IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received November 16, 2019, accepted December 1, 2019, date of publication December 10, 2019,
date of current version December 23, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2958628

Schemes for Privacy Data Destruction in a NAND Flash Memory

NA-YOUNG AHN[1] AND DONG HOON LEE[2]
[1]The Graduate School of Information Security, Korea University, Seoul 02841, South Korea
[2]CIST and The Graduate School of Information Security, Korea University, Seoul 02841, South Korea

Corresponding author: Dong Hoon Lee (donghlee@korea.ac.kr)

This work was supported in part by the Military Crypto Research Center through the Defense Acquisition Program Administration (DAPA) and Agency for Defense Development (ADD) under Grant UD170109ED.

ABSTRACT We propose schemes for efficiently destroying privacy data in a NAND flash memory. Generally, even if privacy data is erased from NAND flash memories, there is a high probability that the data will remain in an invalid block. This is a management problem arising from characteristics of a program operation and an erase operation of NAND flash memories. When updating pages or performing a garbage collection, there is a problem that valid data remains in at least one unmapped memory block. Is it possible to impose an obligation to delete privacy data from an existing NAND flash memory? This paper is the answer to this question. We propose a partial overwriting scheme, an SLC programming scheme, and a deletion duty pulse application scheme for invalid pages to effectively address privacy data destruction issues caused by the remaining data. Such privacy data destruction schemes basically utilize at least one state in which data can be written to programmed cells based on a multi-level cell program operation. Our privacy data destruction schemes have advantages in terms of block management as compared with conventional erasing schemes, and are significantly economical in terms of time and cost. The proposed privacy data destruction schemes may be easily applied to many storage devices and data centers using a NAND flash memory.

INDEX TERMS NAND flash memory, privacy data, destruction, multi-level cell programming, partial overwriting, SLC programming, deletion duty pulse, garbage collection, data center.

I. INTRODUCTION

Nonvolatile memories are divided into an overwritable memory and a non-overwritable memory. NAND flash memory is a non-overwritable memory [1], [2]. NAND flash memory basically adjusts a threshold voltage by moving trapped charges according to an applied voltage. Data corresponding to the adjusted threshold voltage is determined. Also, in terms of management, NAND flash memory has an issue on residual data related to deletion because a writing unit and a erasing unit are different from each other. In general, valid data is collected into a single block through a garbage collection, and remaining invalid data is erased in units of blocks when internally reaching a certain standard [3]–[5]. However, due to the characteristics of NAND flash memory, NAND flash memory may easily obtain meaningful information through digital forensics even if a user deletes the data [6], [7]. If such meaningful information is obtained by investigation agencies such as police/prosecutors, it is the best case. But what about the opposite case? Since digital forensics is easy, anyone can illegally obtain user-related information through digital forensics. If the user-related information is privacy data, it is obviously problematic.

Existing NAND flash memories cannot be overwritten. Therefore, even if the user deletes privacy data stored in NAND flash memory, there is a considerable probability that raw data is still left. Recently, data centers are employing SSDs using NAND flash memories, which is likely to leave unprotected privacy data on SSDs in the data center [8], [9]. Naturally, the duty to delete privacy data is also imposed to NAND flash memory. It should not be considered to be technically impossible or free from these obligations. In fact, technically speaking, even if you don't change the structural parts of existing NAND flash memory, you don't have to impose a lot of obligations to delete them when you use partial program operations. Ahn and Lee have already begun discussions to address these issues [9], and in this paper we conduct a deep study on these issues. We introduced the

The associate editor coordinating the review of this manuscript and approving it for publication was Wei Zhang.

destruction of privacy data (personal information) using an overwritable program operation rather than an erase operation. In 2018, Lin et al. presented a simple method of sanitizing MLC data by one shot programming [10]. Lin's one shot programming scheme, however, is problematic in varying the storage capacity of NAND flash memory. Since Lin's scheme changes from the number of MLC pages to SLC pages, the number of pages in a single memory block suddenly decreases. This refers to a change in the magnitude of the storage capacity. Thus, this method is unlikely to be used from a management point of view.

The duty to delete privacy data from NAND flash memories should be imposed in the near future. There are some limitations to this in existing technology, but we are challenging these limitations using algorithms that impose a simple deletion duty.

The structure of our paper is as follows. In Section II, we explored why privacy data inevitably remains in NAND flash memories and how we can handle the remaining privacy data. In Section III, we confirmed that there are overwritable states in NAND flash memory and that these states can be overwritten in NAND flash memory. In section IV, we proposed a scheme of destroying privacy data using partial overwriting and a scheme of destroying privacy data using a deletion duty pulse. In section V, we compared the performances with respect to processing time and cell degradation characteristics between the conventional block deletion schemes and the proposed deletion schemes.

The proposed data destruction schemes may be immediately applied to existing NAND flash memory and allow simple and quick destruction of privacy data without affecting the cell characteristics. In the future, our data destruction schemes are expected to make a significant contribution to NAND flash memory, which must be subject to the duty to delete.

II. RELATED WORKS FOR NAND FLASH MEMORY

In general, a privacy data life cycle includes collection, storage, storage, use/provision, and destruction. Privacy data destruction is the final stage of the privacy data life cycle. As rights, as the subject of information, become more prominent, the importance of privacy data destruction is increasingly emphasized. Recently, European Union (EU) have legally been obliged to destroy privacy data in terms of enhancement of rights as an individual's information subject [8]. NIST has suggested guidelines for media sanitization. Media sanitization is largely divided into Clear/Purge/Destroy [11].

