

VRVN60-35360

PATENT

**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**

In re application of: G. R. Mohan Rao  
Serial No.: 17/203,385  
Confirmation No.: 6508  
Filed: March 16, 2021  
Group: 2827  
Examiner: Fernando N. HIDALGO  
For: LIFETIME MIXED LEVEL NON-VOLATILE MEMORY SYSTEM

Mail Stop AMENDMENT  
Commissioner for Patents  
P.O. Box 1450  
Alexandria, VA 22313-1450

Dear Sir:

**AMENDMENT AND RESPONSE TO OFFICE ACTION**

This communication is responsive to the Examiner's Office Action dated March 22, 2023. Applicant requests that any previously non-entered Amendments not be entered. Reconsideration is respectfully requested.

**Amendments to the Claims** are reflected in the listing of claims beginning on page 2 of this paper.

**Remarks/Arguments** begin on page 5 of this paper.

AMENDMENT AND RESPONSE  
SN: 17/203,385  
Atty. Dkt. No. VRVN60-35360

Page 1 of 19

### **In the Claims**

This listing of claims will replace all prior versions, and listings, of claims in the application:

1. (Currently Amended) A system for storing data comprising:
  - memory space containing volatile memory space and nonvolatile memory space, wherein the nonvolatile memory space includes both multi-level cell (MLC) space and single level cell (SLC) space;
  - at least one controller to operate memory elements and associated memory space for associated Write access operations to the memory elements and Read access operations from the memory elements;
  - a bank of nonvolatile memory, including:
    - a plurality of MLC memory modules, each including at least one MLC nonvolatile memory element, and
    - at least one SLC memory module including at least one SLC nonvolatile memory element;
  - at least one random access volatile memory element;
  - an FTL flash translation layer, wherein the at least one controller, or FTL, or a combination of both maintain an address table in one or more of the memory elements;
  - the controller having associated controller memory for storing received data therein, the controller controlling access of the MLC and SLC nonvolatile memory elements and the random access volatile memory elements for storage of the received data therein as stored data, the controller, in at least a Write access operation transferring the stored received data from the controller memory to a given one of the MLC nonvolatile memory elements in an associated MLC memory module, operable to store the received data in the given one of the MLC nonvolatile memory elements as stored data and retain such received data in the random access volatile memory as retained data associated with stored data;
  - the controller performing a data integrity test on stored data in the given one of the MLC nonvolatile memory elements in the associated one of the MLC memory modules after the at least a Write access operation performed thereon by reading the stored data to the controller memory and comparing the stored data in the controller memory in the given one of

the MLC nonvolatile memory elements to the retained data that was associated with the stored data in the random access volatile memory by the controller during the Write access operation;

wherein the address table maps logical and physical addresses adaptable to the system, wherein the mapping is performed as necessitated by the system to maximize lifetime, and wherein the mapping maps blocks, pages, or bytes of data in either volatile or nonvolatile, or both, memories; and

wherein a failure of the data integrity test performed on the stored data by the controller results in a remapping of the address space to a different physical range of addresses and transfer of data corresponding to the retained ~~stored~~ data to those remapped physical addresses from those physical addresses determined to have failed the data integrity test ~~to achieve enhanced endurance~~.

2. (Previously Presented) The system of Claim 1, wherein the FTL, flash translation layer, is a software module, or a firmware module containing software updates.

3. (Previously Presented) The system of Claim 1, wherein at least one of the random access volatile memory or the MLC and SLC nonvolatile memory elements are embedded in the at least one controller.

4. (Previously Presented) The system of Claim 1, wherein the MLC and SLC nonvolatile memory elements comprise flash memory.

5. (Cancelled)

6. (Cancelled)

7. (Previously Presented) The system of Claim 1, wherein the random access volatile memory element is dynamic random access memory.

8. (Previously Presented) The system of Claim 1, wherein the random access volatile memory element is static random access memory.

9. (Original) The system of Claim 1, wherein the at least one SLC nonvolatile memory element includes a hard disk drive (HDD).

## **REMARKS**

Applicant has carefully reviewed the Office Action dated March 22, 2023. Applicant has amended Claim 1 to more clearly point out the present inventive concept. Reconsideration and favorable action is respectfully requested.

