IPOS Copy
9703791-5

Specification

1. Title of the Invention:

Method for executing an error recovery procedure

2. Detailed Description of the Invention:

[0001]

[Field of the invention]

The present invention is related to an error recovery method in a storage device. More specifically, it is related to an error recovery procedure (ERP).

[0002]

[Background art]

A disk drive is a device for recording data on the surface of a disk and reading it. A head having a signal transducer is positioned at a desired location on a track on which data is recorded, and it reads or writes data. As a means to recover an error which occurs when data is read or written by the head, there is an error recovery procedure (ERP) comprising various error recovery steps.

[0003]

On a disk, a read error is made to occur by various causes such as a flaw, ununiformity of the magnetic substance, or change of the magnetic substance with time. Generally, for errors occurring in the data portion, an ECC, a conventional error recovery code, is used to execute an error recovery process. Further, various recovery procedures are executed, such as change of the read gain, change of the offtrack, and change of the bias value of an MR (magnetoresistive) element if it is used as the read head. If data is successfully reread by executing such recovery procedures, the data is continuously used. If the error cannot be recovered by a plurality of error recovery procedures, it is recognized as a hard error, or if the data can be reassigned to another area, the corresponding area is made an unusable area and a process for reassigning the data to the other area is performed.

[0004]

In many of the recent disk drives, a magnetoresistive (MR) transducer head is used. This utilizes a very small resistance change due to change in a magnetic field. By causing a d.c.

current to flow through the MR element, a resistance change is converted to a voltage signal to perform the reading of data.

[0005]

However, as one of the read errors which occur in this approach of reading a resistance change, there is thermal asperity. The thermal asperity means that a protrusion or deposit generated on a disk crashes against the read head and generates a resistance change due to a temperature change in the MR element, causing an abnormal signal.

[0006]

As one of the countermeasures against the thermal asperity, there is a technique called low rpm burnish. The low rpm burnish is an approach in which the rpm (revolutions per minute) of the disk is decreased to reduce the fly height of the magnetic head, thereby to positively scrape off the protrusion on the disk which caused the thermal asperity. This also forms one of the abovementioned error recovery procedures (ERPs).

[0007]

There are various countermeasures against errors during data reading or writing, these are generally stored as a series of steps in an error recovery procedure. When the error recovery procedure is started, these steps are sequentially executed.

[0008]

The causes of errors are not always the same, and there may be various causes. Accordingly, error recovery procedures meeting these various causes of errors are required. An error recovery procedure generally performs a rereading by changing and

adjusting the standard read conditions one by one, which are defined among the disk, magnetic head, and HDC (hard disk controller). The read conditions are for instance, offtrack amount which is an offset amount between the center of the magnetic head and the center of a track, bias current value given to an MR element if the magnetic head includes the MR element, adjustment of the auto-gain control (AGC) for making the amplitude of a reproduced signal constant, adjustment of the speed and gain of the PLL circuit for stabilizing the sampling frequency (making the tracking speed a predetermined speed), and the like.

[0009]

Normally, a plurality of error recovery steps are registered as an ERP. Those steps are executed in a predetermined order. A reread is performed for each step, and the ERP terminates when the reread has succeeded. If the reread is not successful, the ERP terminates when a preset maximum number of rereads is reached or the last step of the ERP is ended. Further, the ERP is also caused to terminate by the time-out of a read or write instruction.

[0010]

For a time-out, the host system may perform the retry of the instruction in the time-out state after executing a reset (soft reset or hard reset) for the disk drive. If a write or read error occurs again during the retry, the ERP is executed from the first step.

[0011]

It may take 10 to 20 seconds to execute the steps of the above low rpm burnish. On the other hand, the time elapsing before the

time-out of a read or write instruction depends on the operating system (OS) or BIOS of the host system, and conventionally, it was often about 30 seconds, but it may be only several seconds in the recent OSs such as OS/2 (a trademark) of IBM Corp. For this, the time-out of the read or write instruction may be reached before the all ERP steps are completed. In such a case, only part of the ERP steps is executed because of the limited time to the time-out, and thus there is a high possibility that, even if there is an effective step after that, an error occurs without the execution of that step. Further, for a read or write instruction to a plurality of sectors, if errors occur in a plurality of sectors, the ERP steps are executed in each sector, and thus the read or write instruction may reach the time-out before the completion of the ERP steps for the all sectors in error even if the time to the time-out is relatively long.

