

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

MICRON TECHNOLOGY, INC.,
Petitioner,

v.

YANGTZE MEMORY TECHNOLOGIES COMPANY, LTD.,
Patent Owner.

Case No. IPR2025-00190
U.S. Patent No. 11,568,941
Issue Date: January 31, 2023

Title: MEMORY INCLUDING A PLURALITY OF PORTIONS AND USED
FOR REDUCING PROGRAM DISTURBANCE AND PROGRAM METHOD
THEREOF

REPLY DECLARATION OF DAVID KUAN-YU LIU, PH.D.

Table of Contents

I.	INTRODUCTION	4
II.	CLAIM CONSTRUCTION.....	4
A.	The Phrase “ <i>from bottom-to-top</i> ” Means from “Source to Drain”	5
1.	Both “Bottom” and “Top” are Relative Terms Evaluated Based on a Reference Point	6
2.	The Intrinsic and Extrinsic Evidence Supports Petitioner’s Construction	8
B.	“ <i>program operation</i> ”	17
III.	REBUTTAL ANALYSIS AND OPINIONS	23
A.	Grounds 1 and 2 (Anticipation and Obviousness Over Dong) 23	
1.	A POSITA Would Understand Dong to disclose 3D NAND (with Vertical Memory Strings) in 2020	23
2.	A POSITA Would Understand Dong to disclose 3D NAND (with Vertical Memory Strings) in 2007	35
3.	Dong’s Program Disturb Solutions (in Figs. 5a and 5g) Apply to 3D NAND (with Vertical Strings) and Would Have Been Obvious	41
4.	Enablement and Reasonable Expectation of Success	52
B.	Ground 3 (<i>i.e.</i> , the Dong/Costa Combination) Renders Claims 8 and 9 Obvious.....	56
1.	Dong Combined with Costa Render Obvious the “ <i>the first voltage has a first level if the xth word line is in the first portion</i> ”	56
2.	Ground 3 Satisfies “ <i>Applying a First Voltage</i> ” Under Petitioner’s Interpretation of “ <i>Program Operation</i> ”	66
C.	Ground 4 (<i>i.e.</i> , the Dong/Shirai Combination) Renders Claims 4, 10, and 14 Obvious.....	70
1.	Patent Owner’s Arguments Concerning “V _{PASS-LOW} ” Fail	70
2.	Patent Owner’s Argument About Mismatched Values for “ <i>k</i> ” is a Strawman.....	78

IV. SECONDARY CONSIDERATIONS	79
V. CONCLUSION	80

I. INTRODUCTION

1. I have been retained by counsel for Micron Technology, Inc. (“Micron” or “Petitioner”) in the above-captioned matter as an independent consultant to offer opinions regarding the patentability claims 1, 3-11, and 13-14 (the “Challenged Claims”) of U.S. Patent No. 11,568,941 (the “’941 patent” or the “Challenged Patent”).

2. My opening declaration (Ex. 1004) describes my compensation, educational background and professional experience, etc., which I incorporate herein. I have been asked to reply to the Patent Owner Response (“POR”) and certain opinions in Dr. Sunil Khatri’s (Ex. 2028, the “Khatri Declaration”). The fact that I do not respond to particular assertions made by Patent Owner or opinions of Dr. Khatri should not be taken as agreement with those assertions or opinions.

3. I reviewed and considered the materials listed in my opening declaration, the Board’s Institution Decision (Paper 22), POR (Paper 33) and exhibits thereto (including the Khatri Declaration), the materials listed in Petitioner’s Reply, and any other materials cited herein.

II. CLAIM CONSTRUCTION

4. I have been asked to respond to two claim constructions proffered by Patent Owner and Dr. Khatri regarding the terms *from bottom-to-top* and *program operation*. I disagree with Patent Owner’s constructions for the reasons below.

A. The Phrase “*from bottom-to-top*” Means from “Source to Drain”

5. Each independent claim recites *a memory*. For example, the preamble of claim 1 recites: *A memory comprising:* Ex. 1001, cl.1. Notably, none of the claims say that the *memory* is “3D” or “three-dimensional.” If the claims were limited to vertical memory strings, I would expect them to say as much, especially considering that the specification uses the phrase “three dimensional” in the BACKGROUND section of the ’941 patent, indicating that the patentee knew how to use the term.

6. While nobody says the term *memory* requires vertical 3D memory strings, Patent Owner says the phrase *form bottom to top* does. POR, 12-29. More specifically, Patent Owner says that “bottom means lowest point and top means highest point relative to the substrate.” *Id.*, 14. That is not how a POSITA familiar with NAND flash memory and the ’941 patent would understand that phrase. The thing that extends from *bottom to top* is a set of wordlines that comprise a memory string. A POSITA would understand that indexing and/or programming a memory string *from bottom to top* typically simply means “from source to drain.” Every memory string (regardless of whether it is 2D and 3D) has a source end and a drain end. By far, the most common way to index (viz., assign wordline numbers) and program a memory string is from source to drain. This convention dates back decades. Even a U-shaped 3D NAND string would have a source terminal and a

drain terminal, and programming would necessarily proceed from one terminal end to the other terminal end, and not from the location closest to the substrate.

1. Both “Bottom” and “Top” are Relative Terms Evaluated Based on a Reference Point

7. At the outset, I wish to correct a misconception introduced by Patent Owner—viz., that Patent Owner but not Petitioner applies the plain and ordinary meaning. POR, 14. To do this, I will refer to the dictionaries Patent Owner cites. Neither of them (Ex. 2013 and Ex. 2014) defines “bottom” or “top” relative to a semiconductor “substrate.” However, Patent Owner’s dictionary is clear that “bottom” and “top” are *relative terms*, whose plain and ordinary meaning are evaluated from a “reference point.” Ex. 2014, 3 (defining “bottom” as: “The part closest to a *reference point*”); *id.*, 4 (defining “top” as: “The part farthest from a given reference point” (emphasis in original)).

8. As a sports and basketball fan, I couldn’t help but noting that Patent Owner’s dictionary (Ex. 2014) used the example of a basketball court (which is a *horizontal plane*) to define “top” and “bottom.” This is consistent with the Board’s Institution Decision which also recognized that *bottom to top* was a relative term. Institution Decision, 16-17.

9. Given that both parties agree “top” and “bottom” are relative terms evaluated against a reference point, the only remaining dispute is what the reference point should be. Patent Owner says it is a semiconductor “substrate,” while

Petitioner asserts (and I agree) that the memory string's source/drain are the reference points.

10. Patent Owner also says that “[t]he term ‘from bottom to top’ has no special meaning in the semiconductor manufacturing context or in the ’941 Patent.” POR, 14. If that were correct, there would be no reason to define **bottom to top** relative to a semiconductor substrate. By using specialized terminology from the semiconductor arts (*i.e.*, a semiconductor “substrate”), Patent Owner’s assertion is self-defeating. If anything, what Patent Owner’s proposed construction underscores is the necessity of construing **bottom to top** from the perspective of a POSITA who understands NAND flash memory. From that vantage point, **bottom to top** means from source to drain.

11. The heart of the dispute hinges on whether the claims cover 2D (*i.e.*, horizontal) flash memory strings. The crux of Patent Owner’s argument is that “bottom” and “top” necessarily imply vertically oriented memory strings. But Patent Owner’s own dictionary confirms that is not the case. The American Heritage Dictionary (Ex. 2014) provides the following definitions:

bottom: “The part closest to a reference point: *was positioned at the bottom of the key for a rebound.*”

top: “The part farthest from a given reference point: *took a jump shot from the top of the key.*”

Ex. 2014, 3-4 (emphasis in original). Notably, contrary to the premise of Patent Owner’s construction, the two dictionary examples confirm that a *horizontal plane* (i.e., a basketball court) has “top” and “bottom” locations. Likewise, 2D NAND flash memory strings also have a “top” and “bottom,” which are associated with the “drain” and “source” ends of a memory string, respectively.

2. The Intrinsic and Extrinsic Evidence Supports Petitioner’s Construction

12. Patent Owner says Petitioner’s construction is wrong because the ’941 patent makes “no reference to [] source or drain.” POR, 15-16. Again, Patent Owner’s logic is self-defeating. According Patent Owner’s line of thinking, its own construction cannot be correct because the ’941 patent (and claims) are silent regarding a semiconductor “substrate,” and they certainly do not use the term in a definitional sense as it relates to the phrase *from bottom to top*.

13. The reason the claims recite *from bottom to top* at all is to provide a logical framework (not a physical architecture) for identifying and referencing wordline groups (which the claims and specification call *portion[s]*). Ex. 1094, 174:24-175:10 (Dr. Khatri admitting that the purpose of the ’941 figures is “to describe the wordlines and their – you know, their numerals of their variable names

and their relative orientations, right. WL1 through WL_k is from bottom to top and WL(_k+1) through WL_m is from bottom to top. So that's the purpose of the – that's the purpose of this figure because the figure is -- you know, I mean, the patent is interested in the voltages you would apply to the word lines”), 224:11-15. This is confirmed by the claims' use of variables (such as **x**, **k**, and **m**), which again impart no physical structure. The **wordline[s]** and **portion[s]** are always referred to use relative locations based on variable, not vertical/physical locations.

14. Patent Owner's argument that the claims require vertical memory strings because they recite memory strings indexed **from bottom to top** is circular reasoning. POR, 14-15. And Patent Owner's appeal to the claim language reciting **above** and **below** (*id.*, 15) fares no better, as those terms simply reflect where specified wordlines are relative to other wordlines in the memory string that is already indexed from **bottom to top**. In other words, the **above** and **below** claim language is derivative of the **bottom to top** term and adds nothing of substance to it.

15. By examining only isolated words in the claims, Patent Owner misses the forest for the trees. As noted, the reason the claims recite **bottom to top**, **above**, and **below** is to establish an indexing scheme (using the variables **x**, **k**, **m**, etc.) for specifying wordline locations/groupings (*e.g.*, **portions**) to which different voltages are applied, not to require a physical memory string architecture. In other words, the only purpose the **bottom to top** language serves is to determine the application

of the recited voltages relative to other memory string cells. When viewed holistically, the claim language supports Petitioner.

16. The only place the specification uses the term “three dimensional” is in the BACKGROUND section at col. 1:26-41. While I agree that the claims cover 3D memory, I could find nothing in the specification that would limit them to such an architecture. Indeed, the “SUMMARY OF THE INVENTION” and “DETAILED DESCRIPTION” make no mention of “three dimensional” memory at all, let alone any limiting statements. *See generally* Ex. 1001.

17. Next, in what I can only describe as a double-standard, Patent Owner contends that the ’941 patent’s figures limit the claims to vertical 3D memory strings. POR, 16-18. I disagree. The ’941 figures are analogous to Dong Figs. 5a and 5g in that both show application of specified voltages to particular wordlines. If the ’941 figures impart a vertical/structural orientation, so do Dong Figs. 5a and 5g. And if Dong Figs. 5a and 5g are merely relational/logical, so too are the ’941 figures.

18. Regardless, even if the ’941 figures reflect a vertically oriented memory string, limiting the claims in such a manner would import a limitation from the specification into the claims, which do not recite “three-dimensional memory,” “3D memory,” or “vertical” memory strings.”

19. The ’941 figures depict the relationships between various wordline groupings (*e.g.*, **first portion**, **second portion**, etc.) and wordline **x** (which represents

the wordline being programmed). This does not restrict the physical orientation or arrangement of the memory string. Indeed, the '941 figures all depict a “[c]ontroller” 190 whose placement likewise conveys nothing about the physical placement or orientation. Taking '941 patent Fig. 1 as an example, it shows a “Controller” connected to a single wordline, WL(1). If Fig. 1 were intended to illustrate the physical structure of a memory string, it would not work because the controller would need to access and control **all** wordlines. Further, under Patent Owner’s logic, the controller must be located underneath and to the left of the memory string. There is plainly no such restriction, nor do the figures impose physical restrictions on the memory string implementation.

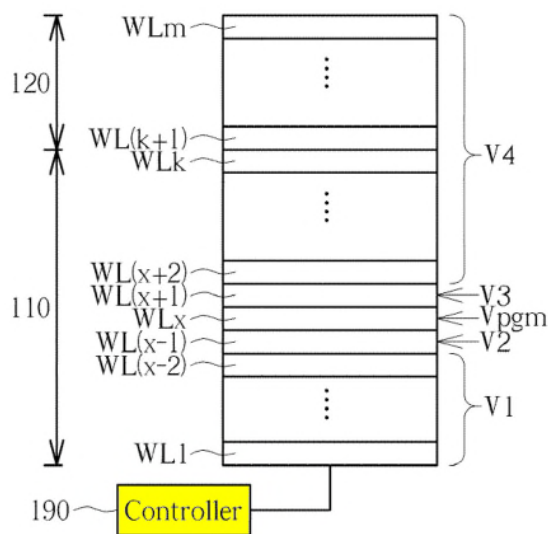


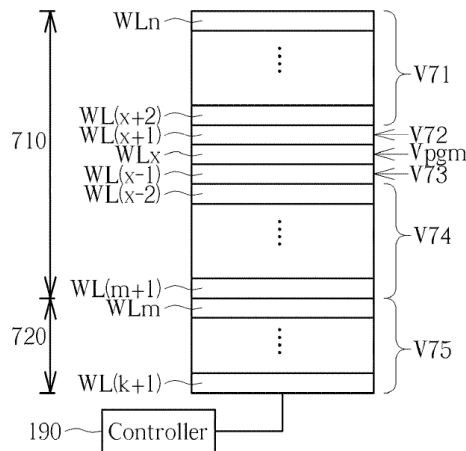
FIG. 1

Ex. 1001, Fig. 1 (annotated).

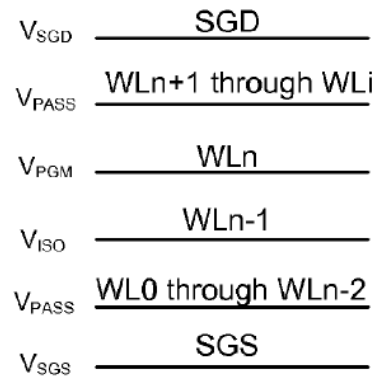
20. Fig. 1 (and most other figures in the '941 patent), show various voltages (*e.g.*, labelled “V1,” “V2,” “V3,” etc.) assigned to wordline groupings. These mappings are analogous to Dong Figs. 5a and 5g, which also depict voltage assignments. In both cases, the voltage mappings are equally applicable to 2D and 3D memory strings to address program disturb on the string being programmed and on an adjacent memory string connected to the same drain select gate. I explain this further below in §III.A.3.

21. The relevance of the '941 figures is to establish a memory string indexing scheme and provide voltage mappings (which is exactly what Dong Figs. 5a and 5g do). The fact that the '941 the figures “illustrate a memory” (Ex. 1001, 2:37-55) does not change anything. Nothing about the '941 patent would change if the figures were labelled “programming approach for a memory” instead of a “memory ... operated in another condition.” *Id.*, 2:38-39. Nor would Dong be meaningfully different if Figs. 5a-5h were labelled “examples of [memory] self-boosting modes.”

22. Patent Owner makes much of the fact that the '941 figures have a vertical orientation. POR, 16-18. But if the orientation of the figures is sufficient to convey a vertical 3D memory string architecture, the same is true of the prior art (*viz.*, Dong Figs. 5a and 5g). The side-by-side figures below show an exemplary image from the '941 patent (on the left) and Dong (on the right).



'941 patent, Fig. 8



Dong, Fig. 5a

Rather than labelling the voltages with names as Dong does, the '941 patent uses a numbering scheme where V71 refers to a “first voltage,” V72 refers to a “second voltage,” V73 refers to a “third voltage,” and so on. Ex. 1001, 4:37-61. Given that '941 patent Figs. 1-11 follow a similar scheme, under Patent Owner’s reasoning, they would all qualify as “voltage charts” subject to the same criticism Patent Owner levels against Dong.

23. In summary, to the extent a POSITA would understand the '941 patent figures to disclose vertical 3D memory strings, a POSITA would understand Dong Fig. 5 the same way. I discuss this further in §III.A.1. In other words, a POSITA would not construe *bottom to top* as limited to vertical memory strings in reliance on the '941 figures, while at the same time interpreting the substantively identical figures in Dong (which expressly discloses “three dimensional” memory) as being limited to 2D memory (and thus excluding vertical memory strings).

