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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
29 June 2006 (29.06.2006)

PCT

(10) International Publication Number
WO 2006/069273 A2

(51) International Patent Classification: Not classified

(21) International Application Number:
PCT/US2005/046688

(22) International Filing Date:
21 December 2005 (21.12.2005)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

60/638,485	21 December 2004 (21.12.2004)	US
11/313,447	20 December 2005 (20.12.2005)	US
11/313,428	20 December 2005 (20.12.2005)	US

(71) Applicant *(for all designated States except US)*: **SANDISK CORPORATION** [US/US]; 140 Caspian Court, Sunnyvale, California 94089 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **HOLTZMAN, Michael** [IL/US]; 7602 Barnhart Place, Cupertino, CA 95014 (US). **COHEN, Baruch, B.** [IL/IL]; Aya 15, 21721 Carmiel (IL). **ISLAM, Muhammed, R.** [BD/US]; 1652 Hope Drive, #1335, Santa Clara, CA 95054 (US). **DAVIDSON, Matthew** [US/US]; 1600 Villa Street, Apt. 345, Mountain View, CA 94041 (US).

(74) Agents: **HSUE, James, S.** et al.; 595 Market Street, Suite 1900, San Francisco, California 94111 (US).

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

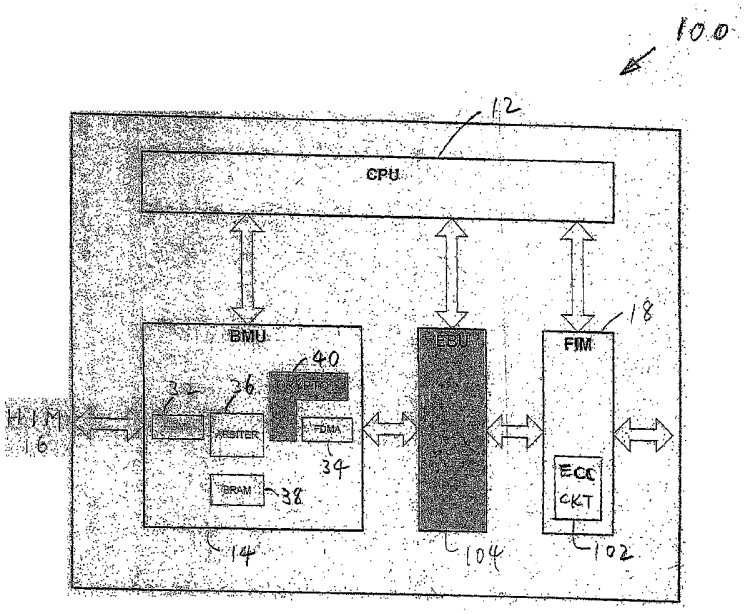
(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) **Title:** MEMORY SYSTEM WITH IN STREAM DATA ENCRYPTION/DECRYPTION AND ERROR CORRECTION



(57) **Abstract:** The throughput of the memory system is improved where error correction of data in a data stream is cryptographically processed with minimal involvement of any controller. To perform error correction when data from the memory cells are read, the bit errors in the data in the data stream passing between the cells and the cryptographic circuit are corrected prior to any cryptographic process performed by the circuit. Preferably the error correction occurs in one or more buffers employed to buffer the data between the cryptographic circuit and the memory where latency is reduced by using multiple buffers.

WO 2006/069273 A2

MEMORY SYSTEM WITH IN STREAM DATA ENCRYPTION/DECRYPTION AND ERROR CORRECTION

BACKGROUND OF THE INVENTION

[0001] This invention relates in general to memory systems, and in particular to a memory system with in stream data encryption/decryption and error correction.

[0002] The mobile device market is developing in the direction of including content storage so as to increase the average revenue by generating more data exchanges. This means that the content has to be protected when stored on a mobile device.

[0003] Portable storage devices are in commercial use for many years. They carry data from one computing device to another or to store back-up data. More sophisticated portable storage devices, such as portable hard disc drives, portable flash memory disks and flash memory cards, include a microprocessor for controlling the storage management.

