X3T10/94-111 r 2

RAID 5 Support on SCSI Disk Drives

Rev. 1.5
7-13-94
Seagate Technology

Contents

Table of Figures

Table of Tables

Note:
For Questions or Comments on this document Contact: Mike Miller (612) 844-5924, Email Mike_Miller@notes.seagate.com or Jay Elrod (612) 844-5978, Email Jay_Elrod@notes.seagate.com

Introduction

In recent years advances in processor performance have far outpaced advances in I/O performance. In addition, the requirement to have 100% on-line protection of user data has become more prevalent. Many companies have recognized this and have developed subsystem architectures that attempt to deal with these problems. One of the most prevalent techniques has been to incorporate a RAID 5 architecture into the subsystem. This is usually done with a special disc controller that has additional hardware functionality to support the RAID 5 architectural requirements. These controllers typically incorporate multiple SCSI buses and an XOR engine for parity generation. If an XOR engine were incorporated into the drive the RAID 5 architecture could be accomplished using standard SCSI controllers. Incorporation of an XOR engine into the Disc drives has been done or at least considered many times in the past. With the advent of higher performance interfaces (such as Fibre Channel) it now makes more sense than ever before to include this functionality in the drive.

This document describes the XOR functionality that is designed into the drive and it shows how this functionality can be used by a RAID controller to accomplish some of the tasks it needs to perform to implement a RAID 5 architecture. The first part of the document describes the XOR related tasks that need to be performed in an environment where multiple SCSI busses are used to control the drives. The second part of the document describes how these same functions would be performed in a single bus architecture (such as Fibre Channel). The third part of the document describes the new SCSI commands that are needed to perform the XOR functions. The forth part of the document describes some additional drive functionality that will increase the performance of write operations in a RAID 5 environment.

RAID 5 Support for Multiple Interface Striping Configurations

There are three primary operations that need to be supported to eliminate the XOR engine from the controller. They are, Read-Modify-Write, Regenerate, & Rebuild. The Read-Modify-Write function is used any time a write operation needs to be performed on the array. The Regenerate function is used to regenerate data from the array when a data drive has malfunctioned. In this situation the Regenerate function takes the place of a normal Read operation. The Rebuild function is used to rebuild a drive that has been added to the array in place of a failed drive. The Rebuilt data is written on the new drive as part of this function.

In multiple interface striping configurations the drives do not necessarily have the capability of peer to peer communication. In this environment all data must pass through the host. This section of the document shows how the three basic Functions (described above) are performed using the XOR engine in the drive along with the new SCSI commands.

READ-MODIFY-WRITE Function

(Drives on separate buses or loops)

