US 20080189485A1

(19) United States

(12) Patent Application Publication

Jung et al.

(10) Pub. No.: US 2008/0189485 A1

(43) Pub. Date: Aug. 7, 2008

(54) COOPERATIVE MEMORY MANAGEMENT

(75) Inventors: **Myoung-Soo Jung**, Suwon-City (KR); **Chan-Ik Park**, Seoul (KR); **Sang-Jin Oh**, Suwon-city (KR)

Correspondence Address:
F. CHAU & ASSOCIATES, LLC
130 WOODBURY ROAD
WOODBURY, NY 11797

(73) Assignee: **SAMSUNG ELECTRONICS CO., LTD.**, Suwon-si (KR)

(21) Appl. No.: **11/833,731**

(22) Filed: **Aug. 3, 2007**

(30) **Foreign Application Priority Data**

Feb. 1, 2007 (KR) P2007-0010573

Publication Classification

(51) **Int. Cl.**
G06F 13/00 (2006.01)

(52) **U.S. Cl.** **711/115**

(57) **ABSTRACT**

A cooperative memory card system includes a memory card device, and a host in signal communication with the memory card device, where the host assumes at least one memory management function for the memory card device; and a corresponding method of cooperative memory management between a host and a memory card device includes selecting at least one of several memory management functions to be performed by the host for the device.

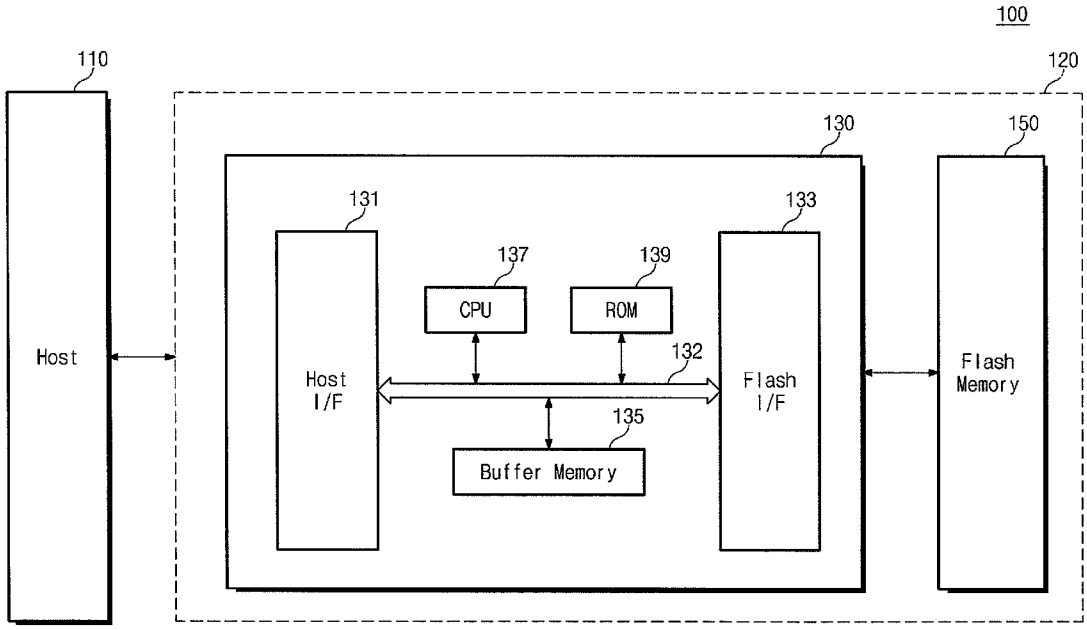


Fig. 1

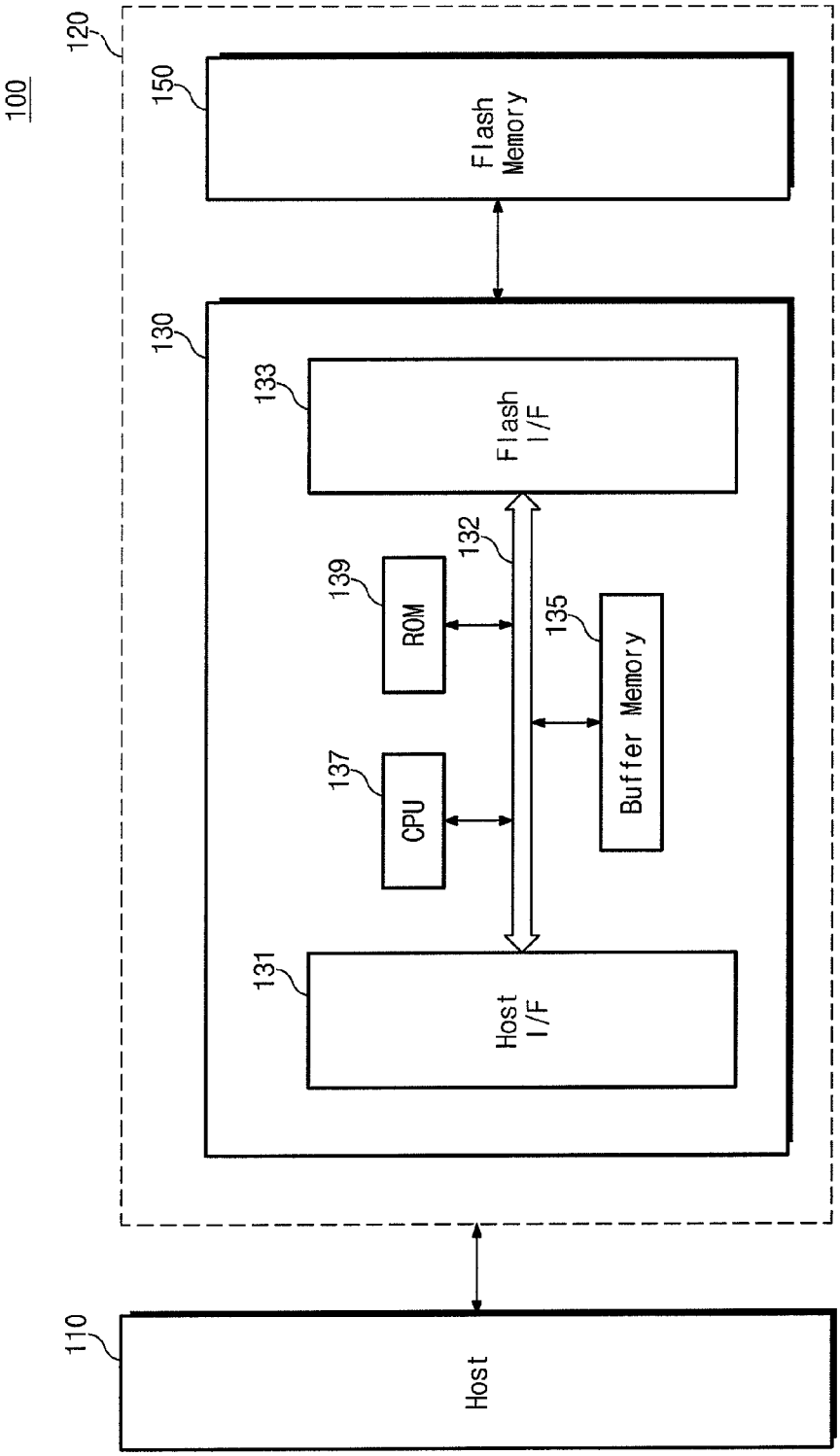

100
120
150
Flash Memory
130
133
Flash I/F
132
139
ROM
135
Buffer Memory
137
CPU
131
Host I/F
110
Host