[0004] In order to protect the contents stored in the portable storage devices, the data stored is typically encrypted and only authorized users are allowed to decrypt the data.

[0005] Since there may be bit errors in the data stored in portable storage devices, it is desirable to employ error correction. Current schemes for error correction may not be compatible with portable storage devices with cryptographic capabilities. It is therefore desirable to provide an improved local storage device where such difficulties are alleviated.

SUMMARY OF THE INVENTION

[0006] The data stored in the memory cells may contain errors for a number of reasons. It is therefore common to perform error correction when data from the memory cells are read. Error correction may also detect the positions of the errors in the data stream. The cryptographic processes performed by a circuit may shift the positions of the bits in the data stream so that if the bit errors in the data stream have not been corrected when such processes are performed, information on the positions of the bit errors will no longer be accurate after the processes so that error correction

may no longer be possible after the cryptographic processes have been performed. Thus one aspect of the invention is based on the recognition that the bit errors in the data in the data stream passing between the cells and the cryptographic circuit are preferably corrected prior to any cryptographic process performed by the circuit. Preferably, at least one buffer is used to store data in the data stream passing between the cells and the circuit and any error or errors in the data stored in the buffer and originating from the cells are corrected prior to cryptographic processing of the data by the circuit.

BRIEF DESCRIPTION OF THE DRAWINGS

[0007] Fig. 1 is a block diagram of a memory system in communication with a host device to illustrate the invention.

[0008] Fig. 2 is a block diagram of some of the blocks of the memory system in Fig. 1.

[0009] Fig. 3 is a circuit diagram illustrating in more detail a preferred configuration of the error correction buffer unit of Fig. 2.

[0010] Fig. 4 is a flow chart illustrating the operation of the system in Fig. 2 to illustrate the preferred embodiment of one aspect of the invention.

[0011] For convenience in description, identical components are labeled by the same numbers in this application.

DETAILED DESCRIPTION OF EXEMPLARY EMBODIMENTS

[0012] An example memory system in which the various aspects of the present invention may be implemented is illustrated by the block diagram of Fig. 1. As shown in Fig. 1, the memory system 10 includes a central processing unit (CPU) 12, a buffer management unit (BMU) 14, a host interface module (HIM) 16 and a flash interface module (FIM) 18, a flash memory 20 and a peripheral access module (PAM) 22. Memory system 10 communicates with a host device 24 through a host interface

bus 26 and port 26a. The flash memory 20 which may be of the NAND type, provides data storage for the host device 24. The software code for CPU 12 may also be stored in flash memory 20. FIM 18 connects to the flash memory 20 through a flash interface bus 28 and port 28a. HIM 16 is suitable for connection to a host system like a digital camera, personal computer, personal digital assistant (PDA), digital media player, MP-3 player, and cellular telephone or other digital devices. The peripheral access module 22 selects the appropriate controller module such as FIM, HIM and BMU for communication with the CPU 12. In one embodiment, all of the components of system 10 within the dotted line box may be enclosed in a single unit such as in memory card or stick 10' and preferably encapsulated in the card or stick.

[0013] The buffer management unit 14 includes a host direct memory access (HDMA) 32, a flash direct memory access (FDMA) controller 34, an arbiter 36, a buffer random access memory (BRAM) 38 and a crypto-engine 40. The arbiter 36 is a shared bus arbiter so that only one master or initiator (which can be HDMA 32, FDMA 34 or CPU 12) can be active at any time and the slave or target is BRAM 38. The arbiter is responsible for channeling the appropriate initiator request to the BRAM 38. The HDMA 32 and FDMA 34 are responsible for data transported between the HIM 16, FIM 18 and BRAM 38 or the CPU random access memory (CPU RAM) 12a. The operation of the HDMA 32 and of the FDMA 34 is conventional and need not be described in detail herein. The BRAM 38 is used to buffer data passed between the host device 24, flash memory 20 and the CPU RAM 12a. The HDMA 32 and FDMA 34 are responsible for transferring the data between HIM 16/FIM 18 and BRAM 38 or the CPU RAM 12a and for indicating sector transfer completion. As will be described below, the FIM 18 also has the capability of detecting errors in the data read from the flash memory 20 and notifying the CPU 12 when errors are discovered.