### **35 USC §112 Rejection**

The Examiner has rejected Claim(s) 1 135 USC §112(b) as being indefinite for failing to particularly point out and distinctly claim the subject matter which the inventor regards as the invention. The Examiner has specifically referred to section three (3) in the current Office Action. However, applicant believes that the Examiner was referring to section four (4), as section three (3) refers to the acceptance of the Terminal Disclaimer. This rejection is respectfully traversed with respect to the claims as currently amended, the withdrawal of which is respectfully requested.

In the Examiner's Response in section 4, the Examiner commented on the fact that the "controller" as set forth in the claim had no purpose in the original claim and the purpose was arguably defined in the middle section of the claim. However, the original claim language for introducing the term "controller" is as follows:

*at least one controller to operate memory elements and associated memory space*

the operation of the "controller" in the middle of the claim previously presented was set forth as follows:

*the controller controlling access of the MLC and SLC nonvolatile memory elements and the random access volatile memory elements for storage of received data therein as stored data, the controller, in at least a Write access operation to a given one of the MLC nonvolatile memory elements in an associated MLC memory module, operable to store the received data in the given one of the MLC nonvolatile memory elements and retain such*

*received data in the random access volatile memory as retained data associated with stored data;*

*the controller performing a data integrity test on stored data in the given one of the MLC nonvolatile memory elements in the associated one of the MLC memory modules after at least a Write access operation performed thereon by comparing the stored data in the given one of the MLC nonvolatile memory elements to the retained data that was associated with the stored data in the random access volatile memory by the controller during the Write access operation;*

Thus, the “controller” is first defined as “to operate memory elements and associated memory space” and then further functioning for “controlling access of the nonvolatile memory elements and the random access volatile memory elements for storage of received data therein.” For clarity, Claim 1 has been amended as follows:

*at least one controller to operate memory elements and associated memory space  
for associated Write access operations to the memory elements and Read access operations  
from the memory elements;*

This clarification is believed to overcome the Examiner’s objection to the structure of the claim.

The Examiner had set forth in section eight (8) the following with respect to the 35 USC §112 rejection:

8. Claim(s) 1 is rejected under 35 U.S.C. 112(b) or 35 U.S.C. 112 (pre-AIA), second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which the inventor or a joint inventor (or for applications subject to pre-AIA 35 U.S.C. 112, the applicant), regards as the invention. **For analysis and reasons for this claim objection, section three (3) above is referenced herein fully and completely and made integral part of this claim rejection.**

The Examiner has further made the comment in section four (4):

Moreover, setting aside the issue of "the controller" in the middle of the claim limitation, the claim limitation, in brief, seems to claim that the controller controls access of nonvolatile and volatile memory elements for storage of received data therein as stored data (note: only data that has been received by memory generally can be stored; if data that has not been received by memory generally were possible to be stored in such memory, the state of the art would be quite different, indeed; and it is argued that such scheme is not one the present Application includes). That is, both nonvolatile and volatile memory elements store received data therein as stored data. And that in a write access operation (note: storing data in memory generally is effectuated in write operations; for example, a read access operation does not effectuate the storing of data in memory), nonvolatile memory elements store the received data and that the received data is retained in the volatile memory elements. That is, both the volatile and nonvolatile memory elements store received data.

In general, Applicant believes the Examiner is considering that the language of storing "received data" as "stored data" makes the claim indefinite. However, Applicant argues that, when data is received by a memory, what is actually stored in a particular memory element may not be the actual received data. In some memories, such as volatile memories, i.e., the DRAMs, error correction bits can be added before storage of any data bit therein, such that any errors or faults in the memory elements, i.e., a bad memory element resulting in an incorrectly stored bit, can be tolerated. Subsequent reading out of this stored data results in corrected data that accurately matches the received data, the point being that the stored data may not actually completely match the data received by the memory. In a similar manner, it is recognized that FLASH nonvolatile memory has an issue as to wearability, wherein certain portions of the memory will wear out as a result of multiple Writes, resulting in bad memory elements. Thus, error correction bits can be added to account for faulty memory elements to tolerate a certain number of bad memory elements or, as is the purpose for the present invention set forth in the current application, data can just be written to another area of the memory and that faulty portion of the memory marked as a bad section of the physical memory. In any event, it should be recognized that the "stored data" may not actually be the exact "received data" due to possible faulty memory elements. For example, received data with a value of "10001010" could be first written to the memory and then read out as a value of "100X1010" with the fifth bit being in error due to a faulty memory element at that location. Thus, what is stored is not actually the received data. The language of the claim prior to the current amendments:

*the controller performing a data integrity test on stored data in the given one of the MLC nonvolatile memory elements in the associated one of the MLC memory modules after at*

*least a Write access operation performed thereon by comparing the stored data in the given one of the MLC nonvolatile memory elements to the retained data that was associated with the stored data in the random access volatile memory by the controller during the Write access operation;*

sets forth that the controller compares the “stored data” with the “retained data” during a Write access operation. If the integrity check fails, then a remapping operation ensues. If the “stored data” was in fact the “retained data,” which “retained data” is the “received data,” the integrity check would never fail. Thus, Applicant believes that there is no indefiniteness with respect to defining “stored data” as being different from “received data.” Applicant respectfully requests withdrawal of the 35 USC §112 rejection respect to Claim 1 on this basis.

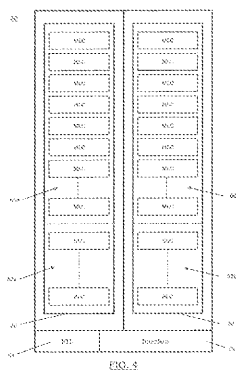
### **35 USC §103 Rejection**

Claims 1 – 4 and 7 – 9 stand rejected under 35 USC §103(a) over *Gorobets* in view of *Goodson* and further in view of *Karamcheti* and/or *Yu* and further in view of *Kawaguchi* and further in view of *Dusija*. This rejection is respectfully traversed in view of the claims as currently amended.

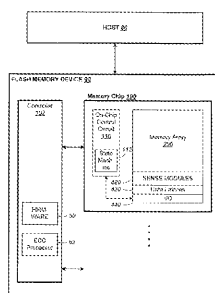
The Examiner has noted the following with respect to the arguments in section four (4) as follows:

a bank of nonvolatile memory, including: a plurality of MLC memory modules, each including at least one MLC nonvolatile memory element[[:]], and at least one SLC memory module including at least one SLC nonvolatile memory element (As found in at least FIG. 15, at least paragraphs [0090], [0135] in the teachings of *Gorobets*: the memory 200 is organized into blocks; and that 200 is partitioned into MLC modules an LSC memory modules (SLC is described as binary mode cells; also refer to [0108]).

Applicant believes that this may be an incorrect interpretation of the claim. The claim requires that there be a plurality of separate MLC memory modules and at least one SLC memory module in the bank of nonvolatile memory. This relates to Figure 4 of current application:



*Gorobets* at ¶[0090] states that “[t]he memory 200 is organized into blocks, each block of cells being a minimum unit of erase.” Thus, a block can only be either an MLC memory block or SLC memory block. Since this is a minimum unit of erase, a block can only exist in one of the memories and calm not include both SLC and MLC memory elements. Applicant believes that *Dusija* is a better reference for this aspect in Figure 1 as follows:



### Karamcheti and Yu References

The Examiner’s next argument relates to the retained data aspect of the claim in volatile memory. The Examiner stated as follows in section four (4) as follows:

the controller controlling access of the MLC and SLC nonvolatile memory elements and the random access volatile memory elements for storage of data therein, the controller, in at least a Write access operation to a given one of the MLC nonvolatile memory elements in an associated MLC memory module, operable to store data in the given one of the MLC nonvolatile memory elements and retain such stored data in the random access volatile memory (While the teachings of Gorobets as modified may not expressly teach the claimed limitation above, US Patent No. 8516172 to Karamcheti et al. ("Karamcheti") and/or US Publication No. 20100318879 to Yu ("Yu"), in relevant and complementary manner at the very least obviate such claim limitation; Karamcheti in at least FIG. 7 and at least Column 3, lines 28-41 ; and Yu in at least paragraph [0057] and at least FIGS. 2, 4-6).

