IN THE UNITED STATES DISTRICT COURT
FOR THE WESTERN DISTRICT OF WISCONSIN

- -

SANDISK CORPORATION,

Plaintiff,

v.

OPINION and ORDER

10-cv-243-bbc

KINGSTON TECHNOLOGY CO., INC. and
KINGSTON TECHNOLOGY CORP.,

Defendants.

- -

In this patent infringement suit, plaintiff SanDisk Corporation contends that defendants Kingston Technology Co., Inc. and Kingston Technology Corp. are infringing plaintiff's United States Patents Nos. 7,397,713 ('713 patent); 7,492,660 ('660 patent); 7,657,702 ('702 patent); 7,532,511 ('511 patent); 7,646,666 ('666 patent); and 7,646,667 ('667), all of which are related to flash memory technology. Now before the court are the parties' cross motions for construction of certain terms found in the claims being asserted in these patents. I construe the terms as provided below.

OPINION

A. Background

In a previous case between the parties, a consolidated action including Cases Nos. 07-cv-605-bbc and 07-cv-607-bbc, plaintiff asserted several patents against defendants that

relate to flash memory technology, and in particular, technology known as "flash EEPROM" (Electrically Erasable Programmable Read Only Memory). In the consolidated action, I construed several claim terms related to the patents at issue in that case, U.S. Patents Nos. 6,757,842 ('842 patent); 6,149,316 ('316 patent); 5,719,808 ('808 patent); 6,426,893 ('893 patent); and 6,763,424 ('424 patent). The patents asserted in this case are related to those patents; each of the patents asserted in this lawsuit is a divisional patent or a continuation of one of the patents that plaintiff asserted against defendants in the consolidated action and shares a specification with those patents. There are three separate "groups" of patents in this case, which relate to the patents in the consolidated action as follows:

- the '713 and '660 patents are entitled "Flash EEPROM System" and share a specification with the '842 patent, '316 patent and'808 patent;
- the '702 patent is entitled "Partial Block Data Programming and Reading Operations in a Non-Volatile Memory" and shares a specification with the '424 patent; and
- the '511, '666 and '667 patents are entitled "Flash EEPROM System with Simultaneous Multiple Data Sector Programming and Storage of Physical Block Characteristics in Other Designated Blocks" and share a specification with the '893 patent.

B. Claims to be Construed:

The parties seek construction of fifteen terms, as follows.

A. Terms from the '713 and '660 patents:

1. "Address register file" ('713 pat., cl. 1) and "register file" ('713 pat., cl. 11); and

2. "defective memory location" ('660 pat., cl. 1) and "defective location" ('660 pat., cl. 15);

B. Terms from the '702 patent:

1. "Logical addresses" ('702 pat., cls. 1, 16, 24 and 33);

2. "Page" ('702 pat., cl. 1);

3. "Sub-array" ('702 pat., cls. 1 and 33)

4. "updatable data structure" ('702 pat., cl. 1) and "updatable address information" ('702 pat., cl. 16);

5. "memory controller" ('702 pat., cls. 24 and 33); and

6. The "Update Programming Step," as the parties call it ('702 pat., cls. 1, 16, 24 and 33).

C. Terms from the '511, '666 and '667 patents:

1. "Defective" ('511 pat., cl. 7);

2. "Attach the calculated redundancy codes" ('667 pat., cl. 5) and "adding the generated code to the user data" ('511 pat., cl. 1);

3. "Generating a redundancy code" ('511 pat., cls. 1 and 15);

4. "Storing, in individual ones of the second group of said blocks" ('511 pat., cls. 14 and 6 and '667 pat., cl. 1);

5. "A record stored in the memory system" ('666 pat., cl. 1);

6. "A controller adapted to [] communicate user data" ('667 pat., cl. 1); and

7. "A controller adapted to [] write data" ('667 pat., cl. 1).

As was the case in the consolidated action, the parties' disputes about the meaning of each term relate to differing views about the scope of the claims more than differing views about which language is clearest and easiest to understand. For that reason, although I will resolve the parties' disputes regarding whether the disputed limitations they identify should apply to the terms at issue, I will not endeavor to provide specific definitions for each term. As I explained to the parties in the consolidated action, providing such specific definitions tends only to encourage disputes about the meaning of those definitions. Sandisk v. Phison Electronics Corp., Case No. 07-cv-607-bbc, dkt. #582, at 5 ("It is counterproductive to resolve claims construction disputes by replacing them with new ones for the parties to dispute about at summary judgment."). To the extent the parties seek to use specific language to describe a given claim term to the jury, they can move in limine.

C. '713 Patent and '660 Patent

1. "Address register file" ('713 pat., cl. 1) and "register file" ('713 pat., cl. 11)

Surrounding Claim Language	Plaintiff's Proposed Construction	Defendants' Proposed Construction
wherein said controller includes an ***address register file***, and is such as to allow a host logical address from said host system to be converted to a physical address of said nonvolatile semiconductor flash memory based on data stored in said address register file, ['713—1] wherein said controller includes a ***register file*** to store defect mapping data; ['713—11]	An "address register file" is a memory structure within the controller having data that allows a host logical address from said host system to be converted to a physical address of said nonvolatile semiconductor flash memory The "register file" is a memory within the controller used to store defect mapping data.	the controller includes a file that can be loaded into a register and is sufficient to map the logical addresses corresponding to a defective memory location to the physical address of a replacement good memory location

For these terms, the parties' principal disputes are (1) whether "address register file" from claim 1 and "register file" from claim 11 of the '713 patent mean the same thing (and thus whether claim 1's "address register file" must be used to store defect mapping data and claim 11's "register file" must allow a logical address from a host system to be converted to a physical flash address); (2) whether the "address register file" from claim 1 must include all the information needed to map the logical address corresponding to a defective location to a physical address of a replacement location; and (3) whether the "register file" from claim 11 must contain a defect map. Defendants seek all of these limitations.

a. The terms "address register file" from claim 1 and "register file" from claim 11 do not mean the same thing

Defendants argue that the two claim terms are identical because "the term 'register file' should have a single meaning for all of the claims." Dkt. #109, at 8. However, the fact that the term "register file" may have a single meaning does not mean that the "address register file" described in detail in claim 1 must include all the features described in detail with respect to the "register file" in claim 11 or vice versa. Indeed, the fact that the requirements of each term are described in detail is evidence that the term "register file" is a general term and the claims describe separate, distinct tasks that the claimed "register file" must perform in each instance.

Defendants' contention that the "address register file" in claim 1 and the "register file" in claim 11 both include the requirements described in each separate claim works only if the requirements listed in each claim were already part of the meaning of "register file," meaning it would be superfluous to list those requirements. The claim language does not support a reading of these two terms as identical for the simple reason that both terms are followed by specific language describing the data that must be stored in the claimed "file" in each instance. The specific language from each of these independent claims cannot be lifted out and transported to the other simply because both refer to a "register file."

Defendants point to the specification as evidence that there is only one type of

"register file," but their sole citation relating to storage in a register file is a statement that "the defect pointer map" is "stored in the register file 509." '713 pat., col. 9, ln. 45. Apparently, their view is that such a "defect pointer map" must include both the defect mapping data described in claim 11 and the address conversion data described in claim 1. Defendants do not explain why this is so, but more important, the statement they rely on occurs in the context of describing Figure 6, which "illustrates the read data path control in the *preferred embodiment*." Id., col. 8, lns. 59-60. Although Figure 6 illustrates both "load[ing] the header information (Head, Cylinder and sector) into a holding register file 509," id., col. 9, lns. 3-5, and "load[ing] . . . bad address locations into the holding register file 509," id., col. 9, lns. 25-26; it remains nothing more than a preferred embodiment. The fact that a preferred embodiment contains limitations is not a sufficient reason to import those limitations into the claim itself, especially when doing so would disregard the broader claim language. Phillips v. AWH Corp., 415 F.3d 1303, 1323 (Fed. Cir. 2005) (claim should not be limited simply because specific embodiment shows requested limitation).

According to defendants, the use of the term "file" in "register file" in claims 1 and 11 and throughout the specification supports a finding that there can be only one such "register file," that all relevant "register" data must be "stored in the same place." Dkt. #109, at 9. However, without any suggestion in the specification or the terms that only one "register file" can exist, there is no basis for reading in such a limitation. Indeed, the

specification refers to "*a* holding register file" used for storing header information, id., col. 9, ln. 4-5, suggesting that a separate "register file" could be used to store other information. In conclusion, the two terms at issue do not mean the same thing. The "address register file" in claim 1 need not be "used to store defect mapping data" as required in claim 11 and the "register file" in claim 11 need not be "such as to allow a host logical address from said host system to be converted to a physical address of said nonvolatile semiconductor flash memory based on data stored" in the address register file as required in claim 1.

b. The "address register file" need not contain all the information needed to convert a host logical address into a physical address

According to defendants, the claimed "file" must contain enough information to allow a host logical address from the host system to be converted to a physical address "without using information stored elsewhere or engaging in any separate calculations or other operations." Dkt. #109, at 10. As a starting point, this argument relates only to claim 1's "address register file" because, as explained above, the "register file" in claim 11 need not contain address conversion data. However, even for claim 1's address register file, the claim language does not reach as far as defendants contend. The claim requires only that the address register file contain enough data to "allow a host logical address . . . to be converted to a [flash] physical address . . . based on data stored in said address register file."

Defendants contend that the requirement that the conversion be "based on data stored in said address register file" means that must be based *only* on such data. However, the word "only" is absent from the claim, and thus the claim language alone does not support its imposition.