[0014] First when data from flash memory 20 is read by the host device 24, encrypted data in memory 20 is fetched through bus 28, FIM 18, FDMA 34, crypto engine 40 where the encrypted data is decrypted and stored in BRAM 38. The decrypted data is then sent from BRAM 38, through HDMA 32, HIM 16, bus 26 to the host device 24. The data fetched from BRAM 38 may again be encrypted by means of crypto engine 40 before it is passed to HDMA 32 so that the data sent to the host device 24 is again

encrypted but by means of a different key and/or algorithm compared to those whereby the data stored in memory 20 is decrypted. Preferably, and in an alternative embodiment, rather than storing decrypted data in BRAM 38 in the above-described process, which data may become vulnerable to unauthorized access, the data from memory 20 may be decrypted and encrypted again by crypto engine 40 before it is sent to BRAM 38. The encrypted data in BRAM 38 is then sent to host device 24 as before. This illustrates the data stream during a reading process.

[0015] When data is written by host device 24 to memory 20, the direction of the data stream is reversed. For example if unencrypted data is sent by host device, through bus 26, HIM 16, HDMA 32 to the crypto engine 40, such data may be encrypted by engine 40 before it is stored in BRAM 38. Alternatively, unencrypted data may be stored in BRAM 38. The data is then encrypted before it is sent to FDMA 34 on its way to memory 20. Where the data written undergoes multistage cryptographic processing, preferably engine 40 completes such processing before the processed data is stored in BRAM 38.

[0016] While the memory system 10 in Fig. 1 contains a flash memory, the system may alternatively contain another type of non-volatile memory instead, such as magnetic disks, optical CDs, as well as all other types of rewrite-able non volatile memory systems, and the various advantages described above will equally apply to such alternative embodiment. In the alternative embodiment, the memory is also preferably encapsulated within the same physical body (such as a memory card or stick) along with the remaining components of the memory system.

ERROR CORRECTION

[0017] Data stored in a non-volatile (e.g. flash) memory may become corrupted and contain errors. For this reason, FIM 18 may contain an error correction (ECC) circuit 102 that detects which bit or bits of the data stream from memory 20 contain errors, including the locations of the errors in the bit stream. This is illustrated in Fig. 2, which is a block diagram of a memory system 100 to illustrate another aspect of the invention. FIM 18 sends an interrupt signal to CPU 12 when error(s) is detected in the bit stream, and circuit 102 sends information concerning the locations of the bits in error to CPU 12. In conventional memory systems without cryptographic features,

the errors are corrected by the CPU in BRAM 38. However, if the data from the data stream is first cryptographically processed before the correction is made, the cryptographic process(es) may cause the locations and/or value(s) of the data bits in the processed data stream to change, so that the location(s) and/or value(s) of the bit errors after the cryptographic processing may be different from those sent to the CPU 12 by circuit 102. This may render it impossible to correct the errors when the cryptographically processed data reach the BRAM 38. An aspect of the invention stems from the recognition that the error(s) detected is corrected before the data is cryptographically processed, so that this problem is avoided.

[0018] An error buffer unit (EBU) 104 is used to store data from the data stream passing between the BMU 14 and FIM 18, so that when the CPU 12 receives an interrupt from FIM 18 indicating the presence of error(s) in the data stream, the CPU corrects the error(s) in EBU 104, instead of at the BRAM 38. To correct digital data, the bits in error are simply "flipped" (i.e. turning "1" to "0" and "0" to "1") at the locations of error(s) detected by circuit 102.