'Clear' refers to a technique for overwriting non-sensitive data using software or hardware. This overwrites the new value on the storage device using a normal read or write command. In the case of a device having no overwriting function, such as a NAND flash memory, 'clear' means that it is set to a factory state [11]. 'Purge' refers to a physical or logical technique using lab schemes to make target data unrecoverable [11]. 'Destroy' refers to the use of laboratory technique to render target data unrecoverable while at the same time not functioning as a medium for storing the data.

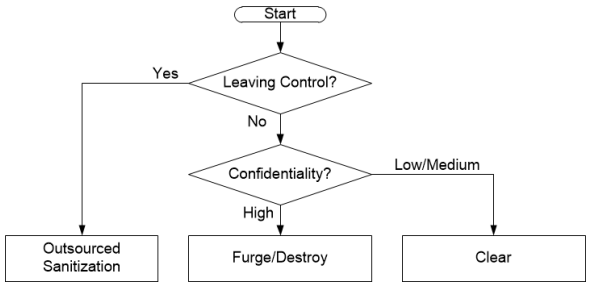

FIGURE 1. Media sanitization process is shown.

Conventionally, there is a guideline for destroying privacy data in NAND flash memory. As shown to FIG. 1, according to NIST standard, there is a description of flash memory according to Clear/Purge. As mentioned in the description of Clear in NIST standard, a flash memory-based storage device may not be able to completely delete old data even if it includes spare cells and performs wear leveling. This is because the storage device does not specify direct addressing of all areas in which important data is stored. The clear operation is to use software/hardware products to overwrite addressable space with non-sensitive data according to conventional read/write commands.

NIST standard also mentions the destruction of privacy data called 'Purge'. The 'Purge' operation includes overwriting, block deletion, and encryption deletion. However, since NAND flash memories do not normally support overwriting and is likely to store privacy data in a non-addressable space, it is difficult to accept that the NIST standard has appropriately performed privacy data destruction. Privacy data, unmapped addresses, still remains in NAND flash memories.

A. GARBAGE COLLECTION

In the case of a typical flash memory, if a free block in the user block area is insufficient due to a bad block or the like, a free block in the spare block area is allocated as a user block area [12]. In this case, since the allocated size is block units (or two or more blocks), a space larger than the required storage space is allocated. Also, the free block in the spare block area becomes short in a relatively short time, and the flash conversion layer performs a background operation such as garbage collection to secure free blocks. Garbage collection is a process of selecting a specific block in a memory array, copying a valid page of a specific block to a free block, and then erasing the block to be a free block, referring to FIG.2.

The erased block may be used to record the data later. However, according to the internal policy, there is a problem that the erase operation for such a block is not performed rapidly. In this case, if the valid page includes privacy data, the invalid page may include the same privacy data. Privacy data is included in both the valid page in the free block and the invalid page in the block to be deleted. As a result, the privacy

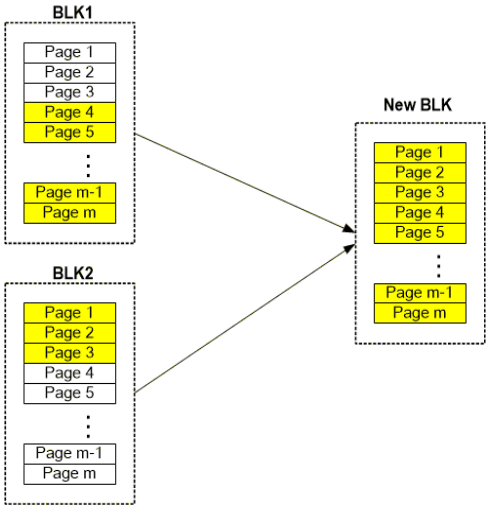

FIGURE 2. Garbage collection for forming a free block is shown. A garbage collection for collecting valid pages of the first block BLK1 and the second block BLK2 to form a new block is shown.

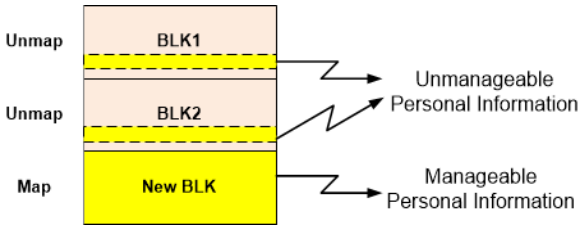

FIGURE 3. Unmanageable privacy data is shown. Unmanageable privacy data remaining in the invalid blocks BLK1 and BLK2 after garbage collection is shown.

data is likely to be included in at least two blocks, only one of which is mapped and the others will remain in the unmapped state. As shown in FIG. 3, if conventional garbage collection is performed, privacy data that cannot be managed exists in at least two blocks. The unmapped first and second blocks BLK1 and BLK2 retain their privacy data until the erase operation is performed.

Our concern is how to destroy the privacy data included in the invalid page in the block to be deleted. One of the simplest ways to destroy privacy data is to immediately erase blocks including invalid pages including privacy data. However, since the erase operation is performed in units of blocks, it is difficult to manage the storage device in terms of policy. In general, a single block includes 1024 pages. Performing an erase operation on a single block due to an invalid page storing privacy data among 1024 pages is significant in terms of management cost.

