(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
9 March 2017 (09.03.2017)

WIPO | PCT

(10) International Publication Number
WO 2017/039841 A1

(51) International Patent Classification:
G06F 11/07 (2006.01) *G06F 3/06* (2006.01)

(21) International Application Number:
PCT/US2016/041975

(22) International Filing Date:
13 July 2016 (13.07.2016)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
14/845,503 4 September 2015 (04.09.2015) US

(71) Applicant: **INTEL CORPORATION** [US/US]; 2200 Mission College Boulevard, Santa Clara, California 95054 (US).

(72) Inventors; and
(71) Applicants *(for US only)*: **RAMANUJAN, Raj K.** [US/US]; 475 SW 345th Pl, Federal Way, Washington 98023 (US). **RAAD, Camille C.** [US/US]; 217 Croce St, Folsom, California 95630 (US). **MANGOLD, Richard P.** [US/US]; 7155 NW Kansas City Rd, Forest Grove, Oregon 97116 (US). **YIGZAW, Theodros** [US/US]; 21505 SW Chapman Rd., Sherwood, Oregon 97140 (US).

(74) Agent: **JORDAN, B. Delano**; Jordan IP Law, LLC, c/o CPA Global, 900 Second Ave. South, Suite 600, Minneapolis, Minnesota 55402 (US).

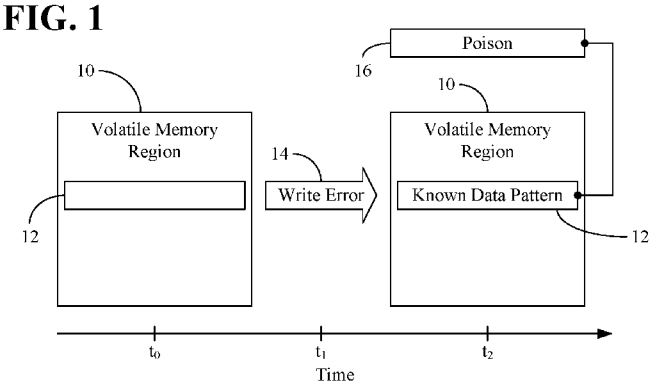
(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report (Art. 21(3))*
— *with amended claims (Art. 19(1))*

(54) **Title**: CLEARING POISON STATUS ON READ ACCESSES TO VOLATILE MEMORY REGIONS ALLOCATED IN NON-VOLATILE MEMORY

FIG. 1



(57) **Abstract**: Systems and methods may provide for detecting that a read operation is directed to a memory region while the memory region is in a poisoned state and clearing the poisoned state if volatile data stored in the memory region does not correspond to a known data pattern. Additionally, the memory region may be maintained in the poisoned state if the volatile data stored in the memory region corresponds to the known data pattern. In one example, an error may be detected, wherein the error is associated with a write operation directed to the memory region. In such a case, the poisoned state may be set for the volatile data in response to the error and the known data pattern may be written to the memory region.

WO 2017/039841 A1

CLEARING POISON STATUS ON READ ACCESSES TO VOLATILE MEMORY REGIONS ALLOCATED IN NON-VOLATILE MEMORY

CROSS-REFERENCE TO RELATED APPLICATOINS

The present application claims the benefit of priority to U.S. Non-Provisional Patent Application No. 14/845,503 filed on September 4, 2015.

TECHNICAL FIELD

Embodiments generally relate to memory structures. More particularly, embodiments relate to clearing poison status on read accesses to volatile memory regions allocated in non-volatile memory.

BACKGROUND

When internal data corruptions are detected in conventional volatile memory, a "poison" state may be set in metadata corresponding to the impacted region (e.g., cache line) of memory to ensure that the corrupted data is not used in future operations. If power is subsequently removed from the volatile memory (e.g., due to a system reset), the poison state may be cleared because the corrupted data may be automatically lost from the impacted memory region the due to the volatile nature of the memory.

BRIEF DESCRIPTION OF THE DRAWINGS

The various advantages of the embodiments will become apparent to one skilled in the art by reading the following specification and appended claims, and by referencing the following drawings, in which:

FIG. 1 is an illustration of an example of a volatile memory region according to an example;

FIG. 2A is a flowchart of an example of a method of handling data corruptions according to an embodiment;

FIG. 2B is a flowchart of an example of a method of handling read accesses according to an embodiment;

FIG. 3 is a flowchart of an example of a method of managing a known data pattern according to an embodiment;

FIG. 4 is a block diagram of an example of a memory controller apparatus according to an embodiment; and

FIG. 5 is a block diagram of an example of a computing system according to an embodiment.

DESCRIPTION OF EMBODIMENTS

Recent developments in memory architectures may provide for non-volatile memory (NVM) that is used to store volatile data. The volatile data may include, for example, data used by an application or operating system, that the application or operating system considers to be stored in a volatile memory and is no longer stored in the volatile memory after a system reset. Examples of NVM may include, for example, phase change memory (PCM), three dimensional cross point memory, resistive memory, nanowire memory, ferro-electric transistor random access memory (FeTRAM), flash memory such as NAND or NOR, magnetoresistive random access memory (MRAM) memory that incorporates memristor technology, spin transfer torque (STT)-MRAM, and so forth. Poison states may typically be cleared for NVM structures only when re-initializations occur (e.g., during manufacturing or reformatting). In such a case, a poison state that has been set for volatile data stored within the NVM may remain after a system reset even though the loss of power may have effectively cleared the underlying data corruption. As a result, regions of volatile memory may appear to be corrupted when they are in fact usable. Techniques described herein may eliminate such a condition.