24. Regardless of whether Ex. 2011 (Dong III) and Ex. 2012 (Ghai) are considered intrinsic or extrinsic evidence, Petitioner’s reliance on those references is misplaced. Both of them show a “U-shaped” memory string where “bottom 238” (Ex. 2011) and “back gate ‘BG’” (Ex. 2012) are in the ***middle*** of the “U-shaped” memory string. Ex. 2011, Fig. 2C, [0007]-[0009] (Figs. 2A-C are “U-shaped NAND embodiment”); Ex. 2012, Fig. 8, [0015] (“FIG . 8 is ... a U-shaped 3D NAND string”). This confirms that Petitioner’s construction—which never places the “bottom” of a memory string directly in the middle of a memory string—is more sensible. A POSITA would not understand the ’941 patent claims to exclude U-shaped 3D memory.

25. As noted above (and confirmed by Patent Owner’s dictionary), ***bottom*** and ***top*** are relative terms. Here, a POSITA would understand that the reference points are the source and drain, as demonstrated by numerous references. Patent Owner does not dispute that some references (including 2D NAND references) use the source/drain memory string nomenclature. POR, 25. Rather, Patent Owner argues that Petitioner’s construction is wrong because it produced some counterexamples that “show different arrangements.” *Id.* That misses the point. The fact that a minority of references disclose indexing memory strings from drain-to-source (top to bottom) instead of source-to-drain (bottom to top) does not change that the far more common convention was to refer to the source as the ***bottom*** of the

memory string and the drain as the *top*.¹ Whether a handful of references deviated from the accepted convention has no bearing on whether the '941 patent did so. It did not, as reflected by the specification's silence on the topic.

26. Numerous references confirm the traditional nomenclature with the source as the bottom and drain as the top of a memory string. *See, e.g.*, Ex. 1009 (Aritome) at 297 ("The bottom of memory string is connected to the common source diffusion."), 38 (discussion of connecting "SGD" to string's drain side and "SGS" to string's source side), 39 (figure showing 32-cell NAND string with SGS at bottom and SGD at top); Ex. 1010 (Micheloni) at p. 122 (showing relationship between structure of 2D and 3D NAND strings by turning 2D NAND string so it "stands up"); Ex. 1011 (Silvagni) at p. 3 (shows relationship between 2D and 3D NAND as standing the string on its end with current flowing from the bottom of the 3D NAND string), 4 ("It can be noted that the NAND string is vertical, and the string current flows in the vertical direction and is collected by a drain contacting the top of the

¹ Drain-to-source programming has the drawback that it is difficult to pass the bitline voltage (*e.g.*, 0V) through the memory string during programming, because doing so requires passing the bitline voltage through memory cells that have already been programmed. For the cells that have already been programmed, a higher pass voltage is needed, which introduces even more disturb issues during programming.

cell stack. The source current must be provided by source diffusion at the bottom.”); Ex. 1008 (Lee II), Figs. 3, 6-9, 11-12, 14-16, 18-20; Ex. 1082, 1:38-2:9 (“NAND arrays are well known in the art and are widely utilized in various consumer devices”), 1:59-63 (The “top and bottom of the string connect to the bit line and a common source line respectively”), Fig. 2A; Ex. 1083, 3:25-33 (“data is written on a memory block in a bottom-up fashion, starting from memory cells on or near the lowermost (or bottom) wordline. Then data is sequentially written page-by-page on memory cells on wordlines above the lowermost wordline until memory cells on the uppermost (or top) wordline are filled up”); Ex. 1084, Fig. 1D (showing 3D NAND architecture with both memory string ends at the same z-axis level due to “U-shaped” architecture); Ex. 1085, 121:41-46 (applicable to both 2D and 3D NAND, “each string comprising M cells connected in series and capped by a top string-select transistor and a bottom string-select transistor, the top string-select transistor connecting its drain to one corresponding [bit line], the bottom string-select transistor connecting its source to a common source line”); Ex. 1086, [0018] (YMTC patent application listing two inventors in common with the ’941 patent, referring to a “top select gate SG_T (on the drain side) and a bottom select gate SG_B (on the source side)”), [0020]-[0021]. Thus, numerous textbooks and other references confirm that Petitioner’s construction is generally accepted.

B. “*program operation*”

27. The term *program operation* appears in all independent claims (*i.e.*, claims 1, 5, and 11). The plain and ordinary meaning of *program operation* to a POSITA includes all of the steps necessary to program a memory cell to store a data bit (*i.e.*, a logic “1” or “0”). The whole point and purpose of performing a *program operation* is to store one or more bits of data in a memory cell. Historically, a NAND flash memory cell stored a single bit of data (*i.e.*, a “1” or a “0”). To increase memory density, so-called “multi-level cells” developed that allowed a single memory cell to store two (or more) bits of data. This was done establishing multiple voltage levels in a cell, where each level corresponded to a different sequence of digital values. This advance require even more carefully controlling (and measuring) the amount of charge stored in a cell.

28. In either scenario (single-level cell or multi-level cell), A POSITA would understand that a *program operation* includes iteratively performing two steps: a charge injection phase (sometimes called a programming phase) where a programming pulse is applied, and a verify phase (*i.e.*, determining if additional charge is needed). The steps are iterative because, if verification confirms that insufficient charge was stored during charge injection, that another charge injection pulse will be applied to the memory cell. This process continues until the cell is programmed. Thus, both phases/steps are necessary to ensure that the appropriate

level of charge is stored. Otherwise, the system will not know whether the correct data value(s) are being stored, which would defeat the whole purpose of the **program operation**. Both the intrinsic and extrinsic evidence confirm that **program operation** includes both phases.

29. To begin, the intrinsic evidence supports Petitioner's construction that **program operation** entails both charge injection and verification. The '941 patent states that "[a] program operation said in the text may be an operation for programming a memory cell." Ex. 1001, 3:6-7. It was (and is) very well-known that in digital systems (like NAND flash memory), memory cells are programmed to store values representing logic "1s" and "0s" (or a combination of "1s" and "0s" in the case of a multi-level cell). A POSITA would not say that a **program operation** had been completed until the memory cell has sufficient charge stored to represent the desired data value(s).

30. Under Patent Owner's incorrect construction, merely storing a small amount of charge insufficient to represent the desired logic value would be considered a **program operation**. That makes no sense, and is not how a POSITA would understand that the term. A **program operation** is not complete until the correct amount of charge is stored, which typically requires multiple iterations of charge injection and verification. This is because it is difficult to store just the right amount of charge on the first try. And over-programming a memory cell is highly

undesirable because, in NAND flash memory, you cannot “erase” just one cell. As is well-known, performing an “erase” operation requires erasing an entire block of memory cells at once. Thus, an incremental programming approach is used in which multiple charge injection and verification steps are iteratively performed.

31. Patent Owner says that “the disclosed voltages [in the ’941 patent] only accomplish the stated aim to reduce program disturb if applied **during** a program operation.”² POR, 31 (emphasis by Patent Owner). But the ’941 patent contradicts this, stating that it is directed to providing “a solution for reducing program disturbance **and pass voltage disturbance** when operating a three-dimensional memory” Ex. 1001, 1:39-41. “Pass voltage disturbance” occurs during both the program phase **and** the verify/read phase. Ex. 1094, 184:5-7 (“So pass voltage disturb would occur on any transistor that has a pass voltage applied to it.”), 185:19-20, 197:4-22, 193:3-8.

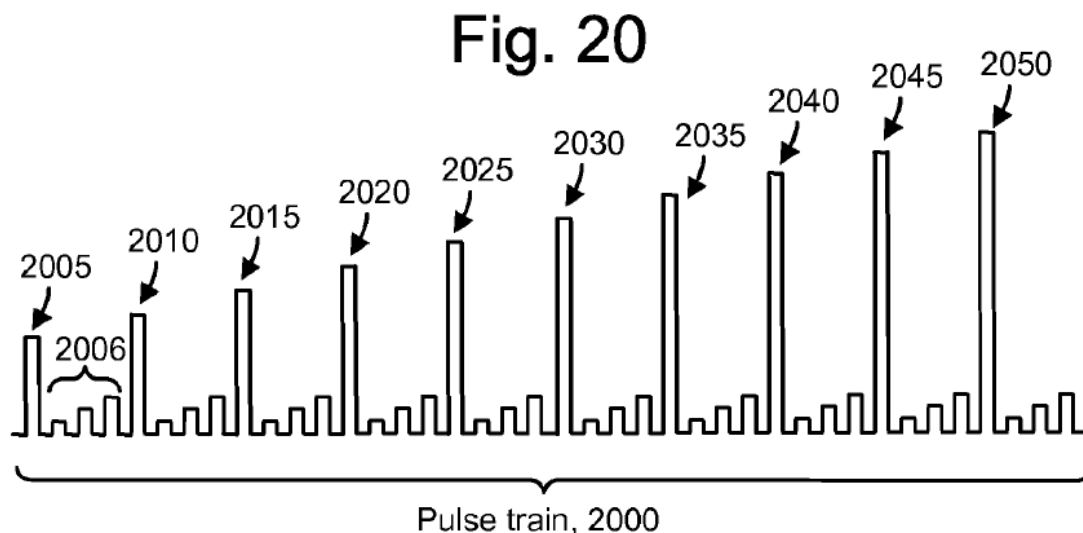
32. Numerous references show that a pass voltage (*e.g.*, V_{pass}) is applied to unselected memory cells during the verification phase of a program operation. Ex. 1012 (Costa), Fig. 14, [0033] (“at least a subset of unselected memory cells that

² PO’s emphasis on “during” is unavailing. None of the claims use that term, but even if they did, PO’s argument is circular as Petitioner’s position is that **program operation** includes verification.

have already been subjected to programming receives the variable pass voltage”), [0112], Fig. 10; Ex. 2024, Fig. 2 (showing that “V_{PASS}” is applied during the “Verify Phase”); Ex. 2027, Fig. 2 (similar). In other words, the voltage techniques disclosed in the ’941 patent are directed to improving the performance of **both aspects** of programming. The ’941 specification explains that “by using a memory sectioned to have a plurality of portions and applying voltages to the word lines of the memory according to the portions, **program disturbance and pass voltage disturbance** may be reduced.” Ex. 1001, 6:39-43. That description precisely corresponds to the claims, which recited various **portions** to which **a first voltage, second voltage, third voltage**, etc. are applied. *See id.*, all claims. Thus, the intrinsic evidence supports Petitioner’s construction which recognizes that a **program operation** is comprised of two parts.

33. Patent Owner argues that Petitioner’s construction cannot be right because “the ’941 Patent does not describe a verify phase.” POR, 32. That is incorrect for the reasons I explained above (*viz.*, discussion of pass voltage disturbance). Regardless, to the extent the ’941 patent does not elaborate on verification, neither does it elaborate on charge injection. The reason the ’941 specification does not provide details on either charge injection or verification is because it was well-known to POSITAs that programming a memory cell entails both phases.

34. Claims 8 and 9 refer to programming memory cells located in different **portions** of a memory string (viz., increasing the **first voltage** as programming progresses along a string). See Ex. 1001, cls. 8-9. This supports Petitioner’s position because programming only advances to the next memory cell if the previous **program operation** is finished, which requires both the charge injection and verification aspects of programming. See, e.g., Dong, Figs. 19-20, [0128]-[0133] (“In between the program pulses are verify pulses.”). Indeed, Dong Fig. 20 (shown below) “depicts an example pulse train 2000 applied to the control gates of non-volatile storage elements **during programming.**” *Id.*, [0132]. As is apparent, the “pulse train 20” includes both the charge injection (program) pulses 2005, 2010, etc. **and** the “verify pulses 2006.” *Id.*



Id., Fig. 20, [0132]. A POSITA looking at Dong Fig. 20 would understand that it represents a single **program operation** for programming just one memory cell, not

ten separate **program operations** for programming one cell ten times (or for programming ten different cells).

35. Numerous NAND flash references confirm that Petitioner’s construction—which includes both phases—is correct. *E.g.*, Ex. 1090, 9:14-16 (“a program operation having cycles of alternate program and verify phases”); Ex. 1091, 21:1-11 (“A program operation can include a ... verify phase.”); Ex. 1092, 5:7-9 (“In the program operation, the address decoder 120 may provide a program/verify voltage V_{pgm}/V_{fy} to a selected word line and a pass voltage V_{pass} to an unselected word line(s).”); Ex. 1093, 27:22-29 (“During a program operation, ... a program voltage or pulse is applied to the selected word line followed by a verify portion in which a verify signal is applied”).

36. I must also comment on Patent Owner’s accusations that my testimony here (regarding the ’941 patent) is inconsistent with my testimony in other proceedings. POR, 33-34. It is not. If you read my testimony in context, I was clearly describing the same two-part program operation I discuss here. *See, e.g.*, Ex. 2022, ¶203 (“Suzuki discloses a program loop comprising a program operation followed by a verify operation.”). I have always maintained that programming a memory involved a charge injection phase **and** a verification phase. Patent Owner’s argument that I sometimes used the term “operation” instead of “phase” when referring to charge injection (in a different reference and context) elevates form over

substance. In other words, the charge injection operation/phase is merely a part (but not the entirety) of the overall program operation used to program a memory cell.

37. Relatedly, Patent Owner’s argument that I “revealed a new position during [my] deposition, [that] ‘a program operation’ [] include[s] both a program phase and verify phase” (POR, 29) is baseless. My original declaration (Ex. 1004) said that “[t]he other phase of a typical programming operation is the verify phase where the memory cell being programmed is read to determine it has been sufficiently programmed.” Ex. 1004, ¶60. Indeed, ¶60 of my original declaration expressly refers to “a typical programming operation,” belying Patent Owner’s assertion (POR, 30) that ¶60 “does not ... even mention ‘a program operation.’” *Id.*

III. REBUTTAL ANALYSIS AND OPINIONS

A. Grounds 1 and 2 (Anticipation and Obviousness Over Dong)

1. A POSITA Would Understand Dong to disclose 3D NAND (with Vertical Memory Strings) in 2020

38. I have been asked to respond to Patent Owner’s assertions that a POSITA reading Dong in 2020 would understand it to exclude 3D memory (*viz.*, with vertical memory strings). See POR, 42-43 (“[a] POSITA would have interpreted Dong’s mention of ‘three dimensional’ memory arrays as naturally referring to 3D memory arrays that ... did not have vertically arranged wordlines”). That assertion is absurd as I explain below. I am informed by counsel for Petitioner that validity (both anticipation and obviousness) is assessed in view of a POSITA’s

knowledge as of the date of the claimed invention (*i.e.*, February 10, 2020 for the '941 patent), not the date of the prior art (*i.e.*, May 7, 2007 for Dong). The notion that a POSITA reading Dong in 2020 would think of stacked 2D NAND is far-fetched. Dr. Khatri's deposition testimony on this point was revealing. When asked when 3D NAND was developed, he admitted that "I just wouldn't know ... at what time frame 2D NAND sort of transitioned to 3D NAND." Ex. 1094, 73:10-25.

39. A POSITA reading Dong in 2020 would immediately envision (and most certainly understand Dong to disclose) vertical 3D memory for multiple reasons. To begin, Dong expressly discloses that "array of storage elements can be three dimensional." (Dong, [0095]). Dong makes this statement when describing Fig. 12, which is a figure of general applicability vis-à-vis the memory strings in Figs. 5a-h. Dong, [0095], Fig. 12.³ Thus, Dong expressly discloses "three dimensional" (3D) memory, the most common form of which (by a long shot) had vertical memory strings.

³ Petitioner expressly identified Dong Figure 12 as disclosing the **memory** recited in the preamble. Pet., 34 (mapping **memory** as including Dong's "memory array 1100" in Figure 12). Dong's "three dimensional" memory disclosure is in the specific context of Figure 12's "memory array 1100" on which Petitioner relies. Dong, [0095].

40. Indeed, 3D memory in 2020 (and for years before) was typically characterized by the number of layers that formed the vertical stack structure through which the vertical memory strings were formed (with the number of memory cells in the vertical string corresponding to the number of layers in the stack). In fact, in 2020, “three dimensional” (3D) memory was practically synonymous with vertical memory strings. Regardless of how well-known vertical 3D NAND was in 2007 (which I discuss below), it was ubiquitous in 2020. As such, in view of a POSITA’s knowledge in 2020, a POSITA would immediately think of and associate Dong “three dimensional” memory disclosure with vertical memory strings.