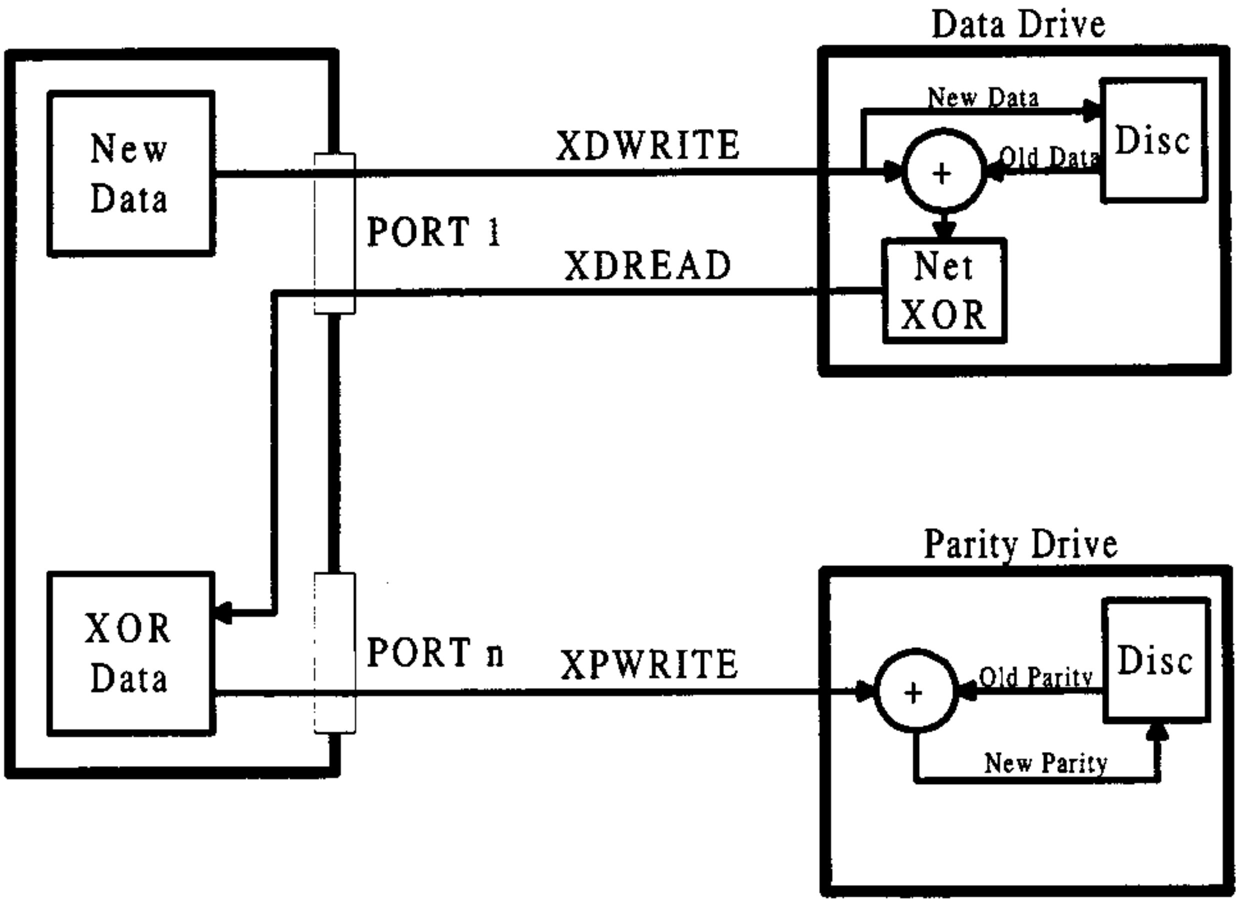
The Read-Modify-Write function is used to write new data to the array. Three new SCSI commands are used to accomplish this. They are XDWRITE, XDREAD, & XPWRITE.

The host begins by sending the data drive new data using an XDWRITE command, with the Secondary Control bits set to 11 to indicate that the host will later be retrieving xor'd data. It also sends the parity drive the XPWRITE command (command phase only at this point - this gets the parity drive started reading old parity from the disc into its buffer). The data drive reads old data from the disc into its buffer, exclusive ors the old data with the new data from the host, stores the xor'd data in its buffer, and writes the new data from the host to the disc. Good XDWRITE ending status is not sent to the host until the exclusive or'd data is available in the buffer.

The host reads the xor'd data by sending the data drive an XDREAD command with the same logical block address and transfer length as in the associated XDWRITE command. (The data drive is required to retain the xor'd data until successfully completing the associated XDREAD command.)

The host then sends the xor'd data to the parity drive during the data phase of the already issued XPWRITE command. The parity drive exclusive ors this data with the old parity data in its buffer. The resulting new parity data is written to the disc.