Turning now to FIG. 1, a volatile memory region 10 is shown, wherein the volatile memory region 10 may generally be used as, for example, dynamic random access memory (DRAM), static RAM (SRAM) or other suitable memory structure that does not have a persistency requirement. In one example, the memory region 10 is a physical volatile memory that is part of a two level memory (2LM) or is a portion of a NVM used to emulate operation of a volatile memory after a system reset. The 2LM may include, for example, cached subsets of system disk level storage in addition to, for example, run-time data. The illustrated volatile memory region 10 has an area 12 (e.g., cache line) that contains either no data or valid data at time t_0. An error 14 associated with a write operation may occur at time t_1, wherein the error 14 may cause a poison state 16 to be set for the volatile memory region 10 at time t_2. The poison state 16 may be set, for example, by toggling a bit and/or field in metadata

corresponding to the area 12 in the volatile memory region 10. The error 14 may also cause a known data pattern (e.g., a "poison pattern") to be automatically written to the area 12 (e.g., replacing the corrupt data) at time t_2. As will be discussed in greater detail below, incrementing the known data pattern on each power cycle may enable the poison state 16 to be subsequently cleared even if the volatile memory region 10 is allocated within a non-volatile memory structure and no re-initialization has taken place. As a result, more efficient memory usage may be achieved.

FIG. 2A shows a method 18 of handling data corruptions such as, for example, the write error 14 (FIG. 1), already discussed. The method 18 may be implemented in one or more modules as a set of logic instructions stored in a machine- or computer-readable storage medium such as RAM, read only memory (ROM), programmable ROM (PROM), firmware, flash memory, etc., in configurable logic such as, for example, programmable logic arrays (PLAs), field programmable gate arrays (FPGAs), complex programmable logic devices (CPLDs), in fixed-functionality logic hardware using circuit technology such as, for example, application specific integrated circuit (ASIC), complementary metal oxide semiconductor (CMOS) or transistor-transistor logic (TTL) technology, or any combination thereof. For example, computer program code to carry out operations shown in method 18 may be written in any combination of one or more programming languages, including an object oriented programming language such as JAVA, SMALLTALK, C++ or the like and conventional procedural programming languages, such as the "C" programming language or similar programming languages.

Illustrated processing block 20 determines whether an error has been detected, wherein the error is associated with a write operation directed to a memory region (e.g., non-volatile/persistent memory region, volatile memory region allocated in non-volatile/persistent memory, etc.). If so, block 22 may set a poisoned state for the memory region in response to the error. Block 22 may include toggling a bit and/or field in metadata corresponding to the memory region. Additionally, illustrated block 24 writes a known data pattern to the memory region. Block 24 may include, for example, repeating the write of a 64-bit known data pattern across a 64B cache line. The known data pattern may be incremented or otherwise changed in a known fashion on each power cycle. Additionally, if a write error is not detected at block 20, the illustrated method 18 repeats.

FIG. 2B shows a method 26 of handling read accesses. The method 26 may also be implemented in one or more modules as a set of logic instructions stored in a machine- or computer-readable storage medium such as RAM, ROM, PROM, firmware, flash memory, etc., in configurable logic such as, for example, PLAs, FPGAs, CPLDs, in fixed-functionality logic hardware using circuit technology such as, for example, ASIC, CMOS or TTL technology, or any combination thereof.

Illustrated processing block 28 may provide for determining whether a read operation has been detected, wherein the read operation is directed to a memory region that is in a poisoned state. Block 28 may therefore include accessing metadata corresponding to the memory region in question. If a read from a poisoned region is detected at block 28, a determination may be made at block 29 as to whether the memory region is a volatile memory region. If the memory region is not a volatile memory region (e.g., non-volatile/persistent memory region), the memory region may be maintained in the poisoned state at block 32. Otherwise, a determination may be made at block 30 as to whether data stored in the volatile memory region corresponds to (e.g., matches) a known data pattern. If so, it may be inferred that a power cycle has not taken place after the volatile memory region was poisoned and illustrated block 32 maintains the volatile memory region in the poisoned state. If, on the other hand, it is determined at block 34 that the data stored in the volatile memory region does not correspond to the known data pattern, then block 34 may clear the poisoned state (e.g., inferring that a power cycle has taken place after the volatile memory region was poisoned). If a read from a poisoned region is not detected at block 28, the illustrated method 26 repeats.

Turning now to FIG. 3, a method 36 of managing a known data pattern is shown. The method 36 may also be implemented in one or more modules as a set of logic instructions stored in a machine- or computer-readable storage medium such as RAM, ROM, PROM, firmware, flash memory, etc., in configurable logic such as, for example, PLAs, FPGAs, CPLDs, in fixed-functionality logic hardware using circuit technology such as, for example, ASIC, CMOS or TTL technology, or any combination thereof.

Illustrated processing block 38 provides for generating a counter value such as, for example, a 64-bit poison pattern, wherein the counter value is a known data pattern. The counter value may be stored to a register such as, for example, a control-status register (CSR) at block 40. The register may be error correction code (ECC)

protected so that an error in the register does not mask out the counter value. Illustrated block 42 determines whether a power cycle (e.g., system reset) has been detected. If so, the counter value may be incremented (e.g., monotonically increased) or otherwise changed by a known amount at block 44. Thus, in the case of a monotonically increased 64-bit poison pattern, the chance of data matching the pattern (e.g., a false positive) would be 2^{64}. The power cycle determination at block 42 may repeat until a power cycle is detected. Other approaches to generating and maintaining the known data pattern may also be used.

FIG. 4 shows a memory controller apparatus 46 (46a-46f) that may be used to support an error-protected computing system. The memory controller apparatus 46 may generally implement one or more aspects of the method 18 (FIG. 2A), the method 26 (FIG. 2B) and/or the method 36 (FIG. 3), already discussed. In the illustrated example, a read monitor 46a detects that a read operation is directed to a volatile memory region while the volatile memory region is in a poisoned state. Additionally, the apparatus 46 may also include a state manager 46b that clears the poisoned state if volatile data stored in the memory region does not correspond to a known data pattern and maintains the memory region in the poisoned state if the volatile data stored in the memory region corresponds to the known data pattern.