41. A simple survey of the state of technology and market trends around the time of the ’941 invention would have shown that virtually all the major 3D NAND flash memory makers were focusing on increasing the number of layers on a vertical string of NAND memory cells, not on stacking different 2D NAND arrays on top of each other.

42. As an illustration to support this conclusion, a summary on the 3D NAND flash memory development status published by Block and Files, a storage news and analysis site, clearly shows 3D vertical NAND strings were the focus. The table below illustrates the trend of increasing the layers (which correspond to cells in a vertical memory string) in 3D NAND flash memory. The full article is entitled Micron reveals Flash roadmap to 500+ layer 3D NAND. Although the article is

published in 2022, but the data and trend date back to 2018.

3D Layer Cake											
SK hynix		SK hynix Solidigm		Western Digital/Kioxia		Samsung		Micron		YMTC	
Generation	Layers	Generation	Layers	Generation	Layers	Generation	Layers	Generation	Layers	Generation	Layers
V3	48	Gen 1	32	BiCS 2	48	V3	48	Gen 1	32		
V4	72	Gen 2	64	BiCS 3	64	V4	64	Gen 2	64	Xtacking 1	64
V5	96	Gen 3	96	BiCS 4	96	V5 (May 2018)	9x	Gen 3	96		
V6 (Q4, 2019)	128	Gen 4 (Q4 2020)	144	BiCS 5 (Q1, 2020)	112	V6 (June 2019)	128	Gen 4 (Q4, 2019)	128	Xtacking 2	128
V7 (2021)	176	Gen 5? (2023)	196?	BiCS 6	162	V7 (April 2021)	176	Gen 5 (Nov 2020)	176		
V7(2023?)	238			BiCS 7	212	V8 (July 2021)	200	Gen 6 (End 2022)	232		
				BiCS 8	?	V9	3xx	Gen 7	2yy		
V? (2025)	500+					V10		Gen 8	3xx		
V? (2030)	800+					V11	5xx	Gen 9	4xx		

<https://blocksandfiles.com/2022/05/13/micron-flash-roadmap/> (Ex. 1106).

43. I note that even YMTC (in the rightmost column in the table above) was focusing on increasing the layers of vertical 3NAND flash memory. It is also noteworthy that despite Patent Owner's assertion of stacked 2D NAND being familiar to a POSITA, YMTC itself only focuses on development of NAND flash memory with vertical memory strings.

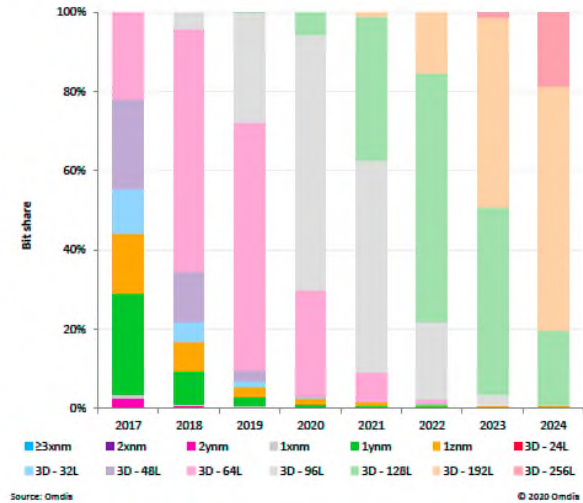
<https://www.techpowerup.com/258862/yangtze-memory-begins-mass-production-of-64-layer-3d-nand-flash-memory> (Ex. 1107).

44. As another example that 3D NAND in 2020 was predominately vertical memory strings (and would not even remotely be 2D stacked arrays) is a chart compiled by OMDIA, a global research and technology advisor firm that provides market intelligence, data and insights.

NAND lithography transitions (1/3)

- The production of 96-layer took over that of 64-layer earlier in 2020 to occupy the NAND bit volume. Omdia projects 96-layer will remain as the volume output through most of 2021, giving the next-generation 128-layer time to mature and for yields to improve. As the industry continues through the layer-count roadmap, yield and maturity will take longer to reach appropriate mass production levels.
- In 2020, 96-layer will account for 65% of total NAND bit production, while 128-layer will account for 5.5%. By 2021, 96-layer will account for 53% of bit production while 128-layer will grow to 37% overall for the year.
 - There will be variances in the industry's 128-layer by vendor. As an example, Kioxia and WD announced a 112-layer while Intel announced a 144-layer.
 - Omdia expects that layers counts beyond "128-layer" will continue to vary by vendor. Omdia currently is showing "192-layer" as the next layer count, then followed by a "256-layer." As vendors announce their next-generation layer counts, Omdia will continue to update the data set.
- The 12nm process node will be the final generation of planar NAND. Planar NAND production will continue, driven by segments that demand lower density die, not offered by 3D NAND technology. The industrial and automotive markets (for example) will likely maintain their demand for planar technology because they are not willing to pay for unneeded higher capacity or to go through long qualification processes.

Figure 35: Industry lithography transition



Page 34

© 2020 Omdia

OMDIA

Ex. 1103, 34.

45. This chart illustrates that in the 2019-2020 timeframe, 2D NAND flash memory (planar NAND) production had only marginal market share. Vertical 3D NAND comprise at least 95% of the NAND flash market in 2019, and approximately 99% in 2020. It was simply not economical to consider stacking 2D NAND into stacked 3D NAND due to lack of availability of high density 2D NAND in the relevant timeframe.

46. Samsung's 3D NAND (VNAND) production timeline (shown below) provides another supporting datapoint that 3D NAND predominately used vertical NAND strings by 2020. Again it is clear that the focus for this 3D NAND flash leader was on vertical strings, not at all on stacked 2D NAND.

Reference: Samsung V-NAND Mass Production Timeline

Date	V-NAND
July 2013	1st-generation (24-layer) 128Gb MLC V-NAND
Aug. 2013	1 st -generation 128Gb MLC V-NAND 960GB SSD
Aug. 2014	2nd-generation (32-layer) 128Gb 3-bit V-NAND
Sept. 2014	2 nd -generation V-NAND SSD
Aug. 2015	3rd-generation (48-layer) 256Gb 3-bit V-NAND
Sept. 2015	3 rd -generation V-NAND SSD '850 EVO', '950 PRO'
Dec. 2016	4th-generation (64-layer) 256Gb 3-bit V-NAND
Jan. 2017	4 th -generation V-NAND SSD
Jan. 2018	4 th -generation 512Gb V-NAND 30.72TB SAS SSD
May 2018	5th-generation (9x-layer) 256Gb 3-bit V-NAND
June 2018	5 th -generation V-NAND SSD
June 2019	6th-generation (1xx-layer) 256Gb 3-bit V-NAND
July 2019	6 th -generation V-NAND SSD

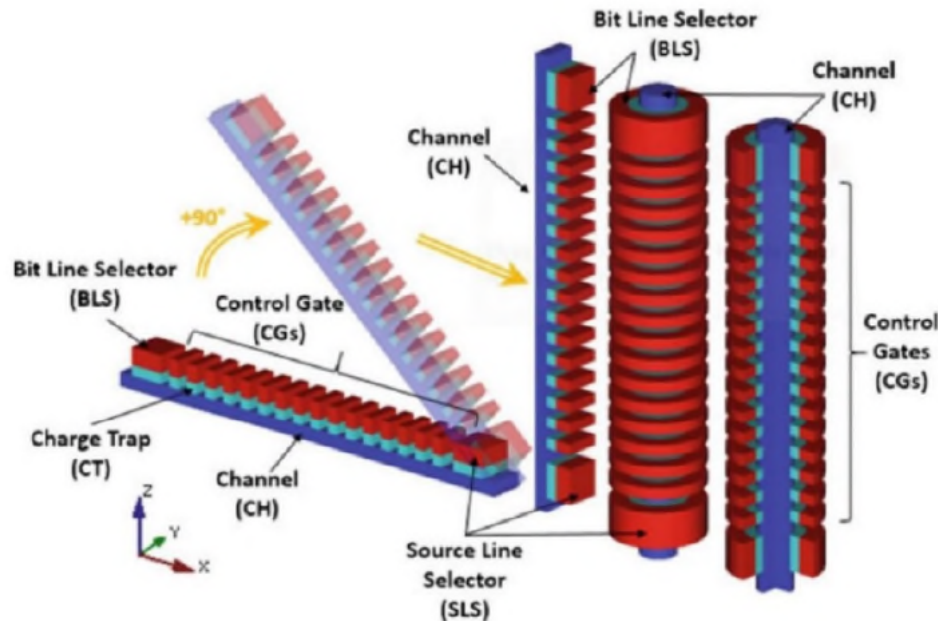
<https://news.samsung.com/global/samsung-electronics-takes-3d-memory-to-new-heights-with-sixth-generation-v-nand-ssds-for-client> (Ex. 1104); *see also* <https://news.skhynix.com/sk-hynix-starts-mass-producing-worlds-first-128-layer-4d-nand/> (Ex. 1105).

47. Any engineer working on 3D memory in 2020 that did know that 3D memory typically had vertical memory strings would soon find themselves out of work. Patent Owner suggestion that a POSITA in 2020 would not think of vertical memory when reading Dong's "three dimensional" disclosure ignores reality. Even

Dr. Khatri admitted that vertical NAND was “used in [the] industry” in the 2020 timeframe. Ex. 1094, 145:19-146:4, 129:24-130:7 (admitting there existed “a large number of 3D NAND offerings by that [2020] time frame”), 132:21-133:2 (“In 2020 ... there was 3D NAND in which cells were arranged vertically from bottom to top”). And he was unaware of anyone making stacked 2D NAND in 2020. *E.g., id.*, 127:12-16, 128:2-11, 129:16-23.

48. And PO does not dispute that a POSITA in 2020 “would have been aware of and generally knowledgeable about NAND memory’s structure and fabrication, its component parts, how it operates, and how it is controlled.” Pet., 22-23; POR, 9 (“[PO] does not dispute Petitioner’s definition of a POSITA for purposes of these proceedings”); Ex. 2028, ¶46 (“I am informed by counsel that Patent Owner does not dispute Petitioner’s definition of a POSITA for the purposes of these proceedings”).

49. This knowledge includes, as my opening declaration explained using pre-2020 textbooks, that “[a] POSITA would have known that ‘[c]onceptually, a modern 3D NAND structure could be conceived by starting from the 2D NAND string building block’ and ‘turning the 2D NAND string upside,’ ‘resulting’ in a ‘3D structure’ with a “vertical channel.” Ex. 1004, ¶52 (citing Ex. 1010, p. 122, Ex. 1011, p. 3).



Ex. 1010, p. 122. As a practical matter, by 2020, the very essence of “three dimensional” NAND was explained to students as embodying vertical memory strings.

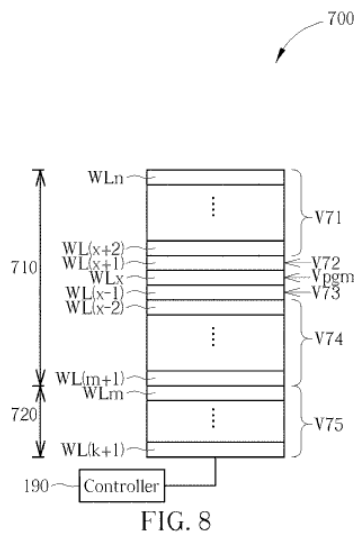
50. In various locations Patent Owner asserts that Dong “teaches wordlines arranged horizontally.” *E.g.*, POR, 36. In support, Patent Owner cites extensively to Dong Figure 4 and reproduces it several times (*see* POR, 10, 37, 39). But there was no reason for Petitioner to cite Fig. 4 because it does not disclose the programming voltages and groups recited in the ’941 claims. Moreover, the purpose of Fig. 4 was to “show[] a program disturb mechanism” (Dong, [0019], [0045]), not

to limit Dong's disclosure to a 2D NAND string.⁴ Thus, Dong Fig. 4 in no way restricts the scope of Dong's additional disclosure of "three dimensional" NAND (Dong, [0095]). Patent Owner reasons that, because Fig. 4 shows a 2D NAND string, Fig. 5c (and, by extension, Figs. 5a and 5g) must necessarily be limited to 2D NAND. But there would be no reason to present Fig. 5c at all if it was only intended to convey the exact same thing as Fig. 4. But a POSITA would understand that the biggest difference between Fig. 4 and 5c is the vertical orientation of the memory string in Fig. 5c (as well as Figs. 5a and 5g). Further, Patent Owner's attempt to restrict Figs. 5a-h (which are vertically oriented in the same manner as the '941 figures) to 2D/horizontal memory ignores Dong's disclosure of "three dimensional" NAND.

51. Patent Owner also faults Petitioner for pointing to Dong's vertically-oriented diagrams in Figs. 5a and 5g. *E.g.*, POR, 37. But these are the diagrams that show the relevant wordline arrangements and programming voltages recited in the '941 claims. As noted, Dong Fig. 4 does not disclose the claimed programming voltages so there was no reason for Petitioner to point to Fig. 4. Patent Owner

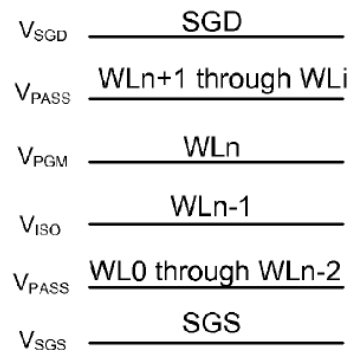
⁴ Patent Owner's argument that Dong Figs. 5a-h are oriented vertically only "for ease of reading" (POR, 37) is belied by Fig. 4 itself, which presents the wordline programming voltages horizontally in legible fashion.

characterizes Dong Figs. 5a-h as “voltage diagrams” which “do not show the physical structure of Dong’s memory,” and thus do not “disclose vertical word lines as recited in the independent claims of ’941 Patent.” Paper 10, 23-24; *see also* POR, §§III.A-B (similar). But these are precisely the types of diagrams found in the ’941 patent, which Patent Owner contends disclose a vertical **bottom to top** arrangement. *E.g.*, POR, 17 (stating that all ’941 figures “illustrate wordlines arranged vertically, as in Figure 8”). The side-by-side figures below compare a representative image from the ’941 patent (on the left) and Dong (on the right).



’941 patent, Fig. 8

Fig. 5a



Dong, Fig. 5a

52. Rather than labelling the voltages with names as Dong does (*e.g.*, V_{ISO} , V_{PASS} , etc.), the ’941 patent uses a numbering scheme where V71 refers to a “first voltage,” V72 refers to a “second voltage,” V73 refers to a “third voltage,” and so on. ’941, 4:37-61. Given that ’941 Figs. 1-11 follow a similar scheme, under Patent

Owner's reasoning, they would all qualify as "voltage diagrams" subject to the same criticism that Patent Owner levels against Dong. Moreover, taking '941 patent Fig. 8 as an example, it shows a "Controller" connected to a single wordline (WL(k+1)). If Fig. 8 were intended to illustrate the physical structure of a memory string, it would not work because the controller would need to access and control *all* wordlines. Thus, PO's reliance on the '941 patent figures to support its interpretation of *bottom to top* employs a double standard.

53. At Institution, the Board stated:

Unlike Figure 8 of the '941 patent, which depicts "memory 700", we do not understand Dong Figures 5a-5h to depict the physical structure of a memory cell, but rather to depict a self-boosting mode. Ex. 1001, 4:29-30 (stating Figure 8 illustrates "memory 700"); Ex. 1005 ¶20 (stating "FIGS. 5a-h depict different examples of self-boosting modes") The parties are invited to address this issue if they so choose.

Institution Decision, fn. 6.

54. The presence or absence of a generic "memory" label/numeral makes no difference to a POSITA's understanding of either Dong or the '941 patent. Even Dong Fig. 4 is not called a "memory," but merely a "NAND string." Dong, [0045]-[0046]. Moreover, Dong Fig. 4 shows programming voltages. Likewise, Figs. 5a-h also depict "storage elements which are being programmed ... [in] selected NAND strings." Dong, [0047]-[0048] et seq. More generally, the entire Dong reference is

about NAND flash memory and is replete with references to “memory.” Dong, [0003] (“The present invention relates to non-volatile memory”); *see generally* Dong. As such, labeling Dong Fig. 4, 5a, or 5g as “memory” would change nothing.