[0012]

[Problems to be solved by the invention]

It is an object of the present invention to provide a method and apparatus which ensures the execution till the last step of the ERP regardless of the time to the time-out of an instruction, thereby to more reliably recover an error.

[0013]

Further, it is an object of the present invention to provide an error recovery method and apparatus suitable for an OS in which the time to time-out is short.

[0014]

Furthermore, it is an object of the present invention to provide a method and apparatus which ensures the execution of the ERP

steps requiring a relatively long execution time such as a low rpm burnish, thereby to more reliably recover an error.

[0015]

[Summary of the invention]

In accordance with the present invention, even if the disk drive receives a reset instruction during the execution of an ERP, it executes the ERP to the last step till the recovery without interrupting the ERP.

That is, the present invention provides a method for executing an error recovery procedure (ERP) comprising a plurality of error recovery steps, the method comprising: executing error recovery steps, and executing the next error recovery step regardless of whether or not a predetermined signal has been received.

Further, in accordance with another aspect of the present invention, if the disk drive receives a reset signal during the execution of an ERP, it stops the execution of the ERP and holds the number K of the step the execution of which has been completed immediately before the stopping, and when receiving a retry instruction thereafter, it sequentially executes the ERP from the K+1-th error recovery step.

That is, the other aspect of the present invention is a method for executing an error recovery procedure (ERP) comprising a plurality of error recovery steps, the method comprising: executing error recovery steps; in response to a first signal, stopping the execution of error recovery steps and holding the number K of the step the execution of which has been completed immediately before the stopping; and in response to a second signal, sequentially executing the ERP from the K+1-th error

recovery step for the sector for which the execution of the ERP has been stopped.

[0016]

[Preferred embodiment]

Figure 1 is a block diagram of the hard disk drive (HDD) to which the present invention is applied. A disk drive 100 consists of a controller unit 110 and a disk unit 130. The controller unit 110 includes a host interface controller (HIC) 112 connected to a host system 10, a hard disk controller (HDC) 114 connected to the host interface controller 112 for controlling the disk unit, a channel 116 connected to the hard disk controller 114 for controlling read and write signals, an MPU 118 connected to HIC 112, HDC 114 and the channel 116 for controlling them, RAM 120 connected to MPU 118 for storing the microcode executed by the MPU. The disk unit 130 includes a motor 134 for rotating a spindle 132. On the spindle 132, disks 136A and 136B are mounted so as to rotate integrally with the spindle 132. The number of disks is two in the figure, but it may be one or more than two.

[0017]

Heads 138A, 138B, 138C, and 138D are disposed and supported by actuator arms 140A, 140B, 140C, and 140D to face the surface of the disk respectively. The actuator arms 140A to 140D are attached to a voice coil motor (VCM) 144 via a pivot shaft 142, and the rotation of the VCM 144 causes the heads 138A to 138D to move to a desired radial position. The motor 134 and the VCM 144 are connected to the HDC 114 to control the rpm, speed, and the like of them. The heads 138A to 138D are connected to the channel 116, and the read write signals are controlled by the channel 116.

[0018]

Generally, in the magnetic disk of a hard disk or the like, data tracks are concentrically formed. The reading or writing of information from/to the disk is performed after the disk is rotated and the head is moved along substantially the radial direction of the disk so as to be positioned at a particular data track (seek). The positioning of the head at a particular data track is carried out by reading the head position identifying information prerecorded on the disk.

[0019]

After the head is positioned at a desired track, data reading or writing is performed. At this time, the failure of the reading or writing of correct data, or an error, may occur. Many of the causes of errors occurring on the disk are the loss of data. The data loss is often caused by a flaw on the disk surface, which occurs with time, a damage of the magnetic substance due to a change with time or the like. Usually, the data reading from the disk is carried out under the standard read conditions set in the disk, head, and HDC.

[0020]

If an error occurs in the reading according to these predetermined standard conditions, the reading is carried out again by intentionally changing the trackability, or maintaining the amplification factor of the auto-gain control (AGC) for the signal amplification. These error recoveries by changing the conditions are performed by activating an ERP (error recovery procedure) in which a plurality of error recovery steps are sequentially executed.

[0021]

If a writing cannot be successfully made even by the ERP, the error is recognized to be a hard error, and data reassignment is performed for recording the data in the particular sector in an alternate sector called a spare sector. Further, in reading, if a data reading is successfully made in predetermined ERP steps, the writing for moving data to a spare sector is performed, and simultaneously data reassignment is made.