Existing NAND flash technology does not support overwriting. A reliable method of data invalidation is to perform an erase operation on BLK1 and BLK2. However, as well known in the art, the erase operation is one of the management operations to be avoided as much as possible because the number of erase cycles for the memory block is limited in terms of wear leveling. Furthermore, the erase operation consumes a large amount of power, and the erase time is

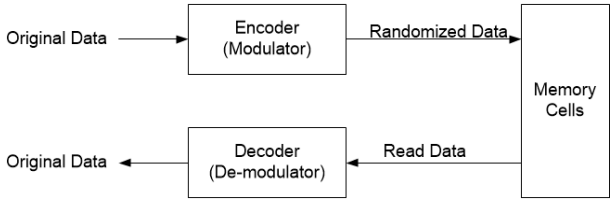

FIGURE 4. Data modulation of NAND flash memory is shown.

also considerable. From the vendor's point of view, the focus is on keeping these erase operations from being performed as soon as possible.

Our research has begun to meet the needs of users who worry about these vendors and want to remove complete privacy data from flash memory. Ahn and Lee attempted to address such an issue by overwriting them in their studies [9]. To summarize their argument: The conventional method of writing NAND flash memory does not support overwriting, but since it is accessed from the viewpoint of privacy data destruction, it is intended to overwrite invalid pages with arbitrary data. Of course, any data should be writeable data to the currently stored invalid page. In general, since the time required for a single page write (program) operation and erase operation for a single block is 1000 hours or more, this method is significantly effective as a privacy data destruction method.

B. DATA MODULATION

In general, data is not stored in the form of original data in NAND flash memory. In order to minimize the possibility of data modification due to interference between data to be programmed, the original data is randomized according to predetermined rules and the randomized data is stored in a physical area [13]. As shown in FIG. 4, an encoder for modulating the data and a decoder for demodulating the data read in the physical area are essentially included.

The program operation may be performed to program the modulated data into the memory cells after modulating the error null data. The read operation may be performed to output the original data after de-modulating the data read from the memory cells. The binary bits corresponding to the original data, that is, the privacy data, are not strictly stored in the memory cells. However, since the modulated data is stored in the memory cells, and the stored modulated data and the original data are mutually exchangeable by a predetermined method, the modulated data is also regarded as privacy data.

C. STATE MAPPING

The data modulated data is mapped depending on a program state corresponding to the threshold voltage Vt of the memory cell.

As shown in FIG. 5, for example, in the case of a 3-level cell, E state is mapped to '111', P1 state is mapped to '011', P2 state is mapped to '001', P3 state is mapped to '000', P4 state is mapped to '010', P5 state is mapped to '110',

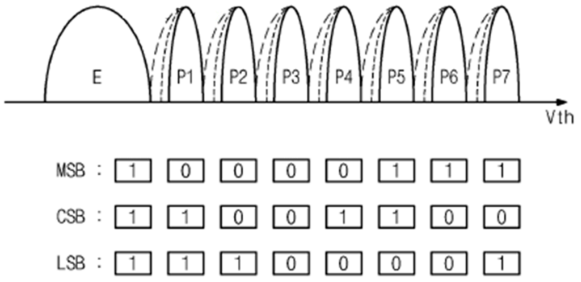

FIGURE 5. State mapping in 3-level cell program is shown. States programmed into the memory cells in the three-level NAND flash memory and the corresponding data bits are shown.

P6 state is mapped to '100', and P7 state is mapped to '101'. That is, program states with MSB (Most Significant Bit), CSB (Center Significant Bit), and LSB (Least Significant Bit) corresponding to the threshold voltage of the memory cell are determined [14].

Performing a write operation in a NAND flash memory refers to forming each memory cell have any one of the eight threshold voltages. NAND flash memory basically includes a plurality of memory blocks, each of the plurality of memory blocks includes a plurality of pages, and each of the plurality of pages includes a plurality of memory cells. In general, NAND flash memory performs a write operation or a read operation in units of pages and performs an erase operation in units of blocks.

III. POSSIBLE OVERWRITING IN NAND FLASH MEMORY

Generally, it is known that NAND flash memories cannot be overwritten. However, there is at least one state in which NAND flash memory can be partially overwritten, and there is a data bit corresponding to the overwritable state. In 3-level programming, there is little probability that programmed states will all be a highest state P7. Since the randomization technique is applied, the probability that data corresponding to the highest state is written is 1/8 or less. This means that at least one of the states lower than the highest state can be overwritten.

A. OVERWRITABLE STATES

As noted in state mapping, each of the memory cells has a threshold voltage having any one of eight states. The write operation of NAND flash memory is repeated by sequentially applying a voltage to the word line corresponding to the page and verifying whether the threshold voltage, associated with the corresponding bit, is exceeded [9]. In this case, since the threshold voltage of the memory cell in which the program is completed is unlikely to be the final state P7, states E to P6 lower than the final state P7 may be overwritten. We are interested in this overwritable state.

B. OVERWRITABLE DATA

As described above, it can be seen that there is an overwritable state. Thus, if it is known which bit of data is currently written in the memory cell, the writable state is determined,

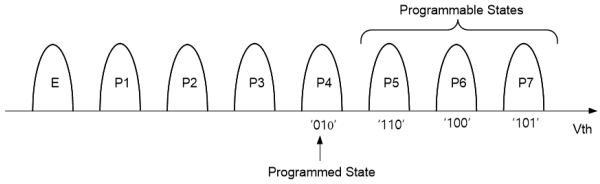

FIGURE 6. Overwritable Data are shown. Programmable states P5, P6, P7 are shown in a memory cell having a particular programmed state P4.

and thus the overwritable data for the writable state may be determined [9].