In one example, a write monitor 46c detects an error associated with a write operation directed to the volatile memory region, wherein the state manager 46b may set the poisoned state for the volatile memory region in response to the error. Moreover, a substitution manager 46d may write the known data pattern to the volatile memory region. Additionally, the apparatus 46 may include a register 46e (e.g., ECC protected CSR) and a pattern manager 46f that generates a counter value and stores the counter value to the register 46e as the known data pattern. The pattern manager 46f may also increment the counter value stored in the register 46e in response to power cycles.

FIG. 5 shows an error-protected computing system 50. The computing system 50 may generally be part of an electronic device/platform having computing functionality (e.g., personal digital assistant/PDA, notebook computer, tablet computer, server), communications functionality (e.g., smart phone), imaging functionality, media playing functionality (e.g., smart television/TV), wearable functionality (e.g., watch, eyewear, headwear, footwear, jewelry), vehicular functionality (e.g., car, truck, motorcycle), etc., or any combination thereof. In the

illustrated example, the system 50 includes a power source 52 to supply power to the system 50 and a processor 54 having an integrated memory controller (IMC) 56 that is coupled to main memory 60 (e.g., volatile "near" memory). The IMC 56 may also be coupled to another memory module 57 (e.g., dual inline memory module/DIMM) containing a non-volatile memory structure such as, for example, NVM 58. The NVM 58 may include "far" memory 61, which may also be used to store volatile data. Thus, the far memory 61 and the main memory 60 may function as a 2LM structure, wherein the main memory 60 generally serves as a low-latency and high-bandwidth cache of the far memory 61, which may have considerably lower bandwidth and higher latency.

The NVM 58 may include, for example, PCM, three dimensional cross point memory, resistive memory, nanowire memory, FeTRAM, flash memory such as NAND or NOR, MRAM that incorporates memristor technology, STT-MRAM, and so forth. As already noted, the memory module 57 may include, for example, volatile DRAM configured as one or more memory modules such as, for example, DIMMs, small outline DIMMs (SODIMMs), etc.

The illustrated system 50 also includes an input output (IO) module 64 implemented together with the processor 54 on a semiconductor die 66 as a system on chip (SoC), wherein the IO module 64 functions as a host device and may communicate with, for example, a display 68 (e.g., touch screen, liquid crystal display/LCD, light emitting diode/LED display), a network controller 70, and mass storage 72 (e.g., hard disk drive/HDD, optical disk, flash memory, etc.). The memory module 57 may include an NVM controller 63 having logic 76 that is connected to the far memory 61 via an internal bus 59 or other suitable interface. The illustrated logic 76 detects read operations directed to memory regions of the far memory 61 while the memory regions are in a poisoned state, clears the poisoned states if volatile data stored in the memory regions does not correspond to a known data pattern, and maintains the memory regions in the poisoned state if the volatile data stored in the memory regions corresponds to the known data pattern. Thus, the NVM controller 63 may have similar functionality to that of the memory controller apparatus 46 (FIG. 4), already discussed. For example, logic 76 may also include a CSR 62 (e.g., single error correction and double error detection/SECDED ECC protected register), wherein the known data pattern may be stored in the CSR 62 as a counter value that

increments with each power cycle. Additionally, the logic 76 may optionally be implemented elsewhere in the system 50.

Additional Notes and Examples:

Example 1 may include an error-protected computing system comprising a memory structure containing a memory region, a bus coupled to the memory structure and a memory controller apparatus coupled to the bus, the memory controller apparatus including a read monitor to detect that a read operation is directed to the memory region while the memory region is in a poisoned state and a state manager to clear the poisoned state if volatile data stored in the memory region does not correspond to a known data pattern and maintain the memory region in the poisoned state if the volatile data stored in the memory region corresponds to the known data pattern.

Example 2 may include the system of Example 1, wherein the memory controller apparatus further includes a write monitor to detect an error associated with a write operation directed to the memory region, wherein the state manager is to set the poisoned state for the volatile memory region in response to the error, and a substitution manager to write the known data pattern to the memory region.

Example 3 may include the system of any one of Examples 1 or 2, further including a control-status register and a pattern manager to generate a counter value and store the counter value to the control-status register, wherein the counter value is the known data pattern.

Example 4 may include the system of Example 3, wherein the control-status register is error correction code protected.

Example 5 may include the system of Example 3, wherein the pattern manager is to increment the counter value in response to a power cycle.

Example 6 may include the system of Example 1, wherein the memory structure is a non-volatile memory structure.

Example 7 may include a memory controller apparatus comprising a read monitor to detect that a read operation is directed to a memory region while the memory region is in a poisoned state and a state manager to clear the poisoned state if volatile data stored in the memory region does not correspond to a known data pattern and maintain the region in the poisoned state if the volatile data stored in the memory region corresponds to the known data pattern.

Example 8 may include the apparatus of Example 7, further including a write monitor to detect an error associated with a write operation directed to the memory region, wherein the state manager is to set the poisoned state for the volatile data in response to the error, and a substitution manager to write the known data pattern to the memory region.

Example 9 may include the apparatus of any one of Examples 7 or 8, further including a pattern manager to generate a counter value and store the counter value to a control-status register, wherein the counter value is the known data pattern.

Example 10 may include the apparatus of Example 9, wherein the control-status register is to be error correction code protected.

Example 11 may include the apparatus of Example 9, wherein the pattern manager is to increment the counter value in response to a power cycle.

Example 12 may include the apparatus of Example 7, wherein the read operation is to be directed to a non-volatile memory structure containing the memory region.