55. Even if using the term “memory” were necessary (it is not), Dong does so in Fig. 12, which “is a block diagram of a **non-volatile memory** system” and plainly shows “MEMORY ARRAY 1100.” Dong, [0095]. The diagram illustrates “a **memory device 1296**,” with “one or more **memory die 1298**” which includes either “a two-dimensional array of storage elements 1100” or a “**three dimensional**” array of memory elements. *Id.* A POSITA would readily understand that Figs. 5a and 5g would be implemented in “MEMORY ARRAY 1100” in Fig. 12, which can be “three dimensional.” Dong, Fig. 12, [0095].

56. In conclusion, in view of the well-developed state of 3D NAND memory in 2020, a POSITA would at once envisage the ’941 patent’s claimed **bottom to top** wordline arrangement upon reading Dong’s disclosure of “three dimensional” NAND (Dong, [0095]) and viewing the vertically oriented memory strings in Figs. 5a and 5g. Certainly, at an absolute minimum, in conjunction with Dong’s disclosure that its memory can be “three dimensional,” the vertically oriented Figs. 5a and 5g can fairly be read as disclosing vertical memory strings.

**2. A POSITA Would Understand Dong to disclose 3D NAND
(with Vertical Memory Strings) in 2007**

57. I am informed by counsel for Petitioner that anticipation and obviousness are to be evaluated as of the '941 patent's priority date in 2020 in view of the knowledge of a POSITA at that time. Nevertheless, I have been asked to opine on how a POSITA would understand Dong's disclosure in 2007 to address Patent Owner's assertion that "at the time of Dong's May 7, 2007 filing, 'three-dimensional' memory would not have been understood by a POSITA to disclose a memory structure with word lines arranged vertically." POPR (Paper 11), 47-49; *see also* POR, 42-43 (similar). I disagree with Patent Owner that a POSITA would understand Dong's disclosure (*viz.*, that its "array of storage elements can be three dimensional" (Dong, [0095]) and its vertically oriented memory strings diagrams in Figs. 5a and 5g) to exclude vertical 3D memory. To the contrary, a POSITA would readily appreciate that vertical memory strings were within the scope of Dong's disclosure.

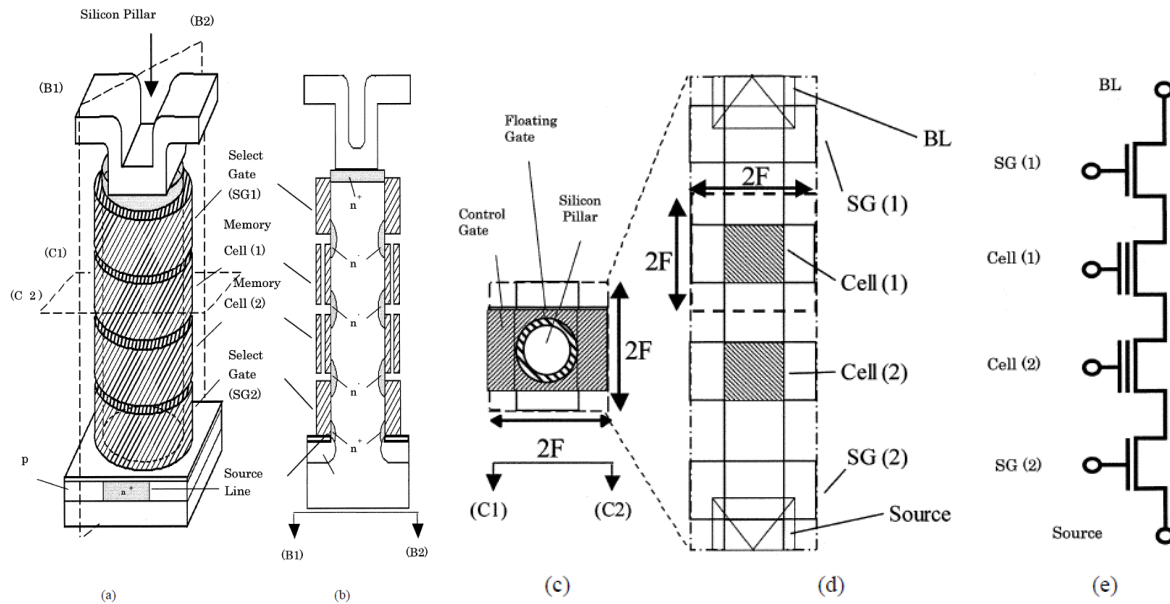
58. In this regard, Patent Owner's principal argument is that "three dimensional" NAND memory in 2007 "***included***" stacked 2D NAND. POR, 5. But that does not mean the "three dimensional" memory ***excluded*** vertical memory strings. It did not. Multiple references (discussed in the next several paragraphs) confirm this. And Dr. Khatri admitted as follows at deposition:

Q. Do you have an opinion one way or another, Dr. Khatri, whether 3D NAND at the time of Dong's filing ***excluded*** vertical 3D NAND memory strings?

A. ***Again, I have not said that.*** My statement is simply that ... 3D NAND at the time of Dong's filing ***includes*** stacked 2D NAND.

Ex. 1094, 103:6-15.

59. As early as 2001, researchers were publishing about using vertical memory strings for flash memory to increase density. Endoh1 (Ex. 1101) published in 2003 in a well-respected IEEE journal, and was a continuation of Endoh2 (Ex. 1102) by the same lead author which had published in 2001. Endoh1 disclosed “[t]he structure of the proposed S-SGT structured Flash memory cell arranges bit line contact, first select gate, two SGT type memory cells, second select gate and source line in series vertically in one silicon pillar.” Ex. 1101, 2. “The SGT type memory cell is structured that floating gate and control gate surround the silicon pillar [7] and gates, source, and drain are arranged on a silicon pillar vertically. All devices for the proposed cell are vertically stacked in one silicon pillar.” *Id.* Endoh1's vertical flash memory string is shown in Figs. 3(a)-(e) below.



Id., Figs. 3(a)-(e). Endoh2's disclosure from 2001 is quite similar. Ex. 1102, Fig. 1(a)-(e), 1 (previous "two-dimensional memory array architecture doesn't use three dimensional space sufficiently").

60. The Kito reference (Ex. 1087) provides a more comprehensive example of a wholistic memory device. Kito was filed on January 18, 2007 and published later that same year (on November 1, 2007). Kito is a long reference and discloses extensive information about "three dimensional" 3D NAND, including vertical memory strings and details about fabrication. Right in the Abstract, Kito discloses a "semiconductor memory device ... in which memory cells are laminated in a **three dimensional state.**" Ex. 1087, Abstract. Kito contains numerous figures (such as Fig. 17 below) illustrating the overall 3D architecture, including vertical memory strings.

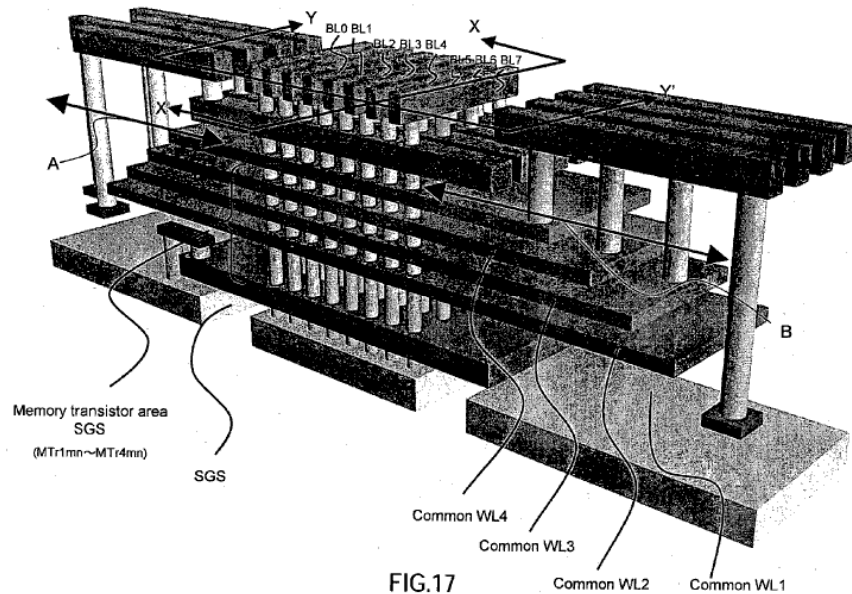


FIG.17

Id., Fig. 17; *id.*, Figs. 1, 2-3 (vertical strings).

61. Other figures, like Fig. 2 and 3(a) below show additional close-up details of vertical memory strings.

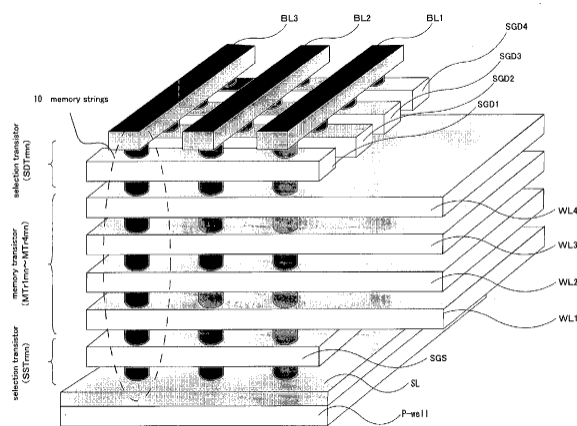


FIG.2

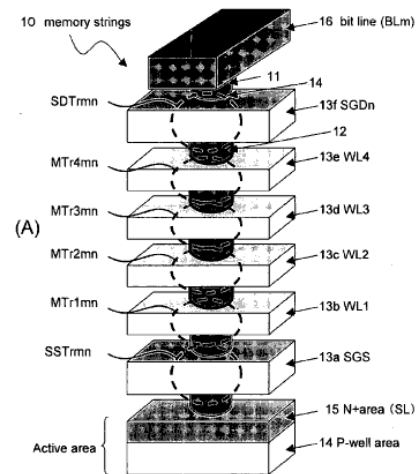


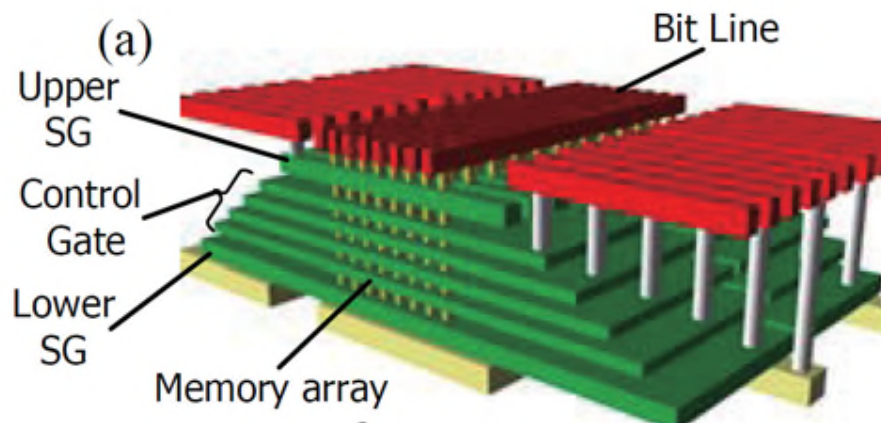
FIG.3

Ex. 1087, Fig. 2 (at left) and 3(a) (at right).

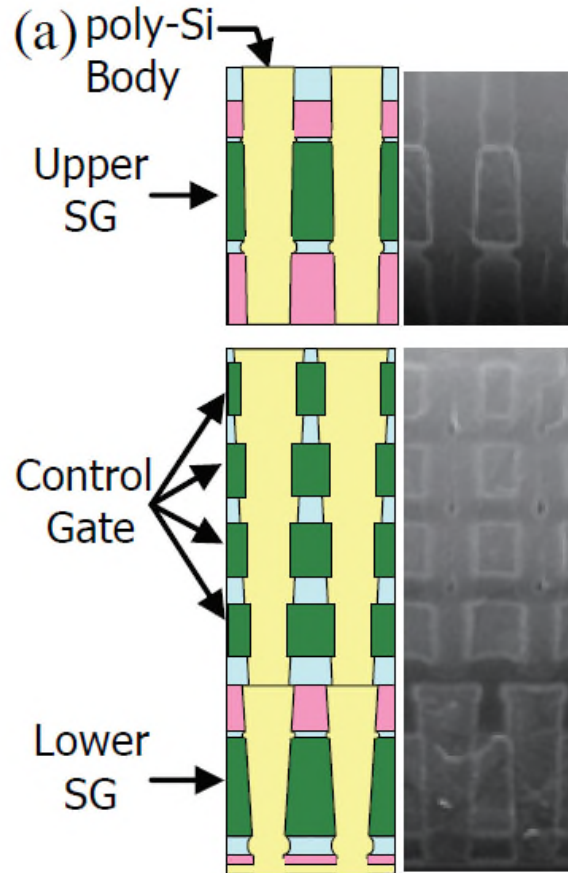
62. I found it telling the Dr. Khatri was unable to say at his deposition whether these figures represented a vertical or horizontal memory string. Ex. 1094,

104:24-122:5. Such a response is simply not credible as any POSITA would immediately recognize these images as vertical 3D memory strings.

63. The 2007 Tanaka article (Ex. 1075) also discloses a vertical memory string architecture for a 3D memory device.



Ex. 1075, Fig. 3(a). Tanaka is further notable because it also shows the cross section of a vertical memory string from a real-life semiconductor chip. This confirms that not only were 3D vertical memory strings known in the art at a theoretic level, but had actually been manufactured as well.



Ex. 1075, Fig. 7(a); *see also id.*, Figs. 4-6.

64. Any assertion that “three dimensional” vertical strings were unknown in 2007 is unsupported. Thus, when a POSITA read Dong’s disclosure that its “array of storage elements can be three dimensional” it would immediately call to mind a vertical memory string, especially when taking into account Dong’s vertically oriented wordline diagrams in Figs. 5a and 5g.

3. Dong's Program Disturb Solutions (in Figs. 5a and 5g) Apply to 3D NAND (with Vertical Strings) and Would Have Been Obvious

65. Even if Dong's programming techniques were originally made in the context of 2D memory (they were not), they apply equally to vertical 3D NAND memory strings and a POSITA would have found it obvious to apply them in that context.

66. Patent Owner asserts that a "POSITA would not apply [Dong's programming techniques] in vertical 3D NAND" because "[p]rogram disturb phenomena and mechanisms of a 3D NAND cell are much different from that of a 2D NAND cell." POR, 50 (quoting Ex. 1009, p. 362). In relevant part, PO could not be more incorrect. The reason is simple: ***Patent Owner focuses on the wrong disturb modes.***

67. It is well known that 3D and 2D NAND flash have some structural distinctions, and as a result 3D NAND will have additional possible program disturb mechanisms. But it is misleading to categorically say "the program disturb conditions and characteristics of 3D cells are greatly changed from that of 2D cells." Ex. 2028, ¶72. A more accurate way to characterize the distinction would be that 2D NAND program disturb mechanisms are a subset of the 3D program disturb mechanisms. A fitting analogy would be 3D tic-tac-toe vs. 2D tic-tac-toe. Just because 3D tic-tac-toe has more possibilities for connecting 3 squares in a row does

not allow one to neglect the possibilities of connecting 3 in a given 2D plane.