Defendants contend that the specification also supports their position, pointing to the following statement:

> When the controller is given an address to access data, the controller compares this address against the sector defect map. If a match occurs then access to the defective sector is denied and the substitute address present in the defect map is entered and the corresponding substitute sector is accessed instead.

'713 pat., col. 11, lns. 47-52. Although defendants suggest that this language is used to describe the "invention," it appears in a paragraph describing "another embodiment." Again, without more, the presence of certain features in an embodiment do not support reading a limitation into the claim itself. In short, defendants' citations fail to support a requirement that claim 1's "address register file" contain all the information needed to perform the claimed address conversion.

c. The "register file" in claim 11 need not contain a "defect map"

Defendants also contend that the claimed "register file" must contain a defect map, by which they seem to mean that it must contain *all* defect mapping data. Because, as

explained above, the two terms at issue do not have the same meaning, this argument would apply only to claim 11. However, neither the claim language nor the specification supports requiring a defect map to be stored in the "register file" in claim 11. Defendants point to the following passages:

> One feature of the invention allows defect mapping at cell level in which a defective cell is replaced by a substitute cell from the same sector. The defect pointer which connects the address of the defective cell to that of the substitute cell is stored in a defect map. Every time the defective cell is accessed, its bad data is replaced by the good data from the substitute cell.
>
> ***
>
> The present invention also has provision for defect mapping of the whole sector, but only after the number of defective cells in the sector has exceeded the cell defect mapping's capacity for that specific sector. . . . When the number in a sector exceeds a predetermined value, the controller marks that sector as defective and maps it to another sector. The defect pointer for the linked sectors may be stored in a sector defect map.

'713 pat., col. 2, lns. 30-36 and col. 11, lns. 31-39.

Neither passage establishes that claim 11 requires the register file to contain a complete defect map. The passages provide only that the claimed invention provides for defect mapping at both the cell and sector level and that a cell-level defect pointer is stored in a defect map and a sector-level pointer "may be" stored in such a map. Neither passage ties the defect map to the claimed "register file."

Even if the discussion of a defect map suggested storage of the map in the claimed

register file, at most, this would make such storage a preferred embodiment of claim 11, not an element of the claim. The claim language states that the "register file" in claim 11 must store "defect mapping data," not a defect map. In the specification, the inventor referred to both a "defect map" and "defect mapping data," without ever equating the two. At most, a "defect map" is a specific sort of "defect mapping data." Because the claim language uses broader language, the narrower "defect map" is not a limitation of the claim. Thus, claim 11's "register file" need only store "defect mapping data," not a defect map.

The parties do agree on one point about the meaning of the claim terms: "the 'defect mapping data' in the asserted claims contains enough information to directly link each defective memory location to a replacement memory location." Defs.' Br., dkt. #109, at 10-11; Plt.'s Resp. Br., dkt. #117, at 5. However, neither party is seeking that construction, and in light of their agreement, there is no dispute to resolve with respect to this aspect of the claim term.

2. "defective memory location" ('660 pat., cl. 1) and "defective location" ('660 pat., cl. 15)

Surrounding Claim Language	**Plaintiff's Proposed Construction**	**Defendants' Proposed Construction**

receiving a logical address at a controller for the flash memory array and determining that the logical address corresponds to a ***defective memory location*** ['660—1] a selecting unit receiving a logical address for the flash memory array, determining that the logical address corresponds to a ***defective location*** ['660—15]	The term "defective" refers to a condition whereby the block (as a whole) is deemed unusable. A memory location may be deemed defective when the bit fails to program and/or erase.	determining that the received logical address corresponds to a memory location which has accumulated such a number of defective cells that the defects cannot be corrected by the error correction codes (ECC) corresponding to data stored in the memory location

The parties agree that the claimed "defective memory location" in claim 1 of the '660 patent and "defective location" from claim 15 of the '660 patent have the same meaning, but they disagree about what that meaning must be. The parties' disputes for this term are: (1) whether the claimed "defective memory location" must always be an entire block or could also be a single cell; and (2) whether a memory location can be "deemed defective" even if the defects can still be corrected by error correction code. (Plaintiff suggests that there is a third dispute, whether a memory location can be deemed defective other than by use of the error correction code. However, defendants do not seek a construction that would impose any limitation on "the reason a certain memory location becomes defective." Defs. Resp. Br., dkt. #119, at 7.)

a. "Defective memory location" is not limited to a "block"

As defendants point out, the patent refers both to defective blocks, also known as sectors, and defective cells, or bits. '660 pat., col. 11, lns. 28-30 (describes marking "sector" as defective and maps it to another sector); id., col. 11, lns. 18-21 (describes writing "collection of bits" flagged as defective into memory at "alternative defects data locations"). Moreover, the claim uses the term "memory location" instead of "block."

Defendants' citations support reading the claimed "memory location" more broadly than "block," as defendants contend. Plaintiff contends that the surrounding claim language shows otherwise. ACTV, Inc. v. Walt Disney Co., 346 F.3d 1082, 1088 (Fed. Cir. 2003) (court must consider context of surrounding words of claim when construing term). As plaintiff points out, in claim 1 of the '660 patent, "defective memory location" appears in step (a), which involves determining that a logical address received by the controller "corresponds to a defective memory location." The next step in claim 1 involves "determining . . . a block of the flash memory array to access based on the received logical address." This citation does not support a requirement that the claimed memory location be of a "block." The mere fact that a block is ultimately accessed "based on the received logical address" does not mean the address must refer to an entire block; a logical address of a single cell within a block could allow a controller to determine which block to access.

b. Meaning of "defective"

The parties' principal disagreement is about what is required for a memory location to be deemed "defective." The term is not defined in the claims or specification, and the surrounding language of the claims provides no guidance on what the term means. Plaintiff contends that the term must be read broadly, citing its expert's assertion that a person of ordinary skill in the art would understand that a block is "defective" when it is deemed "unusable by the controller of the system." In addition, plaintiff points to two instances in the specification describing designating memory locations as defective, contending these disclosures show that the controller decides whether a block is defective. First, the specification explains that "[i]f a bit fails to verify after prolonged program/verify cycling, the controller will designate that bit as defective and update the defect map accordingly." '660 pat., col. 11, lns. 4-6. Next, the specification adds that "[w]hen the number [of defective cells] in a sector exceeds a predetermined value, the controller marks that sector as defective and maps it to another sector." Id., col. 11, lns. 28-30.

Plaintiff's citations support their position that the controller determines when a block is defective. However, plaintiff is not merely seeking a construction that the controller determines when a block is defective. Instead, they are attempting to argue that the criteria a controller may apply when determining this may be extremely broad: a *block* may be deemed defective when a *bit* fails to program or erase. In other words, when a single bit goes

bad, the controller may deem the entire block "defective." (Defendants point out that a single block averages 2.16 million bits these days, but the specification was dealing with a less drastic scale, refer to "sectors," or blocks, of only 512 bytes, or 4,096 bits. '660 pat., col. 5, lns. 1-2.)

The cited language does not support plaintiff's broad construction. Although the controller may deem a block defective at some "predetermined value," nothing suggests that value could be *any* number of bits, including a single bit. Indeed, plaintiff's suggestion that any value can be set as the "predetermined value" would make the term "defective" meaningless: the predetermined value could just be set at 0 and then all fully functional blocks would be defective blocks.

Moreover, plaintiff's reading of "defective" would result in large numbers of non-defective cells going unused because they reside within a block containing a single defective cell. Although such "wastefulness" in itself is not necessarily a problem, it is a problem in the context of this patent because the specification explains that avoiding wastage is a key aspect of the invention:

> In a disk system made from such [flash] memory devices, low cost considerations necessitate efficient handling of defects. [Aside from the use of flash memory chips instead of conventional memory, a]nother important feature of the invention enables the error correction scheme to conserve as much memory as possible. Essentially, it calls for the defective cells to be remapped cell by cell rather than by throwing away the whole sector (512 bytes typically) whenever a defect occurs in it.

'660 pat., col. 7, lns. 42-51. Elsewhere, the specification explains that "the present invention has provision for defect mapping of the whole sector, but only after the number of defective cells in the sector has exceeded the cell defect mapping's capacity for that specific sector." Id., col. 11, lns. 24-27; see also id., col. 7, ln. 65-col. 8, ln. 5 ("One important feature of the present invention is the ability for the system to correct for hard errors whenever they occur. . . . Also during read operation, defective cells are detected and located by the [error correction code]. As soon as a defective cell is identified, the controller will apply defect mapping to replace the defective cell with a space cell located usually within the same sector. This dynamic correction of hard errors, in addition to conventional error correction schemes, significantly prolongs the life of the device.").

Thus, plaintiff's construction will not be adopted. The patent specification's description of the "present invention" as "conserv[ing] as much memory as possible" and providing for defect mapping "only after the number of defective cells has exceeded the cell defect mapping's capacity" makes it clear that an entire block should be deemed defective only in limited circumstances; the presence of one single defective bit would not suffice. Trading Technologies International, Inc. v. eSpeed, Inc., 595 F.3d 1340, 1353-54 (Fed. Cir. 2010) (describing limitation as part of "the present invention" or "the invention" is strong evidence that claims should be limited); see also Honeywell International, Inc. v. ITT Industries, Inc., 452 F.3d 1312, 1318 (Fed. Cir. 2006) (four references to fuel filter as "this

invention" or "the present invention" warranted limiting the invention to fuel filter).