As shown in FIG. 6, for example, if it is P4 state corresponding to '010' of the programmed memory cell, the overwritable state is the upper states, namely P5 to P7 states. Therefore, data that can be overwritten on the memory cell is '110', '100', and '101'.

In MLC program operations, it has been found that overwriting is possible for higher states than the programmed states. However, such overwriting is based on a program operation by default, which can lead to program disturbances. The program disturbances affect valid data of neighboring pages by a current program operation. Our deletion schemes are designed to enable overwriting while minimizing the effects of these program disturbances.

In addition, in order to overwrite a page, it is inevitable to read the data of the current page, check the overwritable state in the read data, and generate data for overwriting. However, if the costs for all of the above-described processes are not greater than the cost of destroying personal information immediately in NAND flash memory, we will inevitably have to use this approach.

IV. PROPOSED PRIVACY DATA DESTRUCTION SCHEMES

We propose privacy data destruction schemes in NAND flash memory using overwritable states. The proposed privacy data destruction schemes include a partial overwriting scheme, an SLC programming scheme and a deletion duty pulse application scheme.

A. PARTIAL OVERWRITING SCHEME

It is assumed that the invalid data of the memory block to be deleted includes privacy data. There will be binary data corresponding to the privacy data, and there will be data that modifies it. Information on the program state may be obtained through the respective state mappers of the modulated data. When program states corresponding to privacy data are obtained, data corresponding to states equal to or higher than the programmed states may be generated. Then, a write operation may be performed on a page on which invalid data are programmed with such data. Privacy data may be discarded by such an overwriting operation [9].

In general, program operation of NAND flash memory is performed by ISPP (Incremental Step Pulse Programming) [15], [16]. A specific pulse is applied to the word line and a pulse incremented by a predetermined value is applied to the word line in each step. This operation is repeated until

TABLE 1. Privacy data destruction process using partial overwrite.

Personal Inform.	661004															
Binary Bits (48)	001101100011011000110001001100000011000000110100															
Random Bits (48)	110110001101100011000100110000001100000011010000															
State Mapping (16)	P5	P5	P2	P7	P6	P1	P3	P6	P5	P3	P2	P6	P3	P1	P4	P3
Partial Overwritable States (16)	P6 P7	P6 P7	P3 P4 P5 P6 P7		P7	P2 P3 P4 P5 P6 P7	P4 P5 P6 P7	P7	P6 P7	P4 P5 P6 P7	P3 P4 P5 P6 P7	P7	P4 P5 P6 P7	P2 P3 P4 P5 P6 P7	P5 P6 P7	P4 P5 P6 P7
Selected State (16)	P6	P7	P3	P7	P7	P4	P5	P7	P6	P4	P5	P7	P5	P2	P5	P4
PO Bits (48)	100101000101101010110101100010110101110001110010															
Derandom Bits (48)	001100010011000000110000001100010011000000110001															
Purge	100101															

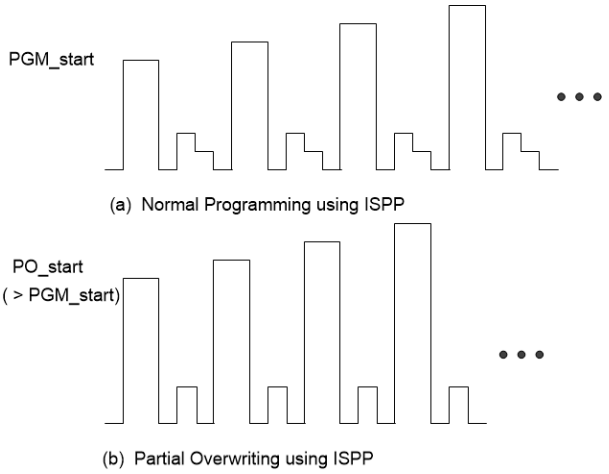

FIGURE 7. Partial overwriting using ISPP is shown. Note that (a) shows pulses related to normal programming using ISPP, and (b) shows pulses related to partial overwriting using ISPP.

state corresponds to the desired threshold voltage. After the initial program pulse PGM_start, verify pulses are applied to verify state. When such a program set is advanced, a predetermined number of program pulses are applied, and then verify pulses are applied.

We suggest performing partial overwriting using ISPP described above. Of course, the initial overwrite pulse PO_start will be higher than the initial program pulse PGM_pulse of normal program operation. In addition, in the partial overwrite operation, the verify pulse may be set to less than the number of verify pulses of normal program operation. This is because the program state in which the overwrite operation is performed is expected to have a relatively high threshold voltage in most cases.

In the following, we will describe how to delete a personal dentification number using partial overwriting, for example. For convenience of description, it will be assumed that the personal identification number is '661004'. The personal identification number can be changed in binary as follows. A state chain of Table 1 is selected with a selected partial overwrite state. Binary bits corresponding to '661004' are '001101100011011000110001001100000011000000110100'. By randomizing the binary bits through random algorithm, the randomized bits are '110110001101100011000100 11000 000110000011010000'.