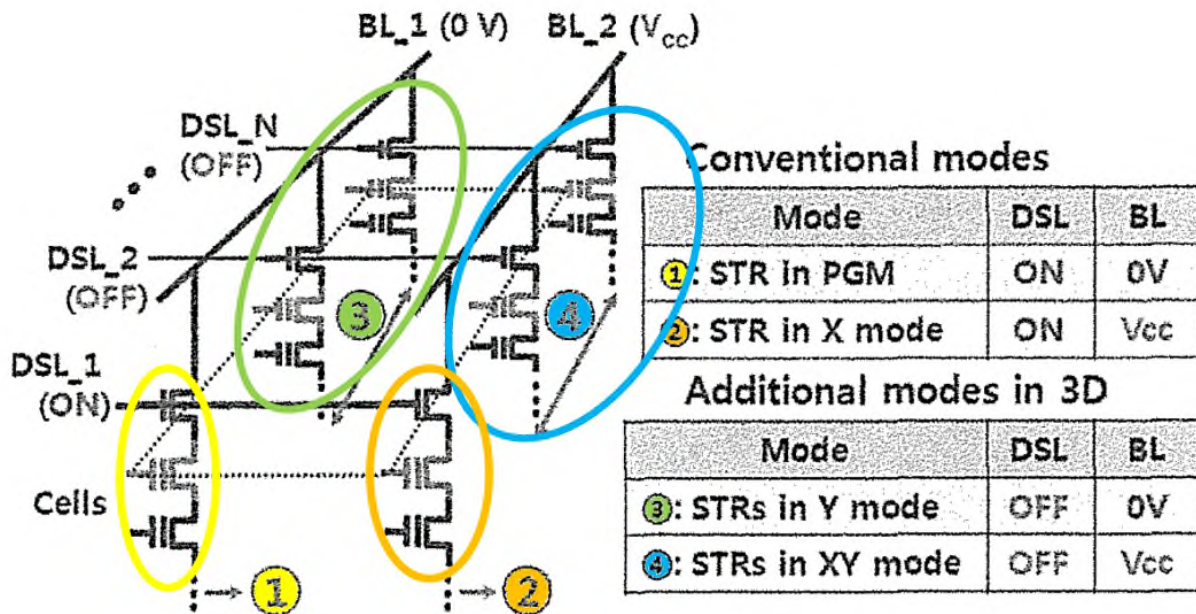
68. As discussed below, this distinction of 3D vs. 2D is supported by Aritome. The Aritome textbook Patent Owner cites (Ex. 1009) directly contradicts Patent Owner's conclusion that Dong is not applicable to 3D memory strings. Section 8.4.1. of Aritome (which is titled "New Program Disturb Modes") explains, compares, and contrasts four program disturb modes, two of which are "[c]onventional" (*i.e.*, they exist in both 2D and 3D memory) and two "[a]dditional" program disturb modes which only exist in 3D memory. Ex. 1009, §8.4.1 (pp. 362-364).

Conventional modes		
Mode	DSL	BL
①: STR In PGM	ON	0V
②: STR in X mode	ON	Vcc
Additional modes in 3D		
Mode	DSL	BL
③: STRs in Y mode	OFF	0V
④: STRs in XY mode	OFF	Vcc

Id., Fig. 8.11 (excerpted). As Fig. 8-11 shows, Aritome labels the four program disturb modes ① ("PGM"), ② ("X mode"), ③ ("Y mode"), and ④ ("XY mode"),

based on which memory string(s) are being programmed. *Id.*, p. 363, Fig. 8.11.⁵

The corresponding memory strings where each mode manifests itself are also shown in Fig. 8.11.

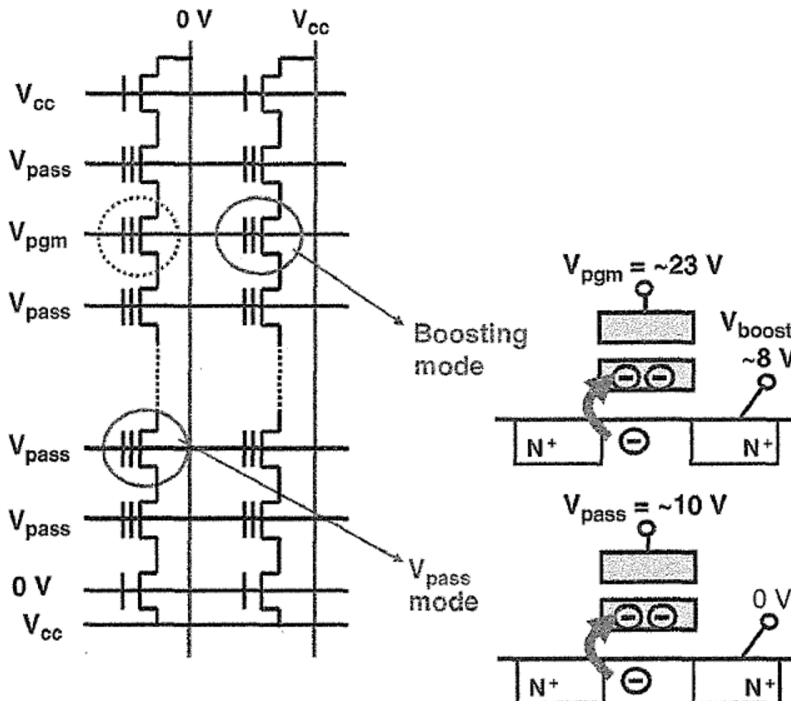


Id., Fig. 8.11 (annotated). Fig. 8.11’s caption explains that “[i]n a 3D NAND cell, two new modes of Y-mode and XY-mode are added.” *Id.* Program disturb modes ① and ②, in addition to program disturb modes ③ and ④, all need to be addressed in order to reliably operate a 3D NAND during programming.

69. The two “new program disturb modes” (*i.e.*, modes ③ and ④) are not the disturb modes Dong was addressing. As Aritome explains, the “[c]onventional”

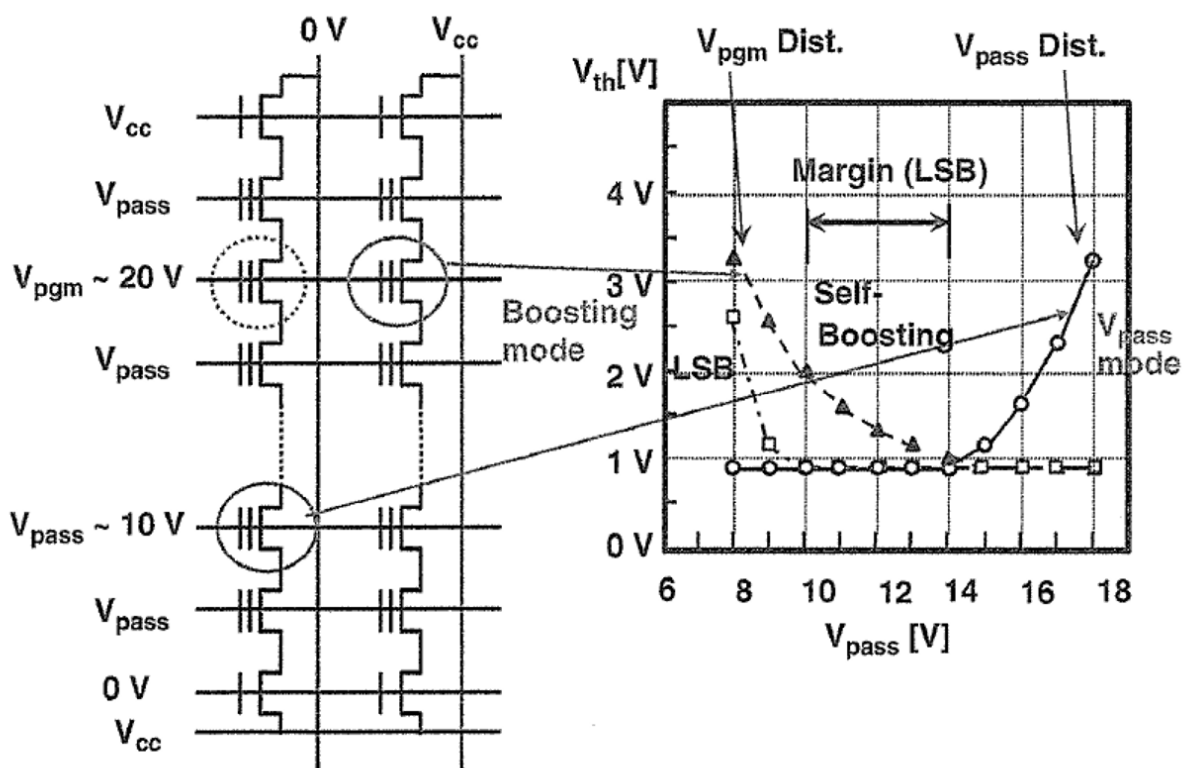
⁵ In Aritome, “STR” refers to a “NAND [memory] string,” “BL” is bitline, and DSL is drain select line/transistor. Ex. 1009, p. 362.

program disturb modes (*i.e.*, the PGM and X modes, labelled ① and ②) exist in both 2D and 3D memory. Ex. 1009, Fig. 8.11; *id.*, p. 363 (“Disturb X mode is the same as a conventional program disturb mode in a 2D NAND cell.”). The reason Aritome §8.4 does not elaborate on PGM and X modes (*i.e.*, modes ① and ②) is because Aritome already explained those two modes in detail in §2.2.4 (*id.*, pp. 52-55). Aritome illustrates these two disturb modes (labelled “Vpass mode” for the selected memory string and “Boosting mode” for the unselected memory string) in Fig. 2.21 below. “Vpass mode” in Fig. 2.21 corresponds to PGM ① in Fig. 8.11, and “Boosting mode” corresponds to X mode ②. Fig. 2.21 does not show either Y mode ③ or XY mode ④ because Fig. 2.21 is for a conventional 2D NAND array.



Id., Fig. 2.21 (excerpted).

70. And, as Fig. 2.22 (below) shows, reducing program disturb for “V_{pass} mode” (PGM ①) exacerbates “Boosting mode” (X mode ②) and vice versa. Aritome expressly says this: “V_{pass} mode dominates [at] higher V_{pass} [voltage]. Boosting mode dominates [at] lower V_{pass} [voltages].” Ex. 1009, p. 54. Thus, there is a trade-off and tension between the two modes.



Id., Fig. 2.22.

71. Thus, even if Dong’s program disturb solutions in Fig. 5a and 5g were made in the context of 2D NAND, it would have been obvious that they apply equally (or with greater force) to 3D NAND to address the “[c]onventional” “PGM” and “X mode” disturb. Ex. 1009, p. 363. There should be no dispute about this.

Even Dr. Khatri “first describe[d] the program disturb modes that are similar to those of 2D NAND flash” in his declaration. Ex. 2028, ¶77. In doing so, he cited Aritome Fig. 8.11 and said that “[t]he **first program disturb (labeled as ‘STR in PGM’** in the figure above) can occur in flash transistors in the selected string (a string selected for programming, such as string 1), for flash transistors that have a pass voltage V_{pass} applied on their control gates. The **second program disturb (labeled as ‘STR in X mode’** in the figure above) can occur in flash transistors in the unselected string (a string not selected for programming, such as string 2).” *Id.*

72. When confronted with his own declaration at deposition, Dr. Khatri tried to backtrack, saying that there were only 3 disturb modes, not 4. Ex. 1094, 280:23-281:10. To try and save face in view of his repeated assertions at deposition that PGM ① was not a disturb mode (*i.e.*, that program disturb never occurred on the string being programmed), Dr. Khatri threw his declaration under the bus, saying that “this is an error, and I had missed it.” *Id.*, 282:2-14. Dr. Khatri even went so far as to say that when he was preparing his declaration “I didn’t really read this because I assumed they [the lawyers] were right.” *Id.*, 283:8-12. As it turns out, the lawyer[s] who wrote Dr. Khatri’s declaration (and Aritome) got it right that there are two conventional 2D disturb modes that apply to 3D NAND. Dr. Khatri is the one who got it wrong. Dong itself expressly says that “[p]rogram disturb can occur at inhibited NAND strings during programming of other NAND strings, **and**

sometimes at the programmed NAND string itself.” Dong, [0011].

73. Indeed, Aritome Fig. 8.15 confirms there are 2 “Conventional 2D” disturb modes and 2 additional “3D” disturb modes.

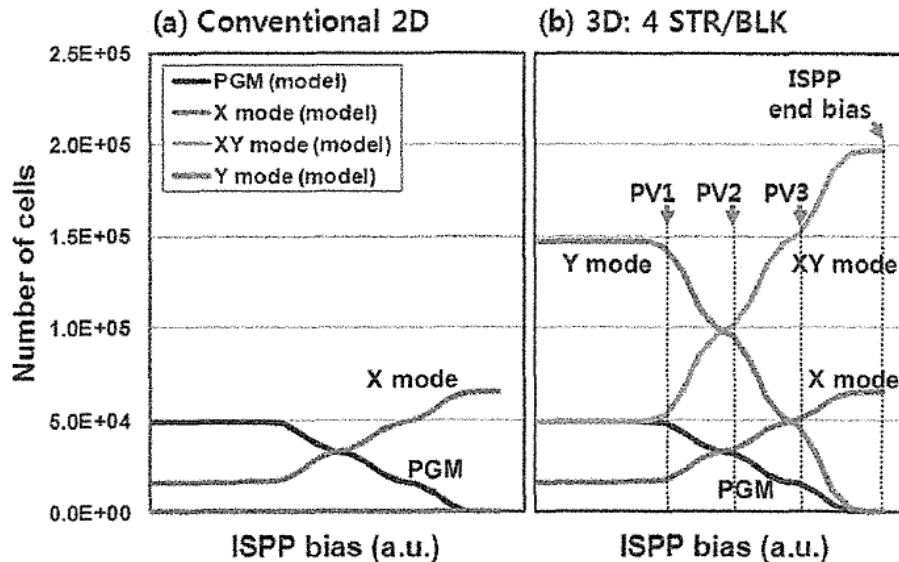
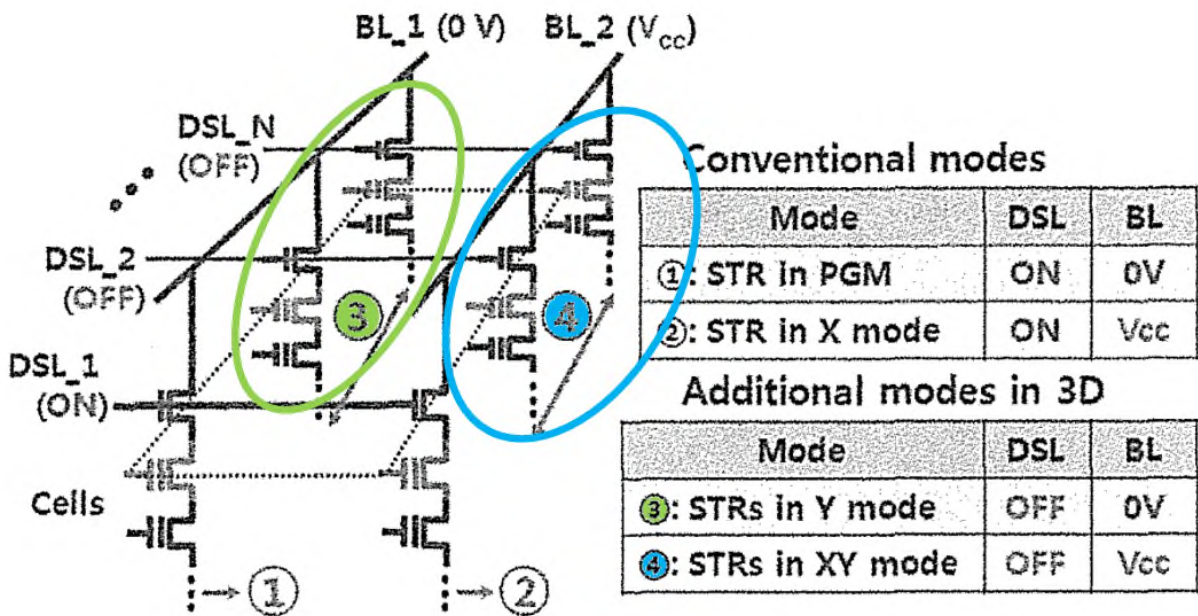


FIGURE 8.15 Probability (number of cells) of each program disturbance mode with increasing ISPP bias.

Ex. 1009, Fig. 8.15 (show disturb error rates for all four modes, including “X mode” and “PGM” mode). As Aritome Fig. 8.15 shows, the “PGM” and “X-mode” program disturb curves are identical for 2D and 3D memory.

74. But even if there were just one conventional 2D mode (there is not), Aritome could not be clearer that “[d]isturb X mode is the same as a conventional program disturb mode in a 2D NAND cell.” Ex. 1009, p. 363. Thus, Patent Owner loses even under Dr. Khatri’s self-contradictory positions regarding the number of conventional disturb modes.

75. Patent Owner’s fundamental problem is that it focused on **the wrong program disturb** modes (i.e., program disturb modes ③ and ④), while Dong was addressing program disturb modes ① and ②. Aritome Fig. 8.11 is clear that both “Y mode” ③ and “XY mode” ④ occur in memory strings located in the Y-direction (which does not exist in 2D NAND) where the drain select line (DSL) is “OFF.”



