



- (51) **International Patent Classification:**
G06F 12/02 (2006.01) *G06F 3/06* (2006.01)
- (21) **International Application Number:**
PCT/FI201 1/05 1168
- (22) **International Filing Date:**
29 December 2011 (29.12.2011)
- (25) **Filing Language:** English
- (26) **Publication Language:** English
- (71) **Applicant** (for all designated States except US): **NOKIA CORPORATION** [FI/FI]; Keilalahdentie 4, FI-02150 Espoo (FI).
- (72) **Inventor; and**
- (75) **Inventor/Applicant** (for US only): **MYLLY, Kimmo** [FI/FI]; Liisanniityntie 20, FI-33480 Ylqjarvi (FI).
- (74) **Agent:** **ESPATENT OY**; Kaivokatu 10 D, FI-00100 Helsinki (FI).
- (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, 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, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, 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, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, 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, ML, MR, NE, SN, TD, TG).

Published:

— with international search report (Art. 21(3))

- (54) **Title:** METHOD FOR ERASING DATA ENTITY IN MEMORY MODULE

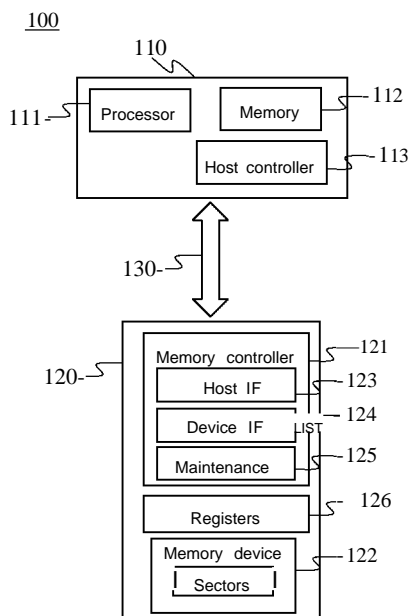


Fig. 1

(57) **Abstract:** A method including storing a data entity using at least two sectors of a memory device, the at least two sectors associated to the same data entity, and maintaining, at a memory controller, context information of the data entity comprising a pointer to at least one of the at least two sectors of the memory device. The method further includes erasing the at least two sectors of the memory device using the context information.

METHOD FOR ERASING DATA ENTITY IN MEMORY MODULE

TECHNICAL FIELD

5

The present invention relates generally to memory modules. The invention relates particularly, though not exclusively, to erasing a data entity in the memory module.

BACKGROUND ART

10

Today's memory module interfaces may hide the used memory technology. The host software does not have to adapt to various memory technologies and architectures. However, this poses a problem that memory modules are not able to efficiently handle the wear leveling and erasing functions. For example, if the memory card is once written full, it will internally appear as full even if all the files that were stored in it will be deleted. The reason for this is that in a normal file system implementation, data will just be written to the memory card and new data will just overwrite the old one. Files are deleted only from the file allocation table by writing the reserved file/cluster entries to be non-reserved.

20

In a managed NAND memory subsystems a memory controller takes care of flash management functions like bad block management and wear leveling. The memory controller is not aware of file system level details of data but only managing reading/writing sectors of data without understanding the possible relationship between different sectors (e.g. if certain sectors belong to same or different file).

25

30

When an external host deletes a file the host marks the corresponding allocation in file allocation table (FAT) as free. A sophisticated host operating system and/or file system may also perform a so-called trimming function to sectors which the file covered. Depending on how fragmented the file allocation table (FAT) is, several commands may be needed to trim all sectors belonging to a file.

SUMMARY

According to a first example aspect of the invention there is provided a method comprising:

- 5 storing a data entity using at least two sectors of a memory device, the at least two sectors associated to the same data entity;
maintaining, at a memory controller, context information of the data entity comprising a pointer to at least one of the at least two sectors of the memory device; and
10 erasing the at least two sectors of the memory device using the context information.

In an embodiment, the method may further comprise:

- defining, at the memory controller, a signal indicative of erasing at least one
15 of the at least two sectors from the memory device.

In an embodiment, the method may further comprise:

- deleting, by the host, a data entity entry from a file allocation table in response to a delete signal from an external host application; and
20 generating the signal indicative of erasing at least one of the at least two sectors from the memory device in response to the deletion of the data entity entry from the file allocation table.

- The pointer may comprise a start address indicating a logical address of a first
25 sector of the memory device and a start address indicating a logical address of a second sector of the memory device. The pointer may also comprise a start address indicating a physical address of a first sector of the memory device and a start address indicating a physical address of a second sector of the memory device.

30

A context identifier may be defined for the context information. Furthermore, a group number or an associative command may be defined for the context information.

In an embodiment, the method may further comprise erasing the at least two sectors of the memory device in response to the signal indicative of erasing at least one of the at least two sectors, associated with the context identifier.

5

In an embodiment, the method may further comprise erasing, by an erasing function, the at least two sectors of the memory device using at least one of the following commands:

- ERASE;
- 10 - TRIM;
- SecureTRIM;
- SecureERASE;
- DISCARD;
- Sanitize; and
- 15 - Unmap (UFS).

In an embodiment, the method may further comprise generating, by the erasing function, a first signal indicative of erasing the first sector and a second signal indicative of erasing the second sector. The sector may comprise at least one of

20 the following:

- an erase block;
- a write block;
- a page;
- a super page (set of pages); and
- 25 - a sector (e.g. of a hard drive).

According to a second example aspect of the invention there is provided a memory module, comprising:

- at least one memory device;
- 30 a memory controller having an interface to an external host and to the at least one memory device;
- the memory device including computer program code, the computer program

code being configured to, with the memory controller, cause the memory module at least to:

store a data entity using at least two sectors of a memory device, the at least two sectors associated to the same data entity;

5 maintain, at a memory controller, context information of the data entity comprising a pointer to at least one of the at least two sectors of the memory device; and

erase the at least two sectors of the memory device using the context information.