States corresponding to the randomized bits are selected. The selected states are P5-P5-P2-P7-P6-P1-P3-P6-P5-P3-P2-P6-P3-P1-P4-P3. Each of the selected states is programmed into a corresponding memory cell of a page. As described above, program operations for states corresponding to privacy data are performed on the memory cells. Thereafter, according to a request for destroying privacy data of the programmed memory cells, the partial overwriting may be performed as follows.

The partial overwritable states for personal identification number 661004 may be configured in any one of various overwrite enabled states, as shown in Table 1. In the highest program state P7, no action will be taken. Therefore, the chain of program states selected for the partial overwriting is P6-P7-P3-P7-P7-P4-P5-P7-P6-P4-P5-P7-P5-P2-P5-P4. Thus, the selected state chain is programmed in the memory cells. State to be used for partial overwriting may be randomly selected.

However, it is not necessarily limited to this, and may be selected as the highest state P7, or a directly lower state P6 may be selected. When the highest state P7 is selected as a state of partial overwriting, it should be noted that there is a possibility of physical destruction of the memory cell due to an excessive write operation.

The partial overwriting scheme may verify that the destruction of privacy data was successful. When performing a state read operation from a partially overwritten memory cell, PO bits are '100101000101101010110101100010110101 11000 1110010.' When derandomizing the PO bits, the derandomized bits are '001100010011000000110000001100010011 000000110001.' Therefore, ASCII values corresponding to the derandomized bits are '100101'. Thus, the privacy data 661004 is completely destroyed in the memory cells.

TABLE 2. Privacy data destruction process using partial overwrite.

Personal Inform.	661004															
Binary Bits (48)	001101100011011000110001001100000011000000110100															
Random Bits (48)	110110001101100011000100110000001100000011010000															
State Mapping (16)	P5	P5	P2	P7	P6	P1	P3	P6	P5	P3	P2	P6	P3	P1	P4	P3
Partial Overwritable States (16)						P2								P2		
			P3			P3					P3			P3		
			P4			P4	P4			P4	P4		P4	P4		P4
			P5			P5	P5			P5	P5		P5	P5	P5	P5
	P6	P6	P6			P6	P6		P6	P6	P6		P6	P6	P6	P6
	P7	P7	P7		P7	P7	P7	P7	P7	P7	P7	P7	P7	P7	P7	P7
Selected State (16)	P5	P7	P5	P7	P6	P5	P5	P6	P5	P5	P5	P6	P5	P5	P5	P5
PO Bits (48)	100101000101101010110101100010110101110001110010															
Derandom Bits(48)	110001001100000011000100110001001100000011000100															
Purge	ÄÀÄÄÀÄ															

To reduce the burden of selecting any one of the above overwritable states, the partial overwriting proceed only to the program state lower than the specific program state. For example, the partial overwriting may proceed with the batch program operation to the fifth program P5 for states E, P1, P2, P3, and P4 lower than the fifth program state P5. That is, only the memory cells having a state lower than the fifth program state P5 are programmed to the fifth program state P5, and the memory cells having states higher than the fifth program state P5 maintain their current states.

A state chain of Table 2 is selected with a selected partial overwrite state higher than a specific program state. According to this method, the chain of program states selected for the partial overwriting is P5-P7-P5-P5-P5-P5-P5-P6-P5-P5-P5-P6-P5-P6-P5-P5-P5-P5. PO bits corresponding to these states chain are '100101000101101010110101100010110101110001110010', and are programmed into the memory cells. ASCII values corresponding to this binary code is 'ÄÀÄÄÀÄÀ". Therefore, privacy data may be completely destroyed.

The above-described method is actually possible because a randomization technique is applied to equalize the number of bits to be programmed. When modulating the original data, the number less than a specific state, for example, the intermediate program state, is similar to the remaining number because it is randomized. Therefore, the reverse analysis attack, conducted by the partial overwriting by the attacker, will be blocked as a source.

B. SLC PROGRAMMING

In general, NAND flash memory performs multi-level cell program operation. When the target page is invalidated by an operation such as garbage collection, an on-the-fly invalid page program operation is performed to destroy privacy data before the corresponding memory block is processed into an invalid block. In this case, the on-the-fly invalidation page program operation constitutes a single-level cell program operation based on a specific reference level rather than a multi-level cell program operation.

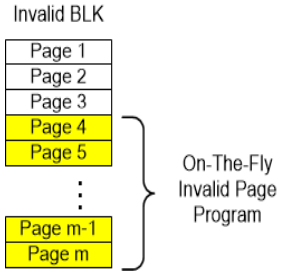

FIGURE 8. On-the-fly invalid page program is shown. An on-the-fly invalid page program for a page with privacy data in an invalid block is shown.

Our proposed on-the-fly invalid page program will perform a single-level cell program operation based on a certain threshold voltage, referring to FIG.8.

The memory controller may force an invalid page program on the fly on invalid pages prior to invalid block processing. This on-the-fly invalid page program operation may be achieved by performing a single-level cell program operation on random data in programmed multi-level cells. The concept is slightly different from the above-mentioned partial overwriting technique. In the partial overwriting technique, three pages of data are generated and programmed for privacy data destruction, but an on-the-fly invalid page programming technique is sufficient to generate and program only one page of data. In addition, there is no need to verify whether the programming is successful.

In FIG. 9, on-the-fly invalid page program operation using an SLC program (MLC PGM → SLC PGM) is shown.

For example, the privacy data stored by a 3-level cell program operation is destroyed by a 1-level cell program operation, referring to FIG. 8. Therefore, it may be called "SLC programming technique".