Moreover, these descriptions support defendant's view that determining when a block or other memory location becomes "defective" should relate to the capacity of the error correction code (or at least the "error correction scheme" to the extent there is a difference) to correct the defects within that block. As mentioned above, the invention limits the availability of sector defect mapping to "after the number of defective cells in the sector has exceeded the cell defect mapping's capacity for that specific sector." '660 pat., col. 11, lns. 24-27. Although the specification does not explain what that "capacity" must be except to note that it is some "predetermined value," another aspect of the invention is that it includes an "error correction scheme" aimed at conserving "as much memory as possible" and "essentially" remapping defects cell-by-cell within a sector. Id., col. 7, lns. 42-51.

Plaintiff contends that the reference to conserving "as much memory as possible" using an "error correction scheme" need not amount to a limitation on block defect mapping and suggests that the reference is related to aspects of the invention other than the claim term in dispute. However, when plaintiff was asked at the claims construction hearing to identify some other part of the patent that might "carry out" the concept of conserving as much memory as possible, plaintiff could not identify any. Hrg. Tr., dkt. #127, at 14-18. (The closest plaintiff comes is to note that the "columns of the patent," the specification, refers to both sector mapping and cell mapping.)

Thus, the invention includes a requirement that the controller must wait until the memory location's cell defect mapping "capacity" has been reached to deem the memory location defective and in the meantime must use an "error correction scheme" to "essentially" remap cell-by-cell.

D. '702 Patent

1. "Logical addresses" ('702 pat., cls. 1, 16, 24 and 33)

Surrounding Claim Language	**Plaintiff's Proposed Construction**	**Defendants' Proposed Construction**

programming a first group of a plurality of pages in at least the first and second blocks with original data, the pages of original data having ***logical addresses*** associated therewith ['702—1] programming original data into individual ones of a first plurality of pages in at least a first block, the original data having ***logical addresses*** associated therewith ['702—16] programming the received plurality of pages of original user data into a first plurality of pages of storage elements ['702—24] programming the received plurality of pages of original data into a first plurality of pages of storage elements ['702—33]	— non-physical address associated with the data — "logical address" is not limited to an address sufficient to identify an individual page of data — A "logical address" is not necessarily programmed into a first plurality of pages in a first block.	programming a first plurality [group] of pages in a first block, each individual page being programmed with original data and an address sufficient to identify the individual page of data

There are two issues related to this term: (1) whether the logical address must be of a specific page of data; and (2) whether the logical address must be programmed into the page or pages to which it is designated. The parties agree that a logical address is a non-physical address associated with certain data, but disagree about where the logical address must be located and what details it must include.

a. "Logical address" need not refer to a specific "page"

According to defendants, the claimed "logical address" must refer to a specific "page" as opposed to only part of a page or multiple pages. Defendants point out that the asserted claims repeatedly refer to data in terms of pages. However, the claim language at issue does not restrict the language as defendants suggest. For example, claim 1 requires "pages of original data having logical addresses associated therewith." '702 pat., cl. 1. (Claim 16 leaves out the word "pages.") However, the language does not suggest that there can be only one page for each logical address or one logical address for each page.

Defendants do not cite any reference in the claim language or the specification that would prevent such a setup. Instead, they argue only that the '702 patent is a "page-based system" and for such a system to work, "the logical address of each page must be sufficient to identify that individual page." Defs.' Resp. Br., dkt. #119, at 11. This is not the right time to be arguing about whether the patent will "work." Indeed, the law prohibits construing claims more narrowly just to fix an unworkable patent. Chef America, Inc. v. Lamb-Weston, Inc., 358 F.3d 1371, 1374 (Fed. Cir. 2004) ("[C]ourts may not redraft claims, whether to make them operable or to sustain their validity."). To the extent there is an enablement problem because the disclosures in the patent would not create a workable product, that is a matter for summary judgment or trial.

In short, defendants offer no persuasive reason why the claimed "logical addresses"

must be limited to referring to a single page and why a page could not have more than one such "logical address."

b. "Logical address" need not be programmed into the original page or pages

Defendants recognize that the claim language does not specify that the claimed "logical address" must be stored in the original page or pages to which it was designated. They contend that the limitation applies nonetheless because (1) each relevant figure depicts in-page storage of a logical page address; (2) all the embodiments describe in-page storage; and (3) the specification does not contemplate any alternatives. See, e.g., '702 pat., col. 2, lns. 39-41 and 46-49 and Figs. 8, 10 and 11.

Defendants are fighting an uphill battle. As explained above, generally, the use of one or more embodiments in the specification to illustrate how the patent works is not a ground for circumscribing broader claim language. Phillips, 415 F.3d at 1323. Even if an embodiment is the only one disclosed, it may serve to limit a claim only if it is clear that the patentee intended to limit the scope of the claims to the disclosed embodiment. Id.; see also On Demand Machine Corp. v. Ingram Industries, Inc., 442 F.3d 1331, 1339-40 (Fed. Cir. 2006) (claim limitation warranted because specification used the term "customer" repeatedly in specialized context); Nystrom v. TREX Co., Inc., 424 F.3d 1136, 1144-45 (Fed. Cir. 2005) (claim limitation warranted because written description and prosecution history used

term "board" consistently to refer to wood decking materials cut from log). One of the ways such an intent may be found is from the presence of "repeated and definitive remarks" in a specification that a particular limitation applies to the claims. Computer Docking Station Corp. v. Dell, Inc., 519 F.3d 1366, 1374 (Fed. Cir. 2008).

Defendants contend that the use of the term "logical address" in the '702 patent is akin to the use of the term "spike" in ICU Medical Inc. v. Alaris Medical Systems, Inc., 558 F.3d 1368, 1374-76 (Fed. Cir. 2009). In that case, the court of appeals concluded that the term "spike" required an elongated, pointed tip for piercing a seal. Id. at 1375. In so doing, the court noted that "each figure depicts the spike as elongated and pointed," each figure depicting an activated valve showed a spike piercing the seal and the patents did not describe piercing as optional or describe any non-piercing item as a spike. Id. Defendants do not discuss the next observation by the court of appeals, however, which is that the plaintiff had "not offer[ed any] support from any intrinsic or extrinsic source in support of its claim that the ordinary meaning of spike would include a non-pointed structure such as a tube or straw." Id. Thus, ICU Medical does not stand for the proposition that the mere presence of particular examples and absence of other types of examples would support limiting the claim language. Indeed, if that were all it took to impose a limitation, the exception would swallow the rule.

Looking at the examples defendants provide of usage of the term "logical address,"

they are limited to examples that happen to include in-page storage of logical page address; they include no counterexamples. Nothing about these examples supports a finding that the patentee intended to limit the claims of the '702 patent to storage in-page. Moreover, as plaintiff points out, in a sense there was a "counterexample": Figure 4 shows the prior art approach to updating, which involved rewriting all the pages and replacing the new pages in order. Because the pages remained in order, there was no need to store an in-page logical address and none is shown. Under these circumstances, had the patentee wanted to limit the invention to in-page storage, it would have been an easy matter to include an express statement in the claim language, or even distinguish that aspect of the prior art in the specification.

Defendants devote much of their argument to explaining why the page-based system of the '702 patent would not work without in-page storage, and plaintiff attempts to offer counterexamples to needing in-page storage, in the form of Figures 9 and 12 of the '702 patent. It is not necessary to resolve the parties' disagreements about the meaning of these figures and whether the claimed invention would "work" because, as explained above, whether a patent "works" is a question of enablement, not claim construction. Defendants attempt to frame the matter in terms of what the "disclosed invention relies upon . . . to function," Defs.' Br., dkt. #109, at 18, citing ICU Medical, 558 F.3d at 1375 ("It is entirely proper to consider the functions of an invention in seeking to determine the meaning of

particular claim language). However, the cited passage is taken out of context. It refers not to *whether* something disclosed in the patent will function, but *how* it functions in the context of the patent. Id. (concluding that it was appropriate to add the functional language "for piercing the seal" as part of the definition of "spike" because the term "spike" alone failed to suggest the degree to which the spike must be pointed).

At any rate, defendants' position is not that storage out of page will never work in a structure practicing the asserted claims; otherwise, they would not be facing infringement claims for their products storing logical addresses out of the original page, or at least would not need to seek a limiting construction to fend off the infringement claims. Instead, their view is that the patent does not disclose how such a structure would work in a page-based system. As explained above, that is not an argument for construing the claim, but rather for lack of enablement, and now is not the time for that determination. In conclusion, defendants' proposed limitations for "logical address" are not adopted. (Plaintiffs' proposed constructions were just denials of those limitations.)

2. "Page" ('702 pat., cl. 1)

Surrounding Claim Language	**Plaintiff's Proposed Construction**	**Defendants' Proposed Construction**

the plurality of blocks being individually divided into a given number of a plurality of ***pages*** of charge storage elements that are programmable together ['702—1]	— A sub-division of a block, that may comprise one or more user data sectors plus overhead data and which serves as the unit of programming and reading — A "page" does not necessarily store an address sufficient to identify the individual page of data.	a sub-division of a block, comprising one or more user data sectors plus overhead data including an address sufficient to identify the individual page of data

The parties agree that a "page" is a "subdivision of a block" that serves as the unit of programming and reading and agree that the page should include overhead data. What they disagree about is whether each subdivision must store an address sufficient to identify the individual page of data. As defendants acknowledge, their arguments for including this limitation are identical to those they asserted for imposing that limitation into the term "logical address." Hrg. Tr., dkt. #127, at 70 (what overhead data must be included "takes us back to the discussion we just had [about 'logical address'] in which every embodiment and every teaching of this patent is that in order for this system to work, the overhead data is necessarily going to be a logical block address and a logical page address."). As I explained above, the patent and specification does not support a requirement that the claimed "logical address" be included in each page. For the same reasons, there is no reason to require each "page" to contain page-identifying information. To the extent such a setup would not

"work," defendants will just have to argue that at a later date.

