0** contains user data that had a large margin when initially programmed (FIG. **9**C) but has experienced significant read disturb (FIG. **9**D) such that corrective action must be taken to preserve that data. The memory cells of the upper sub-block SB**1** are erased.

[0210] Before read failure is detected, instead of finding another memory block to erase and program, the data in lower sub-block SB**0** is re-programmed into the upper sub-block SB**1**. That is, data are relocated from part of the physical memory block (the lower sub-block SB**0**) to another part of the same physical memory block (the upper sub-block SB**1**). This is highly predictable, avoids the need to have to find an available memory block, and avoids the need to manage the relocated data in a logical-to-physical mapping table. After re-programming the data from the lower sub-block SB**0** to the upper sup-block SB**1**, the threshold voltage Vt distribution has a margin similar to when it was initially programmed, e.g. compare FIGS. **9**C and **9**E. The lower sub-block SB**0** can then be erased.

[0211] When the data that is now in the upper sub-block SB**1** has suffered a significant amount of read disturb, this process can then be repeated in reverse to move the data from the upper sub-block SB**1** back to the lower sub-block SB**0**. This process can be repeated as many times as necessary and in any number (including all) of the memory blocks in the memory device.

[0212] Unlike erase, which happens on a memory block or sub-block level, programming occurs on a word line-by-word line basis from one side of the memory block towards an opposite side. In other words, the programming direction can start from a drain side of a sub-block and proceed towards the source side of the sub-block or vice versa. More specifically, in a normal order programming (NOP) direction, programming begins on the source side of the sub-block and proceeds in a direction towards the drain side of the sub-block. For example, in a memory block that has one hundred and sixty-two (162) word lines (WL**0**-WL**161**) with physical word lines WL**0**-WL**80** being in the lower sub-block SB**0** and physical word lines WL**81**-WL**161** being in the upper sub-block SB**1**, NOP of the upper sub-block SB**1** begins at WL**81** and ends at WL**160**. In a reverse order programming (ROP) direction, programming begins on the drain side of the sub-block and proceeds towards the source side. For example, ROP begins at WL**80** and ends at WL**0**. According to an exemplary embodiment of the present disclosure, the lower sub-block SB**0** is programmed using ROP, and the upper sub-block SB**1** is programmed using NOP.

[0213] Further, as depicted in FIG. **9**B, when relocating data from the lower sub-block SB**0** to the upper sub-block SB**1**, the order of the data is reversed from a first order in the lower sub-block SB**0** to an opposite second order in the upper sub-block SB**1**. More specifically, the relocating operation follows the following steps, in order. The data in the memory cells of the upper-most word line (e.g., WL**80**) of the lower sub-block SB**0** is read and then programmed into the memory cells of the lower-most word line of the upper sub-block (e.g., WL**81**). Next, the data in the second highest word line (e.g., WL**79**) of the lower sub-block SB**0** is read and programmed into the second lowest word line of the upper sub-block (e.g., WL**82**). Then, the data in the third highest word line (e.g., WL**78**) of the lower sub-block SB**0** is read and programmed into the third lowest word line of the upper sub-block (e.g., WL**83**). This process continues until either all of the data has been copied from the lower sub-block SB**0** to the upper sub-block SB**1** or until the data of the lowest word line WL**0** of the lower sub-block SB**0** has been relocated to the highest word line WLN (e.g., WL**161**) of the upper sub-block SB**1**. In other words, the data in the upper sub-block SB**1** has a reverse order from what it had in the lower sub-block SB**0**. The lower sub-block SB**0** can then be erased in an erase operation so that its memory cells are ready to receive the same data back when the upper sub-block SB**1** experiences significant read disturb.

[0214] As illustrated in FIG. **9**F, when relocating data from the upper sub-block to the lower sub-block, the data in the memory cells of the lowest word line (e.g., WL**81**) of the upper sub-block SB**1** is first read and copied into to the memory cells of the highest word line (e.g., WL**80**) of the lower sub-block SB**0**. Then, the data in the second lowest word line (e.g., WL**82**) of the upper sub-block SB**1** is copied into the next highest word line (e.g., WL**79**) of the lower sub-block SB**0**. This process continues until either all of the data has been migrated from the upper sub-block SB**1** to the lower sub-block SB**0** or until the data of the highest word line (e.g., WL**161**) of the upper sub-block SB**1** has been relocated to the lowest word line WL**0** of the lower sub-

block SB0. The upper sub-block SB1 can then be erased in an erase operation so that its memory cells are ready to receive the same data back when the lower sub-block SB0 experiences significant read disturb.