10

In an embodiment, the computer program code being further configured to, with the memory controller, cause the memory module at least to:

define, at the memory controller, a signal indicative of erasing at least one of the at least two sectors from the memory device.

15

A data entity entry may be deleted, by the host, from a file allocation table in response to a delete signal from an external host application; and the signal indicative of erasing at least one of the at least two sectors from the memory device may be generated in response to the deletion of the data entity entry from the file allocation table.

20

In an embodiment, the sector comprises at least one of the following:

- an erase block;
- a write block;
- 25 - a page;
- a super page (set of pages); and
- a sector (e.g. of a hard drive).

25

In an embodiment, the memory module comprises at least one of the following:

30

- an Embedded MultiMediaCard (eMMC);
- a memory card (SD);
- an Advanced Technology Attachment (ATA); and
- a Universal Flash Storage (UFS).

According to a third example aspect of the invention there is provided a computer program embodied on a computer readable medium comprising computer executable program code which, when executed by at least one controller of a memory module, causes the memory module to:

store a data entity using at least two sectors of a memory device, the at least two sectors associated to the same data entity;

maintain, at a memory controller, context information of the data entity comprising a pointer to at least one of the at least two sectors of the memory device; and

erase the at least two sectors of the memory device using the context information.

Any foregoing memory medium may comprise a digital data storage such as a data disc or diskette, optical storage, magnetic storage, holographic storage, opto-magnetic storage, solid state memory, phase-change memory, resistive random access memory, magnetic random access memory, solid-electrolyte memory, ferroelectric random access memory, organic memory or polymer memory. The memory medium may be formed into a device without other substantial functions than storing memory or it may be formed as part of a device with other functions, including but not limited to a memory of a computer, a chip set, and a sub assembly of an electronic device.

Different non-binding example aspects and embodiments of the present invention have been illustrated in the foregoing. The above embodiments are used merely to explain selected aspects or steps that may be utilized in implementations of the present invention. Some embodiments may be presented only with reference to certain example aspects of the invention. It should be appreciated that corresponding embodiments may apply to other example aspects as well.

BRIEF DESCRIPTION OF THE DRAWINGS

The invention will be described, by way of example only, with reference to the accompanying drawings, in which:

5

Fig. 1 shows some details of the system architecture in which various embodiments of the invention may be applied;

Fig. 2 shows some details of the memory device, in which various embodiments of the invention may be applied;

10 Fig. 3 presents an example block diagram of a user apparatus in which various embodiments of the invention may be applied; and

Fig. 4 shows a flow diagram showing operations in a memory module in accordance with an example embodiment of the invention.

15

DETAILED DESCRIPTION

In the following description, like numbers denote like elements.

20 Fig. 1 shows some details of the system 100, in which various embodiments of the invention may be applied. The system comprises an external host 110, such as a user apparatus, a user equipment (UE), and a memory module 120. Additionally the system comprises an interface 130 for communicating data, control signals and clock signal between the host 110 and the memory module 120.

25

The memory module 120 may comprise for example a memory card, a MultiMediaCard (MMC), a secure digital (SD) card or an embedded memory. The memory module 120 may comprise a memory controller 121 and at least one memory device 122. The memory device 122 may comprise memory blocks with
30 multiple sectors for storing data. The memory controller 120 may comprise a host interface controller 123 for facilitating communications with the host device 110, a memory interface controller 124 for facilitating communications and operations (e.g., read, write or erase operations) with the memory device 122 and a memory

maintenance controller 125. In an embodiment, the controllers 121, 123-125 may be logical circuits.

In an embodiment, when the memory module 120 is connected to a host (electronic) device 110, the host controller 113 may provide a command signal over the interface 130 to the host interface controller 123. In the memory device 122, a data entity, such as a file, is stored using at least two sectors of the memory device 122.

10 In an embodiment, a sector may comprise at least one of the following:

- an erase block (NAND);
- a write block (eMMC);
- a page (NAND);
- a super page (set of pages); and
- 15 - a sector (e.g. of a hard drive).

To better differentiate between large sequential operations and small random operations, and to improve multitasking support, contexts can be associated with groups of read or write commands. Associating a group of commands with a single context allows the device to optimize handling of the data. A faster solution to erase fragmented sectors is also needed.

A context can be seen as an active session, configured for a specific read/write pattern (e.g. sequential in some granularity). Multiple read or write commands are associated with this context to create some logical association between them, to allow device to optimize performance. For example, a large sequential write pattern may have better performance by allowing the device to improve internal locality and reduce some overheads (e.g. if some large unit of data is allowed to be affected by a power failure as a whole while it is being written to, all of the commands that fill this unit can work faster because they can reduce the overhead usually required to protect each write individually in case of a power failure).

Device may support one or more concurrent contexts, identified by a context identifier (context ID). To use a context, an available context ID shall be picked. Then, it shall be initialized by writing to a configuration register. Then, data can be read/written associated to the context by specifying the context ID in command
5 before sending the read/write command. When the context is no longer used, the configuration register should be updated to close the context ID. A context shall be closed prior to re-configuring it for another configuration/use.

10 In an embodiment, a memory device location, such as a sector, may be indicated using a pointer. The pointer may comprise for example a logical address or a physical address of the memory device location.

At the memory controller 121 of the memory module 120, context information may be maintained of the data entity. The context information may comprise logical
15 addresses of the at least two sectors of the memory device 122. At the memory controller 121, a signal may be defined indicative of erasing the at least two sectors from the memory device 122. The at least two sectors of the memory device 122 may be then erased using the context information.