Defendants point out that plaintiff "distinguished" one of the '702 patent's parents from prior art by stating that the prior art managed data on a block level while that patent focused on "the management of pages of data, rather than entire blocks." Defs.' Resp. Br., dkt. #119, at 15 (quoting prosecution history of United States Patent No. 7,818,490, page 6 of 1/8/2007 Amendment). Defendants suggest that these statements amount to an admission that the "invention" manages data on a page level. Assuming that management on a page level meant each page would have to have in-page logical addresses (a big assumption), defendants do not explain why a parent patent's "invention" should be treated as the invention of a patent in suit, or why the statement could be read so broadly as to include the inventions disclosed in the '702 patent.

3. "Sub-array" ('702 pat., cls. 1 and 33)

Surrounding Claim Language	**Plaintiff's Proposed Construction**	**Defendants' Proposed Construction**

"the array being divided into a plurality of ***subarrays*** in which the charge storage elements within individual sub—arrays are programmable independently, wherein the ***sub-arrays*** are individually divided into a plurality of blocks of charge storage elements that are erasable together" ['702—1] an array of re-programmable non-volatile storage elements arranged in a plurality of ***subarrays*** of storage elements in which programming operations may be performed independently, the ***sub-arrays*** individually being divided into a plurality of blocks of storage elements that are erasable together as a unit, the individual blocks being divided into a plurality of pages of storage elements that have specified offset positions within their respective blocks ['702—33]	— The blocks in a physically distinct subdivision of memory cells of an array on an integrated circuit chip — Two or more subarrays may be read and/or programmed together as part of a single operation.	A subdivision of an array

For this term, the parties' main disagreement is whether more than one claimed sub-array may be read or programmed together as part of a single operation. The parties also disagree about whether the "array" from which the "subarray" is divided must be present on a single integrated circuit chip.

a. Sub-arrays must be simultaneously programmable

The parties' dispute about simultaneous programming comes down to a disagreement about the claims' requirement that the sub-arrays be "programmable independently" (cl. 1) or have "programming operations" that "perform independently" (cl. 33). According to defendants, plaintiff's construction of sub-arrays as simultaneously programmable would "vitiate" the "independent programming" requirement. However, defendants appear to conflate "programmable" with "programmed." Plaintiff does not seek a construction requiring the sub-arrays to *always* program simultaneously. Indeed, plaintiff does not suggest that any given sub-array must *ever* program simultaneously. The only limitation plaintiff seeks to include is the requirement that the sub-array be *capable* of being programmed simultaneously. (At first blush, it may appear that plaintiff is not asking for a limitation at all, but rather only fending off a possible construction requiring non-simultaneous programming. However, the parties' discussions make it clear that plaintiff seeks to require simultaneous programmability; in addition, defendants do not seek any counter-requirement of only *non*-simultaneous programming.)

Defendants' argument fails because they do not explain why a sub-array cannot be both "independent" and simultaneously programmable. There is nothing inherent in the notion of "independent" that would preclude such a concept. At most, such simultaneous programmability would require interconnectedness between sub-arrays; this alone would not interfere with the claimed "independence" of each so long as each sub-array could operate

on its own. (Analogous would be an individual that can work either "independently" or in a team.) Indeed, as the specification notes, the sub-arrays need only be "largely independent." '702 pat., col. 11, lns. 60-67.

Defendants point out that the claim language does not expressly require such simultaneous programmability. Instead, the claims simply leave open the possibility of simultaneous programming, with claim 1 providing that the sub-array need only be programm*able* independently and claim 33 providing that the programming operations "may" be performed independently. Thus, neither claim precludes simultaneous programming, but neither requires such capability. However, plaintiff points to a passage in the specification that supports reading in such a limitation:

> Another principal aspect of the present invention groups together two or more blocks positioned in separate units of the memory array (also termed sub-arrays) for programming and reading together as part of a single operation.

'702 pat., col. 3, lns. 8-10. The disclosed "grouping" of two or more sub-arrays "for programming and reading together" could work only if the sub-arrays were programmable simultaneously. As explained above, descriptions of the "present invention" are strong evidence that a limitation was intended. Trading Technologies International, 595 F.3d at 1353-54. Defendants do not respond to this citation at all, or suggest that it might relate to other aspects of the invention beside the claimed "sub-array." I am persuaded that the statement shows that the patentee intended to limit the claims to sub-arrays that are capable

of being programmed and read "together as part a single operation."

b. There is no requirement that an entire array be present on a single chip

Next, the parties appear to disagree about whether the entire array to which a given sub-array belongs must be present on a single chip. However, at the claims construction hearing, plaintiff acknowledged that at least "theoretically," a single array can appear on multiple chips. Hrg. Tr., dkt. #127, at 32. Plaintiff seeks only to insure that all the sub-arrays, on one chip or more, must all be "part of the same array," rather than simply a random assortment of pieces of arrays on different chips. However, plaintiff does not articulate what sort of limitation would address their concerns and comport with the claim language. Plaintiff will have to wait until a later stage of the case to articulate what it means when it says the sub-arrays must all be "part of the same array."

4. "updatable data structure" ('702 pat., cl. 1) and "updatable address information" ('702 pat., cl. 16)

Surrounding Claim Language	**Plaintiff's Proposed Construction**	**Defendants' Proposed Construction**

an ***updatable data structure*** that links one or more physical addresses of the first group of pages with one or more of the logical addresses associated with the data stored therein ['702—1] ***updatable address information*** that links physical addresses of the first and second blocks storing original and updated data with the logical addresses associated with the stored data ['702—16]	— An "updatable data structure" may consist of one or more tables. — The "updatable address information" may also consist of one or more tables and includes the physical address of multiple blocks, but need not include the physical addresses of pages.	The "updatable data structure" or "updatable address information": 1. need not take the form of a table or map, 2. is something more than simply storing a pointer to the physical address of the first group of pages [or blocks], and 3. links the logical addresses of pages (not blocks) to the physical addresses of those pages (not blocks)

The parties dispute whether (1) the two terms mean the same thing; (2) the claimed "linking" in either case can be as little as storing a pointer to the physical address of the first group of pages; and (3) the linking must be page-to-page as opposed to block-to-block. (The parties appear to dispute whether the claimed structures must take the form of a table or a map, but not so: plaintiff argues they "may" take such a form and defendants argue they "need not." Those two views are consistent with one another so there is no real dispute.)

a. The terms "updatable data structure" and "updatable address information" are not synonymous

Defendants contend that these two terms have the same meaning, but go nowhere with that argument except to argue that the terms should have identical limitations. On the other hand, plaintiff does not suggest that the meaning of either claim term differs in a way that relates to plaintiff's arguments against defendants' proposed limitations, so this disagreement is probably a non-issue. To the extent defendants are seeking to transport limitations across the claims, that will not work. As plaintiff points out, the claim language surrounding each of the different terms at issue requires different things from each of those terms. It would be improper to impose requirements described in one claim on another claim just because the claims use certain words having the same meaning.

b. The claimed "linking" must be more than simply storing a pointer in the physical address that points to the location of the logical address.

Defendants seek to require that the claimed linking be "something more than simply storing a pointer to the physical address of the first group of pages [or blocks]," a limitation drawn from a claim construction from the consolidated action. In particular, in that case I addressed the meaning of the term "linking" in a different flash EEPROM patent (the '842 patent) and concluded that

> "linking the address of such unusable blocks with addresses of other blocks that are useable" need not take the form of a map or table, but must be something more than simply storing a pointer in the unusable block that

points to a usable block.

Case No. 07-605, dkt. #604, at 12. Defendants seek to import this construction into this term, and have agreed to the court's proposal that the language be altered to track the specifics of this claim. In particular, defendants agree to a construction that "The claim structures cannot link the physical and logical addresses by simply storing a pointer in the physical address that points to the location of the logical address." Hrg. Tr., dkt. #127, at 73. This alteration tracks the fact that, unlike in the previous case, one of the two addresses being linked in this case is a logical address.

Plaintiff contends that nothing in the language of either of those claims precludes linking by pointers in the fashion described, but the rationale for so limiting the claims is that such "linking" is too indirect, as I explained in the earlier case. Blocks linked by storing a pointer in one block that points to the other block do not link the *addresses* of the two blocks as claimed. Id. The same applies here: storing a pointer that points to a logical address links that particular location with the given logical address, but it does not link the *physical address describing* that physical location with the logical address, except indirectly.

c. There is no requirement that the claimed "linking" be at the page level

Defendants again rely solely on their theory that the claimed "logical addresses" operate only at the page level to support their theory that the claimed "linking" must occur

at the page level. As explained above, their theory does not support imposing any limitations on the claim language; at most it supports a defense for lack of enablement.

5. "memory controller" ('702 pat., cls. 24 and 33)

Surrounding Claim Language	**Plaintiff's Proposed Construction**	**Defendants' Proposed Construction**
the ***memory controller*** being characterized by performing at least the following operations: (a) responds to receipt . . . by programming the received plurality of pages of original user data into a first plurality of pages of storage elements in the preset order in at least a first one of the blocks ['702—24] a ***memory controller*** connected with the interface and with the plurality of blocks of storage elements, the memory controller being characterized by performing at least the following operations: . . . (b) responds to receipt . . . by programming the received plurality of pages of original data into a first plurality of pages of storage elements in a first plurality of blocks forming a first metablock, ['702—33]	The memory controller may "program[] the received plurality of pages." Such programming includes by providing the plurality of blocks, over a bus, a program command, the appropriate addresses, and the received pages.	the controller programs the received plurality of pages by applying the necessary programming voltage to the individual memory cells of the plurality of pages

The parties dispute whether the claimed "memory controller" must apply voltage to

perform the "programming" claimed in claims 24 and 33 of the '702 patent. As plaintiff points out, the claims do not discuss the application of voltage at all, remaining silent on whether the controller must apply voltage to the cells or whether the flash chip could do it instead. In addition, the term "program" as a term of art does not require a controller to apply actual voltage. Even at the time of the invention, it was known that a flash chip could provide the actual voltage instead of the controller, and a controller's "programming" role could be to simply load "command, address and transfer data to and from the flash memory array." '702 pat. col. 5, lns. 38-41; Rhyne Decl., dkt. #105 , ¶ 23.