[0215] Turning now to FIG. **11**, a flow chart **1100** is provided depicting the steps of relocating data between sub-blocks according to an exemplary embodiment of the present disclosure. These steps could be performed by the controller; a processor or processing device or any other circuitry, executing instructions stored in memory; and/or other circuitry described herein that is specifically configured/programmed to execute the following steps.

[0216] At step **1102**, it is determined that the memory cells of a first sub-block (for example, the lower sub-block SB0 illustrated in FIGS. **9**A, **9**B, and **9**F) have experienced significant read disturb. This could include detecting the read disturb or establishing that a read cycle count since the data was programmed into the first sub-block exceeds a predetermined threshold.

[0217] At step **1104**, starting with the innermost data-containing word line of the first sub-block and continuing sequentially towards (and possibly all the way to) the outermost word line of the first sub-block, the data of a selected word line of the first sub-block is read and then programmed into one of the word lines of a second sub-block (for example, the upper sub-block SB1 illustrated in FIGS. **9**A, **9**B, and **9***f*) starting with the innermost word line of the second sub-block and continuing towards the outermost word line of the second sub-block. Thus, the data has a first order in the first sub-block and an opposite second order in the second sub-block.

[0218] At step **1106**, the first sub-block is erased.

[0219] At step **1108**, it is determined that the memory cells of the second sub-block have experienced significant read disturb. This could include detecting the read disturb or establishing that a read cycle count since the data was programmed into the second sub-block exceeds a predetermined threshold.

[0220] At step **1110**, starting with the innermost data-containing word line of the second sub-block and continuing sequentially towards (and possibly all the way to) the outermost word line of the second sub-block, the data of a selected word line of the second sub-block is read and then programmed into one of the word lines of the first sub-block starting with the innermost word line of the first sub-block and continuing towards the outermost word line of the first sub-block. Thus, the data has a second order in the second sub-block and an opposite first order in the first sub-block.

[0221] At step **1112**, the second sub-block is erased. The process then can return to step **1102**.

[0222] For purposes of this document, reference in the specification to "an embodiment," "one embodiment," "some embodiments," or "another embodiment" may be used to describe different embodiments or the same embodiment.

[0223] For purposes of this document, a connection may be a direct connection or an indirect connection (e.g., via one or more others parts). In some cases, when an element is referred to as being connected or coupled to another element, the element may be directly connected to the other element or indirectly connected to the other element via intervening elements. When an element is referred to as being directly connected to another element, then there are no intervening elements between the element and the other element. Two devices are "in communication" if they are directly or indirectly connected so that they can communicate electronic signals between them.

[0224] For purposes of this document, the term "based on" may be read as "based at least in part on."

[0225] For purposes of this document, without additional context, use of numerical terms such as a "first" object, a "second" object, and a "third" object may not imply an ordering of objects, but may instead be used for identification purposes to identify different objects.

[0226] For purposes of this document, the term "set" of objects may refer to a "set" of one or more of the objects.

[0227] The foregoing detailed description has been presented for purposes of illustration and description. It is not intended to be exhaustive or to limit to the precise form disclosed. Many modifications and variations are possible in light of the above teaching. The described embodiments were chosen in order to best explain the principles of the proposed technology and its practical application, to thereby enable others skilled in the art to best utilize it in various embodiments and with various modifications as are suited to the particular use contemplated. It is intended that the scope be defined by the claims appended hereto.

What is claimed is:

1. A method of operating a memory device, comprising the steps of:

preparing a memory device that includes a memory block with an array of memory cells that are arranged in a plurality of word lines, the plurality of word lines being divided into a first sub-block and a second sub-block, the memory cells of the first sub-block containing data, and the memory cells of the second sub-block being erased;

determining that the memory cells of the first sub-block have experienced significant read disturb; and

programming the data in the memory cells of the first sub-block into the memory cells of the second sub-block.

2. The method as set forth in claim **1**, further including the steps of counting a number of read cycles to establish a read cycle count; and

comparing the read count cycle to a predetermined threshold; and

wherein the step of determining that the memory cells of the first sub-block have experienced significant read disturb occurs in response to the read cycle count exceeding the predetermined threshold.

3. The method as set forth in claim **1**, wherein the memory block has a source side and a drain side;

wherein prior to the step of programming the data in the memory cells of the first sub-block to the memory cells of the second sub-block, the data has a first order within the first sub-block; and

after step of programming the data in the memory cells of the first sub-block to the memory cells of the second sub-block, the data has a second order within the second sub-block, the second order being opposite of the first order.

4. The method as set forth in claim **3**, further including the steps of:

erasing the memory cells of the first sub-block;

determining that the memory cells of the second sub-block have experienced significant read disturb; and

programming the data in the memory cells of the second sub-block into the memory cells of the first sub-block.