Ex. 1009, Fig. 8-11 (annotated).

76. The solutions to “Y mode” ③ and “XY mode” ④ disturb in 3D NAND are **complementary** to (and do not replace) Dong’s solution for PGM ① and X-mode ② disturb. Ex. 1009, pp. 363-364. This is facially apparent from Aritome itself, which proffers **three solutions for the new disturb modes**: “(1) high V_t of DSL, (2) applying negative bias to DSL, and (3) inserting dummy word lines between DSL and edge word line.” *Id.* Because Aritome’s DSL line for the selected memory

string (and all strings connected to the same drain select line) must be turned on for programming, Aritome's solutions 1 and 2 (which change the DSL voltage) necessarily only apply to the 3D memory strings located in the Y-direction, and thus change nothing about Dong's programming scheme in Fig. 5a and 5g. And Aritome's solution 3 self-evidently deals with "dummy word lines" which are not at issue here.

77. As mentioned above, the additional 3D program disturb modes do not in any way lessen the importance of addressing the program disturb modes ① and ② that are commonly shared between 2D NAND and 3D NAND. In fact, it turns out that Patent Owner's additional references that apply to 3D NAND employ Dong's teachings as discussed below.

78. Patent Owner's argument about a "floating" body/channel/node also fail upon scrutiny. POR, 50-54 (citing Exs. 2024-2027).⁶ Indeed, they actually support obviousness. Patent Owner says that, in 3D NAND, "during programming, the inhibited channel becomes a floating node and the natural local self-boosting (NLSB) effect occurs." Ex. 2028, ¶81 and Ex. 2024, Fig. 1(b). But Patent Owner's own exhibits (*e.g.*, Ex. 2027) say otherwise, recognizing that NLSB introduces hot-

⁶ In relevant part, these exhibits are substantively similar (which is unsurprising since Myounggon Kang is an author on all of them).

carrier injection (HCI) in adjacent cells (due to a high voltage gradient) and thus cannot be used to solve program disturb. Ex. 2027, 2 (“when the NLSB effect becomes small, program disturb occurs, and when the NLSB effect becomes too large, the HCI phenomenon occurs [*i.e.*, another form of program disturb]. Therefore, it is important to maintain an appropriate channel potential in 3D NAND flash memory”).

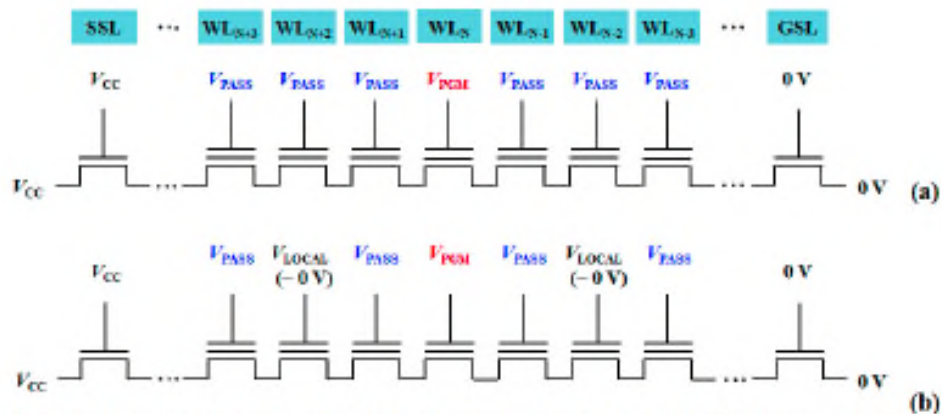


Fig. 1. Bias conditions of SBPI schemes. (a) global SBPI scheme. (b) local SBPI scheme.

Ex. 2024, Fig. 1.

79. Due to the nature of 3D and its inherent thin body region, the natural local boosting would be implemented with a bias scheme similar to Figure 1(b) shown above. The benefit of such bias scheme was described as “with the local SBPI scheme, the channel of the program-inhibited cell is isolated from its full its full string channel by lowering the voltages of the adjacent WL’s from V_{pass} to V_{local} before V_{PGM} is applied.” Ex. 2024, 2. This is the same bias scheme that Dong is teaching, for example in Fig. 5c of Dong, except that Dong tries to distribute the

channel potential difference using more than one memory cell.

80. From the above discussion, it is clear that it would be incorrect to suggest that Dong's teachings (even if originally intended for 2D NAND, which I do not agree with) would not be applicable to address the program disturb mechanisms of 3D NAND.

81. In the unselected memory strings that share a DSL with the selected memory string, the 2D and 3D program disturb mechanism (*i.e.*, X mode ② disturb) is the same as Aritome states. Ex. 1009, p. 363. But the lack of a grounded substrate (which PO's exhibits call a "floating" body/channel/node) only exacerbates the program disturb effects, because 2D memory would also be subjected to a "floating" body. During the program inhibit in mode ② for a 2D structure, the V_{CC} on the unselected would essentially cause the drain select transistor to be shut off. The channel of the 2D string will undergo a boosting effect and lose its connection with the grounded substrate (thus making it largely irrelevant that the substrate is grounded). The only difference between 2D and 3D vis-à-vis a self-boosting channel is that the depletion capacitor (formed between the channel and floating gate) in 3D is much larger due to the thin body being fully depleted, thus allowing the channel to boost to a higher channel voltage. The fact that in 2D the channel is not tied to a grounded substrate is fundamental to facilitating program inhibit on unselected strings (*i.e.*, mitigating program disturb on unselected strings) while

programming the selected string. As a matter of fact, the enhanced X mode ② disturb effects makes Dong's teachings even more applicable (not less applicable) to the 3D NAND scenario. Moreover, if PO were correct that the programming voltage schemes disclosed in Dong Fig. 5a and 5g did not improve program disturb in 3D memory, the '941 patent's claimed programming schemes (which are identical to Dong 5a and 5g) would have no beneficial effect.

82. The explanation above also gets rid of PO's cursory motivation to combine challenge (POR, 47) which asserts that PO's motivation to combine is too generic. As discussed in detail above, Aritome explains that the relevant program disturb mechanisms (*viz.* "[c]onventional" PGM ① and X mode ②) are the same for 2D and 3D NAND strings.

4. Enablement and Reasonable Expectation of Success

83. I was asked to opine on Patent Owner's assertions (POR, 44) that a POSITA would not have had a reasonable expectation of success (or that it would have required undue experimentation)⁷ to implement Dong's programming techniques in a 3D memory with vertical strings. A POSITA most certainly would have had a reasonable expectation of success in doing so.

⁷ I am informed by counsel for Petitioner that a prior art reference is entitled to a presumption that it is enabled.

84. In 2020 (and for years beforehand), 3D NAND memory with vertical memory strings was ubiquitous and extremely well-known to POSITAs. Several major semiconductor manufacturers (like Micron, Samsung, Toshiba/Kioxia, and others) all had commercial 3D NAND products on the market with vertical memory strengths with varying numbers of cells/layers.

- <https://investors.micron.com/news-releases/news-release-details/micron-and-intel-unveil-new-3d-nand-flash-memory> (dated March 26, 2015) (Ex. 1108)
- <https://news.samsung.com/global/samsung-starts-mass-producing-industrys-first-3d-vertical-nand-flash> (dated August 6, 2013) (Ex. 1109)
- <https://overclock3d.net/news/storage/western-digital-announces-its-bics5-112-layer-3d-nand-with-kioxia/> (dated January 31, 2020) (Ex. 1110)
- <https://news.samsung.com/global/samsung-electronics-takes-3d-memory-to-new-heights-with-sixth-generation-v-nand-ssds-for-client-computing> (dated August 6, 2019) (Ex. 1104)

Indeed, by 2019, Samsung was already on its 6th generation of 3D NAND products per the timeline below from a Samsung press release. Samsung explains that its 6th generation 3D NAND technology “is achieved by building an electrically conductive mold stack comprised of 136 layers, then vertically piercing cylindrical holes from top to bottom, creating uniform 3D charge trap flash (CTF) cells.” *Id.*, p. 2.

Reply Declaration of David Kuan-Yu Liu, Ph.D.

Reference: Samsung V-NAND Mass Production Timeline

Date	V-NAND
July 2013	1st-generation (24-layer) 128Gb MLC V-NAND
Aug. 2013	1 st -generation 128Gb MLC V-NAND 960GB SSD
Aug. 2014	2nd-generation (32-layer) 128Gb 3-bit V-NAND
Sept. 2014	2 nd -generation V-NAND SSD
Aug. 2015	3rd-generation (48-layer) 256Gb 3-bit V-NAND
Sept. 2015	3 rd -generation V-NAND SSD '850 EVO', '950 PRO'
Dec. 2016	4th-generation (64-layer) 256Gb 3-bit V-NAND
Jan. 2017	4 th -generation V-NAND SSD
Jan. 2018	4 th -generation 512Gb V-NAND 30.72TB SAS SSD
May 2018	5th-generation (9x-layer) 256Gb 3-bit V-NAND
June 2018	5 th -generation V-NAND SSD
June 2019	6th-generation (1xx-layer) 256Gb 3-bit V-NAND
July 2019	6 th -generation V-NAND SSD

Id., p. 3.

85. Patent Owner's assertions ignore a POSITA's knowledge and the level of ordinary skill in the art, which is undisputed in this case: a POSITA "would have had a bachelor's degree in electrical engineering or a similar discipline, along with 2-3 years of professional experience working with (*e.g.*, researching, designing, or teaching) NAND memory devices (including issues relating to the operation, *e.g.*, programming, of said devices) ... [and] generally knowledgeable about NAND memory's structure and fabrication, its component parts, how it operates, and how

it is controlled.” Pet., 22-23; POR, 9 (“[PO] does not dispute Petitioner’s definition of a POSITA for purposes of these proceedings”); Ex. 2028, ¶46 (“I am informed by counsel that Patent Owner does not dispute Petitioner’s definition of a POSITA for the purposes of these proceedings”).

86. Any assertion that applying wordline voltages in a vertical memory string instead of a horizontal one requires undue experimentation lacks any credibility. I explained in §III.A.3 why Dong’s programming techniques (even if made in the context of 2D memory strings) is readily applicable to 3D memory. Three dimensional NAND flash memory already includes controllers, etc. for applying different voltages to memory strings. It was well with a POSITA’s knowledge and skill to make any minor tweaks necessary to the software/firmware in the controller to change the voltage levels applied to the wordlines.

87. Although the physical manufacture of a vertical 3D memory is not at issue (only programming a memory string is), the ’941 patent itself presumes that a POSITA can construct a vertical 3D memory as it discloses nothing about that process. Moreover, even references in 2007 provided detailed manufacturing steps for how to fabricate 3D NAND memory, reflecting the level of skill in the art. *See, e.g.,* Ex. 1087.

B. Ground 3 (*i.e.*, the Dong/Costa Combination) Renders Claims 8 and 9 Obvious

1. Dong Combined with Costa Render Obvious the “*the first voltage has a first level if the xth word line is in the first portion*”

88. It is telling that, for claims 8-9, Patent Owner does not dispute that Costa discloses that “the pass voltage that is applied [to unselected memory cells] is dependent upon the position of the selected memory cell,” and that the pass voltage increases “[a]s programming progresses from a memory cell at a first end of a NAND string toward a memory cell at a second end of the NAND string.” Costa, Abstract.

89. Instead, Patent Owner says that “Petitioner’s application of Dong to independent claim 5 fails when carried forward to address claims 8 and 9.” POR, 55. This argument ignores that Petitioner’s invalidity analysis was not limited to just the specific example of programming only the fourth wordline (*viz.*, when *x*=4). I also note that the ’941 patent failed to provide any examples at all. As I said in my opening declaration, “I wish to emphasize that the mappings below are just examples and that other mappings could also satisfy claim 5.” Ex. 1004, ¶115; *see also id.*, ¶¶110-111 (“I note that memory strings with lengths of 32, 64, and 128 cells were well-known to POSITAs and commonplace by the Priority Date and all of them would have been obvious to use, as reflected by a number of references. ... I note that other memory string lengths would work just as well and would have been

obvious to a POSITA to achieve higher memory densities”); *see also* Pet., 33 (“[T]hese mappings are merely exemplary and other mappings also satisfy [claim 5]”).

90. Indeed, Petitioner specifically discussed that Dong Fig. 10 (among other things) discloses that “programming continues sequentially until all memory cells in a string are programmed.” Pet. 31; *see also id.*, 34 (quoting Dong, [0048] (programming “proceed[s] in a programming sequence one wordline at a time”)); Dong, Fig. 10, [0089] (“If programming for the current word line is complete but programming for all word lines is not complete, at decision step 1065, the next word line is selected for programming at step 1075.”). Costa teaches the same. Costa, Abstract (“programming progresses from a memory cell at a first end of a NAND string toward a memory cell at a second end of the NAND string”); *see also* Pet., 62 (“A POSITA would understand that memory string programming typically advances one cell at a time”). Thus, all cells/wordlines in Dong/Costa combination would be programmed. This necessarily includes all memory cells in the **first, second, and third portions**, regardless of how those portions are designated (which in the ’941 patent is arbitrary).

91. Nor would it make sense to limit Petitioner’s analysis to only the example provided for claims 5 and 7 (where **k**=2, **m**=7, **x**=4, and **n**=32) where claims 8 and 9 refer to programming wordlines in each of the **first, second, and/or third**

portion[s]. Even though no concrete example (like that provided for claim 5) is necessary for claims 8 and 9, numerous examples apply. As the Petition explained “Costa’s zones/range correspond to the claimed portions in the ’941 patent” and “Costa provides ‘a simple example,’ where ‘a NAND string [is divided] into two equal-sized zones.’” Pet., 75 (quoting Costa, [0115]). Using this arrangement from the Petition and applying it to Dong (see Pet., 75-76), the **first portion** and **second portion** each have 16 wordlines (meaning $k=16$ and $m=32$). This arrangement is illustrated below, showing one exemplary wordline being programmed in each portion (viz. $x=4$ (**first portion**) and $x=20$ (**second portion**)).

Second Portion	WL31	(Vpass)	(Vpass)	m=32
	WL30	(Vpass)	(Vpass)	
	WL29	(Vpass)	(Vpass)	
	WL28	(Vpass)	(Vpass)	
	WL27	(Vpass)	(Vpass)	
	WL26	(Vpass)	(Vpass)	
	WL25	(Vpass)	(Vpass)	
	WL24	(Vpass)	(Vpass)	
	WL23	(Vpass)	(Vpass)	
	WL22	(Vpass)	(Vpass)	
	WL21	(Vpass)	(Vpass)	
First Portion	WL20	(Vpass)	V3 (Vpass)	X+1
	WL19	(Vpass)	VPGM	X=20
	WL18	(Vpass)	V2 (Viso)	X-1
	WL17	(Vpass)	V1 (Vpass)	X-2
	WL16	(Vpass)	V1 (Vpass)	k=16
	WL15	(Vpass)	V1 (Vpass)	
	WL14	(Vpass)	V1 (Vpass)	
	WL13	(Vpass)	V1 (Vpass)	
	WL12	(Vpass)	V1 (Vpass)	
	WL11	(Vpass)	V1 (Vpass)	
	WL10	(Vpass)	V1 (Vpass)	
	WL9	(Vpass)	V1 (Vpass)	
	WL8	(Vpass)	V1 (Vpass)	
	WL7	(Vpass)	V1 (Vpass)	
	WL6	(Vpass)	V1 (Vpass)	
	WL5	(Vpass)	V1 (Vpass)	X+2
	WL4	V3 (Vpass)	V1 (Vpass)	X+1
	WL3	VPGM	V1 (Vpass)	X=4
	WL2	V2 (Viso)	V1 (Vpass)	X-1
WL1	V1 (Vpass)	V1 (Vpass)	X-2	
WL0	V1 (Vpass)	V1 (Vpass)		
VSGS				

Dong, Fig. 5a (modified and annotated).