[0019] In order to reduce the amount of interruption in the data stream when errors are detected, two or more buffers may be employed in the EBU 104, such as shown in Fig. 3. As shown in Fig. 3, two buffers 104a and 104b are used, where one of the two buffers is receiving data from the memory 20 through FIM 18 and the other is sending data to the Crypto-Engine 40 through FDMA 34 in BMU 14. In Fig. 3, two switches 106a and 106b are used. When the two switches are in the solid line positions as shown in Fig. 3, buffer 104a is supplying data to the BMU 14 and buffer 104b is receiving data from FIM 18. When the two switches are in the dotted line positions as shown in Fig. 3, buffer 104b is supplying data to the BMU 14 and buffer 104a is receiving data from FIM 18. Each of the buffers can first be filled with data before data stored in it is sent to the BMU. The CPU corrects the error(s) in the buffer(s) 104a and 104b when data is sent from or received by them. In this manner, the only latency is the time required to fill one of the two buffers when the data stream is started. After that, there will be no interruption in the data stream even when error(s) have been detected by circuit 102, if the time taken by the CPU to correct the error(s) is small compared to the time needed to fill each buffer.

[0020] If correcting the data takes longer then filling a buffer, the data stream will be interrupted only when errors are detected and the data stream will flow without interruption when no errors are detected. A buffer-empty signal (not shown) connecting between the EBU 104 and the FDMA 34 signals the latter that the data stream is interrupted and no more data is available. The FDMA 34 as well as the crypto engine 40 will then pause and wait for the data stream to resume.

[0021] When data is written by the host device 24 to memory 20, there may be no need for error correction, so that it would be desirable to bypass the EBU. This may be accomplished by switch 108. When switch 108 is closed, the data from HIM 16 (not completely shown in Fig. 2) simply bypasses the two buffers 104a and 104b. Switch 108 may also be closed in a bypass mode where no cryptographic processing is needed when data is read from or written to memory 20. In this mode, HDMA and FDMA are connected directly to arbiter 36 as if crypto-engine 40 is eliminated from system 10, and the data stream bypasses both the EBU 104 and the Crypto-Engine 40. This may be accomplished also by using switches. Hence, in the bypass mode, a logic circuit (not shown) in system 100 under the control of CPU 12 causes the data stream to bypass block 40 and causes switch 108 to close.

[0022] The error correction process is illustrated by the flow chart of Fig. 4. The CPU 12 starts a read operation after receiving a read command from the host device 24 (ellipse 150). It then configures the Crypto-Engine 40 using appropriate security configuration information, and configures the BMU 14 for a reading operation, and other parameters such as the allocation of memory space in BRAM 38 for the operation (blocks 152, 154). It also configures the FIM 18, such as by specifying the locations in memory 20 where data is to be read (block 156). The HDMA and FDMA engines 32 and 34 are then started. See Block 158. When the CPU receives an interrupt, it checks to see whether it is a FIM interrupt (diamond 160). When a FIM interrupt is received, the CPU checks to see whether the interrupt is one indicating that there is one or more errors in the data stream (162). If error(s) is indicated, it proceeds to correct the error(s) (block 164) in buffers 104a and/or 104b and returns to configure the FIM 18 to change the locations in memory 20 where data is to be read next (block 156). When the FIM interrupt does not indicate error(s) in the data stream, it means the FIM has completed its operation and the CPU also returns to

block 156 to re-configure and restart the FIM. If the interrupt detected by the CPU is not a FIM interrupt, it checks to see if it is an end of data interrupt (diamond 166). If it is, then the read operation ends (ellipse 168). If not, this interrupt is irrelevant to the cryptographic processing of the data (i.e. clock interrupt) and the CPU 12 services it (not shown) and returns to diamond 160 to check for interrupts.

[0023] Fig. 4 needs only to be modified slightly for a write operation. Since there is no handling of ECC errors in the data to be written to memory 20, the CPU 12 can skip the processes in diamond 162 and block 164 in a write operation. If a FIM interrupt is received by the CPU 12 during a write operation, this means that the FIM completed its operation and the CPU also returns to block 156 to re-configure the FIM. Aside from this difference, the write operation is substantially similar to the read operation.

[0024] While the invention has been described above by reference to various embodiments, it will be understood that changes and modifications may be made without departing from the scope of the invention, which is to be defined only by the appended claims and their equivalent. All references referred to herein are incorporated by reference.