C. DDP APPLICATION SCHEME

Generally, even when a power supply voltage is applied to a body region of a memory cell, if a word line voltage of a predetermined value or more is applied to a word line connected to the memory cell, charge transfer may occur due to F-N tunneling. This implies that the operation for invalid

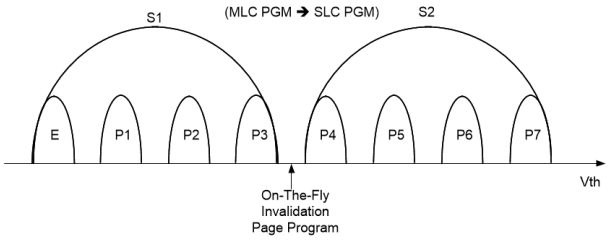

FIGURE 9. On-the-fly invalid page program using SLC programming is shown. The on-the-fly invalid page program includes an SLC program operation for performing a single-level cell program operation from a multi-level cell program.

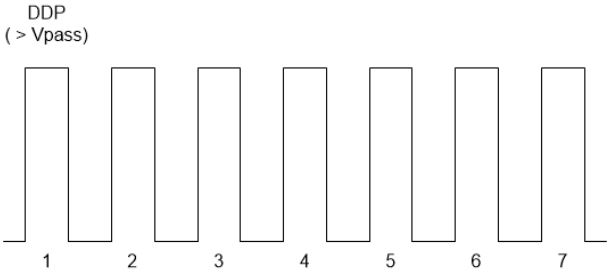

FIGURE 10. Deletion duty pulse (DDP) is shown. Note that DDN pulse does not require a verify pulse and has the same pulse amplitude.

data destruction may be performed significantly easily. To put it another way, when the word line pulse is applied to the word line corresponding to the invalid page having the privacy data without applying the power supply voltage to the body side, the privacy data may be easily destroyed as a whole. This is because the threshold voltage may change into completely unknown states.

These wordline pulses are referred to as DDP (Deletion Duty Pulses), referring to FIG.10. By a request from the host or by NAND flash memory's own request, DDP may be applied to the invalid page. Since the data state is not required to be checked after such a DDP is applied, a verify pulse is not required. The level of DDP may be higher than a pass voltage Vpass. In general, when a voltage lower than the pass voltage Vpass is applied to the word line, no charge transfer occurs in the channel. Also, the number of DDPs should be determined experimentally. For example, the number of DDPs may be determined to be smaller than the level or number of conventional program pulses. The number of DDPs may be determined to significantly reduce program disturbances.

Generally, NAND flash memory uses ECC (Error Correction Code) technique to correct data due to deterioration of data. The number of erasure duty pulses should be determined to be greater than the error correction level of this ECC technique. The number of such experimental DDPs will depend on NAND chip manufacturer's cell characteristics.

D. REQUEST TO DESTROY PRIVACY DATA

Upon receiving the privacy data destruction request from the host, the memory controller preferentially identifies a block having an invalid page for the unaddressed area

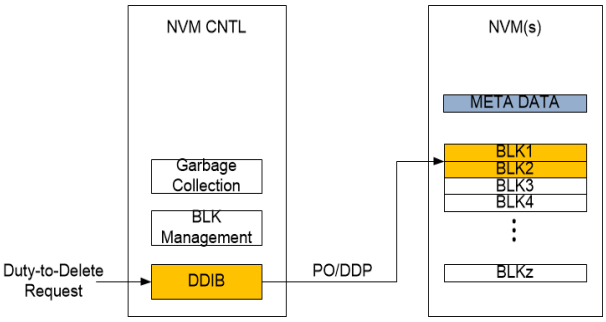

FIGURE 11. Proposed storage device for processing duty-to-delete request is shown. The storage device may receive a privacy-related delete request from the host and may operate the privacy data destruction schemes in response to the request.

(e.g., BLK1, BLK2) in NAND flash memory. Thereafter, a partial overwriting (PO) operation or a delete duty pulse (DDP) operation is performed on the block having an invalid page.

As shown in FIG. 11, the memory controller may include a garbage collection unit, a block management unit, and a delete duty execution unit DDIB. The block management unit may specially manage invalid blocks having stored privacy data, as compared with the block management unit. For example, the block management unit may store a physical address for an invalid page for the invalid block. A deletion duty execution unit may apply a privacy data destruction technique (e.g., PO / DDP) to the physical address for an invalid page.

E. VERIFICATION OF PRIVACY DATA DESTRUCTION

In response to the host's privacy data destruction request, NAND flash memory may perform a privacy data destruction operation. Thereafter, NAND flash memory may perform a verify operation on the request for destroying privacy data. The verify operation of the privacy data destruction request determines whether the information to be restored of the privacy data requested to be destroyed or the information to be restored exists in the valid page or the invalid page of NAND flash memory. For example, a determination is made as to whether data, in which privacy data and privacy data are modulated, exists in the invalidation area or the validation area of NAND flash memory. If there is no the privacy data or data corresponding to the privacy data in NAND flash memory, the privacy data destruction request is determined to be successful. Then, the NAND flash memory transmits this fact in response to the host. Basically, privacy data search for unmapped memory blocks as well as mapped memory blocks should be performed to verify the privacy data destruction.