[0022]

The error recovery steps included in the ERP, include for instance, as steps which can be processed in a short time, a. AGC hold, b. offtrack read, c. skip reading of the servo region, and the like. Further, as steps requiring a relatively long time, there are d. butterfly seek, and e. low rpm burnish. In the butterfly seek, a seek operation is repeated several times in the vicinity of the track in which an error has occurred, to give positional control of the head to a target track again. Further, the low rpm burnish is to reduce the fly height of the head to remove the protrusions, dusts or the like on the disk by the structure of the head for error recovery.

[0023]

Figure 2 shows the method for executing the ERP in the background art. The operation is described below.

(1) A read or write instruction is transferred from the host system to the HDD, which responds to it to execute a read or write.
(2) If a read or write error occurs in a logical block address (LBA) A, the ERP is invoked from the RAM, and the MPU

sequentially executes the ERP from step 1 of the ERP through the channel.

(3) In a system in which the time elapsing before time-out is shorter than the execution time of the all steps of the ERP (assumed to be M steps), if the error is still not recovered when step K (<M) of the ERP is executed, the MPU receives a reset instruction from the host system, the ERP stops at step K, and the HDD sends a reset completion signal to the host. If the error is recovered while the ERP is executed to step K, the HDD sends the recovered data to the host.

(4) A retry instruction is communicated from the host system to the HDD, which responds to it for executing a reread or rewrite. The number of retry instructions depend on the host system.

(5) If a read or write error occurs again in the LBA A, the ERP is invoked from the RAM, and the MPU sequentially executes the ERP again from step 1 of the ERP through the channel. If the error is not recovered, the probability of an error occurring again in the same LBA is generally high.

(6) The time to time-out is usually constant in a system, and accordingly, if the error is still not recovered when step K of the ERP is executed, the MPU receives a reset instruction from the host system, the ERP stops at step K of the ERP, and the HDD transmits a reset completion signal to the host.

(7) If a retry instruction is further transmitted, (4) to (6) are repeated.

[0024]

These steps are stored as a microcode program in a memory such as a ROM on the circuit board. Further, the all or part of the steps may be stored in the disk. In this case, they are read out from the disk when the hard disk is started up, and stored in a memory such as a RAM for execution.

[0025]

In the above described background art, since the time to time-out is shorter than the execution time of the all steps and the ERP is executed again from step 1 in a retry, steps K+1 to M of the ERP are not executed. For this, even if a step effective for an error exists among steps K+1 to M of the ERP, the possibility of the error being recovered is low because that step is not executed.

[0026]

Figure 3 shows the first embodiment of the present invention. The operation is described below.

(1) A read or write instruction is transferred from the host system to the HDD, which responds to it for executing a read or write.
(2) If a read or write error occurs at the LBA A, the ERP is invoked from the RAM, and the MPU sequentially executes the ERP from step 1 through the channel.
(3) In a system in which the time to time-out is shorter than the execution time of the all steps of the ERP (M steps), if the error is still not recovered when step K (<M) of the ERP is executed, the MPU receives a reset instruction from the host system. At this point, the ERP does not stop, and step K+1, and the steps after that are continuously executed. Thus, in the first embodiment, the next step of the ERP is executed regardless of whether or not a reset instruction is received from the host system.
(4) In reading, if the error is recovered during the execution of steps K+1 to M of the ERP, the recovered data is stored in a buffer memory such as a RAM in the HDD, and the host system is not notified of the recovery. In addition, a reassignment operation for writing the read data to another physical block

address (PBA) is performed. In writing, if the writing is successfully made during the execution of steps K+1 to M of the ERP, the host system is not notified of the success.

(5) If the error is still not recovered when step M of the ERP is executed, it is recognized to be a hard error for a read instruction, and for a write instruction, data is reassigned to another PBA and data is written to the reassigned PBA.

(6) After performing a reset operation, the HDD transmits a reset completion signal to the host. This step may be executed before step (4) or (5), or in the middle of step (4) or (5).

(7) A retry instruction is communicated from the host system to the HDD, which responds to it for executing a reread or rewrite. A) If, in reading, the error is recovered in the previous steps, the data of the LBA A, which is stored in the buffer memory, is read out from the buffer memory rather than the disk. B) If, in reading, the error is not recovered in the previous steps and if an error occurs in the reread, then the ERP is executed from step 1. C) If, in writing, the write is successfully made in the previous steps, data is overwritten. D) If, in writing, the write fails in the previous steps, or reassignment is made, then data is overwritten to the reassigned PBA.