Fig. 2

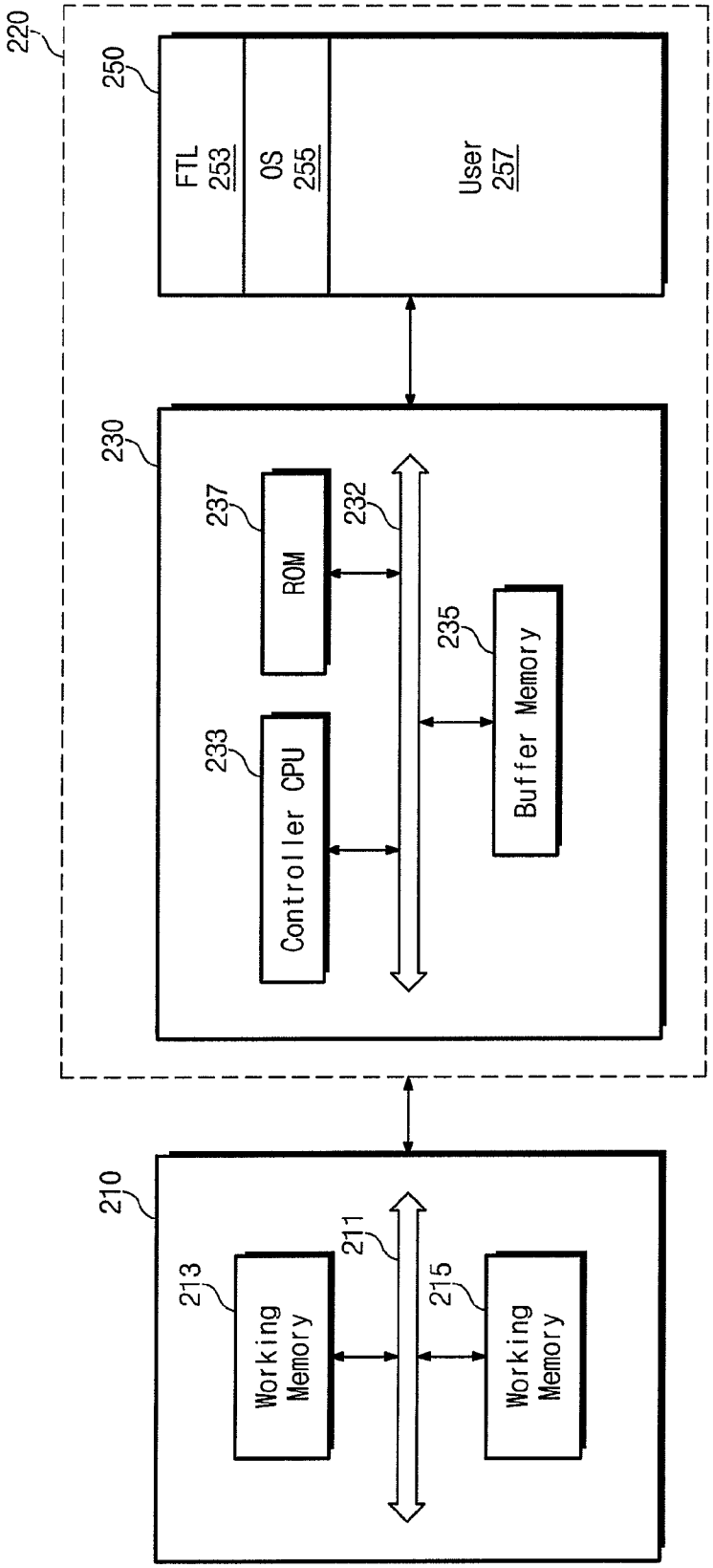
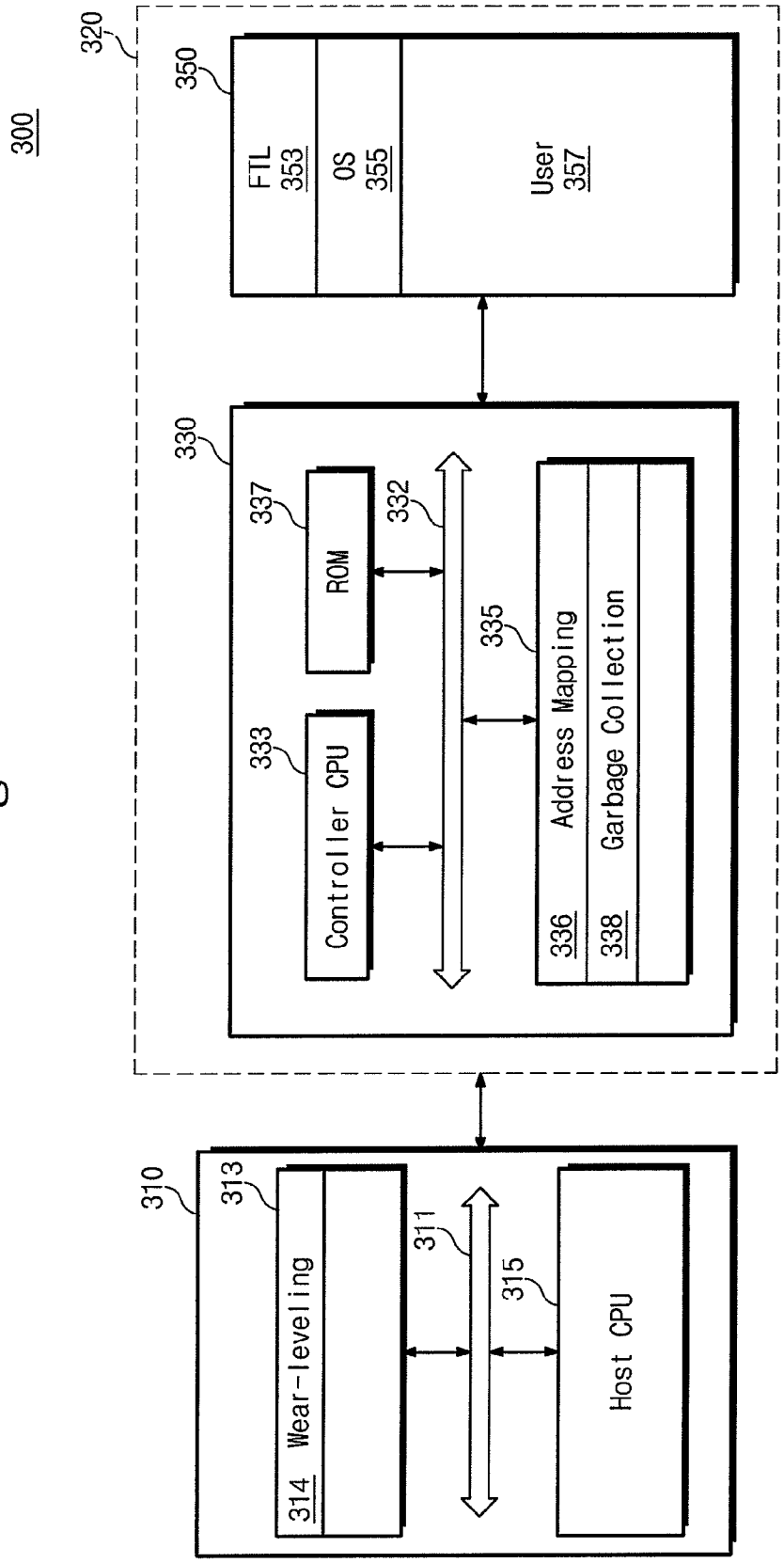

220
250
FTL
253
OS
255
User
257
230
237
ROM
232
235
Buffer Memory
233
Controller CPU
210
213
Working Memory
211
215
Working Memory