20 As described herein, the signal received from the host 110 may comprise a context ID for a data entity to be erased from the memory device 122 comprised by at least two sectors. The context ID identifies the context information comprising a pointer. The pointer may comprise a start address indicating a logical address of a first sector of the memory device 122 and a start address indicating a
25 logical address of a second sector of the memory device 122. Erasing of the data entity may comprise erasing all the sectors comprising data relating to the data entity or at least one sector comprising data relating to the data entity.

30 In an embodiment, erasing may be implemented using at least one of the following functions:

- ERASE (eMMC);
- TRIM (eMMC/ATA);
- SecureTRIM/SecureERASE (eMMC);

- DISCARD (eMMC);
- Sanitize (eMMC); and
- Unmap (UFS).

5 In an embodiment, the erasable unit may be defined as an erase group. The erase group is measured in write blocks, which are the basic writable units. The size of the erase group is a device specific parameter. The content of an explicitly erased memory range shall be '0' or '1' depending on different memory technology. Once the ERASE command completes successfully, the mapped device address range
10 that was erased shall behave as if it was overwritten with all '0' or all '1' depending on the different memory technology. The impact of the ERASE command should be simply moving the mapped host address range to the unmapped host address range. The host can erase a contiguous range of Erase Groups. Starting the erase process is a three steps sequence. First the host defines the start address of the
15 range using the ERASE_GROUP_START command, next it defines the last address of the range using the ERASE_GROUP_END command and finally it starts the erase process by issuing the ERASE command with argument bits set to zero. The address field in the erase commands is an erase group address, in byte units for densities up to 2GB, and in sector units for densities greater than 2GB.

20

In an embodiment, a trimming function TRIM is similar to the ERASE function described above. The function applies the erase operation to write blocks instead of erase groups. The trimming function TRIM allows the host to identify data that is no longer required and to erase the data if necessary during background erase
25 events. The contents of a write block where the trim function has been applied may be '0' or '1' depending on different memory technology.

Once the trimming function completes successfully, the mapped device address range that was trimmed shall behave as if it was overwritten with all '0' or all '1'
30 depending on the different memory technology. The impact of the trim command should be simply moving the mapped host address range to the unmapped host address range. Completing the TRIM process is a three steps sequence. First the host may define the start address of the range using the ERASE_GROUP_START

command, next it defines the last address of the range using the ERASE_GROUP_END command and finally it starts the erase process by issuing the ERASE command with argument bit 0 set to one and the remainder of the arguments set to zero.

5

In an embodiment, a sanitize function may be used, to remove data from the device. The use of the sanitize operation requires the device to physically remove data from the unmapped user address space. After the sanitize operation is completed, no data should exist in the unmapped host address space. Since the
10 region being operated on is not accessible by the host, applications requiring this feature must work with individual device manufacturers to ensure this operation is performing properly and to understand the impact on device reliability.

In an embodiment, a discard function may be used for erasing. The DISCARD is
15 similar operation to TRIM. The discard function allows the host to identify data that is no longer required so that the device can erase the data if necessary during background erase events. The contents of a write block where the discard function has been applied shall be 'don't care'. After discard operation, the original data may be remained partially or fully depend on device. The device will decide the
20 contents of discarded write block. The distinction between DISCARD and TRIM is the device behavior, where the device is not required to guarantee that host would not retrieve the original data from one or more logical block address's that were marked for erase when a read operation is directed. The TRIM shall response with either '0' or '1', depending on different memory technology. Completing the discard
25 process is a three steps sequence. First the host defines the start address of the range using the ERASE_GROUP_START command, next it defines the last address of the range using the ERASE_GROUP_END command and finally it starts the erase process by issuing the ERASE command with argument bit 0 and bit 1 set to one and the remainder of the arguments set to zero. In the case of the
30 discard operation the commands may identify the addresses of write blocks rather than erase groups.

In an embodiment, a secure erase function may be used. A secureERASE command indicating a secure purge may be performed on an erase group. The secure erasing should be done not only to the data in this erase group but also on any copies of this data in separate erase groups. This command is executed immediately, and the memory device will not return control to the host until all necessary erase groups are purged. One erase group is the minimum block of memory that can be erased on a particular NAND flash, for example.

In an embodiment, a secure trim function may be used. Similar to the secureERASE, the secureTRIM command operates on write blocks instead of erase groups. To handle this properly, the operation may be divided into two steps. The first step marks blocks for secure purge that can be done to multiple sets of blocks before the second step is called. The second step is an erase with a separate bit flag sequence that performs all the requested secure trims.

Erasing the sectors of the memory device 122 may be utilized by the host 110, for example, according to an embodiment of the present invention. As described herein, the memory maintenance controller 125 may manage a memory maintenance comprising possible erasing sectors or blocks. After deleting the data item in file system level from the file allocation table (FAT), the host 110 may be further configured to perform collecting together all sectors marked as deleted and possibly logically/physically erasing the plurality of sectors marked as deleted. The host 110 may also perform garbage collection and memory wear leveling of the memory device 122 after deleting. Furthermore, the memory maintenance controller 125 may be further configured to perform collecting together all sectors marked to be erased, based on instructions from the host, and possibly logically/physically erasing the plurality of sectors. Based on the instructions from the host 110, the memory maintenance controller 125 may perform memory wear leveling of the memory device 122 after erasing.

In an embodiment, the memory controller 121, the host interface controller 123, the memory device controller 124, the memory maintenance controller 125, and the memory device 122 may be implemented as a software module, a hardware

module or a combination thereof. Furthermore, each of the modules 121-125 may be implemented as a separate module/block or can be combined with any other standard module/block of the memory module 120, or it can be split into several blocks according to their functionality. All or selected modules/blocks of the
5 memory module 120 may be implemented using one integrated circuit.