(8) If the reread is successfully made, the HDD sends the read data to the host, and if the rewrite is successfully made, the HDD transmits an instruction completion signal to the host.

[0027]

In the above described first embodiment, since the steps of the ERP are executed until data is recovered regardless of the time to time-out, there is not the drawback that the steps after a particular step are always not executed. Further, since the all ERP steps are given a chance that they are executed until data is recovered, the probability of error recovery is high. Further, in reading, since the data in the LBA A, which is recovered by the ERP steps after time-out, is stored in the

buffer and read out from the buffer memory in a retry, it is not necessary to execute the ERP again in the reading of the LBA A. On the other hand, the steps including the reassignment are completed if the write fails, and thus, also in wiring, it is not necessary to execute the ERP again in the writing of the ERP. Accordingly, a fast and reliable response can be expected for a retry.

[0028]

Figure 4 shows the second embodiment of the present invention. The operation is described below.

(1) A read or write instruction is transmitted from the host system to the HDD, which responds to it for executing a read or write.

(2) If a read or write error occurs in the LBA A, the ERP is invoked from the RAM, and the MPU sequentially executes the ERP from step 1 through the channel.

(3) In a system in which the time to time-out is shorter than the execution time of the all steps of the ERP (M steps), if the error is still not recovered when step K (<M) of the ERP is executed, the MPU receives a reset instruction from the host system. At this point, the MPU stops the execution of the ERP, and holds in a memory the number K of the step the execution of which has been completed immediately before the stopping.

(4) After performing a reset operation, the HDD transmits a reset completion signal to the host.

(5) A retry instruction is transmitted from the host system to the HDD, which responds to it for executing a reread or rewrite.

(6) If a read or write error occurs again in the LBA A, the number K of the step the execution of which has been completed immediately before that is invoked from the memory, and the MPU sequentially executes the ERP from step K+1 of the ERP through the channel. If no read or write errors occurs in the LBA A, the process goes to step (7). In addition, if a new read or write

error occurs in an LBA other than the LBA A, the ERP is executed from step 1 as usual.

(7) If the reread is successfully made, the HDD sends the read data to the host, and if the rewrite is successfully made, the HDD transmits an instruction completion signal to the host.

(8) If the error is not recovered when step M of the ERP is executed, it is recognized to be a hard error for a read instruction, and for a write instruction, data is reassigned to another physical block address (PBA), and data is written to the reassigned PBA.

[0029]

In the above described second embodiment, even if the execution of the ERP is interrupted by time-out, the ERP is executed from the next step in a retry, and thus, there is not the defect that the steps after a particular step are always not executed, as in the first embodiment. Further, since the all ERP steps are given a chance that they are executed until a recovery is obtained, the probability of error recovery is high.

[0030]

[Advantages of the invention]

In accordance with the present invention, there is not the disadvantage that the steps after a particular step of ERP are always not executed, and the all ERP steps are given a chance that they are executed until a recovery is obtained. Accordingly, the probability of error recovery is high. Thus, the various steps included in the error recovery procedure can be efficiently executed, and the reliability of error recovery can be increased.

3. Brief Description of the Drawings:

Figure 1 is a block diagram of the hard disk drive (HDD) to which the present invention is applied;

Figure 2 is a diagram showing the method for executing the ERP in the background art;

Figure 3 is a diagram showing the first embodiment of the present invention; and

Figure 4 is a diagram showing the second embodiment of the present invention.

[Description of symbols]

10 ... Host system
100 ... Disk drive
110 ... Disk drive unit
112 ... Host interface controller (HIC)
114 ... Hard disk controller (HDC)
116 ... Channel
118 ... MPU
120 ... RAM
130 ... Disk unit
132 ... Spindle
134 ... Motor
136A, 136B ... Disks
138A to 138D ... Heads
140A to 140D ... Actuator arms
142 ... Pivot shaft
144 ... Voice coil motor (VCM)



4. Claims:

(1) A method for executing an error recovery procedure (ERP) comprising a plurality of error recovery steps on a storage device connected to a host system, said method comprising:

executing error recovery steps; and

executing the next error recovery step regardless of whether or not a predetermined signal has been received.

(2) A method for executing an error recovery procedure as set forth in claim 1, wherein said predetermined signal is a signal corresponding to a reset instruction from said host system.