92. The red text shows the applied voltages (per Dong Fig. 5a) when $x=4$ (viz., when a wordline in the *first portion* is being programmed) and the blue text shows the applied voltages (per Dong Fig. 5a) when $x=20$ (viz., when a wordline in

the **second portion** is being programmed). This satisfies all conditions of claim 9 and underlying claim 5. As the figure above shows, Dong's "Vpass" voltage corresponds to the claimed **first voltage** (V1), Dong's "Viso" voltage corresponds to the claimed **second voltage** (V2), and Dong's "Vpass" voltage corresponds to the claimed **third voltage** (V3). Each voltage is applied to the wordlines recited in claim 5 (i.e., V1 spans the **first word line to an (x-2)th wordline**, V2 spans **an (x-1)th word line**, and V3 spans **an (x+1)th word line**) and claim 9 (i.e., the analysis and mappings hold true for when a wordline (e.g., WL3) is being programmed in the **first portion** and when a wordline (e.g., WL19) is being programmed in the **second portion**). Simple math confirms that **x, k and m are positive integers, $1 < k < m$ and $3 \leq x$** .

93. Many other examples satisfy claim 9 (and base claim 5). Using the same string length (i.e., 32-bit) and same **portion** sizes (i.e., 16 bits each), another example is **x=8** (for a wordline being programmed in the **first portion**) and **x=23** (for a wordline being programmed in the **second portion**). Using the same color scheme as before, Dong Fig. 5a is shown below.

		VSGD		
Second Portion	WL31	(Vpass)	(Vpass)	m=32
	WL30	(Vpass)	(Vpass)	
	WL29	(Vpass)	(Vpass)	
	WL28	(Vpass)	(Vpass)	
	WL27	(Vpass)	(Vpass)	
	WL26	(Vpass)	(Vpass)	
	WL25	(Vpass)	(Vpass)	
	WL24	(Vpass)	(Vpass)	
	WL23	(Vpass)	V3 (Vpass)	X+1
	WL22	(Vpass)	VPGM	X=23
	WL21	(Vpass)	V2 (Viso)	X-1
	WL20	(Vpass)	V1 (Vpass)	X-2
	WL19	(Vpass)	V1 (Vpass)	
	WL18	(Vpass)	V1 (Vpass)	
	WL17	(Vpass)	V1 (Vpass)	
	WL16	(Vpass)	V1 (Vpass)	
First Portion	WL15	(Vpass)	V1 (Vpass)	k=16
	WL14	(Vpass)	V1 (Vpass)	
	WL13	(Vpass)	V1 (Vpass)	
	WL12	(Vpass)	V1 (Vpass)	
	WL11	(Vpass)	V1 (Vpass)	
	WL10	(Vpass)	V1 (Vpass)	
	WL9	(Vpass)	V1 (Vpass)	X+2
	WL8	V3 (Vpass)	V1 (Vpass)	X+1
	WL7	VPGM	V1 (Vpass)	X=8
	WL6	V2 (Viso)	V1 (Vpass)	X-1
	WL5	V1 (Vpass)	V1 (Vpass)	X-2
	WL4	V1 (Vpass)	V1 (Vpass)	
	WL3	V1 (Vpass)	V1 (Vpass)	
	WL2	V1 (Vpass)	V1 (Vpass)	
	WL1	V1 (Vpass)	V1 (Vpass)	
	WL0	V1 (Vpass)	V1 (Vpass)	
		VSGS		

Dong, Fig. 5a (modified and annotated). Another example (not shown) that works equally well is **x**=10 as the wordline in the **first portion** and **x**=25 as the wordline in **second portion**.

94. Claim 8 (through its dependency on claim 7) requires a *third portion* that *comprises an (m+1)th word line to an nth word* and *applying a fourth voltage to the (x+2)th word line to the nth word line*. Costa says that “the size of the zones may not be equal ... [and] more than two zones may be used.” Costa, [0116]. Designating another “zone” in Dong Fig. 5a (per Costa’s teachings) results in a *third portion*, an example of which is below (where *k*=16, *m*=24, and *n*=32). In this example, one exemplary wordline is programmed in each portion (viz. *x*=4 (*first portion*), *x*=20 (*second portion*), and *x*=29 (*third portion*)).

		VSGD			
Third Portion	WL31	(Vpass)	(Vpass)	V4 (Vpass)	n=32
	WL30	(Vpass)	(Vpass)	V4 (Vpass)	X+2
	WL29	(Vpass)	(Vpass)	V3 (Vpass)	X+1
	WL28	(Vpass)	(Vpass)	VPGM	X=29
	WL27	(Vpass)	(Vpass)	V2 (Viso)	X-1
	WL26	(Vpass)	(Vpass)	V1 (Vpass)	X-2
	WL25	(Vpass)	(Vpass)	V1 (Vpass)	
Second Portion	WL24	(Vpass)	(Vpass)	V1 (Vpass)	
	WL23	(Vpass)	(Vpass)	V1 (Vpass)	m=24
	WL22	(Vpass)	(Vpass)	V1 (Vpass)	
	WL21	(Vpass)	(Vpass)	V1 (Vpass)	X+2
	WL20	(Vpass)	V3 (Vpass)	V1 (Vpass)	X+1
	WL19	(Vpass)	VPGM	V1 (Vpass)	X=20
	WL18	(Vpass)	V2 (Viso)	V1 (Vpass)	X-1
First Portion	WL17	(Vpass)	V1 (Vpass)	V1 (Vpass)	X-2
	WL16	(Vpass)	V1 (Vpass)	V1 (Vpass)	
	WL15	(Vpass)	V1 (Vpass)	V1 (Vpass)	k=16
	WL14	(Vpass)	V1 (Vpass)	V1 (Vpass)	
	WL13	(Vpass)	V1 (Vpass)	V1 (Vpass)	
	WL12	(Vpass)	V1 (Vpass)	V1 (Vpass)	
	WL11	(Vpass)	V1 (Vpass)	V1 (Vpass)	
	WL10	(Vpass)	V1 (Vpass)	V1 (Vpass)	
	WL9	(Vpass)	V1 (Vpass)	V1 (Vpass)	
	WL8	(Vpass)	V1 (Vpass)	V1 (Vpass)	
	WL7	(Vpass)	V1 (Vpass)	V1 (Vpass)	
	WL6	(Vpass)	V1 (Vpass)	V1 (Vpass)	
	WL5	(Vpass)	V1 (Vpass)	V1 (Vpass)	X+2
	WL4	V3 (Vpass)	V1 (Vpass)	V1 (Vpass)	X+1
	WL3	VPGM	V1 (Vpass)	V1 (Vpass)	X=4
	WL2	V2 (Viso)	V1 (Vpass)	V1 (Vpass)	X-1
	WL1	V1 (Vpass)	V1 (Vpass)	V1 (Vpass)	X-2
	WL0	V1 (Vpass)	V1 (Vpass)	V1 (Vpass)	
		VSGS			

Dong, Fig. 5a (modified and annotated). This satisfies all conditions of claim 8 (and base claims 5 and 7). As the figure above shows, Dong’s “Vpass” voltage

corresponds to the claimed *first voltage* (V1), Dong's "Viso" voltage corresponds to the claimed *second voltage* (V2), Dong's "Vpass" voltage corresponds to the claimed *third voltage* (V3), and (where applicable) Dong's "Vpass" voltage also corresponds to the claimed *fourth voltage* (V4). Each voltage is applied to the wordlines recited in claim 5 (i.e., V1 spans the *first word line to an (x-2)th wordline*, V2 spans *an (x-1)th word line*, and V3 spans *an (x+1)th word line*), claim 7 (V4 spans *the (x+2)th word line to the nth wordline*), and claim 8 (i.e., the analysis and mappings hold true for when a wordline (e.g., WL3) is being programmed in the *first portion*, when a wordline (e.g., WL19) is being programmed in the *second portion*, and when a wordline (e.g., WL28) is being programmed in the *third portion*). Many other examples satisfy claim 8 (and base claims 5 and 7).

95. Using the same string length (i.e., 32-bit) and same *portion* sizes (i.e., 16 bits, 8 bits, and 8 bits for the *first*, *second*, and *third portions*, respectively), another example is $x=5$ (for a wordline being programmed in the *first portion*), $x=21$ (for a wordline being programmed in the *second portion*), and $x=28$ (for a wordline being programmed in the *third portion*). Using the same color scheme as before, Dong Fig. 5a is shown below.

		VSGD				
Third Portion	WL31	(Vpass)	(Vpass)	V4 (Vpass)	n=32	
	WL30	(Vpass)	(Vpass)	V4 (Vpass)		
	WL29	(Vpass)	(Vpass)	V3 (Vpass)	X+2	
	WL28	(Vpass)	(Vpass)	V3 (Vpass)	X+1	
	WL27	(Vpass)	(Vpass)	VPGM	X=28	
	WL26	(Vpass)	(Vpass)	V2 (Viso)	X-1	
	WL25	(Vpass)	(Vpass)	V1 (Vpass)	X-2	
	WL24	(Vpass)	(Vpass)	V1 (Vpass)		
Second Portion	WL23	(Vpass)	(Vpass)	V1 (Vpass)	m=24	
	WL22	(Vpass)	(Vpass)	V1 (Vpass)	X+2	
	WL21	(Vpass)	V3 (Vpass)	V1 (Vpass)	X+1	
	WL20	(Vpass)	VPGM	V1 (Vpass)	X=21	
	WL19	(Vpass)	V2 (Viso)	V1 (Vpass)	X-1	
	WL18	(Vpass)	V1 (Vpass)	V1 (Vpass)	X-2	
	WL17	(Vpass)	V1 (Vpass)	V1 (Vpass)		
	WL16	(Vpass)	V1 (Vpass)	V1 (Vpass)		
	WL15	(Vpass)	V1 (Vpass)	V1 (Vpass)	k=16	
	WL14	(Vpass)	V1 (Vpass)	V1 (Vpass)		
First Portion	WL13	(Vpass)	V1 (Vpass)	V1 (Vpass)		
	WL12	(Vpass)	V1 (Vpass)	V1 (Vpass)		
	WL11	(Vpass)	V1 (Vpass)	V1 (Vpass)		
	WL10	(Vpass)	V1 (Vpass)	V1 (Vpass)		
	WL9	(Vpass)	V1 (Vpass)	V1 (Vpass)		
	WL8	(Vpass)	V1 (Vpass)	V1 (Vpass)		
	WL7	(Vpass)	V1 (Vpass)	V1 (Vpass)		
	WL6	(Vpass)	V1 (Vpass)	V1 (Vpass)	X+2	
	WL5	V3 (Vpass)	V1 (Vpass)	V1 (Vpass)	X+1	
	WL4	VPGM	V1 (Vpass)	V1 (Vpass)	X=5	
	WL3	V2 (Viso)	V1 (Vpass)	V1 (Vpass)	X-1	
	WL2	V1 (Vpass)	V1 (Vpass)	V1 (Vpass)	X-2	
	WL1	V1 (Vpass)	V1 (Vpass)	V1 (Vpass)		
	WL0	V1 (Vpass)	V1 (Vpass)	V1 (Vpass)		
			VSGS			

Dong, Fig. 5a (modified and annotated). Of course, numerous other *portion* assignments and designated wordlines for programming satisfy claim 8.

96. While the examples above (like claims 5 and 7) use a 32-bit memory string, Petitioner's analysis included longer memory strings. *E.g.*, Pet., 31 "32-cell (*and larger*) memory strings were well-known to POSITAs by the Priority Date and

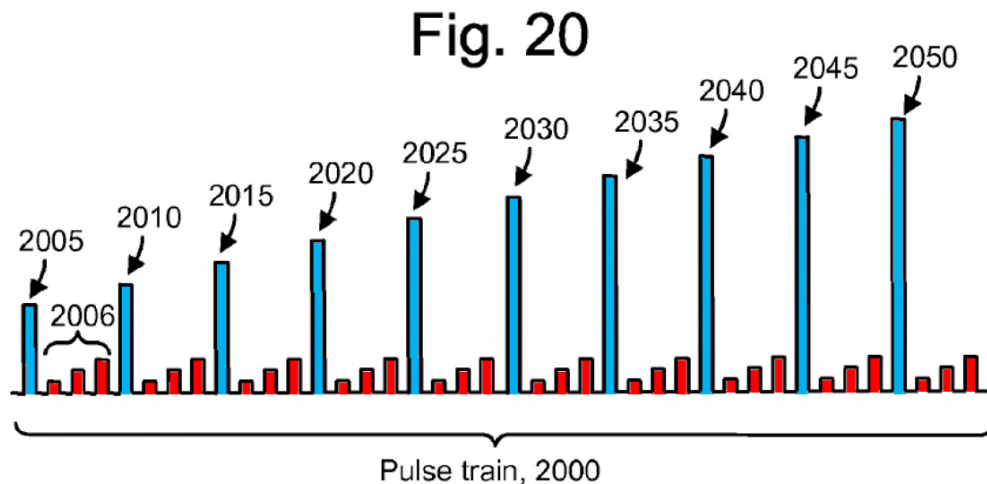
would have been obvious to use”). And Petitioner cited Dong as disclosing a 64-bit memory string (*id.*) and collected numerous references reciting 74-bit, 64-bit, 88-bit, etc. memory strings (*id.*). These memory strings (all longer than 32-bits presents) present numerous additional examples of mappings that satisfy claims 8 and 9, and the number of examples grows exponentially when different **portion** sizes are used. Indeed, as memory string lengths grow, so does the need to account for increased channel resistance using Costa’s teachings.

2. Ground 3 Satisfies “*Applying a First Voltage*” Under Petitioner’s Interpretation of “*Program Operation*”

97. Claims 8 and 9 both depend (directly or indirectly) on independent claim 5 which recites ***applying a first voltage [] to the first word line to an (x–2)th word line when an xth word line is used to perform a program operation.***

98. To begin, Costa says that “the pass voltage that is applied [to unselected memory cells] is dependent upon the position of the selected memory cell,” and that the pass voltage of unselected cells closer to the source increases “[a]s programming progresses from a memory cell at a first end of a NAND string toward a memory cell at a second end of the NAND string.” Costa, Abstract. This satisfies the ***first voltage*** requirement (viz., that ***the first [voltage] level is lower than the second [voltage] level, and the second [voltage] level is lower than the third [voltage] level***) under Petitioner’s construction. Ex. 1001, cl.8.

99. Dong Fig. 20 (shown below) “depicts an example pulse train 2000 applied to the control gates of non-volatile storage elements *during programming*.” Dong, [0132]. Dong’s “pulse train 2000”—which depicts “programming”—plainly includes both the charge injection pulses 2005, 2010, etc. (in blue) *and* the “verify pulses 2006” (in red). *Id.* And it plainly depicts a single *program operation* for one memory cell, not ten separate *program operations*.



Dong, Fig. 20 (annotated), [0132].

100. In view of this clear disclosure, Patent Owner argues that a POSITA would not have been motivated to combine Dong and Costa “because Dong already discloses a verification operation” POR, 65. This reasoning is circular. Under Patent Owner’s logic, the existence of one approach (not matter how inferior or incomplete) would foreclose obviousness with respect to all other solutions. The engineering profession is largely built on efforts to improve upon existing approaches and technologies, including in obvious ways by combining known,

complementary approaches, which is precisely the case here. In particular, a POSITA would have combined Costa's teachings with Dong to improve the overall operation of Dong, including its verification process to account for increased channel resistance.

101. Patent Owner fails to understand (or at a minimum misstates) Petitioner's argument (and my opinion) by saying that Petitioner relied on "silence" to establish the motivation to combine Dong and Costa. POR, 66. That is incorrect. The motivation to combine is Costa's disclosure of an improved programming method that accounts for the known problem of increasing channel resistance. Pet., 72-73. The reason for noting that "Dong is silent regarding how V_{PASS} ... change[s]" (*id.*, 72) was to dispel any notion that Dong teaches away from Costa's variable Vpass teachings.

102. Patent Owner further argues that "a POSITA would not look to Costa" because "it does not focus on program disturb." POR, 66. Even if that were true (it is not), it misses the whole point—*i.e.*, that Costa teaches a solution to the **verification phase** of programming. In other words, if Petitioner's view of **program operation** is correct (*i.e.*, as including both charge injection and verification), there is plainly a motivation to combine Costa's Vpass teachings vis-à-vis the verify pulses with Dong's teachings vis-à-vis charge injection pulses. Patent Owner's contention that a POSITA would ignore Costa's teachings to reduce errors vis-à-vis

half of the programming process is a head-in-the-sand approach and not how a POSITA (*i.e.*, an engineer) would think.