In an embodiment, registers 126 shown in Fig. 1 are configured to store register information, such as earlier described configuration register. The registers 126 may also be located in the memory device 122, for example. The registers may be
10 accessed by the memory controller 121.

Fig. 2 shows some details of the memory device 122, in which various embodiments of the invention may be applied.

15 In an embodiment, in a file system implementation, data will be written to the memory device, for example, and new data will just overwrite the old one. Files are deleted only from a file allocation table (FAT) by writing the reserved file/cluster/sector entries to be non-reserved. Erasing data from the memory device is a separate function.

20

When writing data to the flash memory via a logical interface, the host typically assigns unique logical addresses to sectors, clusters or other units of data within a continuous virtual address space of the memory system. The host typically maintains a file system and allocates file data to logical clusters, where the cluster
25 size is typically fixed. A flash device is divided into plural logical sectors and the host allocates space within the clusters comprising of a plurality of logical sectors. A cluster is a subdivision of logical addresses and a cluster map may be designated as a file allocation table (FAT). The file allocation table (FAT) is normally stored on the storage device itself.

30

A continuous logical address space is large enough to provide addresses for all the data that may be stored in the memory device 122. The host address space is typically divided into increments of sectors or clusters of data. Each cluster may be

designed in a given host system to contain a number of sectors of data, somewhere between 4 and 64 sectors being typical. A standard sector contains 512 or 4096 bytes of data, for example.

- 5 Fig. 2 shows an example of how memory device 122 is typically partitioned. Storage device 122 may include a plurality of partitions, and in Fig. 2 a single partition 210 is shown for simplicity. The partition 210 includes a system area 220 and a data area 230. The system area 220 has various segments, for example, the master boot record ("MBR") area, partition boot record ("PBR") area, FAT1
10 area, FAT2 area and root directory area.

MBR area stores overall partition information, and if the media is a bootable device, then MBR includes instructions to jump from MBR area to PBR area. MBR area also includes hidden area, which is reserved space between MBR area and
15 PBR area.

PBR area includes partition/boot information for a partition. For example, PBR area includes information for the type of the file allocation table (FAT) (for example, 12/16/32 bits); a label (i.e. name of the drive), size of the drive; cluster
20 size (i.e. the number of sectors per allocation unit); number of file allocation table (FAT) areas (2 FAT areas FAT1 and FAT2 shown in Fig. 2) and the number of sectors per file allocation table (FAT). File allocation table (FAT) areas contain sector/cluster information for each file. For example, for FAT-12, each entry in file allocation table (FAT) area contains 12 bits and there are a total of 4096 entries.

25

Root directory contains entries for each file. Each directory entry includes a certain number of bytes for file name or directory (for example, 8 bytes); a number of bytes for extension (for example, 3 bytes), a number of bytes (for example, 1 byte) for file attributes (for example, if a file is read only, hidden, system file, volume
30 label, directory or modified); a number of bytes indicating the time and date when a file was created; a certain number of bytes (for example 2) for a starting sector/cluster; and a certain number of bytes (for example 4) indicating the file length. The number of bytes for the starting sector/cluster points to the first

sector/cluster in the file allocation table (FAT) and the last sector/cluster may be indicated by 0xFFF or 0xFFFF, for example.

5 In order to write a file name in the directory, the host system finds free sector/cluster space in FAT1 area and data is written in the data area 230. Both FAT1 and FAT2 entries are then updated. The directory entry i.e., the date/time/starting sector/cluster/file length is also updated.

10 In an embodiment, a data entity, for example a file, is stored to the memory device 122 in a partition 210. The data entity is stored in two sectors 231, 232. A context configuration may be defined for the data entity, identified by a context ID. Configuration may be done by writing to the configuration register of the specific context ID required. Then, all read commands or write commands that are associated with this context ID shall be sent with the ID. When the context is no longer needed, it should be closed by writing a zero byte to the configuration register. The configuration registers are an array of 15 byte registers, for example, one for each context (except the fixed, pre-defined #0). Different rules may be defined also for new data to be later on added to the same file/context ID. The ID index of the configuration register can be re-opened after closing it and the ID index can always create new entity. However, after data has been successfully written to an ID index and the context ID has been closed, the data is logically grouped together. There is no exposure of this grouping to the host but the memory controller has means for figuring this out from spare pages of NAND, headers of pages or stored data, for example.

25

In an embodiment, a host decides to delete a data entity, such as a file, in response to receiving a deleting signal from the host application, the host marks the corresponding allocation in the file allocation table (FAT) as free. Furthermore, erasing the data entity located in the sectors 231, 232 of the memory device may also be desired. The sectors 231, 232 may be erased using separate trimming functions for each sector 231, 232.

30

In an embodiment, more efficient way to erase a file comprised by a plurality of sectors is introduced by reducing overhead of sending more than one trimming commands from the host to the memory device. The memory controller may be informed that certain parts of data entity (file) belong together and a specific single
5 trimming function may be defined to apply to such data as a whole. Such function is independent of the number/address of the sectors covered by the file. The function of context ID may be used for erasing the sectors 231 , 232 comprising the data entity, such as the file. At least following options may exist.

- 10 First option is to introduce a specific context ID erase function. Either separate specific commands or a special parameter in existing command may be used. Furthermore, register setting may be used. The first option may utilize existing erase functions as is. Such erase functions may also be used for erasing single sectors of certain data entity, ID or file. Erase functions may comprise commands,
15 such as Erase, SecureErase, TRIM, SecureTRIM, or Discard in JEDEC eMMC and UFS memory standards.