A stronger reason for imposing the requirement that defendants request comes from the surrounding claim language. The claim requires the controller to be "connected with the interface and with the plurality of blocks of storage elements," '702 pat., cls. 24, 33, the "interface" being the means by which "pages of user data, logical addresses associated with the page of user data, commands and status signals" pass through. Id. As defendants explain, the fact that "logical addresses" pass through this interface shows that the claimed interface is located between the controller and the host, not the controller and the flash memory. Plaintiff has no response to this point.

Thus, the claims recite a host-to-controller interface without reciting any controller-to-block interface, instead requiring the controller to be "connected with the . . .plurality of blocks of storage elements." Plaintiff contends that such "connection" could be "indirect,"

in the sense that the controller could be "connected" to the blocks through an interface. However, the claim language weighs against such a reading by leaving out any reference to a flash interface and instead describing a connection with the blocks themselves. Not only that, but the same claim language describes a connection with a *host* interface. By omitting reference to connection with a flash interface and instead referring to a connection with the blocks themselves, the claim language shows that the patentee intended to require the controller to have a direct connection with the blocks.

Moreover, some of the same references plaintiff relies on to show that it was known that a flash memory could apply voltage also show that it was known that a controller could be connected with a flash interface instead of directly to the blocks. '702 pat., Fig. 1 (showing connection between controller and "data register"); id., Fig. 2 (showing connection between controller and flash memory device through "flash media interface"). The failure of the claim to account for such an "indirect" connection despite its prevalence in the prior art weighs even further against reading the claim as broadly as plaintiff requests.

Thus, I agree with defendants that the required "connection" between the controller and the blocks is direct instead of through an interface. This conclusion does not establish that the controller must apply the voltage, however. The presence of a direct connection between a controller and a block does not prevent the controller from also having a connection with an interface allowing flash application of voltage (although this arrangement

might be unlikely or impractical).

To take their argument the rest of the way, defendants rely on disclosures in two patents "incorporated by reference" into the '702 patent: United States Patents Nos. 6,426,893 and 5,602,987. Zenon Envtl., Inc. v. U.S. Filter Corp., 506 F.3d 1370, 1378 (Fed.Cir. 2007) ("Incorporation by reference provides a method for integrating material from various documents into a host document . . . by citing such material in a manner that makes clear that the material is effectively part of the host document as if it were explicitly contained therein.") (internal quotations omitted). (Technically, the '702 patent incorporates U.S. Application No. 09/505,555, which issued as the '893 patent, and the '893 patent incorporates the '987 patent. However, neither party suggests that these details would require treating the disclosures in either patent any differently from the way they would be treated as part of the specification of the '702 patent.)

These specifications do not establish that the controller *must* apply voltage; instead, they describe embodiments by which a controller does apply voltage. These include the reference in the '893 patent to "byte 2 a programming voltage" that "the controller uses" and the reference in the '987 patent to the controller's "appl[ying] a pulse of programming (or writing) voltages." Defendants contend that the claims will have to be ruled "invalid as lacking written description" if these embodiments are not imported. However, the possibility of a ruling of invalidity does not give the court the power to redraft the patent to save it from

invalidity. If it turns out that the patents do not sufficiently describe the claimed "programming," then the relevant claims will have to be ruled invalid; however, the language as it is written leaves open what sort of "programming" must be performed. Thus, although I agree that the controller must be connected directly to the blocks, I disagree that the claimed controllers must apply voltage.

6. The Update Programming Step ('702 pat., cls. 1, 16, 24 and 33)

Surrounding Claim Language	**Plaintiff's Proposed Construction**	**Defendants' Proposed Construction**
programming an updated version of some of the original data into a second group of one or more pages less than said given number in at least one update block [702—1] thereafter programming into individual one of a second plurality of pages in a second block, an updated version of less than the given number of pages of the original data programmed into the first plurality of pages [702—16] programming the received one or more pages of updated user data into a second one or more pages of storage elements in at least a second block [702—24, — 33]	Must occur without marking the original (now superseded) pages of data in the first block with an invalid data flag.	No construction needed, or, alternatively, programming one or more pages in a second block [group] with updated data

The parties disagree whether the claimed update programming step must occur without marking pages of data in the first block with an invalid data flag. Plaintiff contends that the limitation should be imposed because the specification describes use of invalid data flags as part of the prior art, '702 pat., col. 6, lns. 32-50, and then explains that

> [w]hat is needed is a mechanism by which data that partially supercedes data stored in an existing block can be written without either copying unchanged data from the existing block or programming flags to pages that have been previously programmed.

Id., col. 7, lns. 14-18. The specification adds that "a principal aspect of the present invention" includes avoiding "the need to update flags within the original block." Id., col. 2, lns. 34-35.

As an initial matter, defendants do not respond to this argument of plaintiff's. They focus on plaintiff's position that an ancestor patent, the '424 patent, disclaimed the use of invalid data flags. Indeed, defendants state that plaintiff's "rationale for its construction of this term is based solely on a prosecution disclaimer," Defs.' Resp. Br., dkt. #119, at 25, despite the fact that plaintiff had identified the block quote above as supporting its position that the claims should be limited as plaintiff requests. Plt.'s Br., dkt. #102, at 42. Also, although plaintiff cited all of these references in its response brief, defendants continued to focus on the ancestor disclaimer issue without addressing the passages from the '702 patent. Thus, defendants have waived any argument that the passages cited fail to warrant

imposition of a limitation.

Even if defendants' failure to respond were not waiver, however, I would side with plaintiff. The passages plaintiff cites in the '702 patent support the requested limitation because they show that patentee intended to disclaim the use of invalid data flags. (Because I conclude that the '702 patent alone establishes this limitation, I need not consider whether the presence of a disclaimer in the '424 patent could establish the same limitation.)

E. '511, '666 and '667 Patents

1. "Defective" ('511 pat., cl. 7)

Surrounding Claim Language	**Plaintiff's Proposed Construction**	**Defendants' Proposed Construction**
7. The method according to claim 6, wherein storing data of information related to physical characteristics of the plurality of memory cell blocks or their operation comprises storing an indication of whether any of the plurality of memory cell blocks is ***defective***. ['511—7]	— Condition whereby the block (as a whole) is deemed unusable. — A memory location may be deemed defective when the bit fails to program and/or erase.	a block which has accumulated such a number of unusable cells that the defects cannot be corrected by the error correction codes (ECC) corresponding to data stored in the block

As might be expected, the parties' disputes regarding this claim term are very similar to their disputes regarding the claim term as it appears in the '660 patent, discussed above. Indeed, the cited passages from the '660 patent are relevant to construing the term as used

in the '511 patent because that patent was incorporated by reference into the '511 patent. That said, there are differences worth mentioning. First, with this term, the parties agree that the term involves what is defective at the block level. This is unsurprising because the claim language makes that point quite clearly.

Next, defendants' rationale for imposing an error correction code limitation is weakened by disclosures in the '511 patent. In particular, Figure 10 illustrates a block that has "exceeded its useful lifetime or otherwise has been determined by the controller to be a defective block." '511 pat., col. 15, lns. 7-9. In other words, a block can be defective even if it has not "exceeded its useful lifetime," if the controller so determines. This language shows that in the '511 patent, the term "defective" is not tied to a need to conserve as it was in the '660 patent. Defendants point to some language in the '511 patent aimed at "conservation," but it works more against defendants than for them:

> Information of defects in the memory, such as those discovered during the manufacturing process, may also be stored in separate blocks devoted for this purpose and used by the controller so that the imperfect memory circuit chips may be included in the memory system rather than discarding them. This is particularly an advantage when a single defect record affects many blocks.

Id., col. 3, lns. 43-49. Notably, the '511 patent suggests that special storage of defects is an *option* ("may . . . be stored in separate blocks") and even then is recommended only when wastefulness reaches a certain point: "when a single defect record affects *many* blocks."

Defendants point out that one disclosure from the '511 patent that plaintiff cited in

support of a broader construction for "defective" actually describes an instance consistent with defendants' construction. Plaintiff points out that the specification refers to marking a unit as defective when "there are more than four bad columns in that unit," suggesting this as an example of deeming a block defective independently of the error correction concerns defendants identify. However, the same example in the specification involves a block that has only eight spare bytes, with each bad column using two bytes. Thus, any bad column after the fourth would not be reparable by error correction code. '511 pat., col. 15, lns. 40-47. (Defendants treat the spare bytes as part of the error correction feature of the invention and plaintiff does not challenge it.)

Although the example does not help plaintiff, it also does not help defendants establish that the patent *requires* a block to be uncorrectable before deeming it defective. At most, it is just a single embodiment in line with their construction. Unlike the term "defective" as it is used in the '660 patent, the '511 patent contains no assertion that the "present invention" must be as conservative as possible with memory space. Instead there is a statement that only once multiple blocks of wastage occur would it be "particularly advantageous" to conserve. (Although the '660 patent, incorporated by reference, did contain such "present invention" statements, there is no reason to think statements in a prior art patent about "the present invention" are intended to apply to the invention in which it has been incorporated.) Therefore, I will not read the error correction limitation into the

claim language of the '511 patent, unlike the '660 patent.