Example 13 may include a method of operating a memory controller apparatus comprising detecting that a read operation is directed to a memory region while the memory region is in a poisoned state, clearing the poisoned state if volatile data stored in the memory region does not correspond to a known data pattern, and maintaining the memory region in the poisoned state if the volatile data stored in the memory region corresponds to the known data pattern.

Example 14 may include the method of Example 13, further including detecting an error associated with a write operation directed to the memory region, setting the poisoned state for the volatile memory region in response to the error, and writing the known data pattern to the memory region.

Example 15 may include the method of any one of Examples 13 or 14, further including generating a counter value, and storing the counter value to a control-status register, wherein the counter value is the known data pattern.

Example 16 may include the method of Example 15, wherein the control-status register is error correction code protected.

Example 17 may include the method of Example 15, further including increment the counter value in response to a power cycle.

Example 18 may include the method of Example 13, wherein the read operation is directed to a non-volatile memory structure containing the memory region.

Example 19 may include at least one non-transitory computer readable storage medium comprising a set of instructions, which when executed by a computing system, cause the computing system to detect that a read operation is directed to a memory region while the memory region is in a poisoned state, clear the poisoned state if volatile data stored in the memory region does not correspond to a known data pattern, and maintain the memory region in the poisoned state if the volatile data stored in the memory region corresponds to the known data pattern.

Example 20 may include the at least one non-transitory computer readable storage medium of claim 19, wherein the instructions, when executed, cause the computing device to detect an error associated with a write operation directed to the memory region, set the poisoned state for the volatile memory region in response to the error, and write the known data pattern to the memory region.

Example 21 may include the at least one non-transitory computer readable storage medium of any one of claims 19 or 20, wherein the instructions, when executed, cause the computing device to generate a counter value, and store the counter value to a control-status register, wherein the counter value is the known data pattern.

Example 22 may include the at least one non-transitory computer readable storage medium of claim 21, wherein the control-status register is to be error correction code protected.

Example 23 may include the at least one non-transitory computer readable storage medium of claim 21, wherein the instructions, when executed, cause the computing device to increment the counter value in response to a power cycle.

Example 24 may include the at least one non-transitory computer readable storage medium of claim 19, wherein the read operation is to be directed to a non-volatile memory structure containing the memory region.

Example 25 may include a memory controller apparatus comprising means for detecting that a read operation is directed to a memory region while the memory region is in a poisoned state, means for clearing the poisoned state if volatile data stored in the memory region does not correspond to a known data pattern, and means

for maintaining the memory region in the poisoned state if the volatile data stored in the memory region corresponds to the known data pattern.

Example 26 may include the apparatus of claim 25, further including means for detecting an error associated with a write operation directed to the memory region, means for setting the poisoned state for the volatile memory region in response to the error, and means for writing the known data pattern to the memory region.

Example 27 may include the apparatus of any one of claims 25 or 26, further including means for generating a counter value, and means for storing the counter value to a control-status register, wherein the counter value is the known data pattern.

Example 28 may include the apparatus of claim 27, wherein the control-status register is to be error correction code protected.

Example 29 may include the apparatus of claim 27, further means for including increment the counter value in response to a power cycle.

Example 30 may include the apparatus of claim 25, wherein the read operation is to be directed to a non-volatile memory structure containing the memory region.

Techniques described herein may therefore use a unique data pattern that replaces a location's stored data when the location is poisoned during write operations. When a read operation is encountered, if poison is detected in the metadata bits that encode poison and the location's address is in a persistent memory region, the poison state may be maintained without checking the poison pattern. If, on the other hand, the location's address is in a memory region, the memory controller may compare the data with the poison pattern. Thus, the poison state might be maintained only if the comparison indicates a match. In the case of non-matching data and poison pattern, the memory controller may clear the poison state from the returning data.

Embodiments are applicable for use with all types of semiconductor integrated circuit ("IC") chips. Examples of these IC chips include but are not limited to processors, controllers, chipset components, programmable logic arrays (PLAs), memory chips, network chips, systems on chip (SoCs), SSD/NAND controller ASICs, and the like. In addition, in some of the drawings, signal conductor lines are represented with lines. Some may be different, to indicate more constituent signal paths, have a number label, to indicate a number of constituent signal paths, and/or have arrows at one or more ends, to indicate primary information flow direction. This, however, should not be construed in a limiting manner. Rather, such added

detail may be used in connection with one or more exemplary embodiments to facilitate easier understanding of a circuit. Any represented signal lines, whether or not having additional information, may actually comprise one or more signals that may travel in multiple directions and may be implemented with any suitable type of signal scheme, e.g., digital or analog lines implemented with differential pairs, optical fiber lines, and/or single-ended lines.

Example sizes/models/values/ranges may have been given, although embodiments are not limited to the same. As manufacturing techniques (e.g., photolithography) mature over time, it is expected that devices of smaller size could be manufactured. In addition, well known power/ground connections to IC chips and other components may or may not be shown within the figures, for simplicity of illustration and discussion, and so as not to obscure certain aspects of the embodiments. Further, arrangements may be shown in block diagram form in order to avoid obscuring embodiments, and also in view of the fact that specifics with respect to implementation of such block diagram arrangements are highly dependent upon the platform within which the embodiment is to be implemented, i.e., such specifics should be well within purview of one skilled in the art. Where specific details (e.g., circuits) are set forth in order to describe example embodiments, it should be apparent to one skilled in the art that embodiments can be practiced without, or with variation of, these specific details. The description is thus to be regarded as illustrative instead of limiting.

The term “coupled” may be used herein to refer to any type of relationship, direct or indirect, between the components in question, and may apply to electrical, mechanical, fluid, optical, electromagnetic, electromechanical or other connections. In addition, the terms “first”, “second”, etc. may be used herein only to facilitate discussion, and carry no particular temporal or chronological significance unless otherwise indicated.