Second option is to introduce an address of a context ID to be used by an erase function. The erase function may use a start address or any address of the context
20 ID and automatically erase all sectors belonging to the same context ID.

Third option is to introduce hierarchical erasing features. A context may comprise a superset of other contexts corresponding to a context ID providing a superset of other context ID's. Such hierarchy may enable that data written to context ID1
25 could be e.g. a database and data written to context ID2 could be an item in the ID1 database. Thus a single trimming command to an address of ID2 may invalidate and erase the item from the database, no matter how fragmented the locations of sectors belonging to the item are. Furthermore, a single trimming command to an address of ID1 may invalidate the whole database, also all the
30 items belonging to it but written originally to ID2. There could also be a control register related to the hierarchy of the indexes.

Fig. 3 presents an example block diagram of a user apparatus 300 in which various embodiments of the invention may be applied. This may be a user equipment (UE), user device or apparatus, such as a mobile terminal, a laptop, a tablet, or other communication device.

5

The general structure of the user apparatus 300 comprises a communication interface module 350, a processor 310 coupled to the communication interface module 350, and a memory 320 coupled to the processor 310. The user apparatus further comprises software 330 stored in the memory 320 and operable to be loaded into and executed in the processor 310. The software 330 may comprise one or more software modules and can be in the form of a computer program product. The user apparatus 300 further comprises a user interface controller 360 coupled to the processor 310.

15 The communication interface module 350 implements at least part of the user data radio discussed in connection with various embodiments of the invention. The communication interface module 350 may be, e.g., a radio interface module, such as a WLAN, Bluetooth, GSM/GPRS, CDMA, WCDMA, or LTE (Long Term Evolution) radio module. The communication interface module 350 may be integrated into the user apparatus 300 or into an adapter, card or the like that may be inserted into a suitable slot or port of the user apparatus 300. The communication interface module 350 may support one radio interface technology or a plurality of technologies. Fig. 3 shows one communication interface module 350, but the user apparatus 300 may comprise a plurality of communication interface modules 350.

The processor 310 may be, e.g., a central processing unit (CPU), a microprocessor, a digital signal processor (DSP), a graphics processing unit, or the like. Fig. 3 shows one processor 310, but the user apparatus 300 may comprise a plurality of processors.

The memory 320 may be for example a non-volatile or a volatile memory, such as a read-only memory (ROM), a programmable read-only memory (PROM),

erasable programmable read-only memory (EPROM), a random-access memory (RAM), a flash memory, a data disk, an optical storage, a magnetic storage, a smart card, or the like. The user apparatus 300 may comprise a plurality of memories. The memory 320 may be constructed as a part of the apparatus 300 or
5 it may be inserted into a slot, port, or the like of the user apparatus 300 by a user. The memory 320 may serve the sole purpose of storing data, or it may be constructed as a part of an apparatus serving other purposes, such as processing data.

10 A memory module 340 may be included as a smart card used in the user apparatus 300. The memory module may be for example a non-volatile or a volatile memory, such as a read-only memory (ROM), a programmable read-only memory (PROM), erasable programmable read-only memory (EPROM), a random-access memory (RAM), a flash memory, a data disk, an optical storage, a
15 magnetic storage, a smart card, or the like. The user apparatus 300 may comprise a plurality of memory modules 340. The memory module 340 may be inserted into a slot, port, or the like of the user apparatus 300 by a user. The memory 340 may serve the sole purpose of storing data, or it may be constructed as a part of an apparatus serving other purposes, such as processing data.

20 The user interface controller 360 may comprise circuitry for receiving input from a user of the user apparatus 300, e.g., via a keyboard, graphical user interface shown on the display of the user apparatus 300, speech recognition circuitry, or an accessory device, such as a headset, and for providing output to the user via, e.g.,
25 a graphical user interface or a loudspeaker.

A skilled person appreciates that in addition to the elements shown in Fig. 3, the user apparatus 300 may comprise other elements, such as microphones, displays, as well as additional circuitry such as input/output (I/O) circuitry, memory chips,
30 application-specific integrated circuits (ASIC), processing circuitry for specific purposes such as source coding/decoding circuitry, channel coding/decoding circuitry, ciphering/deciphering circuitry, and the like. Additionally, the user apparatus 300 may comprise a disposable or rechargeable battery (not shown) for

powering the user apparatus 300 when external power if external power supply is not available.

Fig. 4 shows a flow diagram showing operations in a memory module in accordance with an example embodiment of the invention. In step 400, the method is started. In step 410, a data entity is stored using at least two sectors of a memory device, the at least two sectors associated to the same data entity. In step 420, at a memory controller of a memory module, context information of the data entity comprising a pointer to at least one of the at least two sectors of the memory device is maintained. At the memory controller, a signal is defined indicative of erasing the at least two sectors from the memory device in step 430. In step 440, and the at least two sectors of the memory device are erased using the context information. The method ends in step 450.

Various embodiments have been presented. It should be appreciated that in this document, words comprise, include and contain are each used as open-ended expressions with no intended exclusivity.

The foregoing description has provided by way of non-limiting examples of particular implementations and embodiments of the invention a full and informative description of the best mode presently contemplated by the inventors for carrying out the invention. It is however clear to a person skilled in the art that the invention is not restricted to details of the embodiments presented above, but that it can be implemented in other embodiments using equivalent means or in different combinations of embodiments without deviating from the characteristics of the invention.

Furthermore, some of the features of the above-disclosed embodiments of this invention may be used to advantage without the corresponding use of other features. As such, the foregoing description shall be considered as merely illustrative of the principles of the present invention, and not in limitation thereof. Hence, the scope of the invention is only restricted by the appended patent claims.