However, such verification is not as easy as expected. This is because the above-described destroy request verify operation is effective only under the assumption that privacy data exists only in a non-volatile memory such as a NAND flash memory. Actually, there is a possibility that the privacy data exists in volatile memory (DRAM, SRAM, etc.) temporarily uploaded. In order to verify the existence of privacy data, there should be data to be reference data, and the reference

TABLE 3. Garbage collection case.

Schemes	Degree of Cell Gradation (Memory Block)			Privacy Data Destruction Time
	PGM count	Erase count	Total	
Block Erase Scheme	0	1	a	x (until GG, >M*T_{PGM})
Lin's Scheme	x	x	x	x
Partial Overwriting Scheme	< N	0	< b*N	M*T_{PGM} +N* T_{RDG} +T_{POW}
SLC Programming Scheme	< N	0	< b*N	M*T_{PGM} + N*T_{SDG} +T_{SLCP}
DDP Scheme	< N	0	< b*N	M*T_{PGM} +N* T_{DDP}

TABLE 4. One page update case.

Schemes	Degree of Cell Gradation (Memory Block)			Privacy Data Destruction Time
	PGM count	Erase count	Total	
Block Erase Scheme	0	1	a	x (until GG)
Lin's Scheme	1	0	b	M*T_{PGM}+$T_{ONESHOT}$
Partial Overwriting Scheme	1	0	b	M*T_{PGM} + T_{RDG} +T_{POW}
SLC Programming Scheme	1	0	b	M*T_{PGM} + T_{SDG} +T_{SLCP}
DDP Scheme	1	0	b	M*T_{PGM} + T_{DDP}

data may also exist in the volatile memory. Therefore, after the verify operation is completed, it is basically necessary to refresh the volatile memory in the storage device having NAND flash memory. There is a need for more researches into this issue.

V. PERFORMANCE COMPARISON

We compared performances between the conventional Block Erase scheme, the partial overwriting, SLC programming, and DDP scheme. The performances include the degree of degradation and elapsed time of the memory cell until privacy data is destroyed in NAND flash memory. For the sake of convenience, we have compared the schemes with the garbage collection situation mentioned in section II.

Assume that there are M valid pages and N invalid pages in two memory blocks, the M valid pages will be programmed into a new memory block. Assume that there are totally N invalid pages in the two memory blocks, in all schemes, M program operations are performed on a new memory block by the garbage collection. In addition, a read operation for reading valid pages is not mentioned in the performance index because the degree of degradation is inferior to an erase operation or a program operation. In Table 3, ''a'' denotes the degree of cell degradation caused by the erase operation and ''b'' denotes the degree of cell degradation caused by the program operation. In general, ''a'' is 1000 times higher than ''b'' [9], [17]. In other words, the degree of degradation caused by the erase operation is more severe than the degree of degradation caused by the program operation.

In the case of a block erase scheme, no program operation is performed on two memory blocks to be invalidated. Instead, a program operation is performed on M valid pages for a new memory block. Therefore, in the block to be invalidated, the erase operation is performed once to delete privacy data. Typically, block erase operations in garbage collection are performed in accordance with NAND flash memory's internal policy. The internal policy is managed to significantly increase the lifespan of memory blocks. Therefore, as shown in Table 3, it is substantially impossible to predict privacy data destruction time due to various operating variables.

Meanwhile, the proposed schemes can predict privacy data destruction time by applying the deletion schemes. For example, time of the partial overwriting scheme is $M^*T_{PGM}+N^*T_{RDG}+T_{POW}$, time of SLC programming scheme is $M^*T_{PGM}+ N^*T_{SDG}+T_{SLCP}$, and time of DDP scheme is $M^*T_{PGM}+N^*T_{DDP}$. Where T_{PGM} is the program operation time, T_{RDG} is random data generation time, T_{SDG} is SLC data generation time, T is the overwriting time, T_{POW} is the SLC programming time, and T_{DDP} is the DDP application time.

In summary, our proposed schemes can predict the privacy data destruction time. In addition, by comparing the degree of cell degradation for one memory block, the block erase scheme has the degree of degradation corresponding to one erase operation.

Lin's approach considers only the update situation, rather than the garbage collection situation. This suggests that, according to Lin's approach, personal information is still in the invalidation blocks in a garbage collection situation.

Also, we compared the performances of the deletion schemes assuming a case in which an update is performed to a single page of privacy data in a single memory block. On the other hand, Lin's scheme and our proposed schemes do not need to perform an erase operation until the privacy data is destroyed. Therefore, until the destruction of privacy data, our schemes have the degree of degradation less than that in N program operations.

Even in a single page update, the conventional block erase scheme cannot predict privacy data destruction time. On the other hand, both Lin's scheme and our schemes can predict the destruction time. And the degradation characteristic up to privacy data destruction is also better in the case of the proposed schemes than the conventional erase scheme. As shown in Table 4, for example, time of Lin's scheme is $M^*T_{PGM}+T_{ONESHOT}$, time of the partial overwriting scheme is $M^*T_{PGM}+T_{RDG}+T_{POW}$, time of SLC programming scheme is $M^*T_{PGM}+T_{SDG}+T_{SLCP}$, and time of DDP scheme is $M^*T_{PGM}+T_{DDP}$. Where $T_{ONESHOT}$ is the one shot programming time.

We confirmed that the proposed schemes can significantly reduce the privacy data destruction time, as compared to the conventional block erase scheme. In addition, we can see that the proposed schemes have many advantages in terms of degradation characteristics of memory cells. As described above, our proposed method of destroying privacy data can be easily applied to existing NAND flash memory, and seems to be useful in terms of management and economics.