Those skilled in the art will appreciate from the foregoing description that the broad techniques of the embodiments can be implemented in a variety of forms. Therefore, while the embodiments have been described in connection with particular examples thereof, the true scope of the embodiments should not be so limited since other modifications will become apparent to the skilled practitioner upon a study of the drawings, specification, and following claims.

CLAIMS

We claim:

1. An error-protected computing system comprising:

a memory structure containing a memory region;

a bus coupled to the memory structure; and

a memory controller apparatus coupled to the bus, the memory controller apparatus including:

a read monitor to detect that a read operation is directed to the memory region while the memory region is in a poisoned state, and

a state manager to clear the poisoned state if volatile data stored in the memory region does not correspond to a known data pattern and maintain the memory region in the poisoned state if the volatile data stored in the memory region corresponds to the known data pattern.

2. The system of claim 1, wherein the memory controller apparatus further includes:

a write monitor to detect an error associated with a write operation directed to the memory region, wherein the state manager is to set the poisoned state for the volatile memory region in response to the error; and

a substitution manager to write the known data pattern to the memory region.

3. The system of any one of claims 1 or 2, further including a control-status register and a pattern manager to generate a counter value and store the counter value to the control-status register, wherein the counter value is the known data pattern.

4. The system of claim 3, wherein the control-status register is error correction code protected.

5. The system of claim 3, wherein the pattern manager is to increment the counter value in response to a power cycle.

6. The system of claim 1, wherein the memory structure is a non-volatile memory structure.

7. A memory controller apparatus comprising:

a read monitor to detect that a read operation is directed to a memory region while the memory region is in a poisoned state; and

a state manager to clear the poisoned state if volatile data stored in the memory region does not correspond to a known data pattern and maintain the memory region in the poisoned state if the volatile data stored in the memory region corresponds to the known data pattern.

8. The apparatus of claim 7, further including:

a write monitor to detect an error associated with a write operation directed to the memory region, wherein the state manager is to set the poisoned state for the volatile memory region in response to the error; and

a substitution manager to write the known data pattern to the memory region.

9. The apparatus of any one of claims 7 or 8, further including a pattern manager to generate a counter value and store the counter value to a control-status register, wherein the counter value is the known data pattern.

10. The apparatus of claim 9, wherein the control-status register is to be error correction code protected.

11. The apparatus of claim 9, wherein the pattern manager is to increment the counter value in response to a power cycle.

12. The apparatus of claim 7, wherein the read operation is to be directed to a non-volatile memory structure containing the memory region.

13. A method of operating a memory controller apparatus, comprising:

detecting that a read operation is directed to a memory region while the memory region is in a poisoned state;

clearing the poisoned state if volatile data stored in the memory region does not correspond to a known data pattern; and

maintaining the memory region in the poisoned state if the volatile data stored in the memory region corresponds to the known data pattern.

14. The method of claim 13, further including:

detecting an error associated with a write operation directed to the memory region;

setting the poisoned state for the volatile memory region in response to the error; and

writing the known data pattern to the memory region.

15. The method of any one of claims 13 or 14, further including:

generating a counter value; and

storing the counter value to a control-status register, wherein the counter value is the known data pattern.

16. The method of claim 15, wherein the control-status register is error correction code protected.

17. The method of claim 15, further including increment the counter value in response to a power cycle.

18. The method of claim 13, wherein the read operation is directed to a non-volatile memory structure containing the memory region.

19. At least one non-transitory computer readable storage medium comprising a set of instructions, which when executed by a computing system, cause the computing system to:

detect that a read operation is directed to a memory region while the memory region is in a poisoned state;

clear the poisoned state if volatile data stored in the memory region does not correspond to a known data pattern; and

maintain the memory region in the poisoned state if the volatile data stored in the memory region corresponds to the known data pattern.

20. The at least one non-transitory computer readable storage medium of claim 19, wherein the instructions, when executed, cause the computing device to:

detect an error associated with a write operation directed to the memory region;

set the poisoned state for the volatile memory region in response to the error; and

write the known data pattern to the memory region.

21. The at least one non-transitory computer readable storage medium of any one of claims 19 or 20, wherein the instructions, when executed, cause the computing device to:

generate a counter value; and

store the counter value to a control-status register, wherein the counter value is the known data pattern.

22. The at least one non-transitory computer readable storage medium of claim 21, wherein the control-status register is to be error correction code protected.

23. The at least one non-transitory computer readable storage medium of claim 21, wherein the instructions, when executed, cause the computing device to increment the counter value in response to a power cycle.

24. The at least one non-transitory computer readable storage medium of claim 19, wherein the read operation is to be directed to a non-volatile memory structure containing the memory region.

25. A memory controller apparatus comprising means for performing the method of any one of claims 13 or 14.

AMENDED CLAIMS
received by the International Bureau on 04 January 2017 (04.01.17)

CLAIMS

We claim:

1. An error-protected computing system comprising:

a memory structure containing a memory region;

a bus coupled to the memory structure; and

a memory controller apparatus coupled to the bus, the memory controller apparatus including:

a read monitor to detect that a read operation is directed to the memory region while the memory region is in a poisoned state, and

a state manager to clear the poisoned state if volatile data stored in the memory region does not correspond to a known data pattern and maintain the memory region in the poisoned state if the volatile data stored in the memory region corresponds to the known data pattern.

2. The system of claim 1, wherein the memory controller apparatus further includes:

a write monitor to detect an error associated with a write operation directed to the memory region, wherein the state manager is to set the poisoned state for the memory region in response to the error; and

a substitution manager to write the known data pattern to the memory region.

3. The system of any one of claims 1 or 2, further including a control-status register and a pattern manager to generate a counter value and store the counter value to the control-status register, wherein the counter value is the known data pattern.