Claims:

1. A method comprising:

storing a data entity using at least two sectors of a memory device, the at least two sectors associated to the same data entity;

5 maintaining, at a memory controller, context information of the data entity comprising a pointer to at least one of the at least two sectors of the memory device; and

erasing the at least two sectors of the memory device using the context information.

10

2. The method of claim 1, further comprising:

defining, at the memory controller, a signal indicative of erasing at least one of the at least two sectors from the memory device.

15 3. The method of claim 1 or 2, further comprising:

deleting, by a host, a data entity entry from a file allocation table in response to a delete signal from an external host application; and

generating the signal indicative of erasing at least one of the at least two sectors from the memory device in response to the deletion of the data entity entry
20 from the file allocation table.

4. The method of any of claims 1 to 3, wherein the pointer comprises a start address indicating a logical address of a first sector of the at least two sectors of the memory device and a start address indicating a logical address of a second
25 sector of the at least two sectors of the memory device.

5. The method of any of claims 1 to 3, wherein the pointer comprises a start address indicating a physical address of a first sector of the at least two sectors of the memory device and a start address indicating a physical address of a second
30 sector of the at least two sectors of the memory device.

6. The method of any of claims 1 to 5, further comprising:
defining a context identifier for the context information.

7. The method of any of claims 1 to 5, further comprising:
defining a group number for the context information.
- 5 8. The method of any of claims 1 to 5, further comprising:
defining an associative command for the context information.
9. The method of claim 6, further comprising:
erasing the at least two sectors of the memory device in response to the
10 signal indicative of erasing at least one of the at least two sectors, associated with
the context identifier.
10. The method of any of claims 1 to 9, further comprising:
erasing, by an erasing function, the at least two sectors of the memory device
15 using at least one of the following commands:
- ERASE;
 - TRIM;
 - SecureTRIM;
 - SecureERASE;
 - 20 - DISCARD;
 - Sanitize; and
 - Unmap (UFS).
11. The method of claim 10, further comprising:
25 generating, by the erasing function, a first signal indicative of erasing the first
sector and a second signal indicative of erasing the second sector.
12. The method of any of claims 1 to 11, wherein the sector comprises at least
one of the following:
- 30 - an erase block;
- a write block;
- a page;
- a super page; and

- a sector of a hard drive.

13. A memory module, comprising:

at least one memory device;

5 a memory controller having an interface to an external host and to the at least one memory device;

the memory device including computer program code, the computer program code being configured to, with the memory controller, cause the memory module at least to:

10 store a data entity using at least two sectors of a memory device, the at least two sectors associated to the same data entity;

maintain, at a memory controller, context information of the data entity comprising a pointer to at least one of the at least two sectors of the memory device; and

15 erase the at least two sectors of the memory device using the context information.

14. The memory module of claim 13, wherein the computer program code being further configured to, with the memory controller, cause the memory module at least to:

20 define, at the memory controller, a signal indicative of erasing at least one of the at least two sectors from the memory device.

15. The memory module of claim 13 or 14, wherein the computer program code being further configured to, with the memory controller, cause the memory module at least to:

delete, by the host, a data entity entry from a file allocation table in response to a delete signal from an external host application; and

30 generate the signal indicative of erasing at least one of the at least two sectors from the memory device in response to the deletion of the data entity entry from the file allocation table.

16. The memory module of any of claims 13 to 15, wherein the sector comprises at least one of the following:

- an erase block;
- a write block;
- a page;
- a super page; and
- a sector of a hard drive.

17. The memory module of any of claims 13 to 16, wherein the memory module comprises at least one of the following:

- an Embedded MultiMediaCard (eMMC);
- a memory card (SD);
- an Advanced Technology Attachment (ATA); and
- a Universal Flash Storage (UFS).

18. A computer program embodied on a computer readable medium comprising computer executable program code which, when executed by at least one controller of a memory module, causes the memory module to:

store a data entity using at least two sectors of a memory device, the at least two sectors associated to the same data entity;

maintain, at a memory controller, context information of the data entity comprising a pointer to at least one of the at least two sectors of the memory device; and

erase the at least two sectors of the memory device using the context information.