5. The method as set forth in claim **4**, wherein after the step of programming the data in the memory cells of the second sub-block into the memory cells of the first sub-block, the data has the first order.

6. The method as set forth in claim **5**, wherein the first sub-block is a lower sub-block and wherein the second sub-block is an upper sub-block;

wherein the step of programming the data in the memory cells of the lower sub-block into the memory cells of the upper sub-block includes programming according to a normal order programming direction.

7. The method as set forth in claim **6**, wherein the step of programming the data in the memory cells of the upper sub-block into the memory cells of the lower sub-block includes programming according to a reverse order programming direction.

8. A memory device, comprising:

a memory block with an array of memory cells that are arranged in a plurality of word lines, the plurality of word lines being divided into a first sub-block and a second sub-block, the memory cells of the first sub-block containing data, and the memory cells of the second sub-block being erased; and

circuitry configured to;

determine that the memory cells of the first sub-block have experienced significant read disturb, and

program the data in the memory cells of the first sub-block into the memory cells of the second sub-block.

9. The memory device as set forth in claim **8**, wherein the circuitry is further configured to count a number of read cycles to establish a read cycle count; and

compare the read count cycle to a predetermined threshold; and

wherein the circuitry determines that the memory cells of the first sub-block have experienced significant read disturb in response to the read cycle count exceeding the predetermined threshold.

10. The memory device as set forth in claim **8**, wherein the memory block has a source side and a drain side;

wherein prior to programming the data in the memory cells of the first sub-block to the memory cells of the second sub-block, the data has a first order within the first sub-block; and

after programming the data in the memory cells of the first sub-block to the memory cells of the second sub-block, the data has a second order within the second sub-block, the second order being opposite of the first order.

11. The memory device as set forth in claim **10**, wherein the circuitry is further configured to:

erase the memory cells of the first sub-block;

determine that the memory cells of the second sub-block have experienced significant read disturb; and

program the data in the memory cells of the second sub-block into the memory cells of the first sub-block.

12. The memory device as set forth in claim **11**, wherein after programming the data in the memory cells of the second sub-block into the memory cells of the first sub-block, the data has the first order.

13. The memory device as set forth in claim **12**, wherein the first sub-block is a lower sub-block and wherein the second sub-block is an upper sub-block; and

wherein the circuitry is configured to program the data in the memory cells of the lower sub-block into the memory cells of the upper sub-block according to a normal order programming direction.

14. The memory device as set forth in claim **13**, wherein the circuitry is configured to program the data in the memory cells of the upper sub-block into the memory cells of the lower sub-block according to a reverse order programming direction.

15. A computing system, comprising:

a processor unit;

a plurality of high bandwidth flash (HBF) packages in electrical communication with the processor unit;

at least one of the HBF packages including a memory block with an array of memory cells that are arranged in a plurality of word lines, the plurality of word lines being divided into a first sub-block and a second sub-block, the memory cells of the first sub-block containing data, and the memory cells of the second sub-block being erased; and

the at least one of the HBF packages further including circuitry that is configured to;

determine that the memory cells of the first sub-block have experienced significant read disturb, and

program the data in the memory cells of the first sub-block into the memory cells of the second sub-block.

16. The computing system as set forth in claim **15**, wherein the circuitry is further configured to count a number of read cycles to establish a read cycle count; and

compare the read count cycle to a predetermined threshold; and

wherein the circuitry determines that the memory cells of the first sub-block have experienced significant read disturb in response to the read cycle count exceeding the predetermined threshold.

17. The computing system as set forth in claim **15**, wherein the memory block has a source side and a drain side;

wherein prior to programming the data in the memory cells of the first sub-block to the memory cells of the second sub-block, the data has a first order within the first sub-block; and

after programming the data in the memory cells of the first sub-block to the memory cells of the second sub-block, the data has a second order within the second sub-block, the second order being opposite of the first order.

18. The computing system as set forth in claim **17**, wherein the circuitry is further configured to:

erase the memory cells of the first sub-block;

determine that the memory cells of the second sub-block have experienced significant read disturb; and

program the data in the memory cells of the second sub-block into the memory cells of the first sub-block.

19. The computing system as set forth in claim **18**, wherein after programming the data in the memory cells of the second sub-block into the memory cells of the first sub-block, the data has the first order.

20. The computing system as set forth in claim **19**, wherein the first sub-block is a lower sub-block and wherein the second sub-block is an upper sub-block; and

wherein the circuitry is configured to program the data in the memory cells of the lower sub-block into the memory cells of the upper sub-block according to a normal order programming direction.

* * * * *