4. The system of claim 3, wherein the control-status register is error correction code protected.

5. The system of claim 3, wherein the pattern manager is to increment the counter value in response to a power cycle.

6. The system of claim 1, wherein the memory structure is a non-volatile memory structure.

7. A memory controller apparatus comprising:

a read monitor to detect that a read operation is directed to a memory region while the memory region is in a poisoned state; and

a state manager to clear the poisoned state if volatile data stored in the memory region does not correspond to a known data pattern and maintain the memory region in the poisoned state if the volatile data stored in the memory region corresponds to the known data pattern.

8. The apparatus of claim 7, further including:

a write monitor to detect an error associated with a write operation directed to the memory region, wherein the state manager is to set the poisoned state for the memory region in response to the error; and

a substitution manager to write the known data pattern to the memory region.

9. The apparatus of any one of claims 7 or 8, further including a pattern manager to generate a counter value and store the counter value to a control-status register, wherein the counter value is the known data pattern.

10. The apparatus of claim 9, wherein the control-status register is to be error correction code protected.

11. The apparatus of claim 9, wherein the pattern manager is to increment the counter value in response to a power cycle.

12. The apparatus of claim 7, wherein the read operation is to be directed to a non-volatile memory structure containing the memory region.

13. A method of operating a memory controller apparatus, comprising:

detecting that a read operation is directed to a memory region while the memory region is in a poisoned state;

clearing the poisoned state if volatile data stored in the memory region does not correspond to a known data pattern; and

maintaining the memory region in the poisoned state if the volatile data stored in the memory region corresponds to the known data pattern.

14. The method of claim 13, further including:

detecting an error associated with a write operation directed to the memory region;

setting the poisoned state for the memory region in response to the error; and

writing the known data pattern to the memory region.

15. The method of any one of claims 13 or 14, further including:

generating a counter value; and

storing the counter value to a control-status register, wherein the counter value is the known data pattern.

16. The method of claim 15, wherein the control-status register is error correction code protected.

17. The method of claim 15, further including incrementing the counter value in response to a power cycle.

18. The method of claim 13, wherein the read operation is directed to a non-volatile memory structure containing the memory region.

19. At least one non-transitory computer readable storage medium comprising a set of instructions, which when executed by a computing system, cause the computing system to:

detect that a read operation is directed to a memory region while the memory region is in a poisoned state;

clear the poisoned state if volatile data stored in the memory region does not correspond to a known data pattern; and

maintain the memory region in the poisoned state if the volatile data stored in the memory region corresponds to the known data pattern.

20. The at least one non-transitory computer readable storage medium of claim 19, wherein the instructions, when executed, cause the computing device to:

detect an error associated with a write operation directed to the memory region;

set the poisoned state for the memory region in response to the error; and

write the known data pattern to the memory region.

21. The at least one non-transitory computer readable storage medium of any one of claims 19 or 20, wherein the instructions, when executed, cause the computing device to:

generate a counter value; and

store the counter value to a control-status register, wherein the counter value is the known data pattern.

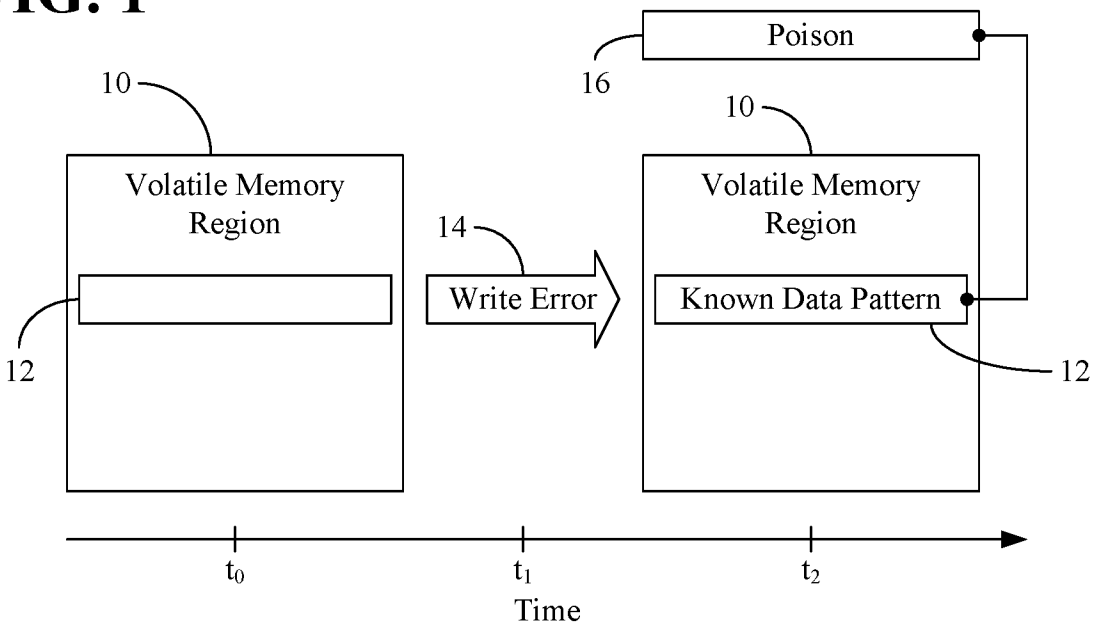
22. The at least one non-transitory computer readable storage medium of claim 21, wherein the control-status register is to be error correction code protected.

23. The at least one non-transitory computer readable storage medium of claim 21, wherein the instructions, when executed, cause the computing device to increment the counter value in response to a power cycle.

24. The at least one non-transitory computer readable storage medium of claim 19, wherein the read operation is to be directed to a non-volatile memory structure containing the memory region.