At the same time, I am not satisfied with plaintiff's construction. Once again, plaintiff contends that "defective" is broad enough to allow a block to be deemed defective any time a single bit is deemed defective. Although the '511 patent does not emphasize conserving memory to the same degree as the '660 patent, the specification of the '511 patent does explain that there is "particularly an advantage" to avoiding a circumstance in which "a single defect record affects many blocks." Id., col. 3, lns. 43-49. In other words, the patent specification recognizes that a block should not necessarily be deemed defective any time a bit fails because at times it will be advantageous to avoid losing that much memory. Thus, I conclude that for a block to be deemed defective, there must be circumstances aside from a single bit failing that would warrant deeming the block defective. The parties will have to address what those circumstances must be at a later date.

2. "Attach the calculated redundancy codes" ('667 pat., cl. 5) and "adding the generated code to the user data" ('511 pat., cl. 1)

Surrounding Claim Language	**Plaintiff's Proposed Construction**	**Defendants' Proposed Construction**

attach the calculated redundancy codes to the units of user data from which they are calculated['667 – 5] adding the generated code to the user data from which they are generated['511 – 1]	— The calculated redundancy codes are stored together in the same block as the units of user data from which they are calculated. — The redundancy codes need not be immediately adjacent to their respective units of user data.	joining the generated code to the user data from which it is generated

The parties disagree about whether the claimed "redundancy codes" or "generated code" must be stored immediately adjacent to the user data from which it is generated. This dispute is very similar to a dispute the parties had in the consolidated action about the meaning of "appended to the ends of." I concluded that that term required immediate adjacency in the context of the '893 patent in light of the phrase "to the ends of." However, these terms are quite different. Here, the claim language requires only that the codes be "attach[ed] . . . to" or "add[ed] . . . to the user data from which they are generated. '667, cl. 5; '511, cl. 1. The terms "attached to" and "added to" are more akin to "appending to" than "appending to the ends of," and as I pointed out when I construed the latter term, had the term simply been "appending to," that term would not have required immediate adjacency. Case No. 07-cv-605, dkt. #604, at 23 ("Think of a set of exhibits 'appended to'

a brief; not all the exhibits are immediately adjacent, but any of them could be said to be 'appended to' the brief.").

Defendants contend that the terms "attaching to" and "adding to" connote a closer spatial relationship than mere storage in the same block, as plaintiff would have it. In particular, according to defendants, the redundancy code must be "joined" to the user data. In this context, defendants say, joining would amount to immediate adjacency. In addition, the specification of the '511 and '667 patents shows an example of a redundancy code (called ECC) that is immediately adjacent to the user data. '511 pat., Fig. 4.

I am not persuaded that the claim language or specification requires immediate adjacency, but I am also not persuaded that the claim language is so permissive as to treat any storage in the same block as "attaching to" or "adding to" that data, which is what plaintiff requests. Even if the claims must "join" the code with the data, this does not explain what counts as "joining" in the setting of data storage in flash memory. "Adding to" or even "attaching to" data in flash memory is a misnomer: data is not physically dragged through the chip and placed on top of other data, like files of paper. Instead, as the parties have explained, signals are transmitted to cells located in memory, changing those cells to indicate that they are storing data. Thus, data is never "attached" to other data; at most, it is "located" somewhere in relationship to other related data. Both the patent and the parties are silent as to how data is "attached to" or "added to" other data. Should it be placed in

the next location or locations that the controller would look for data? Or should it be placed in the next location or locations in a row or column? Under what rationale? These questions remain unanswered.

The claim language itself suggests a relationship that the parties do not address. In particular, claim 1 of the '511 patent provides that, after the redundancy code is "add[ed] to" the user data, both must be moved "in parallel." It would make sense that "adding to" or "attaching to" would involve a functional relationship between the data more than a spatial one: the data should be "together" for reading or writing purposes, for example. It remains unclear exactly what that means in terms of location: does it go so far as to include all data in the same block, or is there a narrower set of locations within the block that function together in a way that could be considered "joining" data? At this point, I will deny plaintiff's request that the claim be construed to include any storage within the same block, just as I will deny defendants' request to seek the requirement that the storage be "immediately adjacent." The parties may return to this construction question in their motions for summary judgment, but if they do so, they should address the questions I have raised.

3. "Generating a redundancy code" ('511 pat., cls. 1 and 15)

Surrounding Claim Language	Plaintiff's Proposed Construction	Defendants' Proposed Construction
generating a redundancy code from the user data of the individual sectors and adding the generated code to the user data from which they are generated ['511—1] wherein the information of said at least one characteristic of the user data that is stored along with sectors of data includes ***redundancy codes that have been generated*** from the user data while the user data is being transferred to the plurality of the first group of blocks, individual ones of the redundancy codes being added to the user data from which they are generated ['511—15/14]	— The generated user data and redundancy codes are stored together in the same block. — The generated code need not be immediately adjacent the user data from which are generated. — The redundancy code need not be generated from the user data alone.	The redundancy code is (1) generated from the user data alone, and (2) joined to the user data from which it is generated

The parties' disputes include (1) whether the generated redundancy code must be immediately adjacent to the user data; and (2) whether the code must be generated from the user data alone. I have addressed the first dispute above and the parties do not add any new arguments in the context of this claim term. As for the second dispute, plaintiff has two arguments in support of its view that there is no requirement that the redundancy code be generated solely from the user data. Plaintiff contends that the source of the code is not limited to user data because the claim language is silent on whether additional sources may

be used and the claims are method claims that use the transitional phrase "comprising" to introduce the requirements related to the claim terms at issue.

A careful look at what "redundancy codes" are shows that the claims' "silence" on other sources of data does not leave open a possibility that other sources could be included. As defendants explain (and plaintiff do not dispute), a redundancy code (also known as an error correction code) is a number calculated using data that contains properties related to that data such that, if some of the data becomes defective, it can be reconstructed from the remaining good data and the code. Hrg. Tr., dkt. #127, at 62. Defendants explain that an "oversimplified example" of such code is the sum of four numbers. Id. If one of those numbers cannot be read, its value can be determined using the remaining numbers and the sum that had been calculated. Id. Thus, the source of the error correction code is key in shaping the value of the code itself.

Under these circumstances, the claims' description of "redundancy codes that have been generated from the user data" is not simply a description of one source that must be used to generate redundancy code; it is an explanation of what kind of code it is. In this setting, silence about using other sources to generate the code does not support a construction that other sources may also be used.

Plaintiff contends that the claims' use of the term "comprising" means the claim cannot be limited to generating redundancy data from user data alone. Crystal

Semiconductors v. Tritech Microelectronics, 246 F.3d 1336, 1348 (Fed. Cir. 2001) (holding that use of term "comprising" "creates a presumption that the recited elements are only a part of the device, that the claim does not exclude additional, unrecited elements"). Plaintiff's theory misses the mark in the present setting. The rule does not mean steps already present described in the claim can be broadened; it means only that an accused product may perform additional steps *not* claimed. Moleculon Research Corp. v. CBS, Inc., 793 F.2d 1261, 1271 (Fed. Cir. 1986), abrogated on other grounds by Egyptian Goddess, Inc. v. Swisa, Inc., 543 F.3d 665 (Fed. Cir. 2008). In Moleculon, the court of appeals concluded that a "comprising" method claim involving "engaging eight cubes" would not be met by a product that involved a 27-piece cube or a 64-piece cube. The court explained that the presence of the term "comprising" opens the door to additional steps or elements, but does not open up the scope of a particular structure recited within the claim's step. Id.

In this case, the claims' requirement that the redundancy code be generated from user data is analogous to the requirement in Moleculon that a product "engag[e] eight cubes" because, like the "eight cubes," a redundancy code created from user data is a "structural recitation." Id. Thus, for the same reason that "eight cubes and then some" could not be read into the claim language in Moleculon, neither can "user data and possibly something else" be read into the claims at issue. I conclude that defendants are correct that the redundancy code must be generated from the user data alone.

4. "Storing, in individual ones of the second group of said blocks" ('511 pat., cls. 14 and 6 and '667 pat., cl. 1)

Surrounding Claim Language	Plaintiff's Proposed Construction	Defendants' Proposed Construction
Storing, in individual ones of the second group of said blocks, information of physical characteristics of the first group of blocks or their operation ['511—14] The method according to claim 1, additionally comprising storing, in one or more of the distinct memory cell blocks other than the plurality of memory cell blocks, data of information related to physical characteristics of the plurality of memory cell blocks or their operation ['511—6/1] a second group of one or more of the blocks storing records of physical characteristics of thefirst group of blocks or their operation ['667—1]	A block within the first group of blocks may include (a) physical characteristics and/or (b) information concerning its status.	all information related to physical characteristics of the first group of blocks is stored in blocks other than the first group of blocks

For this term, the parties dispute whether *no* physical characteristics of the first group of blocks can be stored in the first group of blocks. (Although the term focuses on storage in the "second group of blocks," both parties seek a construction about storage in the "first

group of blocks.") The language states that physical characteristics of the first group are to be stored in the second group of blocks, but it does not say that *all* physical characteristics must be. Nonetheless, defendants contend that this limitation applies in light of the specifications.[1] As defendants point out, the specifications emphasize the use of separate blocks to store physical block characteristics. The Abstract states that "[o]ne feature is the storage in separate blocks of the characteristics of a large number of blocks of cells in which user data is stored." Later, the specifications explain that "storing a block's overhead data outside of that block" avoids having to rewrite "the overhead data each time the user data is rewritten into the block," reduces access and read times for the overhead data and allows a single redundancy code to be generated for the "large number of overhead records that are stored in this way." '511 pat., col. 3, lns. 22-29.