Fig. 3


300
320
350
FTL
353
OS
355
User
357
330
337
ROM
332
333
Controller CPU
335
336
Address Mapping
338
Garbage Collection
310
313
314
Wear-leveling
311
315
Host CPU

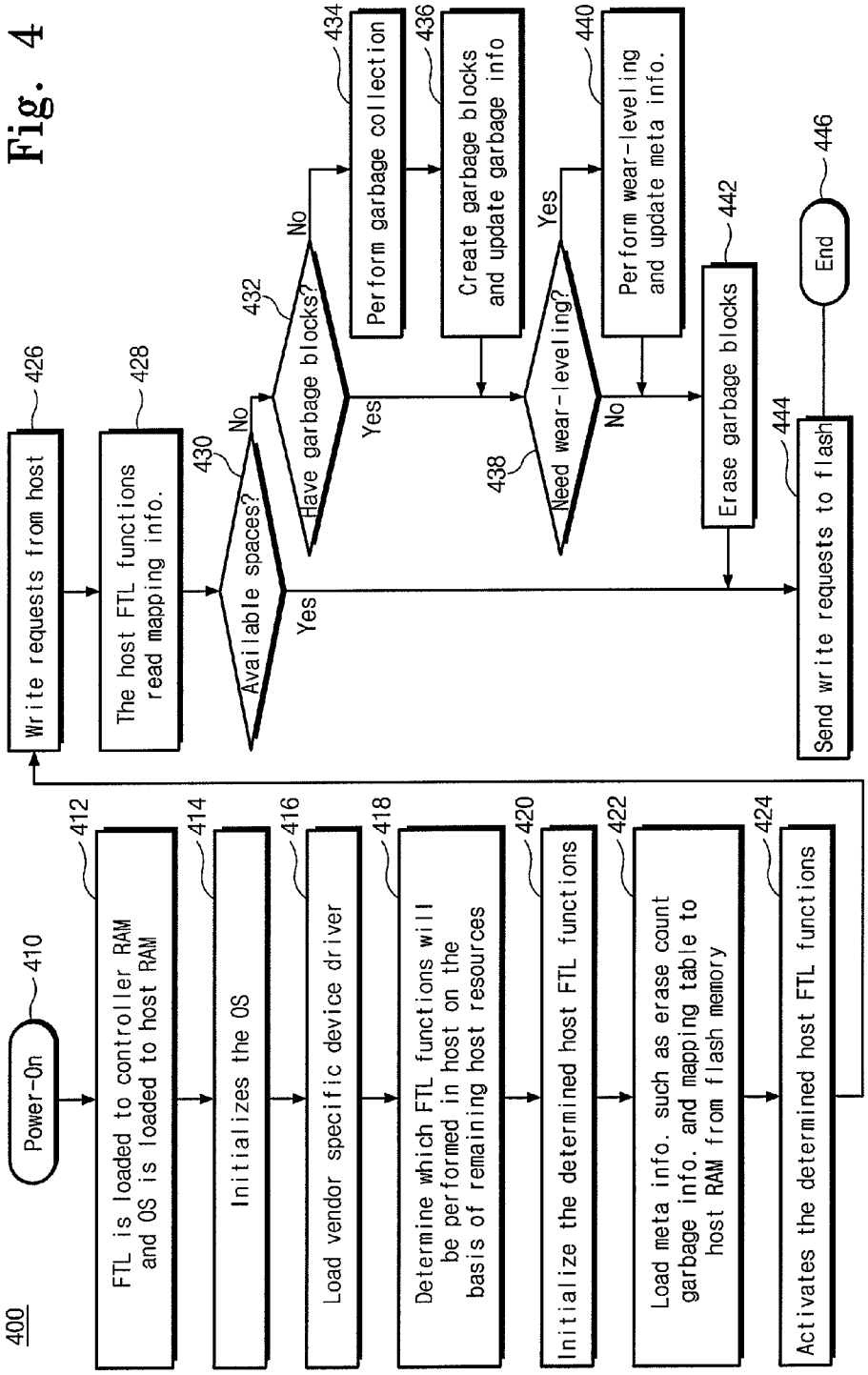

400
Power-On
410
FTL is loaded to controller RAM and OS is loaded to host RAM
412
Initializes the OS
414
Load vendor specific device driver
416
Determine which FTL functions will be performed in host on the basis of remaining host resources
418
Initialize the determined host FTL functions
420
Load meta info. such as erase count garbage info. and mapping table to host RAM from flash memory
422
Activates the determined host FTL functions
424
Write requests from host
426
The host FTL functions read mapping info.
428
Available spaces?
430
Yes
No
Have garbage blocks?
432
Yes
No
Perform garbage collection
434
Create garbage blocks and update garbage info
436
Need wear-leveling?
438
Yes
No
Perform wear-leveling and update meta info.
440
Erase garbage blocks
442
Send write requests to flash
444
End
446

Fig. 4

Fig. 5

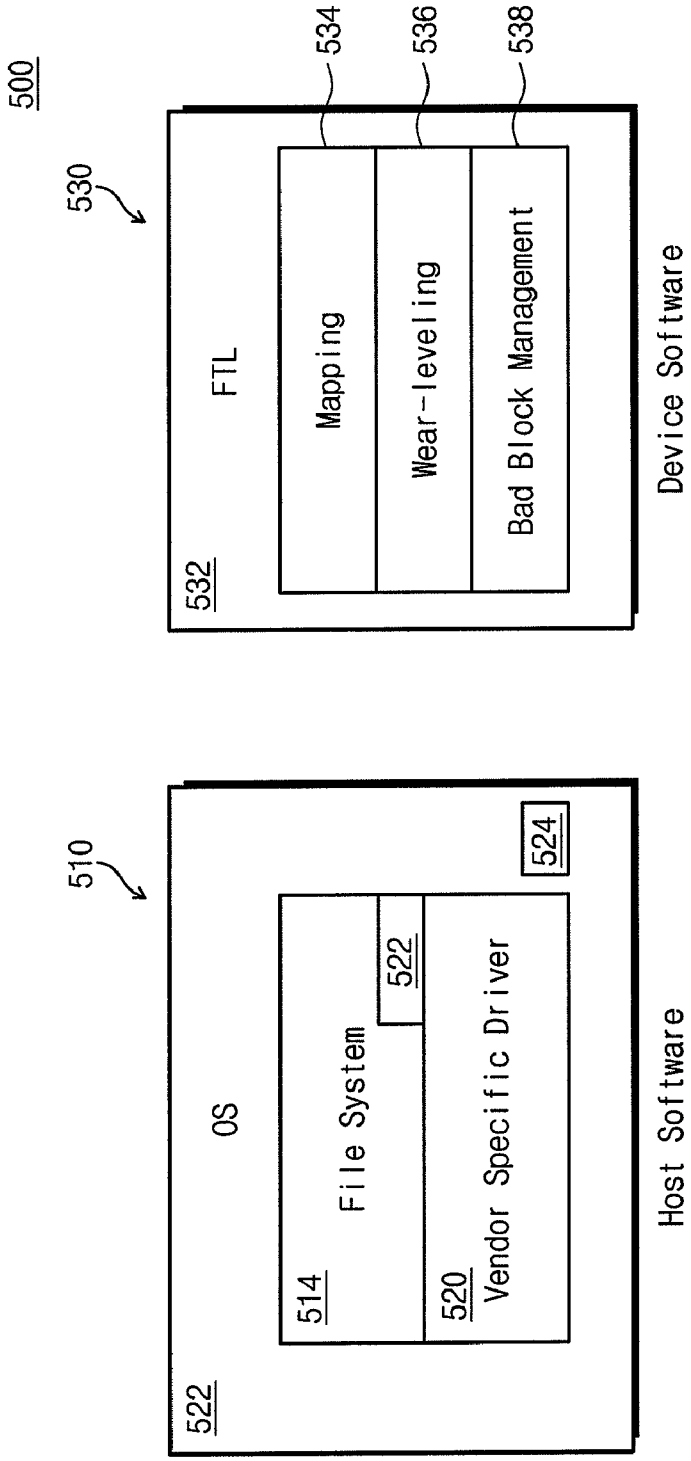

500
530
FTL
532
534
Mapping
536
Wear-leveling
538
Bad Block Management
Device Software
510
OS
522
514
File System
522
520
Vendor Specific Driver
524
Host Software