25. A memory controller apparatus comprising means for performing the method of any one of claims 13 or 14.

FIG. 1



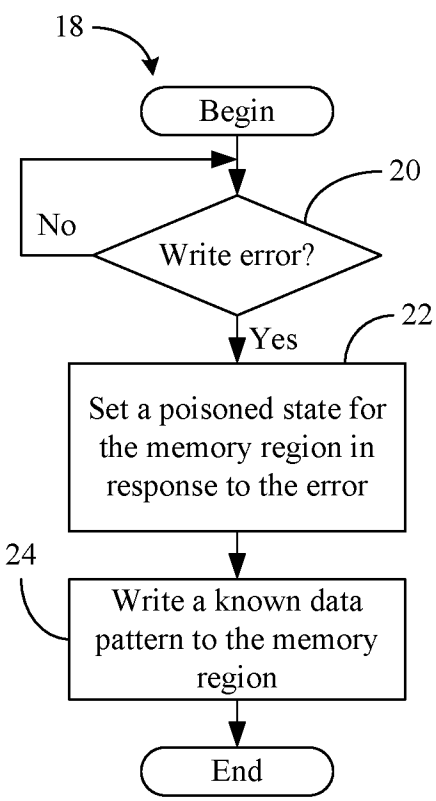


FIG. 2A

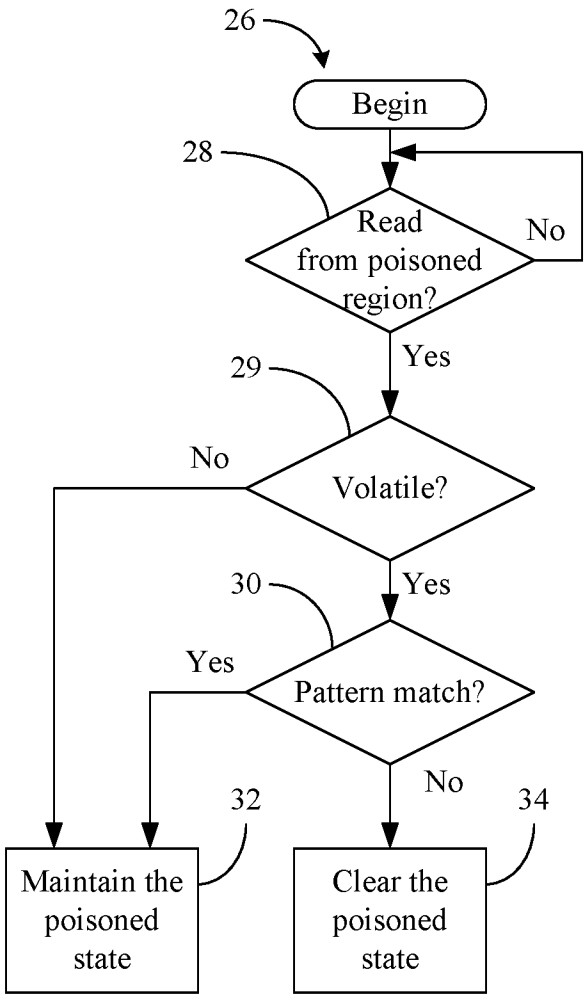


FIG. 2B

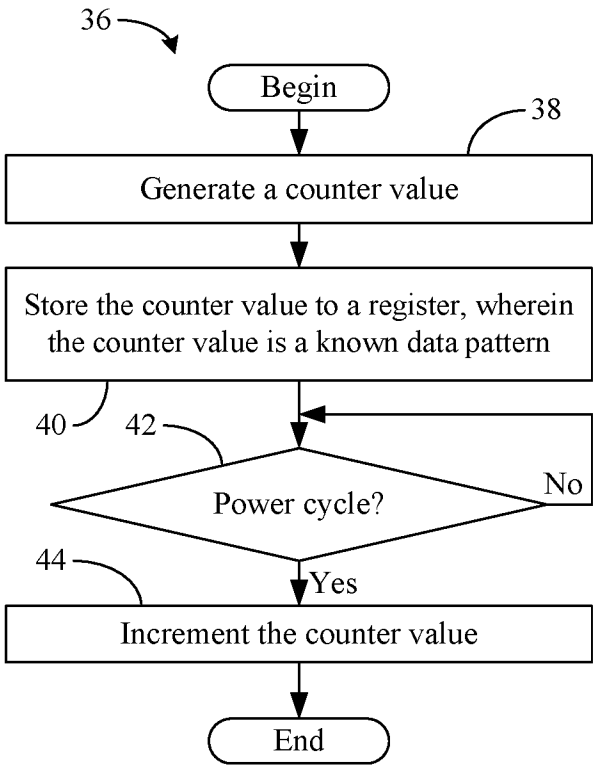


FIG. 3

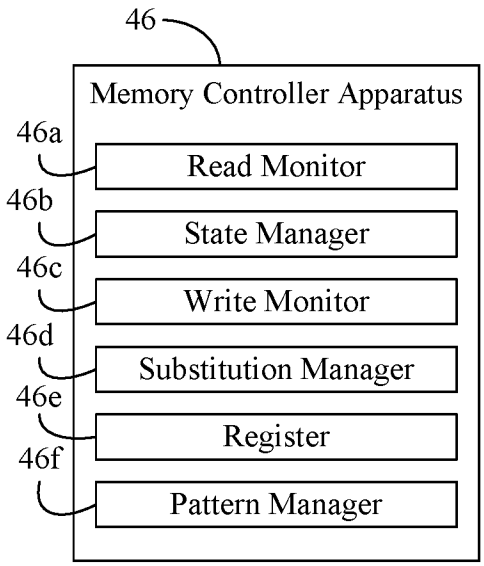


FIG. 4

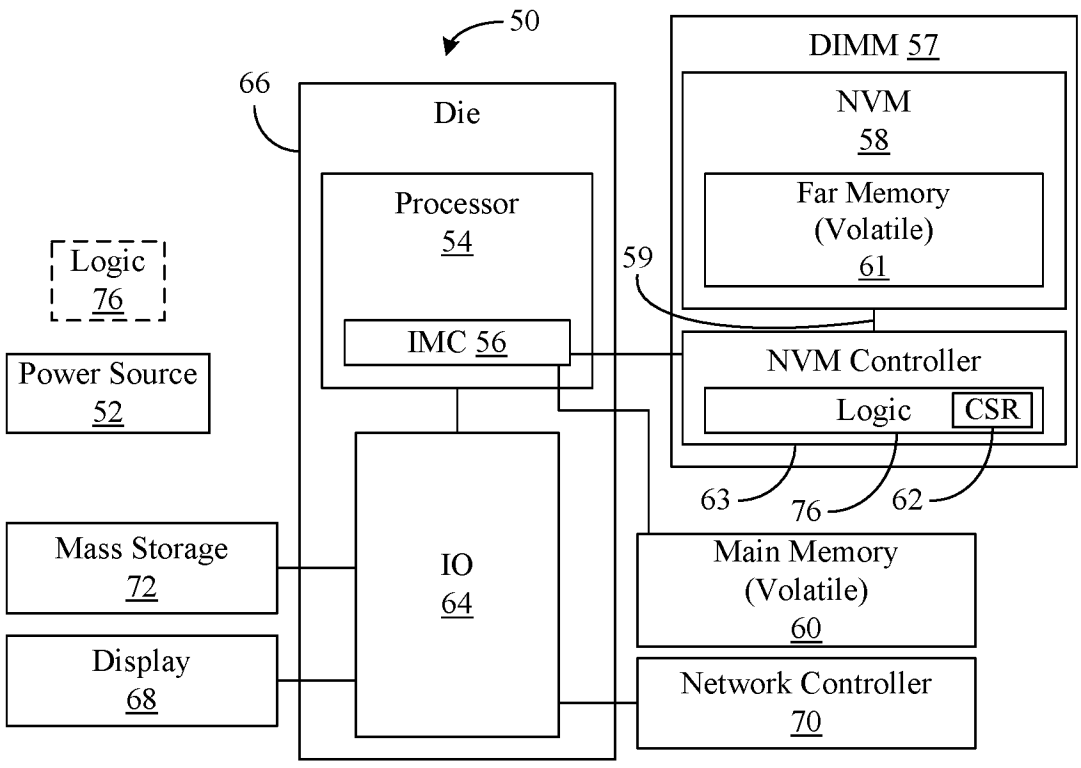


FIG. 5

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2016/041975

A. CLASSIFICATION OF SUBJECT MATTER

G06F 11/07(2006.01)i, G06F 3/06(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F 11/07; G06F 11/14; G06F 11/08; G06F 11/00; G06F 3/06

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Korean utility models and applications for utility models
Japanese utility models and applications for utility models

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
eKOMPASS(KIPO internal) & Keywords: memory, error, poison, pattern, volatile, corrupt, region, location, clear, and similar terms.

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 8,245,087 B2 (DENNIS C. ABTS et al.) 14 August 2012 See column 4, lines 33-34; column 7, lines 7-9; claim 28; and figure 1.	1-25
A	US 2012-0023364 A1 (ROBERT C. SWANSON et al.) 26 January 2012 See paragraphs [0018]-[0027] and figure 1.	1-25
A	US 2014-0136915 A1 (ELWHA LLC, A LIMITED LIABILITY CORPORATION OF THE STATE OF DELAWARE) 15 May 2014 See paragraphs [0067]-[0069] and figures 3A-3B.	1-25
A	US 2014-0157054 A1 (HEWLETT-PACKARD DEVELOPMENT COMPANY, L.P.) 05 June 2014 See paragraphs [0010]-[0012] and figure 1.	1-25