CLAIMS

1. A method for correcting data in a memory system for storing encrypted data comprising non-volatile memory cells and a cryptographic circuit, said method comprising:

using the circuit to perform cryptographic processes on data in a data stream from or to the cells;

providing at least one buffer to store data in the data stream passing between the cells and the circuit; and

correcting any error(s) in the data stored in the buffer and originating from the cells prior to performance of cryptographic processes on the data by the circuit.

2. The method of claim 1, wherein the correcting is in response to a signal indicating the presence of one or more error(s) in data in the data stream from the cells and destined for the circuit, so that the correcting corrects the one or more error(s) before the data reach the circuit.

3. The method of claim 2, said system comprising two buffers storing data in the data stream passing between the cells and the circuit, said method further comprising using the two buffers to alternately store and sent data from the cells to the circuit.

4. The method of claim 3, wherein said using stores data in a first one of the two buffers when data stored in a second one of the two buffers is sent to the circuit.

5. A memory system for storing encrypted data, comprising:

non-volatile memory cells;

a circuit performing cryptographic processes on data in a data stream from or to the cells;

at least one buffer storing data in the data stream passing between the cells and the circuit; and

a controller controlling the cells, the at least one buffer and the circuit to correct any error(s) in the data stored in the buffer and originating from the cells prior to the performing of cryptographic processes on such data by the circuit.

6. The system of claim 5, wherein the controller responds to a signal indicating the presence of one or more error(s) in data in the data stream from the cells and destined for the circuit and corrects the one or more error(s) before the data reach the circuit.

7. The system of claim 6, further comprising an error correction circuit that detects error(s) in the data stream and causes the signal to be sent to the controller when at least one error in the data stream is detected.

8. The system of claim 5, said system comprising two buffers storing data in the data stream passing between the cells and the circuit, wherein the two buffers are alternately used to store and sent data from the cells to the circuit.

9. The system of claim 8, wherein data is being stored in a first one of the two buffers when data stored in a second one of the two buffers is sent to the circuit.

10. The system of claim 5, wherein the controller causes data in the data stream to bypass the buffer and the circuit in an alternate bypass path in a bypass mode.