Fig. 6

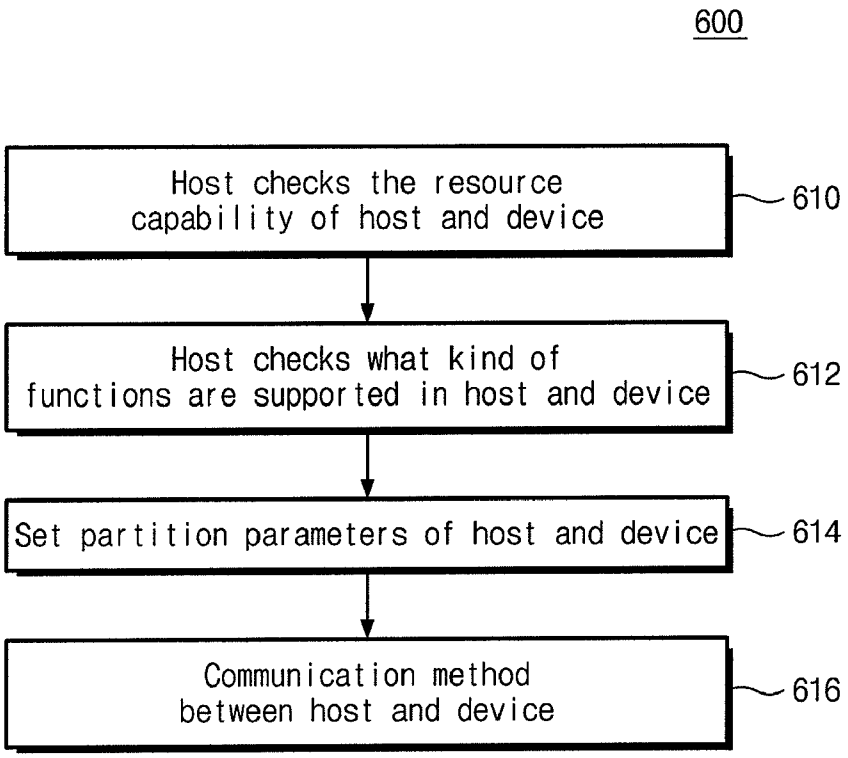

600
Host checks the resource capability of host and device
610
Host checks what kind of functions are supported in host and device
612
Set partition parameters of host and device
614
Communication method between host and device
616

Fig. 7

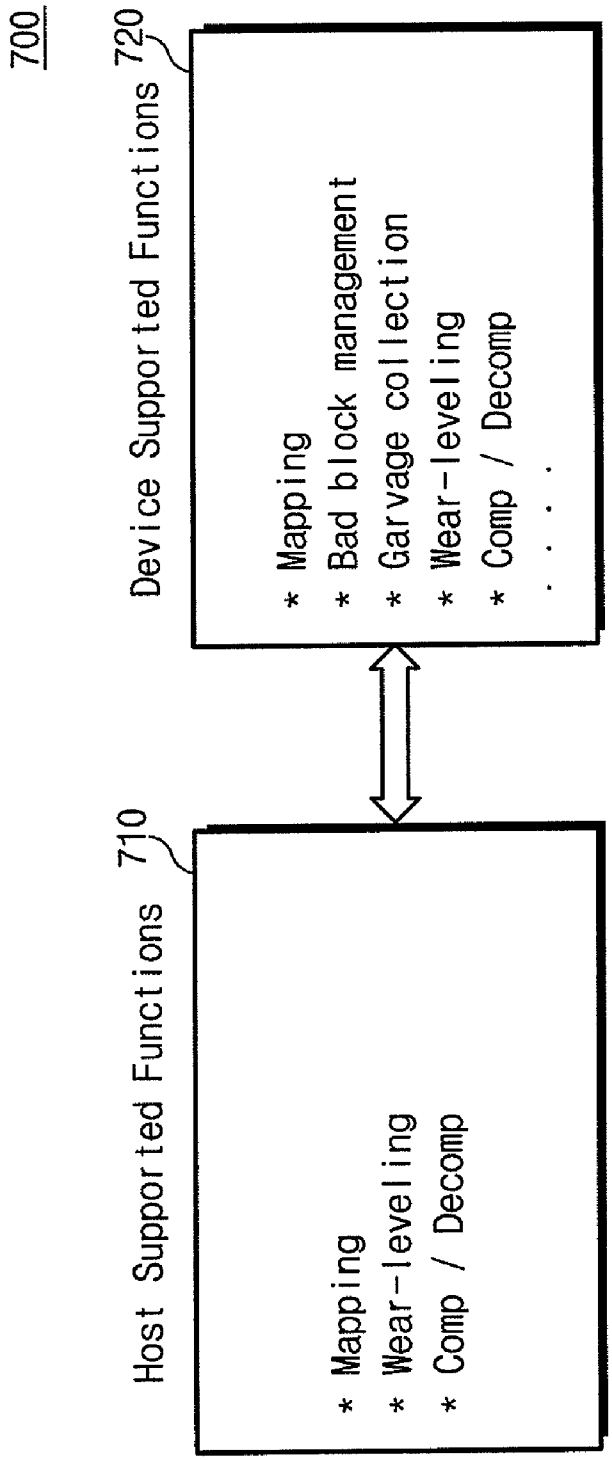

700
720
Device Supported Functions
* Mapping
* Bad block management
* Garvage collection
* Wear-leveling
* Comp / Decomp
. . . .
710
Host Supported Functions
* Mapping
* Wear-leveling
* Comp / Decomp