A	WO 2003-001380 A2 (INTEL CORPORATION) 03 January 2003 See page 6, line 2 - page 8, line 18 and figure 2.	1-25

☐ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents,such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
21 October 2016 (21.10.2016)	**21 October 2016 (21.10.2016)**

Name and mailing address of the ISA/KR
International Application Division
Korean Intellectual Property Office
189 Cheongsa-ro, Seo-gu, Daejeon, 35208, Republic of Korea
Facsimile No. +82-42-481-8578

Authorized officer
NHO, Ji Myong
Telephone No. +82-42-481-8528

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.
PCT/US2016/041975

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 8245087 B2	14/08/2012	US 2009-0177932 A1	09/07/2009
		US 2009-0287889 A1	19/11/2009
		US 2010-0185897 A1	22/07/2010
		US 8065573 B2	22/11/2011
		US 8464007 B2	11/06/2013
US 2012-0023364 A1	26/01/2012	AU 2011-286271 A1	07/02/2013
		AU 2011-286271 B2	07/08/2014
		CN 103140841 A	05/06/2013
		CN 103140841 B	20/01/2016
		EP 2598997 A2	05/06/2013
		EP 2598997 A4	05/08/2015
		JP 2013-535738 A	12/09/2013
		JP 5512892 B2	04/06/2014
		KR 10-1473119 B1	15/12/2014
		KR 10-2013-0033416 A	03/04/2013
		TW 201212037 A	16/03/2012
		US 9063836 B2	23/06/2015
		WO 2012-018529 A2	09/02/2012
		WO 2012-018529 A3	24/05/2012
US 2014-0136915 A1	15/05/2014	US 8996951 B2	31/03/2015
US 2014-0157054 A1	05/06/2014	US 8966348 B2	24/02/2015
WO 03-001380 A2	03/01/2003	TW I259355 B	01/08/2006
		US 2002-0199152 A1	26/12/2002
		US 7168026 B2	23/01/2007
		WO 03-001380 A3	06/11/2003