1 / 3

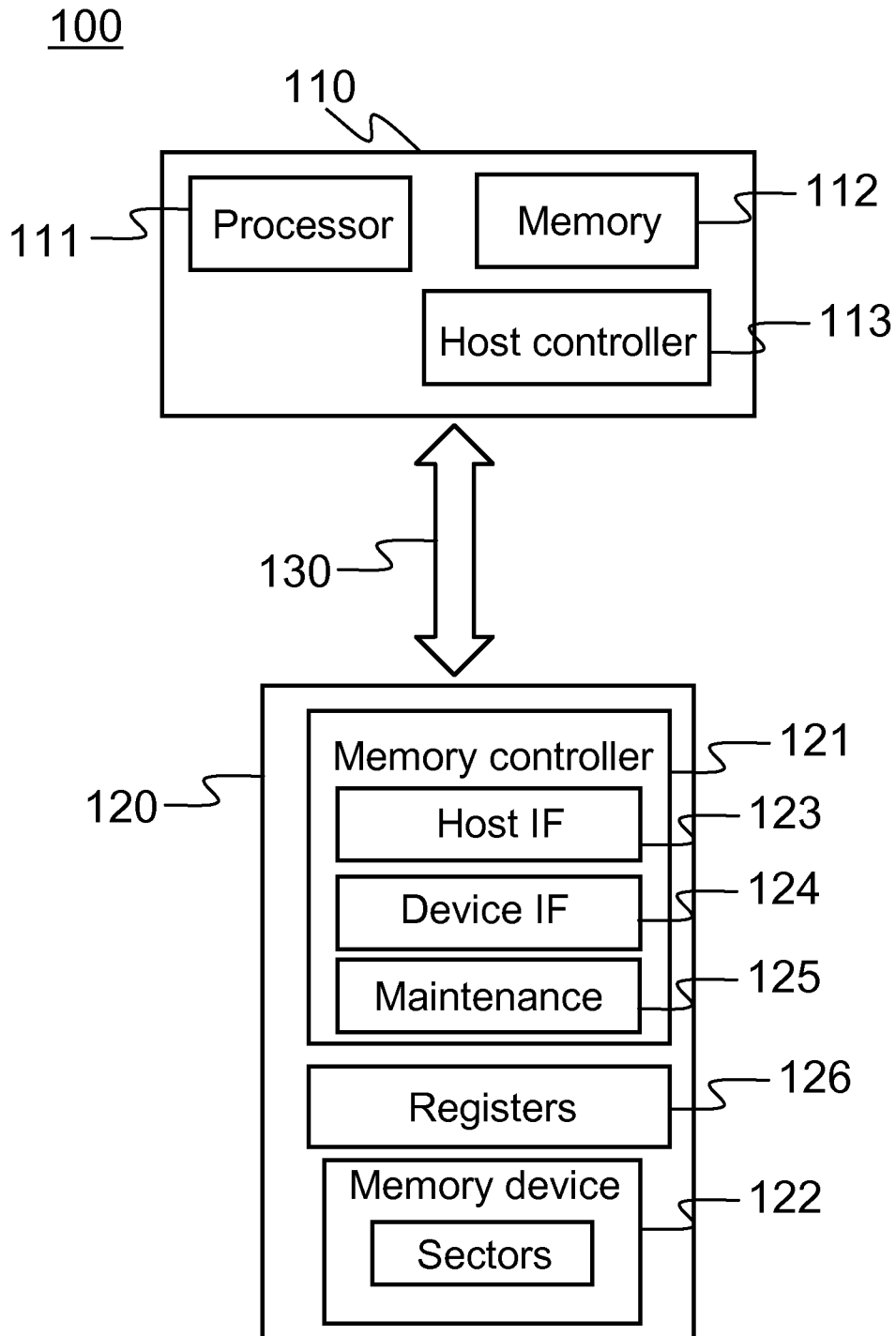


Fig. 1

2 / 3

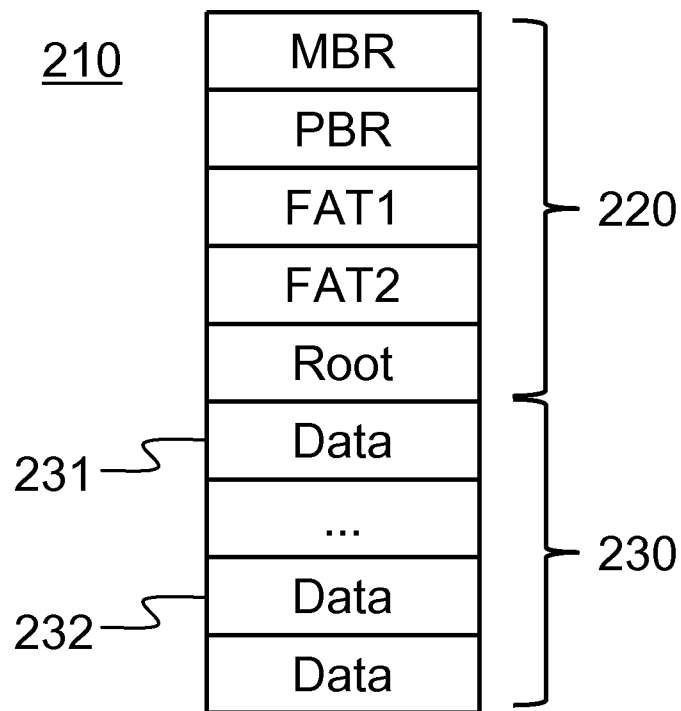


Fig. 2

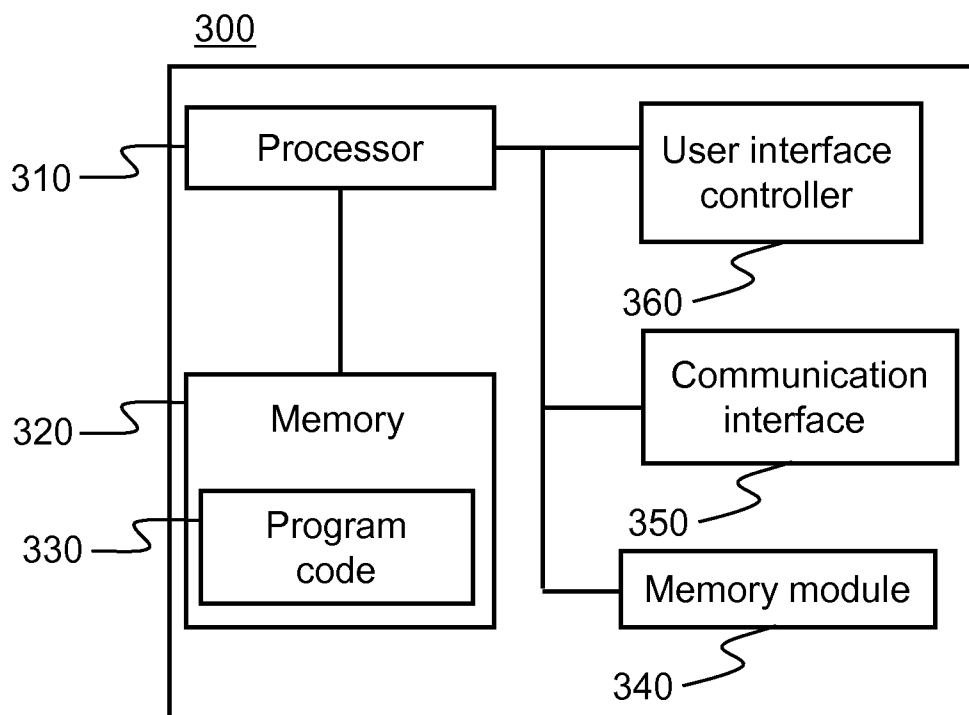


Fig. 3

3 / 3

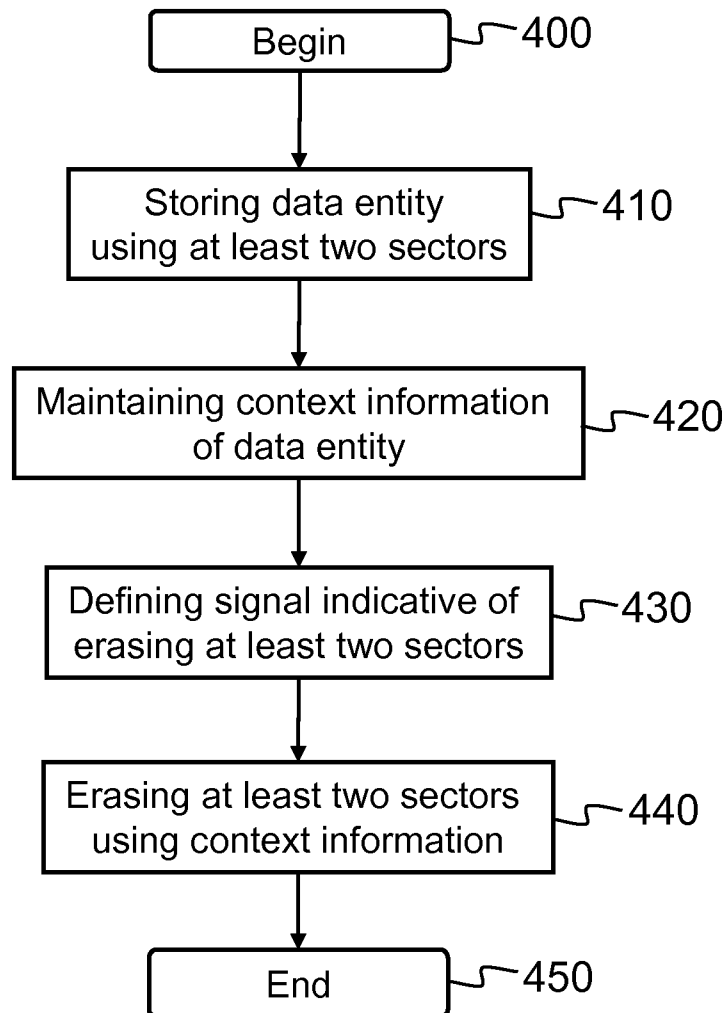


Fig. 4

INTERNATIONAL SEARCH REPORT

International application No

PCT/FI2011/051168

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06F12/02 G06F3/06
ADD.

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

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EPO-Internal , WPI Data, IBM-TDB, COMPENDEX, INSPEC

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	EP 2 006 774 AI (BEIJING WATCH DATA SYS CO LTD [CN]) 24 December 2008 (2008-12-24) abstract paragraph [0007] - paragraph [0028] figures 3-5 claims 4, 5, 8, 13 -----	1-3, 5-16, 18
X	us 2008/059695 AI (TANAKA YOSHIYUKI [JP] ET AL) 6 March 2008 (2008-03-06) abstract paragraph [0021] - paragraph [0052] paragraph [0268] - paragraph [0277] paragraph [0338] - paragraph [0343] figures 16, 20, 40, 41, 49 ----- - / - -	1-4,6-18



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

27 April 2012

Date of mailing of the international search report

14/05/2012

Name and mailing address of the ISA/

European Patent Office, P.B. 5818 Patentlaan 2
NL - 2280 HV Rijswijk
Tel. (+31-70) 340-2040,