Fig. 8

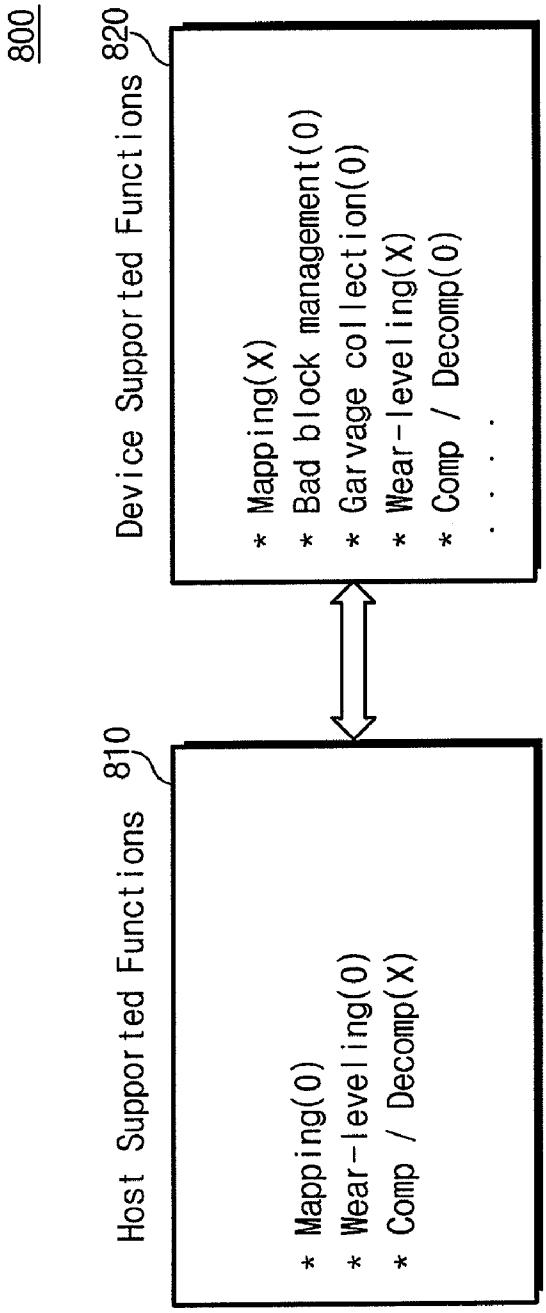
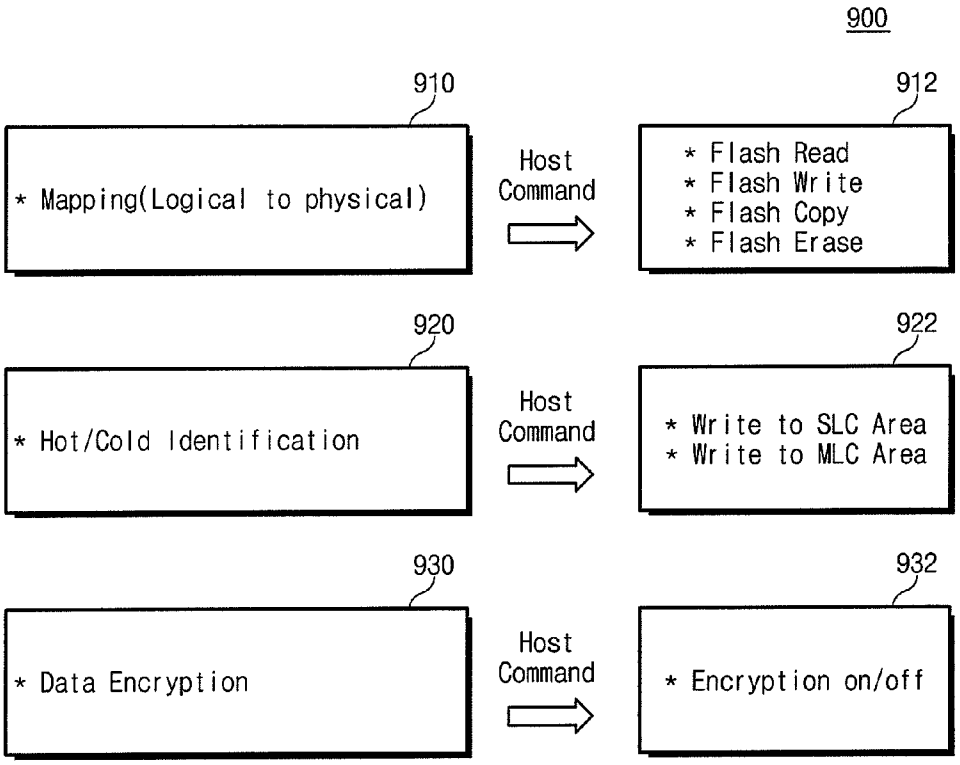

800
Host Supported Functions
810
* Mapping(O)
* Wear-leveling(O)
* Comp / Decomp(X)
Device Supported Functions
820
* Mapping(X)
* Bad block management(O)
* Garvage collection(O)
* Wear-leveling(X)
* Comp / Decomp(O)
. . . .

Fig. 9


900
910
* Mapping(Logical to physical)
Host Command
912
* Flash Read
* Flash Write
* Flash Copy
* Flash Erase
920
* Hot/Cold Identification
Host Command
922
* Write to SLC Area
* Write to MLC Area
930
* Data Encryption
Host Command
932
* Encryption on/off

COOPERATIVE MEMORY MANAGEMENT

CROSS-REFERENCE TO RELATED APPLICATION

[0001] This application claims foreign priority under 35 U.S.C. §119 to Korean Patent Application No. P2007-0010573 (Atty. Dkt. ID-200610-025), filed on Feb. 1, 2007, in the Korean Intellectual Property Office, the disclosure of which is incorporated by reference herein in its entirety.

BACKGROUND OF THE INVENTION

[0002] The present disclosure generally relates to mass digital data storage systems using flash memory technologies, and more specifically relates to techniques for efficiently distributing the resources of such systems.

[0003] Emerging portable electronic devices, such as computers, digital cameras, digital music players, cellular telephones, personal data assistants, and the like, are making increasing use of flash memories and/or flash cards. A flash card may include SSDs, SD cards, MMCs, Memory Sticks, embedded cards such as moviNAND, GBNAND, iNAND, and the like. Hosts communicate with flash memories using a flash translation layer ("FTL"). The FTL is typically stored on the flash card either in a controller or in a flash memory, and a form of firmware is used to effectively manage the card.

[0004] The FTL performs several functions such as address mapping, wear-leveling, garbage collection, and the like. An address mapping operation translates a logical address received from a host into a physical address, which will actually be used to store data within the flash memory. Wear-leveling may be implemented as a firmware technique for balancing the erase counts of physical blocks to fully utilize the lifetime of NAND flash, for example. A garbage collection operation may collect valid data of several blocks and fill in one block, freeing the originals.

[0005] A conventional flash card may perform all FTL functions in the flash controller. Unfortunately, the resources of flash controllers are generally smaller than those of hosts. Thus, a resulting execution speed of such FTL functions performed in the controller is typically slow. In some cases, the FTL functions cannot be performed at all, such as where the controller becomes overloaded.

SUMMARY OF THE INVENTION

[0006] These and other issues are addressed by a cooperative memory management system. Exemplary embodiments are provided.

[0007] An exemplary cooperative memory card system is provided, comprising a memory card device, and a host in signal communication with the memory card device, wherein the host assumes at least one memory management function for the memory card device.

[0008] In addition, similar exemplary memory card systems are provided. Another such system is provided wherein the host issues at least one command to the memory card device. Another system is provided comprising at least one of flash memory, FRAM, PRAM, MRAM, DRAM or EEPROM. Another system is provided wherein the at least one memory management function is a flash translation layer task. Another system is provided wherein the at least one memory management function is an operating system task. Another system is provided wherein the at least one memory management function is a device driver task.

[0009] Yet another system is provided wherein the host comprises a host processor and a working memory in signal communication with the host processor. Another system is provided wherein the host performs resource checking of the memory card device and assumes the at least one task of the memory card device in response thereto. Another system is provided wherein the host checks its own available resources and assumes the at least one task of the memory card device in response thereto. Another system is provided wherein the host performs resource checking of the memory card device, compares the resources of the memory card device with available resources of the host, and assumes the at least one task of the memory card device in response to the comparison.

[0010] A further system is provided wherein the at least one task of the memory card device assumed by the host is at least one of a function for address mapping, wear leveling, garbage collection, power-off recovery, bad block management, error correction coding, write buffering for real-time management, hot versus cold data management, encryption, decryption, compression or decompression. Another system is provided wherein vendor specific memory card device drivers include software modules for performing memory management functions. Another system is provided wherein the memory card device comprises at least one of a solid-state disk, USB card, SD card, MMC, Memory Stick or embedded device. Another system is provided wherein the memory card device comprises a controller and a memory in signal communication with the controller;

[0011] An exemplary method of cooperative memory management between a host and a memory card device is provided, the method comprising selecting at least one of a plurality of memory management functions to be performed by the host for the device.