The Examiner is noted that *Gorobets* does not expressly teach this limitation. This rejection was specifically responded to and the Response filed 3/9/23. The following was set forth by Applicant in that Response with respect to *Karamcheti* and *Yu* on pages 9 and 10:

The determination of whether a Write operation has been successful is first determined and then, if failed, a rewrite is performed from the backup register (516). The information in the backup register or buffer is only used for the rewrite operation and is not used for any type of compare operation. There is no suggestion or disclosure in *Karamcheti* that any data in the NVMD is pulled back up into the controller and utilized for a data integrity check. Rather, the determination of a failure is made at the NVMD. Thus, all that *Karamcheti* discloses is buffering data in a RAM for the purpose of writing to a Flash. This data is not retained in the RAM after writing to the Flash but, rather, it is only buffered or retained in the RAM for the purpose of rewriting when it is determined that data cannot be written to the Flash at that particular sector. Thus, there is no "retained data" that is associated with stored data because, once data is reliably stored, there is no use for the data anymore.

...

Thus, *Yu* does not "retain" the written data from the host in the DRAM but, rather, first stores it in the FLASH memory, then reads it from the FLASH memory, and then stores it in the DRAM. There is no compare operation of data from the FLASH operation with memory from the DRAM but, rather, only a comparison of data from the DRAM with data received from the host.

Applicant had made the argument that the combination of these two references *Karamcheti* and *Yu* does not disclose the limitation of the "retained data" in the volatile memory as a part of the Write access operation. Claim 1 had been amended as follows with respect to the "retained data":

the controller having associated controller memory for storing received data therein, the controller controlling access of the MLC and SLC nonvolatile memory elements and the random access volatile memory elements for storage of the received data therein as stored

*data, the controller, in at least a Write access operation transferring the stored received data from the controller memory to a given one of the MLC nonvolatile memory elements in an associated MLC memory module, operable to store the received data in the given one of the MLC nonvolatile memory elements as stored data and retain such received data in the random access volatile memory as retained data associated with stored data;*

These amendments are supported by the following language in the Specification as filed:

FIGS. 3A and 3B are a flow chart illustrating a method for utilizing a NAND flash memory system incorporating one embodiment of the present disclosure. The method begins in a step 100, when a command to write a quantum of data stored in DRAM to a particular location in NAND flash memory is received, in step 102, the quantum of data is read from DRAM into memory within the device controller (which acts as the memory controller). In step 104, both the logical address range and the NAND flash physical address range to which the quantum of data is to be written, is read into memory of the device controller. In step 106, the quantum of data to be written is combined with the contents of the NAND flash memory. In step 108, the NAND flash physical address range to be written is erased. In step 110, the combined data is written to the appropriate NAND flash physical address range. In step 112 the NAND flash physical address range that was written in step 110 is read into device controller memory.

In accordance with the Specification, there is provided a DRAM for storage of received data therein, but a Write access operation also loads the received data to be stored “into memory of the device controller.” The contents of the controller memory are then written to the FLASH. This is then followed by a step wherein the stored data is read back into the controller memory. The Specification then sets forth the following:

The flowchart continues in FIG. 3B. In step 114 the NAND flash physical address range that was read into device controller memory is compared with the retained data representing the combination of the previous contents of the physical address range and the quantum of data to be written. ...

Thus, there is required at least a controller memory that is used in the Write access operation and a copy of the data from the DRAM, i.e., two data stores.

*Karamcheti* is a reference that operates during a Write Access to Receive Data - Buffer in RAM - Write to FLASH – Verify - Rewrite on Verify. The Verify operation does not involve any compare operation and, if fact, there is no compare operation. The data is maintained in the buffer,

but only for the purpose of a Write operation to the FLASH. Even though it may be held there for multiple Writes, the data is only designated for a single Write. There is no suggestion this buffer could be used for a data store in a compare operation as the data is *discarded* after a Write is complete.