(3) A method for executing an error recovery procedure as set forth in claim 1, wherein said error recovery steps are executed until the error is recovered.

(4) A method for executing an error recovery procedure as set forth in claim 1, wherein said error recovery steps are executed until the last step is completed.

(5) A method for executing an error recovery procedure as set forth in claim 1, wherein data recovered by the error recovery steps after the reception of said predetermined

signal in a read mode is stored in a memory in said storage device.

(6) A method for executing an error recovery procedure as set forth in claim 5, wherein the recovered data stored in said memory is returned in response to a retry instruction from said host system.

(7) A method for executing an error recovery procedure as set forth in claim 1, wherein the position of data to be written is reassigned if the writing is not successful till the completion of the last step in a write mode.

(8) A method for executing an error recovery procedure as set forth in claim 7, wherein data is written to the reassigned position in response to a retry instruction from said host system.

(9) A method for executing an error recovery procedure as set forth in claim 1, wherein the time elapsing before the time-out of an instruction is shorter than the execution time of the all steps of the ERP.

(10) A method for executing an error recovery procedure (ERP) comprising a plurality of error recovery steps in a storage device connected to a host system, said method comprising:

executing error recovery steps;

in response to a first signal, stopping the execution of error recovery steps and holding the number K of the step the execution of which has been completed immediately before the stopping; and

in response to a second signal, sequentially executing the ERP from the K+1-the error recovery step.

(11) A method for executing an error recovery procedure as set forth in claim 10, wherein said first signal is a signal corresponding to a reset instruction from said host system.

(12) A method for executing an error recovery procedure as set forth in claim 10, wherein said second signal is a signal corresponding to a retry instruction from said host system.

(13) A method for executing an error recovery procedure as set forth in claim 10, wherein said error recovery steps are executed until the error is recovered.

(14) A method for executing an error recovery procedure as set forth in claim 10, wherein said error recovery steps are executed until the last step is completed.

(15) A method for executing an error recovery procedure as set forth in claim 10, wherein the time elapsing before the time-out of an instruction is shorter than the execution time of the all steps of the ERP.

(16) An apparatus for executing an error recovery procedure (ERP) comprising a plurality of error recovery steps, said apparatus comprising:

a logic for executing error recovery steps; and

a logic for receiving a signal,

wherein said logic for executing error recovery steps executes the next error recovery step regardless of whether or not said logic for receiving a signal has received a predetermined signal.

(17) An apparatus for executing an error recovery procedure (ERP) comprising a plurality of error recovery steps, said apparatus comprising:

a logic for executing error recovery steps;

a logic for receiving a signal, and

a memory connected to said logic for executing error recovery steps,

wherein said logic for executing error recovery steps responds to a first signal to stop the execution of error recovery steps and store in said memory the number K of the step the execution of which has been completed immediately before the stopping, and responds to a second signal to read out the value of K from said memory and sequentially execute the ERP from the K+1-th error recovery step.

(18) A storage device comprising:

a storage medium;

a transducer for recording information on said storage medium or reproducing it therefrom;

a storage unit for storing error recovery steps;

a logic connected to said storage unit for executing error recovery steps; and

an interface connectable to an external device,

wherein said logic for executing error recovery steps executes the next error recovery steps regardless of whether or not said interface has received a predetermined signal.

(19) A storage device comprising:

a storage medium;

a transducer for recording information on said storage medium or reproducing it therefrom;

a memory for storing error recovery steps;

a logic connected to said storage unit for executing error recovery steps; and

an interface connectable to an external device,

wherein said logic for executing error recovery steps responds to a first signal received by said interface to stop the execution of error recovery steps and store in said memory the number K of the step the execution of which has been completed immediately before the stopping, and responds to a second signal received by said interface to read out the value of K from said memory and sequentially execute the ERP from the K+1-th error recovery step.

(20) A medium in which a program is stored for executing an error recovery procedure (ERP) comprising a plurality of error recovery steps, said program comprising: executing error recovery steps; and

executing the next error recovery step regardless of whether or not a predetermined signal has been received.

(21) A medium in which a program is stored for executing an error recovery procedure (ERP) comprising a plurality of error recovery steps, said program comprising:

executing error recovery steps;

in response to a first signal, stopping the execution of error recovery steps and holding the number K of the step the execution of which has been completed immediately before the stopping; and

in response to a second signal, sequentially executing the ERP from the K+1-th error recovery step.