Fax: (+31-70) 340-3016

Authorized officer

Mandate Davide

INTERNATIONAL SEARCH REPORT

International application No
PCT/FI2011/051168

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2006/020744 AI (SINCLAI R ALAN W [GB] ET AL SINCLAI R ALAN WELSH [GB] ET AL) 26 January 2006 (2006-01-26)	1, 13, 18
A	abstract paragraph [0040] - paragraph [0042] paragraph [0048] - paragraph [0056] figures 1-8 -----	2-12, 14-17
A	WO 2011/095516 AI (ST ERICSSON SA [CH] ; DAS PURKAYASTHA SAUGATA [IN]) 11 August 2011 (2011-08-11) abstract -----	1-18
A	US 5 337 275 A (GARNER RICHARD P [US]) 9 August 1994 (1994-08-09) abstract -----	1-18

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No

PCT/FI2011/051168

Patent document cited in search report		Publication date	Patent family member(s)			Publication date
EP 2006774	A1	24-12-2008	CN	101042696	A	26-09-2007
			EP	2006774	A1	24-12-2008
			KR	20090007357	A	16-01-2009
			US	2009307291	A1	10-12-2009
			WO	2007107073	A1	27-09-2007

US 2008059695	A1	06--03--2008	US	2008059695	A1	06--03--2008
			US	2010023680	A1	28--01--2010

US 2006020744	A1	26--01--2006	CN	101027651	A	29--08--2007
			EP	2254053	A2	24--11--2010
			US	2006020744	A1	26--01--2006

WO 2011095516	A1	11--08--2011	NONE			

US 5337275	A	09-08--1994	NONE			