103. Moreover, Petitioner ***combined Costa with Dong for the express purpose of Costa's Vpass disclosure*** (*viz.*, Vpass increases as programming proceeds to counter increased channel resistance), which applies to the verify pulses (shown in blue in Dong Fig. 20 above) of Dong's programming operation. Pet., 70-74 (citing Costa, Abstract, [0029]-[0030], [0101]-[0104]). Nonetheless, Patent Owner contends that "Petitioner still applies Dong's V_{PASS} —used in a program phase—as the 'first voltage.'" POR, 64. Even if true, this is irrelevant. Vpass is applied both during charge injection (as Dong explains) and during verification (as Costa explains). Dong, Fig. 5a; Costa, Abstract. Even if Dong's Vpass voltage level (applied during charge injection) differed from Costa's Vpass voltage level (applied during verification), nothing in the claim prohibits that. All that matters is that at any given point during a ***program operation***, the specified wordlines (*i.e.*, ***the first wordline to an (x-2)th wordline***) have a common ***first voltage*** applied to them (not that the voltage never change). In other words, all that falls outside the claim is the scenario where two wordlines (in the specified range) have different voltage levels at the same time. Thus, Patent Owner's argument fails.

C. Ground 4 (*i.e.*, the Dong/Shirai Combination) Renders Claims 4, 10, and 14 Obvious

1. Patent Owner’s Arguments Concerning “ $V_{\text{PASS-LOW}}$ ” Fail

104. As an initial matter, Patent Owner’s arguments about “ $V_{\text{PASS-LOW}}$ ” do not apply to claim 10 because Petitioner’s analysis for that claim relies on Dong Fig. 5a which (in contrast to Dong Fig. 5g) lacks “ $V_{\text{PASS-LOW}}$ ” altogether. *Compare* Dong, Figs. 5a and 5g. In any event, Patent Owner’s “ $V_{\text{PASS-LOW}}$ ” arguments for claims 4 and 14 also fail for other reasons explained below.

105. Patent Owner says that Petitioner “never explains how or why a POSITA would omit V_{ISO} (or V_{LOW})” in the Dong/Shirai combination. POR, 70. That is because Petitioner never proposed doing so, and neither Patent Owner nor its expert cite anything showing that Petitioner did remove V_{ISO} or V_{LOW} . *See id.*; Ex. 2028, ¶256 (both lacking cites to the Petition or Petitioner’s expert declaration). But, even if V_{ISO} and/or V_{LOW} were omitted such that $V_{\text{PASS-LOW}}$ extended through all wordlines in the lower deck of the Dong/Shirai combination, that mapping still satisfies claims 4 and 14 (and their underlying independent claims) and is consistent with Dong’s teaching to “gradually reducing the bias voltage” over the channel of multiple wordlines. Dong, [0057]. In such a circumstance, the “gradual” voltage reduction would still be accomplished by the $V_{\text{PASS-HIGH}}$ (on WL11), $V_{\text{PASS-MEDIUM}}$ (on WL10), and $V_{\text{PASS-LOW}}$ voltages (on WL9 to WL0).

106. Similar to Ground 3, Patent Owner tries to limit Petitioner’s invalidity analysis (POR, 66-69) to carrying forward a single example from the independent claim analysis for claims 1 and 11 (where the **second portion** was a single wordline) to dependent claims 4 and 14 (where the **second portion** spans multiple wordlines). For example, Patent Owner says “Petitioner identifies a **single wordline** for Dong’s ‘second portion’ of claim 1.” POR, 67 (emphasis by PO). While that mapping is valid, it was “merely exemplary and other mappings also satisfy the claim.” Pet., 51. Elsewhere, Patent Owner swings to the other extreme, admitting that “Shirai’s Figure 2” “appears to comprise [9 wordlines],” and on that basis says that Petitioner’s combination must include a **second portion** with exactly 9 wordlines. POR, 68-69. But Petitioner’s invalidity analysis is not limited in that regard. Even Dr. Khatri admits Dong is not limited to any specific number of wordlines (Ex. 1094, 233:16-19) and that Shirai Fig. 2 “may have any number of stacked layers” (*id.*, 258:9-15). *See also* Ex. 1018 (Shirai), [0030] (in Fig. 2, “[e]ach of the stacked portions 100 a and 100 b may have any number of stacked layers.”).

107. Like the claims of the ’941 patent, Petitioner’s invalidity theory applies more broadly. As in the ’941 patent, the specific number of wordlines comprising the **second portion** is flexible. Claims 4 and 14 (and their respective independent claims) do not recite a minimum or maximum number of wordlines for the **second portion**. Nor do claims 4 and 14 require that the **second portion** be co-extensive

with a memory deck, only that *the memory has a two-deck structure* (e.g., the *second portion* may be a subset of a memory deck). Ex. 1001, cls. 4, 14.

108. Patent Owner admits that Petitioner provides an example with a single-wordline *second portion*. POR, 67 (“Petitioner identifies a single wordline for Dong’s ‘second portion’”). This example itself satisfies claim 4 (and claim 14). Nothing in the claims 4 and 14 (or their respective base claims) requires that the claimed *second portion* is coextensive with the claimed *lower deck*. Even using a single-wordline *second portion* (e.g., where WL9 is the *second portion*) satisfies all requirements of claims 4 and 14 (and their underlying independent claims) as the example below shows. In this example, all variables have the same values (i.e., *n*=20, *k*=9, *m*=10, and *x*=13) used when analyzing the independent base claims (i.e., claim 1 and 11). Here, the dummy wordlines and connection portion 45 reflect a *two-deck structure* per Shirai.

		VSGD	
	WL31	(Vpass)	
	WL30	(Vpass)	
	WL29	(Vpass)	
	WL28	(Vpass)	
	WL27	(Vpass)	
	WL26	(Vpass)	
	WL25	(Vpass)	
	WL24	(Vpass)	
	WL23	(Vpass)	
	WL22	(Vpass)	
	WL21	(Vpass)	
	WL20	(Vpass)	
First Portion	WL19	V1 (Vpass)	n=20
	WL18	V1 (Vpass)	
	WL17	V1 (Vpass)	
	WL16	V1 (Vpass)	
	WL15	V1 (Vpass)	
	WL14	V1 (Vpass)	X+2
	WL13	V2 (Vpass-high)	X+1
	WL12	VPGM	x=13
	WL11	V3 (Vpass-high)	X-1
	WL10	V4 (Vpass-medium)	X-2
Second Portion		Dummy Wordline	
		Connection Portion 45	
		Dummy Wordline	
	WL9	V5 (Vpass-low)	m=10
	WL8	(Vlow)	k=9
	WL7	(Viso)	
	WL6	(Vlow)	
	WL5	(Vpass-low)	
	WL4	(Vpass)	
	WL3	(Vpass)	
	WL2	(Vpass)	
	WL1	(Vpass)	
	WL0	(Vpass)	
		VSGS	

Dong, Fig. 5g (modified and annotated). Patent Owner’s mistake is that it narrowly interprets the Petition as limited to a mapping that bodily incorporates the entirety of Shirai’s “stacked portion 100a” as the **second portion**. Petitioner never performed such a bodily incorporation. Rather, Petitioner merely made the general statement that “Dong’s second portion corresponds to Shirai’s bottom deck 100a.” Pet, 82. The only reason for mentioning a correspondence at all was not to define the

contours of precisely what constitutes the **second portion** (which Patent Owner admits was already mapped in the underlying base claims), but to discuss the existence and relative locations of the dummy wordline and joint oxide layers.

109. But the Petition was not limited to a single-wordline **second portion**, and the **second portion** may comprise multiple wordlines shown in Shirai Fig. 2's lower memory deck. Pet., 83. While Shirai Fig. 2 showed 9 wordlines, the image in no way limited the analysis to 9 wordlines, nor required that the **second portion** comprise all wordlines in a memory deck. The key aspect of Shirai Fig. 2 was not the precise number of wordlines, but to show dummy wordlines and an oxide layer between **portions**. Neither Petitioner nor myself proposed to bodily incorporate all of Shirai Fig. 2 (or any specific number of wordlines) into Dong.

110. In any event, various wordline assignments exist for claims 4 and 14 that satisfy their respective independent claims. I provide another example here, where each of the claimed **first** and **second portions** has multiple wordlines. Here, everything is the same as the prior example, except the **second portion** contains two wordlines (meaning that $k=8$) and $V_{\text{PASS-LOW}}$ is applied to WL8, which is consistent with Dong's teachings. In particular, Dong teaches that program disturb can be reduced by "increasing the number of word lines in between the selected word line and the V_{ISO} word line, and [] gradually reducing the bias voltage on those word lines." Dong, [0057]. Patent Owner even quotes this passage. POR, 69

(emphasizing that Dong is premised on “**gradually** reducing the bias voltage[s]” of surrounding wordlines). This same logic and analysis applies to expanding the **second portion** (and $V_{\text{PASS-LOW}}$) to include additional wordlines as I explain in paragraph 112.

111. For avoidance of doubt, the mapping below (with $k=8$, $m=10$, $x=13$, and $n=20$) satisfies independent claims 1 and 11 as follows: the **first portion** (from $m+1$ to n) spans WL10-WL19; the **second portion** (from $k+1$ to m) spans WL8 to WL9; the **first voltage** is V_{pass} (labelled V1 below) and spans wordline $x+2$ (WL14) to n (WL19); the **second voltage** is $V_{\text{pass-high}}$ (labelled V2 below) and is applied to wordline $x+1$ (WL13); the **third voltage** is $V_{\text{pass-high}}$ (labelled V3 below) and is applied to wordline $x-1$ (WL11); the **fourth voltage** is $V_{\text{pass-medium}}$ (labelled V4 below) and is applied to wordlines $m+1$ (WL10) to $x-2$ (WL10); and the **fifth voltage** is $V_{\text{pass-low}}$ (labelled V5 below) and is applied to wordlines $k+1$ (WL8) to m (WL9). Basic math confirmed that x, k and m are integers, $0 \leq k < m$, $(m+2) < x < (n-1)$ (i.e., $k=8$, $m=10$, $x=13$, and $n=20$, so $0 \leq 8 < 10$, $12 < 13 < 19$). And Dong confirms that the **fifth voltage** ($V_{\text{pass-low}}$) **is lower than the fourth voltage** ($V_{\text{pass-medium}}$). Dong, [0057].

		VSGD	
	WL31	(Vpass)	
	WL30	(Vpass)	
	WL29	(Vpass)	
	WL28	(Vpass)	
	WL27	(Vpass)	
	WL26	(Vpass)	
	WL25	(Vpass)	
	WL24	(Vpass)	
	WL23	(Vpass)	
	WL22	(Vpass)	
	WL21	(Vpass)	
	WL20	(Vpass)	
First Portion	WL19	V1 (Vpass)	n=20
	WL18	V1 (Vpass)	
	WL17	V1 (Vpass)	
	WL16	V1 (Vpass)	
	WL15	V1 (Vpass)	
	WL14	V1 (Vpass)	
	WL13	V2 (Vpass-high)	
	WL12	VPGM	
	WL11	V3 (Vpass-high)	
	WL10	V4 (Vpass-medium)	
		Dummy Wordline	
		Connection Portion 45	
		Dummy Wordline	
Second Portion	WL9	V5 (Vpass-low)	m=10 k+1=9 k=8
	WL8	V5 (Vpass-low)	
	WL7	(Vlow)	
	WL6	(Viso)	
	WL5	(Vlow)	
	WL4	(Vpass-low)	
	WL3	(Vpass)	
	WL2	(Vpass)	
	WL1	(Vpass)	
	WL0	(Vpass)	
		VSGS	

Dong, Fig. 5g (modified).

112. A wordline voltage mapping that included multiple adjacent $V_{\text{PASS-LOW}}$ wordlines (*e.g.*, in the **second portion**) would provide an even more gradual voltage transition, per Dong’s teachings. Conceptually, this is similar to a set of stairs whose slope (analogous to a voltage gradient or electric field) can be reduced by spreading the stairs across a greater distance. Dr. Katri’s say-so to contrary (POR, 69) gets it backwards, and is belied by Dong itself which teaches that “gradual[]” voltage reduction between “the selected wordline [*i.e.*, V_{PGM}] and V_{ISO} ” is the key. Dong,

[0057]; *see also id.*, [0080]-[0084] (describing “even more gradual transition from V_{PGM2} to V_{ISO} over **one or more intermediate word lines**”). Thus, applying $V_{\text{PASS-LOW}}$ to multiple wordlines (*e.g.*, to WL8) is fully supported by Dong. And any limitation PO wishes to impose on making minor, run-of-the-mill adjustments like this ignores both Dong’s teachings and a POSITA’s capabilities.

113. Patent Owner’s argument that “a POSITA would not have been motivated to apply Dong’s voltage scheme for a single wordline ($V_{\text{PASS-LOW}}$) to [Shirai’s Lower Memory Deck 100a]” (POR, 68-69) restates the arguments discussed above and fails for the same reasons. Putting aside that Dong expressly discloses an example of applying $V_{\text{PASS-LOW}}$ precisely in the manner claimed (*viz.*, to the **second portion**), Patent Owner’s assertion that a POSITA would not apply $V_{\text{PASS-LOW}}$ is premised on bodily incorporation. And it is directly contrary to Dong’s teachings.

114. While the examples above use a 32-bit memory string, Petitioner’s analysis included longer memory strings. *E.g.*, Pet., 31 “32-cell (**and larger**) memory strings were well-known to POSITAs by the Priority Date and would have been obvious to use”). And Petitioner cited Dong as disclosing a 64-bit memory string (*id.*) and collected numerous references reciting 74-bit, 64-bit, 88-bit, etc. memory strings (*id.*). These memory strings (all longer than 32-bits presents) present numerous additional examples of mappings that satisfy the claims.

2. Patent Owner's Argument About Mismatched Values for "*k*" is a Strawman

115. Patent Owner says that Petitioner's Ground 4 analysis fails because it "use[s] a different interpretation for the variable 'k' [for the independent and dependent claims]." POR, 71-73. Neither Petitioner nor myself did any such thing. Patent Owner's assertion misreads the Petition (and my declaration) and rests on the same false premise (which I addressed above in §§III.B and III.C.1) that Petitioner's independent claim analysis was limited to only one specific example (*i.e.*, that only one set of values for the variables *k*, *m*, *x*, etc. satisfies the underlying independent claims).

116. In any event, I gave one example above (in §III.C.1) for claims 4 and 14 where *k*=9, which is the same value of *k* (and the same values for *m*, *x*, etc.) that were expressly used in the example for claims 1 and 11 in my opening declaration. In other words, the analysis from claims 1 and 11 "carries forward" to claims 4 and 14. Neither Petitioner nor myself ever mixed-and-matched an analysis using different *k* values (*i.e.*, "k equals 9" for base claims 1 and 11 and "k equal to 0" for dependent claims 4 and 14). POR, 71.

117. Additionally, I provided a second example above (for claims 4 and 14) where *k*=8 (with $V_{\text{PASS-LOW}}$ applied to WL8). That analysis carries back to the underlying independent claims 1 and 11 and works just fine, as I explained in §III.C.1. This same rationale I explained for claims 4 and 14 rebuts Patent Owner's

copycat mix-and-match argument for claim 10 (and underlying independent claim 5). POR, 71-72.

118. Patent Owner says it is confused as to why it is relevant that memory strings often exceeded 100 cells in 2020. POR, 73. But the Petition explained that Dong contemplated long memory string lengths, and that such strings were “largely achieved by [] 3D NAND memory” (Pet., 77) which would motivate a POSITA to use Shirai’s multi-deck teachings. See <https://news.samsung.com/global/samsung-electronics-takes-3d-memory-to-new-heights-with-sixth-generation-v-nand-ssds-for-client>; <https://news.skhynix.com/sk-hynix-starts-mass-producing-worlds-first-128-layer-4d-nand/> (Ex. 1004).

IV. SECONDARY CONSIDERATIONS

119. As was the case in my opening Declaration, I am unaware of any secondary considerations of non-obviousness that apply to the Challenged Claims. This includes commercial success, skepticism, failure of others, industry praise, unexpected results, or the like relating to the ’941 patent. The POR did identify and such secondary considerations.

V. CONCLUSION

120. I declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true, and further that these statements were made with the knowledge that willfully false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1101 of Title 18 of the United States Code.

Date: February 10, 2026

Location: Arlington, Virginia

Signature: 
David Kuan-Yu Liu