[0012] In addition, similar exemplary methods of cooperative memory management are provided. Another such method includes checking the available resources of at least one of the host or the device and selecting the at least one memory management function in response to the checked available resources. Another method is provided wherein the resources checked are those of the host. Another method is provided wherein the resources checked are those of the device. Another method is provided wherein the resources checked are those of the host and those of the device. Another method is provided wherein checking the available resources comprises the device issuing a request to the host for the host to perform the selected at least one memory management function. Another method is provided wherein the host issues at least one command to the memory card device. Another method is provided comprising loading the selected at least one memory management function from the memory card device to a random access memory of the host.

[0013] Yet another method is provided wherein the at least one memory management function comprises a function of a flash transtation layer. Another method is provided comprising using the host to check the available resources of the memory card device, comparing the available resources of the host with the available resources of the memory card device, and determining which memory management functions will be performed in the host and which in the memory card device in response to the comparison. Another method is provided wherein the available resources of the host comprise at least one of processor type, processor speed, memory size, memory speed, bus type, bus speed, encryption type or encryption speed. Another method is provided wherein the

memory management functions comprise at least one of address mapping, wear leveling, garbage collection or erase count accumulation. Another method is provided wherein the selected at least one memory management function for the host has a parameter set for the host but reset for the device. Another method is provided wherein a communication protocol between the host and the device comprises at least one of a universal serial bus protocol or an advanced technology attachment protocol.

[0014] A further method is provided wherein the selected at least one memory management function is comprised by flash translation layer code from the memory card device. Another method is provided wherein the selected at least one memory management function is comprised by a device driver. Another method is provided wherein the selected at least one memory management function is comprised by an operating system from the memory card device. Another method is provided comprising loading memory management function code from a memory of the memory card device into a controller random access memory of the memory card device, and loading an operating system from the memory of the memory card device into the random access memory of the host.

[0015] Yet another method is provided comprising loading at least one vendor specific memory card device driver into a random access memory of the host. Another method is provided comprising loading meta data from the memory card device to the host. Another method is provided comprising initializing the selected memory management functions in the host. Another method is provided comprising activating the selected memory management functions in the host. Another method is provided comprising producing a write request from the host reading mapping information using host memory management functions, checking for write space available on the memory card device, and sending the write request to the device if there is write space available on the device.

[0016] A further method is provided comprising checking for garbage blocks on the memory card device, if there are no garbage blocks on the device, performing garbage collection, designating garbage blocks and updating garbage collection data, and erasing the garbage blocks. Another method is provided comprising checking the need for wear leveling on the memory card device, and if wear leveling is needed, performing wear leveling and updating meta data for wear leveling.

[0017] The present disclosure will be further understood from the following description of exemplary embodiments, which is to be read in connection with the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

[0018] The present disclosure provides cooperative memory management for a partitioned flash translation layer ("FTL") in accordance with the following exemplary figures, in which:

[0019] FIG. **1** shows a schematic block diagram for a flash memory card system having a flash translation layer;

[0020] FIG. **2** shows a schematic block diagram for a flash memory card system with cooperative memory management in accordance with an exemplary embodiment of the present disclosure;

[0021] FIG. **3** shows a schematic block diagram for a flash memory card system having a partitioned flash translation layer in accordance with an exemplary embodiment of the present disclosure;

[0022] FIG. **4** shows a schematic flowchart for a cooperative memory management method in accordance with an exemplary embodiment of the present disclosure;

[0023] FIG. **5** shows a schematic memory map for a cooperative memory management system in accordance with an exemplary embodiment of the present disclosure;

[0024] FIG. **6** shows a schematic flowchart for a method of cooperative memory management using a partitioned FTL in accordance with an exemplary embodiment of the present disclosure;

[0025] FIG. **7** shows a schematic block diagram for host and device supported functions in accordance with an exemplary embodiment of the present disclosure;

[0026] FIG. **8** shows a schematic block diagram for host and device activated functions in accordance with an exemplary embodiment of the present disclosure; and

[0027] FIG. **9** shows a schematic block diagram for a communication method between a host and a device in a cooperative memory management system in accordance with an exemplary embodiment of the present disclosure.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

[0028] The present disclosure provides for cooperative memory management between a host and a flash memory device. Exemplary embodiments of the present disclosure include systems having partitioned flash translation layers (FTL), and corresponding methods. An exemplary cooperative memory management approach uses a host to check the resources of a flash card, compare the resources of the flash card with those of the host and determine which FTL functions are to be run by the host and which by the card.

[0029] As shown in FIG. **1**, a flash memory card system is indicated generally by the reference numeral **100**. The system **100** may be a portable electronic device, such as a computer, digital camera, digital music player, cellular telephone, personal data assistant (PDA), or the like. The system **100** includes a host **110** in signal communication with a flash card **120**. The flash card **120** may be a solid-state disk ("SSD"), SD card, MMC, Memory Stick, an embedded card such as moviNAND, GBNAND, iNAND, or the like.

[0030] The flash card **120** includes a controller **130** in signal communication with a flash memory **150**. The host **110** communicates with the flash memory **150** using a flash translation layer ("FTL"), which may include logic and/or firmware used to effectively manage the card **120**. The FTL may be stored or implemented in the controller **130** or in the flash memory **150**.

[0031] The controller **130** includes a host interface **131** in signal communication with a controller bus **132**, a flash interface **133** in signal communication with the controller bus **132**, a buffer memory **135** in signal communication with the controller bus **132**, a CPU **137** in signal communication with the controller bus **132**, and a ROM **139** in signal communication with the controller bus **132**. Turning to FIG. **2**, a flash memory card system with cooperative memory management is indicated generally by the reference numeral **200**. The card system **200** includes a host **210** in signal communication with a flash card **220**. The host **210** includes a host bus **211**, a

working memory **213** in signal communication with the host bus **211**, and a host CPU **215** in signal communication with the host bus **211**.

[0032] The flash card **220** includes a controller **230** in signal communication with a flash memory **250**. The controller **230** includes a controller bus **232**, a controller CPU **233** in signal communication with the controller bus **232**, a buffer memory **235** in signal communication with the controller bus **232**, and a ROM **237** in signal communication with the controller bus **232**. The flash memory **250** includes a flash translation layer ("FTL") code block **253**, an operating system ("OS") image block **255** and a user data block **257**. In operation, a host CPU **215** and working memory **213** within a host **210** may be relatively faster than a controller CPU **233** and a buffer memory **235** within a controller **230**. Thus, the host **210** may be used to perform some functions of the FTL. Initially, the FTL code **253** and OS image **255** are stored in the flash memory **250**. The OS image **255** may include a file system and device drivers for executing general functions, for example. The OS image **255** of the present embodiment may further include software modules for executing FTL functions, such as an address mapping function, a wear-leveling function and a garbage collection function.

[0033] Turning now to FIG. **3**, a flash memory card system like the system **200** of FIG. **2** is shown in greater detail with a partitioned FTL, and indicated generally by the reference numeral **300**. The card system **300** includes a host **310** in signal communication with a flash card **320**. The host **310** includes a host bus **311** a working memory **313** in signal communication with the host bus **311**, and a host CPU **315** in signal communication with the host bus **311**. The working memory **313** includes a wear-leveling block **314**.