In addition, the specifications describe this type of storage as an advantage over the prior art, explaining that "[p]rior memory systems . . . have also stored . . . information about the storage media [in the individual user data blocks]" as well as "an ECC generated from the data stored in the block," but "[a]s part of the present invention . . . this type of information about the physical block is stored in another block." Id., col. 14, lns. 6-14. This

[1] The claim terms at issue are found in two separate patents, the '511 patent and the '667 patent. These patents' specifications are identical, so throughout this section, I will cite only the '511 patent specification.

discussion distinguishing the prior art immediately follows a statement that "the overhead information that is stored in a block along with a sector of data is limited to information about the data itself and does not include physical overhead information about the block or its operation." Id., col. 14, lns. 3-6. Further, the specifications explain the scope of the claimed "physical data" stored in the second group of blocks, explaining that "[o]verhead data of the condition, characteristics, status, and the like, of the individual blocks are stored together in other blocks provided in the array for this purpose." Id., col. 3, lns. 9-11.

Defendants' citations seem to support the limitation they request. In particular, the patentee's intent to limit the scope of its invention is apparent from its statement that the only overhead data that may be stored "in a block along with a sector of data [in the first group of blocks] is limited to information about the data itself and does not include physical overhead information about the block or its operation." Plaintiff contends that this discussion appears in the context of describing an embodiment, but that is not correct. The statement appears in the context of a discussion distinguishing the "present invention's" storage of physical characteristics from that of the prior art. Thus, the discussion cannot be treated as one aimed at describing the details of an embodiment, but rather is about the scope of the invention itself.

However, plaintiff points out that the specifications also include an embodiment of a physical characteristic that is stored in the first group of blocks: "flag" overhead

information, which contains a "pre-erase bit." This pre-erase bit is a piece of data "used by the controller to determine whether a block is erased or not." '511 pat., col. 19, lns. 59-67. Defendants contend that this is not truly a physical characteristic of the block "because they simply tell whether . . . there is data there." Hrg. Tr., dkt. #127, at 89. However, as plaintiff explains (and defendants do not deny), the presence of data in any given cell in flash memory is determined by whether a cell "includes electrons on its floating gate." Plt.'s Br., dkt. #102, at 9. Thus, by "telling whether there is data" in a block, a pre-erase bit is doing no more than providing a shorthand answer for whether any of the cells in that block include electrons on their floating gates. Defendants do not explain how such information would not fall squarely within the specification's description of what amounts to "physical characteristics," which is described broadly as "such [o]verhead data of the condition, characteristics, status, and the like, of the individual blocks." '511 pat., col. 3, lns. 9-11.

In short, there is tension between what the specification shows and what it says generally about the storage of physical information. This undermines defendants' position. What it shows is that, despite general statements regarding the storage of physical characteristics, there is no bright-line prohibition on the presence of such data in the first group of blocks. This is still in keeping with the apparent purpose of storage of physical characteristics separately: it avoids having to rewrite "the overhead data each time the user data is rewritten into the block," reduces access and read times for the overhead data and

allows a single redundancy code to be generated for the "large number of overhead records that are stored in this way." '511 pat., col. 3, lns. 22-29. Storage of some physical characteristics data in the same block as user data would not conflict with this purpose, especially if the data is bound to change along with user data. A pre-erase bit is a perfect example.

I conclude that defendants' bright-line rule against any physical characteristics data is not supported by the patent. At the same time, plaintiff's suggestion that any physical characteristics data could be found in the first group of blocks is also wrong because it would conflict with the stated advantages of out-of-block storage of this data generally and with the specifications' repeated statements emphasizing that physical data should generally be stored elsewhere. Therefore, I decline to accept either defendants' too-narrow construction or plaintiff's too-broad one. The exact scope of this claim term will have to wait until another stage of the case, once the parties narrow their dispute.

5. "A record stored in the memory system" ('666 pat., cl. 1)

Surrounding Claim Language	**Plaintiff's Proposed Construction**	**Defendants' Proposed Construction**

a record stored in the memory system that contains non—overlapping ranges of logical addresses of the designated blocks of memory cells within each of the at least two groups, thereby to allow the controller to determine, from a received logical block address, one of the at least two groups in which a corresponding designated block of memory cells is located and the address of the corresponding designated block of memory cells within the determined group ['666—1]	— The "logical addresses" stored within the record individually correspond to a designated block of memory cells — "a record" can refer to one or more records and need not be stored in a single location within the memory system	a single record containing the logical address received from the host for each block designated to store user data . . . and which is sufficient to enable the controller to determine the corresponding physical address from a received logical block address

There are three disputes related to this term: (1) whether the claimed "record" is restricted to a single record; (2) whether the record alone should be able to determine a physical address; and (3) whether the logical address needs to be from the host.

a. "A record" is limited to one record

The first dispute comes down to interpreting the word "a" in claim 1 ("a" record). Case law says "a" generally means "one or more." Defendants' main argument for reading the term otherwise relates to the use of the term "the" in later unasserted claims to refer back to "record." However, plaintiff points to a case that addresses this very issue. In Baldwin

Graphic Systems, Inc. v. Siebert, Inc., 512 F.3d 1338, 1342-43 (Fed. Cir. 2008), the court of appeals held that "the use of a definite article ('said' or 'the') to refer back to an initial indefinite article does not implicate, let alone mandate the singular."

Although the claim language alone does not support defendants' proposed limitation, the specification does. In particular, the patent specification "consistently refers to the record as 'a single record'" when describing the record storing logical addresses. Defs.' Resp. Br., dkt. #119, at 45. For example, the Abstract says that "another feature . . . provides a single system record," including chip capacity and logical address ranges. '666 pat., Abstract. Likewise, the specification describes "storing . . . information [about the memory arrays in the system] as a single record" and "merg[ing] the number of blocks of user data available in each of the memory chips in a way that establishes a continuum of logical block addresses." Id., col. 4, lns. 22-32. Such merger is described as "merging that information [from memory arrays] into a single record." Id. at col. 4, lns. 37-41.

Plaintiff responds by citing other instances in the specification it contends show that the claimed "a record" can sometimes be more than one record. However, in both instances, the specification refers to "individual block overhead records." Id., col. 14, lns. 16-19 ("individual block overhead records . . . are stored in other memory blocks"); col. 16, lns. 27-29 (discussing "one advantage of storing the block overhead data in records.") The claim term at issue does not refer to a "block overhead record," but instead to "a record . . . that

contains non-overlapping ranges of logical addresses of the designated blocks of memory cells. . . ." Even if the logical address record could be considered a type of "block overhead record" (plaintiff does not suggest it could), the specifications' general references to multiple block overhead records does not undermine the specific assertions related to storing all logical block addresses in a chip in a single record.

Plaintiff includes another example, an embodiment in which

> in response to an address from a host system, . . . the controller 11 first calculates a corresponding logical block address. That LBA is then compared with the table (FIG. 14) in reserved block 2 of the first logical memory chip 17 to determine on which of the chips the addressed block lies. The LBA is [sic] of the immediately preceding logical chip is then subtracted from the calculated LBA. The physical block address on that chip is then calculated by the controller reading the designated chip's reserved sector 1 information to shift this differential LBA into a corresponding block of the designated chip that has been designated for user data storage.

'666 pat., col. 17, lns. 53-65. According to plaintiff, this shows the use of two "records," undermining defendants' evidence that the patentee intended only one. However, it does not show the use of two "records" containing ranges of logical addresses. Instead, it shows one such record, "the table . . . in reserved block 2." The later reference to the "chip's reserved sector 1 information" does not show use of a "record" of logical block addresses, but simply shows use of information that is used to "shift this differential LBA into a corresponding block." Plaintiff does not explain how it would be necessary to use additional logical block addresses to shift the differential logical block address. In other words, plaintiff

has failed to identify any mention of multiple records containing logical block addresses.

None of plaintiff's citations to the specification counter the repeated references in the specification of using a single "record" to store logical addresses of the blocks of memory cells. Although isolated examples and statements in a specification are not sufficient reason to limit the scope of a claim, "repeated and definitive remarks" in a specification can warrant the limitation. Computer Docking Station Corp. v. Dell, Inc., 519 F.2d 1366, 1374 (2008). In this instance, defendants' requested limitation is warranted. The multiple references and statements in the specification that logical addresses should be stored in a single record show the patentee's intent to limit the scope of the claimed "a record" to a single record.

b. The record need not be sufficient to determine a physical address

The next dispute is whether the claimed "record" must be "sufficient to determine a corresponding physical address." (Plaintiff does not deny that the record is necessary for determining a physical address, only that it must also be sufficient.) According to defendants, this limitation is warranted in light of the claim's requirement that the record "allow the controller to determine, from a received logical block address, one of the at least two groups in which a corresponding designated block of memory cells is located and the address of the corresponding designated block of memory cells within the determined group." That language does not establish that the record *alone* must determine the physical address.

It requires only that the record "allow" the controller to determine the address from the logical block address it receives from the record. This leaves open whether the controller may receive information elsewhere that is also used to "allow" the controller to determine the physical address.

Aside from the claim language, defendants cite a single discussion in the specification, which explains that

> [t]he controller then first reads the block overhead record in its RAM 29 that corresponds to the first data sector specified by the host while a physical address within the memory array of the user data block is being calculated from its LBA.