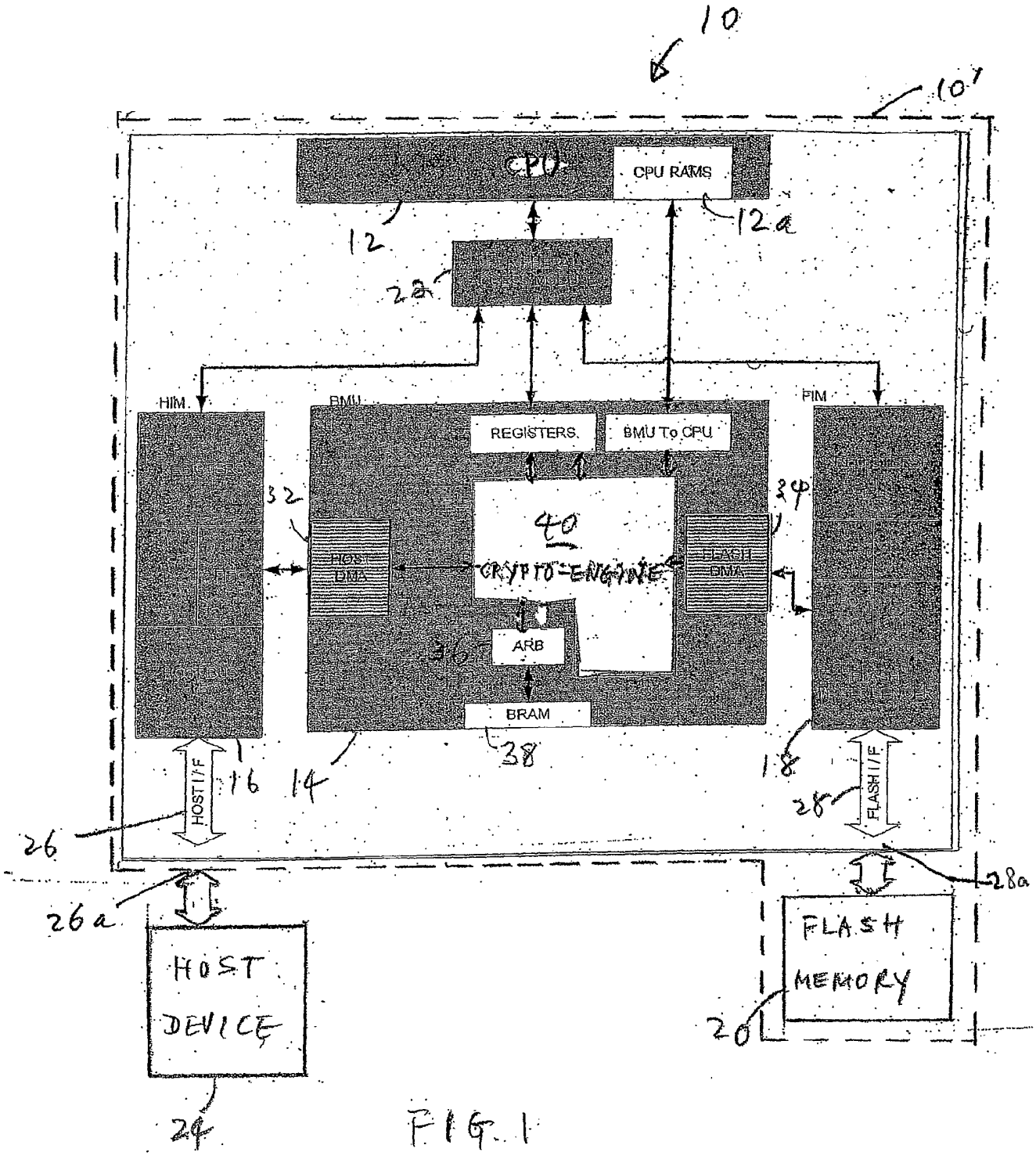

10
10'
CPU
CPU RAMS
12
12a
22
HIM
BMU
PIM
REGISTERS
BMU To CPU
32
34
40
CRYPTO-ENGINE
HOST DMA
FLASH DMA
ARB
BRAM
38
16
14
18
HOST I/F
FLASH I/F
26
26a
28
28a
HOST DEVICE
FLASH MEMORY
20
24