[0034] The flash card **320** includes a controller **330** in signal communication with a flash memory **350**. The controller **330** includes a controller bus **332**, a controller CPU **333** in signal communication with the controller bus **332**, a buffer memory **335** in signal communication with the controller bus **332**, and a ROM **337** in signal communication with the controller bus **332**. The buffer memory **335** includes an address mapping block **336** and a garbage collection block **338**. The flash memory **350** includes a flash translation layer ("FTL") code block **353**, an operating system ("OS") image block **355**, and a user data block **357**. In operation after power on, the FTL code **353** is loaded to the buffer memory **335**, and the OS image **355**, which includes some FTL functions, is loaded to the working memory **313** within the host **310**. In this manners some FTL functions, such as the address mapping **336** and a garbage collection **338** in this example, operate in the controller, and the other functions, such as the wear-leveling **314** in this example, operate in the host.

[0035] Thus, the FTL functions in the host **310** can utilize the resources of the host, which may be relatively larger and/or faster than those in the flash controller **330**, to increase performance of the system while efficiently executing all functions of FTL. Here, the OS image may include software modules for executing any or all functions of the FTL in the host. The FTL functions may be embedded in the file system or device drivers within the OS code. Any or all FTL functions included in the OS image, file system or device drivers may be performed by the host instead of by the controller.

[0036] In alternate embodiments, the OS image **355** might not include software modules for performing some FTL functions. In such cases, vendor specific device drivers may include the FTL software modules. Generally, the device drivers may be provided by hardware suppliers such as SSD manufacturers. USB/SD/MMC application providers, or the like. Thus, the hosts in these embodiments can determine which FTL functions will be loaded to the host from all of the available embedded FTL functions.

[0037] As shown in FIG. **4** a method of cooperative memory management is indicated generally by the reference numeral **400**. The method **400** includes a start or power-on block **410**, which passes control to a function block **412**. The function block **412** toads FTL code into the controller RAM, loads OS code into the host RAM, and passes control to a function block **414**. The function block **414** initializes the OS in the host, and passes control to a function block **418**.

[0038] The function block **418**, in turn, determines which FTL functions will be performed in the host, on the basis of the host's remaining resources, and passes control to a function block **420**. The function block **420** initiatizes the determined FTL functions on the host, and passes control to a function block **422**. The function block **422** loads meta information, such as an erase count, garbage collection data and a mapping table from the flash memory to the host RAM, and passes control to a function block **424**. The function block **424** activates the determined host FTL functions, and passes control to a function block **426**.

[0039] The function block **426**, in turn, processes write requests from the host, and passes control to a function block **428**. At the function block **428**, the host FTL functions read mapping information. Control passes from the function block **428** to a decision block **430**. The decision block **430** determines whether there are any available spaces to write into the flash memory, and if so, passes control to a function block **444**. If not, the decision block **430** passes control to a decision block **432**. The decision block **432** determines whether the flash memory has garbage blocks, and if so, passes control to a decision block **438**. If not, the decision block **432** passes control to a function block **434**.

[0040] The function block **434** performs garbage collection, and passes control to a function block **436**. The function block **436** creates garbage blocks, updates the garbage information and passes control to the decision block **438**. The decision block **438**, in turn, determines whether wear-leveling is needed, and if not, passes control to a function block **442**. If so, the decision block **438** passes control to a function block **440**. The function block **440** performs wear-leveling, updates meta information, and passes control to the function block **442**. The function block **442** erases garbage blocks, and passes control to the function block **444**. The function block **444** sends write requests to the flash memory, and passes control to an end block **446**.

[0041] In operation, a vendor specific device driver may include some or all of the software modules for executing the FTL functions for cooperative memory management of a partitioned FTL. Functionalities of the flash software may include a sector mapping algorithm, garbage collection, power-off recovery, bad block management, wear-leveling, error correction coding ("ECC"), write buffering for real-time management, hot/cold data management, encryption, decryption, compression and decompression.

[0042] Turning to FIG. **5**, a memory map for a cooperative memory management system is indicated generally by the reference numeral **500**. The memory map **500** includes host software **510** and device software **530**. The host software **510** includes an OS image **512**, a file system **514**, and a vendor-specific driver **520**. The vendor-specific driver **520** may be

associated with an additional file system **522** and/or additional OS code **524**. The device software **530** includes FTL code **532**, which may include a mapping function **534**, a wear-leveling function **536** and a bad block management function **538**. Turning now to FIG. **6**, a method for a cooperative memory management system having a partitioned FTL is indicated generally by the reference numeral **600**. The method **600** includes a step **610**, at which a host checks the resource capabilities of the host, and checks the resource capabilities of a device. The resource capabilities may include the CPU speed and the memory size for buffering, for example. At a step **612**, the host checks to determine what kinds of functions are supported by the host, and checks to determine what kinds of functions are supported by the device. At a step **614**, the host and device negotiate and set the partition parameters for the host and for the device. At step **616**, the host and device negotiate and set the communication method and parameters between the host and the device.

[0043] As shown in FIG. **7**, an exemplary set of supported functions is indicated generality by the reference numeral **700**. The supported functions **700** include host supported functions **710** and device supported functions **720**. Here, the host supported functions **710** include a mapping function, a wear-leveling function, and a compression/decompression function.

[0044] The device supported functions **720** include a mapping function, a bad block management function, a garbage collection function, a wear-leveling function, a compression/decompression function, and possibly other functions. A query about the functionality of the host or the device would return an indication of the above supported functions, respectively. Both the host and the device may have one or more overlapping functions, so it remains to be determined which will actually implement each of the overlapping functions. Turning to FIG. **8**, a set of negotiated partitioning parameters is indicated generally by the reference numeral **800**. In the figure, "O" indicates that a local function is activated, and "X" indicates that a local function is not activated. The supported functions **800** include host supported functions **810** and device supported functions **820**. Here, the host supported functions **810** include a mapping function that is activated on the host, a wear-leveling function that is activated on the host, and a compression/decompression function that is not activated on the host.

[0045] The device supported functions **820** include a mapping function that is not activated on the device, a bad block management function that is activated on the device, a garbage collection function that is activated on the device, a wear-leveling function that is not activated on the device, a compression/decompression function that is activated on the device, and possibly other functions that may be activated on the device.

[0046] A query about the active functions of the host or the device would return an indication of the above activated functions, respectively. Both the host and the device may have one or more overlapping functions, so a function activated on one of the host or device will generally not be activated on the other of the host or device, respectively.

[0047] Turning now to FIG. **9**, a communication method at run-time is indicated generally by the reference numeral **900**. The communication method **900** includes associated host commands for the FTL functions activated on the host. Here, the FTL functions activated on the host include a logical to physical mapping function **910**, a hot/cold data identification function **920**, and a data encryption function **930**.

[0048] Invocation of the mapping function **910** causes the host to issue one of a plurality of mapping commands **912**, such as a flash read command, a flash write command, a flash copy command, or a flash erase command. Invocation of the identification function **920** causes the host to issue one of a plurality of identification commands **922**, such as a command for writing to an SLC area, or a command for writing to an MLC area. Invocation of the encryption function **930** causes the host to issue one of a plurality of encryption commands **932**, such as a command for turning encryption on, or a command for turning encryption off.

[0049] Although illustrative embodiments have been described herein with reference to the accompanying drawings, it is to be understood that the present disclosure is not limited to those precise embodiments, and that various other changes and modifications may be effected therein by those of ordinary skill in the pertinent art without departing from the scope or spirit of the present disclosure. All such changes and modifications are intended to be included within the scope of the present disclosure as set forth in the appended claims.

1. A cooperative memory card system comprising:
a memory card device; and
a host in signal communication with the memory card device,
wherein the host assumes at least one memory management function for the memory card device.