However, deeper researches into program disturbance for valid pages according to partial overwriting, SLC programming, and DDP application should be conducted in the future. We proceeded ignoring a read count, but we should also consider a factor of the read count to actually apply the deletion duty. The disturbance issue, caused bya read operation, will also be discussed later.

VI. CONCLUSION

We have identified privacy data that is bound to remain in NAND flash memory. Such residual privacy data increases the risk of leakage of privacy data by illegal acquisition or unauthorized access to NAND flash memory. We discussed how to delete this residual privacy data from NAND flash memory. For example, we have proposed a technique to identify personal information by partially identifying programmable states and using partial overwriting schemes. We have also proposed a method of destroying privacy data by applying a deletion duty pulse several times. We also proposed a method of destroying privacy data using SLC programming. Our privacy data destruction schemes in the NAND flash memory significantly reduce the memory performance degradation, as compared with the conventional erase operation, and expect a considerable effect in terms of time and costs. Our proposed NAND flash memory privacy schemes are expected to have very important and positive consequences for future societies that have the duty to delete. In privacy technology, we are confident that our proposed techniques will be useful as anti-forensic technology of NAND flash memories.

ACKNOWLEDGMENT

The authors would like to thank Prof. Kwon for introducing them to the right to be forgotten.

REFERENCES

[1] Y. Xu and Z. Hou, "NVM-assisted non-redundant logging for Android systems," in *Proc. IEEE Trustcom/BigDataSE/ISPA*, Tianjin, China, Aug. 2016, pp. 1427–1433.

[2] R. Jin and T.-S. Chung, "Dynamic regulation of index implementation for flash memory storages," in *Proc. 2nd Int. Conf. Comput. Autom. Eng. (ICCAE)*, Singapore, 2010, pp. 325–328.

[3] Q. Zhang, X. Li, L. Wang, T. Zhang, Y. Wang, and Z. Shao, "Optimizing deterministic garbage collection in NAND flash storage systems," in *Proc. 21st IEEE Real-Time Embedded Technol. Appl. Symp.*, Seattle, WA, USA, Apr. 2015, pp. 14–23.

[4] D. Chang, W. Lin, and H. Chen, "FastRead: Improving read performance for multilevel-cell flash memory," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 24, no. 9, pp. 2998–3002, Sep. 2016.

[5] C. Matsui, A. Arakawa, C. Sun, and K. Takeuchi, "Write order-based garbage collection scheme for an LBA scrambler integrated SSD," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 2, pp. 510–519, Feb. 2017.

[6] J. P. van Zandwijk and A. Fukami, "NAND flash memory forensic analysis and the growing challenge of bit errors," *IEEE Security Privacy*, vol. 15, no. 6, pp. 82–87, Nov./Dec. 2017.

[7] A. Jonesm, O. Angelopoulo, and L. Noriega, "Survey of data remaining on second hand memory cards in the UK," *Comput. Secur.*, vol. 84, pp. 239–243, Jul. 2019.

[8] *Data Privacy Statement*. Accessed: Aug. 10, 2019. [Online]. Available: https://www.eugogo.eu/data-privacy

[9] N. Y. Ahn and D. H. Lee, "Duty to delete on non-volatile memory," Jul. 2017, *arXiv:1707.02842*. [Online]. Available: https://arxiv.org/abs/1707.02842

[10] P. H. Lin, Y. M. Chang, Y. C. Li, W. C. Wang, C. C. Ho, and Y. H. Chang, "Achieving fast sanitization with zero live data copy for MLC flash memory," in *Proc. Int. Conf. Comput.-Aided Design (ICCAD)*, New York, NY, USA, 2018.

[11] R. Kissel, A. Regenscheid, M. Scholl, and K. Stine, "Guidelines for media sanitization," in *Proc. NIST*, 2014.

[12] J. Gu, C. Wu, and J. Li, "HOTIS: A hot data identification scheme to optimize garbage collection of SSDs," in *Proc. IEEE Int. Symp. Parallel Distrib. Process. Appl. IEEE Int. Conf. Ubiquitous Comput. Commun. (ISPA/IUCC)*, Guangzhou, China, Dec. 2017, pp. 331–3317.

[13] J. Cha, W. Kang, J. Chung, K. Park, and S. Kang, "A new accelerated endurance test for terabit NAND flash memory using interference effect," *IEEE Trans. Semicond. Manuf.*, vol. 28, no. 3, pp. 399–407, Aug. 2015.

[14] M. Kang, W. Lee, and S. Kim, "Subpage-aware solid state drive for improving lifetime and performance," *IEEE Trans. Comput.*, vol. 67, no. 10, pp. 1492–1505, Oct. 2018.

[15] P. Wang, "Three-dimensional NAND flash for vector-matrix multiplication," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 27, no. 4, pp. 988–991, Apr. 2019.

[16] J. Ko, "Variation-tolerant WL driving scheme for high-capacity NAND flash memory," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 27, no. 8, pp. 1828–1839, Aug. 2019.

[17] T. Chen, Y. Chang, C. Ho, and S. Chen, "Enabling sub-blocks erase management to boost the performance of 3D NAND flash memory," in *Proc. 53nd ACM/EDAC/IEEE Design Automat. Conf. (DAC)*, Austin, TX, USA, Jun. 2016, pp. 1–6.