'666 pat., col. 16, lns. 11-15. Assuming this reference could be read as describing the scope of the invention as opposed to an embodiment, it still falls short of limiting the claim term for the same reason that the claim language was insufficient: it leaves open whether the controller uses sources other than the logical block address to calculate the physical address. The passage does nothing more than disclose that the logical block address is part of that equation; it does not say that it must be the only part.

Defendants also cite their expert, who cites the same references they rely on here and then asserts without explanation that a person of ordinary skill in the art would understand the claim to require the record to be sufficient to determine a physical address. An expert's *ipse dixit* assertion of the meaning of a term without any explanation for reaching that

position is useless. In sum, defendants' references fail to support requiring that the record be "sufficient" to determine a corresponding physical address, as opposed to being merely necessary.

c. The recited "logical address" need not be an address received from the host

Defendants also contend that the claimed "logical address" must be the address received from the host. As defendants explain, the specification repeatedly uses the term "logical block address" to refer to "memory system address space," and describes the claimed record as storing logical block addresses, not logical block numbers.

Despite these references, however, the limitation defendants seek is not warranted. As plaintiff points out, the preferred embodiment refers to the controller's receiving "an address from the host system" and then calculating "a corresponding logical block address." '666 pat., col. 17, 53-56. Thus, the term "logical block address" may be from the host or generated by the controller. Defendants' proposed limitation will not be imposed.

6. "A controller adapted to [] communicate user data" ('667 pat., cl. 1)

Surrounding Claim Language	**Plaintiff's Proposed Construction**	**Defendants' Proposed Construction**

a controller adapted to (1) communicate user data between the interface and the first group of blocks with the use of those of the records in the controller memory from the second group of blocks that correspond to those of the first group of blocks with which user data are communicated ['667—1]	The controller need not be adapted to read the records of physical characteristics as part of communicating user data between the interface and the first group of blocks.	the records of physical characteristics in the controller memory are read from the controller memory as part of the communicating user data between the interface and the first group of blocks

For this term, the parties disagree about whether the controller must read records from the controller memory as it communicates user data between the interface and the first group of blocks. The claim requires the controller to "use" records in controller memory from the second group, and the parties agree that, for that use to occur, those records must be "read." The dispute is about whether this read must occur at the time of the claimed communication or could occur before then.

Defendants contend that the read must occur at the same time as the communication in light of the claim language, which requires the "use of . . . the records." According to defendants, if the claim were not intended to refer to same-time reading, it would have referred instead to the "use of the data" from those records. However, nothing about the requirement that the controller "communicate . . . with the use of . . . the records" suggests

that the "use" must occur at the same time.

Defendants point to the specification for further support. In particular, the specification states in the "summary of the invention" that

> [w]hen accessing a specific user data block to perform one or all of programming, reading or erasing, the overhead record for that user data block is first read and its information used in accessing the block.

'667 pat., col. 3, lns. 22-26. It also describes an embodiment in which

> [t]he controller 11 may access a number of user data sectors in response to receipt from a host system of a command containing an address . . . The controller then first reads the block overhead record in its RAM 29 that corresponds to the first data sector specified by the host.

Id., col. 16, lns. 1-15.

These descriptions do not quite establish what defendants seek. First, the latter citation appears in the context of describing embodiments of the invention. Nothing about the language suggests that it was intended to describe the scope of the invention as a whole. As for the former citation, this single statement is ambiguous, meaning it is neither "repeated" nor "definitive." Computer Docking Station, 519 F.2d at 1374 ("[T]his court will not countenance the importation of claim limitations from a few specification statements or figures into the claims. . ."). Although the citation appears to require the "read" to occur "[w]hen accessing a specific user data block," it also says the record "is first read" and the information is "used in accessing the block." This language could be read to require only

that read have occurred by the time the "accessing" occurs so that the information can be used "when accessing" the block. In short, neither the claim language nor the specification requires the records in controller memory to be read "as part of the communicating user data between the interface and the first group of blocks."

7. "A controller adapted to [] write data" ('667 pat., cl. 1)

Surrounding Claim Language	**Plaintiff's Proposed Construction**	**Defendants' Proposed Construction**
a controller adapted to . . . (2) write user data received through the interface into more than one of the blocks of memory cells of the first group of blocks in more than one sub—array at a time. ['667—1]	The controller may write user data by directing a power supply on the memory chip to generate programming voltages.	a controller adapted to apply the actual programming voltage to the memory cells of more than one of the blocks, in more than one sub—array at a time

Once again, the parties dispute whether the controller must actually apply voltage to the memory cells when it "write[s] user data . . . into . . . the blocks of memory cells." Defendants contend that this limitation can be drawn from the claim language, which establishes that the controller must be separate from the array of memory cells and must write user data into the blocks of memory cells. This language would require the controller to apply voltage only if the claimed "writ[e]" it must perform were understood as requiring

application of voltage. It is not. As plaintiff points out, the specification describes embodiments in which either the controller or the flash memory chip applied the voltage, despite the fact that the controller would be considered to have "written" the data in each instance.

Defendants point next to the specification. In its description of Figure 9, the specification explains that that figure

> illustrates example contents of a block overhead record for a good user data block. Byte 0 contains flags including an indication that the user data block is a good one. Byte 1 specifies a voltage for erasing the user data block, and byte 2 a programming voltage."

'667 pat., col. 14, lns. 44-48. The specification then explains that "[t]he controller uses this information when erasing or programming, respectively, the subject user data block for which the record 151 contains overhead data." Id., col. 14, lns. 52-55. The problem with this description is that it is, once again, directed at an embodiment. The same is true with Figure 1A, which shows a flash interface as part of the controller. Nothing about these disclosures would support limiting the terms "controller" or "write" as used in the claim term at issue.

Defendants cite U.S. Patent No. 5,602,987 which is incorporated by reference and describes a controller applying programming voltage, but by the same measure, U.S. Patent No. 5,172,338 is also incorporated by reference and that patent shows data writing and reading circuits (for applying programming voltage) on a flash chip. In addition, another

patent incorporated by reference, U.S. Patent No. 5,532,962, shows a controller writing data by providing the data to a memory device which applies the voltage.

The claim language and the specification leave open the question whether the controller or the flash memory device will apply the actual voltage. All the controller needs do is "write" the data, which may involve as little as directing a power supply on the memory chip to generate voltages.

ORDER

IT IS ORDERED that:

1. From United States Patents Nos. 7,397,713 and 7,492,660:

    a. (1) The terms "address register file" ('713 pat., cl. 1) and "register file" ('713 pat., cl. 11) do not mean the same thing; (2) the "address register file" in claim 1 need not contain all the information needed to convert a host logical address into a physical address; and (3) the "register file" in claim 11 need not contain a "defect map"; and

    b. For "defective memory location" ('660 pat., cl. 1) and "defective location" ('660 pat., cl. 15): (1) the claimed "memory location" is not limited to a "block"; and (2) before a memory location is deemed defective, the memory location's cell defect mapping "capacity" must be reached, and in the meantime memory location must "essentially" be remapped cell-by-cell using an "error correction scheme."

2. From United States Patent No. 7,657,702:

    a. "Logical addresses" ('702 pat., cls. 1, 16, 24 and 33): (1) need not refer to a specific "page"; and (2) need not be programmed into the original page or pages;

b. "Page" ('702 pat., cl. 1) need not store an address sufficient to identify the individual page of data;

c. For "sub-array" ('702 pat., cls. 1 and 33): (1) the "sub-arrays" must be simultaneously programmable; and (2) there is no requirement that an entire array be present on a single chip;

d. For "updatable data structure" ('702 pat., cl. 1) and "updatable address information" ('702 pat., cl. 16): (1) the terms "updatable data structure" and "updatable address information" are not synonymous; (2) the claimed "linking" must be more than simply storing a pointer in the physical address that points to the location of the logical address; (3) there is no requirement that the claimed "linking" be at the page level;

e. The "memory controller" ('702 pat., cls. 24 and 33) must be connected directly to the blocks, but it need not apply voltage to the blocks; and

f. The "Update Programming Step," as the parties call it ('702 pat., cls. 1, 16, 24 and 33), must occur without marking pages of data in the first block with an invalid data flag.

3. From United States Patents Nos. 7,532,511; 7,646,666; and 7,646,667:

a. A block may be deemed "defective" ('511 pat., cl. 7) even if it could still be corrected by error correction codes; however, there must be circumstances aside from a single bit failing that would warrant deeming the block defective;

b. For "attach the calculated redundancy codes" ('667 pat., cl. 5) and "adding the generated code to the user data" ('511 pat., cl. 1): defendants' request to impose a limitation that "the calculated redundancy codes" and "the generated code" must be stored "immediately adjacent" to the user data, and plaintiff's request for a construction of "attach to" and "add to" to include any storage within the same block, are DENIED

c. The claimed "redundancy code" ('511 pat., cls. 1 and 15) must be generated from the user data alone;

d. For "storing, in individual ones of the second group of said blocks"

('511 pat., cls. 14 and 6 and '667 pat., cl. 1): some information related to physical characteristics may be stored in the first group of blocks, but not all sorts of physical characteristics data may be stored in the first group of blocks;

e. For "a record stored in the memory system" ('666 pat., cl. 1): (1) "a record" is limited to one record; (2) the record need not be sufficient to determine a physical address; and (3) the recited "logical address" need not be an address received from the host;

f. For "a controller adapted to [] communicate user data" ('667 pat., cl. 1), the records in controller memory need not be read "as part of" communicating user data between the interface and the first group of blocks"; they may otherwise be read beforehand; and

g. For "a controller adapted to [] write data" ('667 pat., cl. 1), the controller need not apply the actual programming voltage to the memory cells.

Entered this 15th day of March, 2011.

BY THE COURT:
/s/
BARBARA B. CRABB
District Judge