FIG. 1

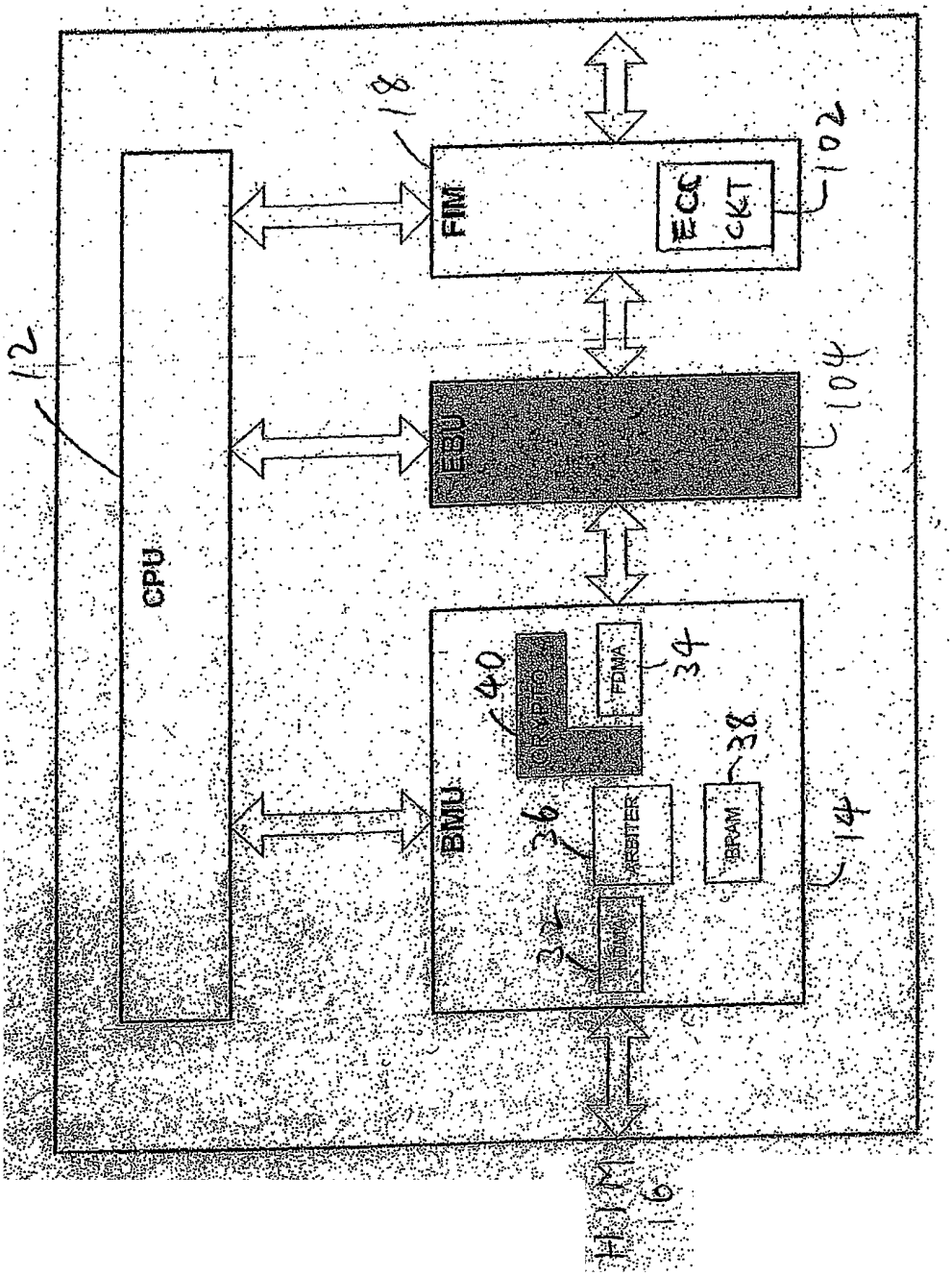

100
18
FIM
ECC CKT
102
12
CPU
EBU
104
BMU
40
CRYPTO
FDMA
34
38
36
ARBITER
BRAM
32
FDMA
14
16

FIG. 2

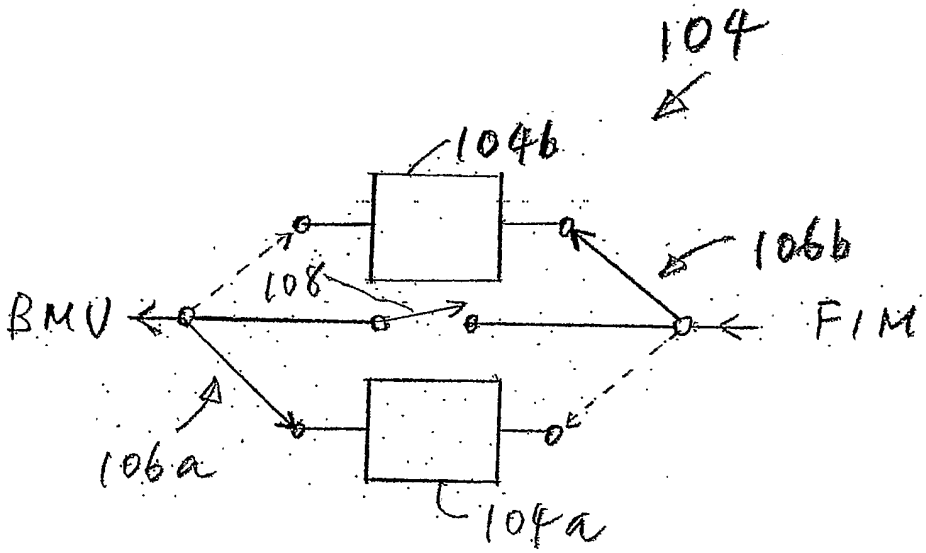
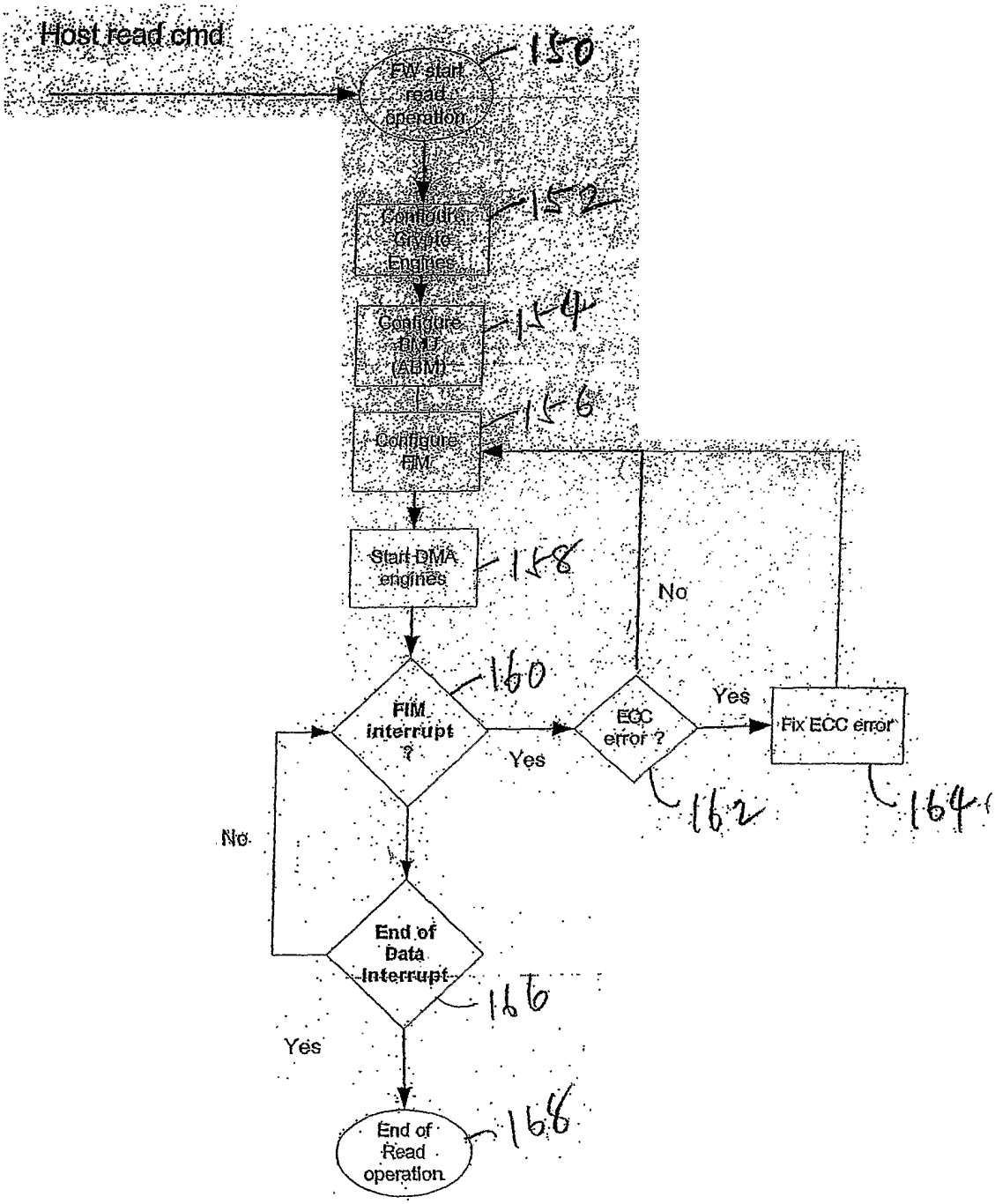


FIG. 3


Host read cmd
FW start read operation
150
Configure Crypto Engines
152
Configure RMT (ABM)
154
Configure FIM
156
Start DMA engines
158
FIM interrupt ?
160
ECC error ?
162
Yes
Fix ECC error
164
No
Yes
No
End of Data Interrupt
166
Yes
End of Read operation
168

F.I.G. 4