*Yu* is an operation that has the primary purpose of reducing the number of data programming operations required for multiple Writes ([0053]):

[0053] As described above with reference to the storage device 100 and the data storing method 200 according certain embodiments of the inventive concept, the write data received from the host is filtered in order to avoid performing a programming operation for the write data when it is already similarly stored in the flash memory 160. Using this write data filtering approach, the number of data programming operations directed to the flash memory 160 may be reduced, and the write endurance of the flash memory 160 may be extended. As described above, if the write endurance of the flash memory 160 is improved, the reliability of the storage device 100 including the flash memory 160 and the lifespan of the storage device 100 may be increased.

During a Write access operation, volatile memory such as RAM is utilized to determine whether data already exists in Flash. This is thus an operation of Receive Data – Verify Data not already stored in FLASH – Write to FLASH if not already stored. There is no operation to determine if data written to FLASH memory is corrupt by comparing to retained data and then rewriting the data from the retained data to a different location in FLASH. The purpose of *Yu* is completely different than the purpose set forth in the current application which is related to first writing data to a FLASH location than, if failed, rewriting that data from a retained data location in volatile memory to another location in FLASH. It is clear that *Yu* does not disclose or suggest a Receive – Retain (in volatile memory) – Write (from controller memory) – Read (to controller memory) – Compare (data in controller memory to retained data in volatile memory) – Rewrite operation. Rather, *Yu* actually teaches against performing a Write operation before any comparison operation. The specific reasoning in *Yu* is to actually reduce the number of data programming operations.

Kawaguchi Reference

The Examiner with respect to the next element of the Claim 1 presented the following argument on Pages 9 - 10 of the current Office Action:

the controller performing a data integrity test on stored data in the given one of the MLC nonvolatile memory elements in the associated one of the MLC memory modules after at least a Write access operation performed thereon by comparing the stored data to the retained data in the random access volatile memory (While the teachings of Gorobets as modified may not expressly teach the claimed limitation above, obviating such claim limitation, CN 101739312 A to Kawaguchi ("Kawaguchi") provides in at least claims 1 and 5: backup data temporarily stored in the volatile memory, keeps the erased area as the backup data in the flash memory, the erased area has no data, and the backup data is written in the erased area ... backing up the backup data newly stored in the said non-volatile memory with recently written in the destination address data comparing the newly storing backup data than the most recently written backup data updating, if the backup data of the most recently written and the newly storing backup data is inconsistent, then the newly storing backup data from the erased area start address is written in the erased area; and the updating destination address and the start address of the erased area).

Claim 1 has been amended with respect to this limitation as follows:

*the controller performing a data integrity test on stored data in the given one of the MLC nonvolatile memory elements in the associated one of the MLC memory modules after the at least a Write access operation performed thereon by reading the stored data to the controller memory and comparing the stored data in the controller memory in the given one of the MLC nonvolatile memory elements to the retained data that was associated with the stored data in the random access volatile memory by the controller during the Write access operation;*

In this Amendment, the data is first read to the “controller memory” and then this read data stored in the controller memory is compared to the retained data. This thus requires two data stores. This Amendment is supported by the following portion of the Specification:

The flowchart continues in FIG. 3B. In step 114 the NAND flash physical address range that was read into device controller memory is compared with the retained data representing the combination of the previous contents of the physical address range and the quantum of data to be written. In step 116 if the retained data matches the newly stored data in the NAND flash memory, the write was a success, and the method exits in step 118. However, if the retained data does not match the newly stored data in the NAND flash memory, the method executes step 120, which identifies the next quantum of available SLC NAND flash memory addresses.

*Kawaguchi* is a new reference and the Examiner has only provided the translation of the claims in support of the rejection with respect to such language. Applicant has had the full Patent translated and provided such as part of the record.

The problem that *Kawaguchi* is faced with is dealing with increased amounts of backup data when monitoring a system such as an automobile, wherein a solution to this increased amount of backup data is to utilize a Flash memory, which would be more desirable than what was used before, EEPROM memory. ¶[0008]. Flash memory is desirable from the standpoint that it is faster to write to as compared to EEPROM. The problem though, is that the number of times it can be written to is less than EEPROM. ¶[0009]. One issue with Flash memory that is noted is that the erasing and writing of data to a particular location can take a long time. The solution is to temporarily store the data in the volatile memory, i.e., RAM, reserve erased areas in the Flash memory for backup data and then write the backup data into a different erased area for each backup operation. ¶[0011].