2. A system as defined in claim **1** wherein the host issues at least one command to the memory card device.

3. A system as defined in claim **2** wherein the at least one command comprises a command for at least one of a read, a write, a copy or an erase.

4. A system as defined in claim **2** wherein the at least one command comprises a command for at least one of a write to a single-level cell region or a write to a multi-level cell region.

5. A system as defined in claim **2** wherein the at least one command comprises a command for at least one of enabling or disabling at least one of encryption or decryption.

6. A system as defined in claim **1**, the memory card device comprising at least one of flash memory, FRAM, PRAM, MRAM, DRAM or EEPROM.

7. A system as defined in claim **1** wherein the at least one memory management function is a flash translation layer task.

8. A system as defined in claim **1** wherein the at least one memory management function is an operating system task.

9. A system as defined in claim **1** wherein the at least one memory management function is a device driver task.

10. A system as defined in claim **1**, the host comprising:
a host processor; and
a working memory in signal communication with the host processor.

11. A system as defined in claim **1** wherein the host performs resource checking of the memory card device and assumes the at least one task of the memory card device in response thereto.

12. A system as defined in claim **1** wherein the host checks its own available resources and assumes the at least one task of the memory card device in response thereto.

13. A system as defined in claim **1** wherein the host performs resource checking of the memory card device, compares the resources of the memory card device with available

resources of the host, and assumes the at least one task of the memory card device in response to the comparison.

14. A system as defined in claim **1** wherein the at least one task of the memory card device assumed by the host is at least one of a function for address mapping, wear leveling, garbage collection, power-off recovery, bad block management, error correction coding, write buffering for real-time management, hot versus cold data management, encryption, decryption, compression or decompression.

15. A system as defined in claim **1** wherein vendor specific memory card device drivers include software modules for performing memory management functions.

16. A system as defined in claim **15** wherein the host determines which of the software modules for performing memory management functions will be loaded to the host.

17. A system as defined in claim **1** wherein the memory card device comprises at least one of a solid-state disk, USB card, SD card, MMC, Memory Stick or embedded device.

18. A system as defined in claim **17** wherein the embedded device comprises at least one of a moviNAND, GBNAND or iNAND.

19. A system as defined in claim **1**, the memory card device comprising:

a controller; and

a memory in signal communication with the controller.

20. A system as defined in claim **19**, the controller comprising:

a controller processor;

a read-only memory in signal communication with the controller processor; and

a buffer memory in signal communication with the controller processor.

21. A system as defined in claim **19**, the memory comprising:

a region having memory management function code; and

a region having user data.

22. A system as defined in claim **21** wherein the region having memory management function code is a flash translation layer.

23. A system as defined in claim **21** wherein the at least one task of the memory card device assumed by the host is a function of the memory management function code.

24. A system as defined in claim **21** wherein the memory management function code comprises at least one of program instructions or data.

25. A system as defined in claim **19**, the memory comprising:

a region having an operating system image; and

a region having user data.

26. A system as defined in claim **25** wherein memory management functions are embedded in a file system of the operating system image.

27. A system as defined in claim **25** wherein memory management functions are embedded in at least one device driver of the host.

28. A system as defined in claim **25** wherein memory management functions are embedded in a flash translation layer of the device.

29. A system as defined in claim **25** wherein the at least one task of the memory card device assumed by the host is a function of the operating system image.

30. A system as defined in claim **25** wherein the operating system image includes software modules for executing all memory management functions.

31. A system as defined in claim **25** wherein the operating system image does not include software modules for performing memory management functions.

32. A method of cooperative memory management between a host and a memory card device, the method comprising:

selecting at least one of a plurality of memory management functions to be performed by the host for the device.

33. A method as defined in claim **32**, further comprising:

checking the available resources of at least one of the host or the device; and

selecting the at least one memory management function in response to the checked available resources.

34. A method as defined in claim **32** wherein the resources checked are those of the host.

35. A method as defined in claim **32** wherein the resources checked are those of the device.

36. A method as defined in claim **32** wherein the resources checked are those of the host and those of the device.

37. A method as defined in claim **32** wherein checking the available resources comprises the device issuing a request to the host for the host to perform the selected at least one memory management function.

38. A method as defined in claim **32** wherein the host issues at least one command to the memory card device.

39. A method as defined in claim **38** wherein the at least one command comprises a command for at least one of a read, a write, a copy or an erase.

40. A method as defined in claim **38** wherein the at least one command comprises a command for at least one of a write to a single-level cell region or a write to a multi-level cell region.

41. A method as defined in claim **38** wherein the at least one command comprises a command for at least one of enabling or disabling at least one of encryption or decryption.

42. A method as defined in claim **38** wherein the at least one command comprises a command for checking available resources of the device.

43. A method as defined in claim **38** wherein the at least one command is unconditional.

44. A method as defined in claim **32**, further comprising loading the selected at least one memory management function from the memory card device to a random access memory of the host.

45. A method as defined in claim **32** wherein the at least one memory management function comprises a function of a flash translation layer.

46. A method as defined in claim **32**, further comprising:

using the host to check the available resources of the memory card device;

comparing the available resources of the host with the available resources of the memory card device; and

determining which memory management functions will be performed in the host and which in the memory card device in response to the comparison.

47. A method as defined in claim **32** wherein the available resources of the host comprise at least one of processor type, processor speed, memory size, memory speed, bus type, bus speed, encryption type or encryption speed.

48. A method as defined in claim **32** wherein the memory management functions comprise at least one of address mapping, wear leveling, garbage collection or erase count accumulation.

49. A method as defined in claim **32** wherein the selected at least one memory management function for the host has a parameter set for the host but reset for the device.

50. A method as defined in claim **49** wherein the remainder of the memory management functions not selected for the host have parameters reset for the host but set for the device.

51. A method as defined in claim **32** wherein a communication protocol between the host and the device comprises at least one of a universal serial bus protocol or an advanced technology attachment protocol.

52. A method as defined in claim **32** wherein the selected at least one memory management function is comprised by flash translation layer code from the memory card device.

53. A method as defined in claim **32** wherein the selected at least one memory management function is comprised by a device driver.

54. A method as defined in claim **32** wherein the selected at least one memory management function is comprised by an operating system from the memory card device.

55. A method as defined in claim **32**, further comprising:

loading memory management function code from a memory of the memory card device into a controller random access memory of the memory card device; and

loading an operating system from the memory of the memory card device into the random access memory of the host.

56. A method as defined in claim **55**, further comprising initializing the operating system in the host.

57. A method as defined in claim **32**, further comprising loading at least one vendor specific memory card device driver into a random access memory of the host.

58. A method as defined in claim **32**, further comprising loading meta data from the memory card device to the host.

59. A method as defined in claim **58** wherein the meta data comprises at least one of erase count data, garbage collection data or mapping table data loaded to the host random access memory from the memory card device.

60. A method as defined in claim **32**, further comprising initializing the selected memory management functions in the host.

61. A method as defined in claim **32**, further comprising activating the selected memory management functions in the host.

62. A method as defined in claim **32**, further comprising:

producing a write request from the host;

reading mapping information using host memory management functions;

checking for write space available on the memory card device; and

sending the write request to the device if there is write space available on the device.

63. A method as defined in claim **32**, further comprising:

checking for garbage blocks on the memory card device;

if there are no garbage blocks on the device, performing garbage collection, designating garbage blocks and updating garbage collection data; and

erasing the garbage blocks.

64. A method as defined in claim **32**, further comprising:

checking the need for wear leveling on the memory card device; and

if wear leveling is needed, performing wear leveling and updating meta data for wear leveling.

* * * * *