The overall backup operation includes a comparison step, which is what the Examiner has referred to with respect to the current Rejection. Current Claim 1 requires that *after a Write access operation of the received data*, and as part of a data integrity check, a Read operation is performed on that written and stored data followed by a compare operation with the retained data stored in volatile memory. If the compare operation is True, then the data integrity check has passed. If False, then the retained data in a volatile memory is then rewritten to another physical location after remapping, it being noted that the same logical address applies to the Write access operation after remapping. Thus, the operation is to – 1) Receive data, 2) Write the received data into volatile memory as retained data and into the Flash as stored data, 3) Read the stored data from Flash and transfer to the controller for comparison with the stored retained data in the volatile memory, 4) If True, do nothing and, if False, remap the logical address to a different physical address, and 5) rewrite the retained data from the volatile memory through the controller to the new physical address, but at the same logical address.

In *Kawaguchi*, the primary purpose of the overall system is to store backup data in nonvolatile memory. Backup data is continually collected and temporarily stored in RAM - volatile memory. This is a periodic operation. This all occurs based upon some type of predetermined

condition as a trigger. ¶[0046]. The backup operation is described with respect to Figure 7 and ¶[0062]. From the Specification, the operation is to backup the collected data in the RAM to the Flash in an area that has been erased, i.e., a free (erased) area. In this manner, it is faster to just Write to a free area rather than perform an erase followed by a Write operation. This just requires a large number of erased sectors which can be sequentially written-two on subsequent Writes. Of course, once the Flash has filled up, is a necessary to erase the contents. It is noted that only one sector will actually have any viable data that is useful, as all of the old sectors that are not erased or free have old backup data stored therein. It is more efficient to just do a bulk erase at some periodic interval for a portion of the Flash when the Flash is determined to be reaching its limit.

A destination address of a designated sector in the Flash is maintained such that, when it is determined that data is to be transferred from the volatile memory to the Flash, i.e., a trigger, that destination address is initially utilized to access the Flash for a Read operation of the previously stored backup information. This particular destination address is the last address to which data was written to the Flash. The reason for this is that it is desirable to determine if the last Write of information resulted in information stored therein that has not changed in the RAM or volatile memory at the current time, i.e., no additional backup information has been collected. Thus, during a backup operation, the first thing to do is check the Flash at the last written-to location by reading the data therefrom and comparing it to the current data in the volatile memory. If the data has changed, then the destination address is changed with a different logical address associated with the different sector and then a Write operation performed to transfer the data from the volatile memory to Flash at that new logical address. Thus, the overall operation is to – 1) Receive data, 2) Store the received data in volatile memory, 3) Read stored data from the Flash at a first destination address which was the last address associated with a Write operation to the Flash, 4) Compare that read data with the data stored in the volatile memory, 5) If False, change the destination address and Write the data from the volatile memory to the Flash at this new destination address. Thus, the operation is a Receive – Store in RAM – Read FLASH – Compare – Write to FLASH.

The primary purpose of *Kawaguchi* is to improve the backup operation in a system for collecting sensor data in a system, such as an automobile. The existing operation involved backing

of the data in non-volatile memory such as a EEPROM. All this required prior to using Flash was buffering information and then just performing a Write operation to the EEPROM. This only required transferring of data to a logical address within the memory space of the system, wherein new data would write over old data, noting that it was not necessary to change the logical address. *Kawaguchi* sought to improve the operation of the prior system by incorporating Flash as the non-volatile memory. This provided some advantages, as noted above, in that it could arguably have faster Writes, be less expensive, etc. The main problem with Flash is that one cannot Write the same logical address every time because that would require writing over previous data. As the previous data is no longer viable, there is no reason to retain such data and writing over of such data would be a normal thing when storing data in the form of backup data, allowing one to keep the same logical address. As noted, a Write access operation with respect to an EEPROM only needs a Write command to a particular logical address in order to write over old and no longer useful backup data with the new backup data. A Flash, on the other hand, if only a single logical address were utilized, would require an erase of the data followed by a Write to the erased location. This was noted as a problem, which made it less desirable than an EEPROM. The solution in *Kawaguchi* was to just provide a large number of writeable (erased) sectors in the Flash and control a changeable destination address to store data in only different erased sectors for each Write operation. Of course, periodically, the Flash memory would have to be erased to create more free and erased sectors. However, there was one more improvement that *Kawaguchi* provided for and that was for first examining the previously written data before transfer of any data and comparing it to the current data in the volatile memory to determine if a Write to the Flash was necessary. This is best stated in ¶[0072] as follows:

In other words, in this embodiment, the flash memory 1h is not rewritten by a simple rewriting process including data erasing and data writing.

In contrast, an erasing process requiring a long processing time is performed only when the backup data is not updated(rewritten), so that an erased area of a predetermined size is always left.

Therefore, only a non-time-consuming writing process is performed when the backup data is actually updated.

**Therefore, the time required to rewrite the backup data stored in the flash memory 1h can be reduced. (Emphasis Added)**

This process utilizing the comparator in the backup process does not relate to any Write access operation; rather, all it does is determine whether to initiate a Write access operation. The above-noted claim limitation specifically requires the compare operation to occur after a Write access operation. *Kawaguchi* utilizes this compare operation to determine whether to transfer or Write data to Flash. It is not part of the Write operation of the device with the Flash. Further, the comparison of the previously captured backup data is made with different data than what was stored in the previous Write operation to the Flash. By definition, if the comparison results in a false comparison, that means the data in the volatile memory is different than the data which was previously stored in the volatile memory and which was transferred to the Flash in the previous Write operation. As such, Applicant believes that *Kawaguchi* does not disclose or suggest any type of compare operation of “retained data” after a Write access operation. *Kawaguchi* does nothing more than disclose the operation of comparing data in one memory before writing to a second memory to reduce the number of data transfer operations if the first memory has not had the contents thereof changed between the two sequential Write operations.

Applicant believes that none of the references to *Gorobets*, *Goodson*, *Karamcheti* and/or *Yu*, *Kawaguchi* or *Dusija*, taken singly or in combination, obviate or anticipate the invention as defined by the amended claims. No combination of these claims describes or suggests the operation of performing a Write operation of non-volatile memory by utilizing a volatile memory for storage of received data as retained data, storing received data in nonvolatile memory at a first location in accordance with a first logical address and then reading this stored information back to the controller in the controller memory and comparing it with the retained data in the volatile memory at the controller level. If false, the physical address is remapped to the same logical address and a rewrite performed. None of the references discloses or suggests the use of the two memories at the controller level for the overall operation and doing a comparison at the controller level with the data stored in these two memories. Applicant therefore respect request withdrawal of the 35 USC §103 rejections with respect to Claims 1 – 4 and 7 – 9.

Applicant has now made an earnest attempt in order to place this case in condition for allowance. For the reasons stated above, Applicant respectfully requests full allowance of the claims as amended.

If any issues arise, or if the Examiner has any suggestions for expediting allowance of this Application, the Applicant respectfully invites the Examiner to contact the undersigned at the telephone number indicated below or at [ghowison@munckwilson.com](mailto:ghowison@munckwilson.com).

Recognizing that Internet communications are not secure, Applicant hereby authorizes the USPTO to communicate with the undersigned and practitioners in accordance with 37 CFR 1.33 and 37 CFR 1.34 concerning any subject matter of this application by video conferencing, instant messaging, or electronic mail. Applicant understands that a copy of these communications will be made of record in the application file.

Please charge any additional fees or deficiencies in fees or credit any overpayment to  
Deposit Account No. 50-0208/VRVN60-35360.

Respectfully submitted,  
MUNCK WILSON MANDALA, LLP  
Attorneys for Applicant

/Gregory M. Howison, Reg. #30646/

Gregory M. Howison  
Registration No. 30646

GMH/jsr  
P.O. Drawer 800889  
Dallas, Texas 75380  
Tel: 972-628-3600  
Fax: 972-628-3616  
September 22, 2023

AMENDMENT AND RESPONSE  
SN: 17/203,385  
Atty. Dkt. No. VRVN60-35360

Page 19 of 19