Figure 1: PARALLEL READ-MODIFY-WRITE



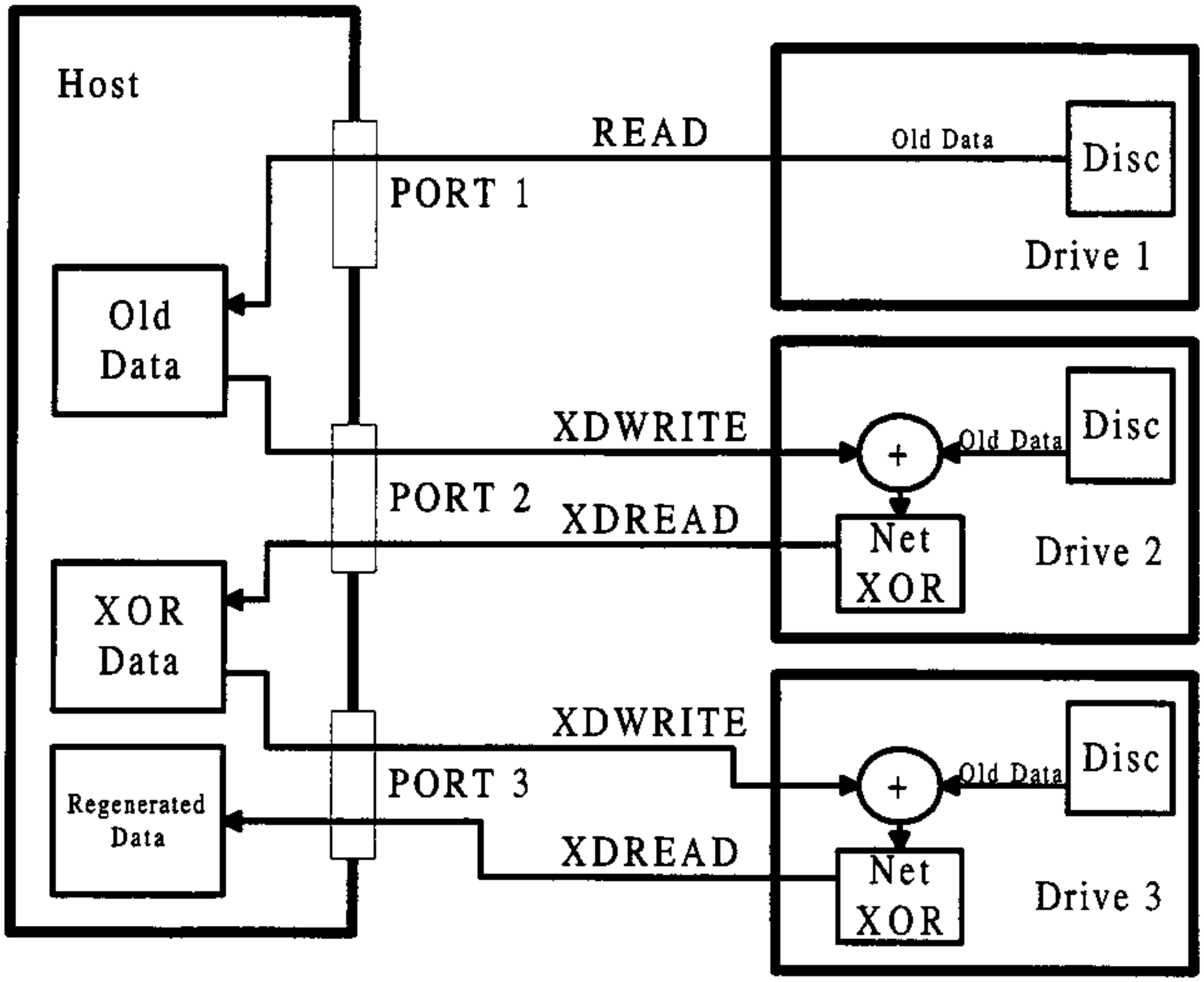
REGENERATE Function

(Drives on separate buses or loops)

This function is used in place of a Read command when one data drive in the array has malfunctioned. The host begins by sending the first source drive a READ command. The data received from this drive is sent to the second source drive using an XDWRITE command with the NDISC bit set (to prevent data from being written to disc). The second source drive reads old data from the disc, exclusive ors this old data with the data sent from the host, and stores the result in its buffer. The host retrieves the xor'd data by sending the second drive an XDREAD command with the same logical block address and transfer length as in the associated XDWRITE command. (The second drive is required to retain the xor'd data until successfully completing the associated XDREAD command.) The host issues the XDWRITE and XDREAD commands in a likewise manner for each successive source drive.

The xor'd data from the last source drive is the regenerated data.

Figure 2: PARALLEL REGENERATE



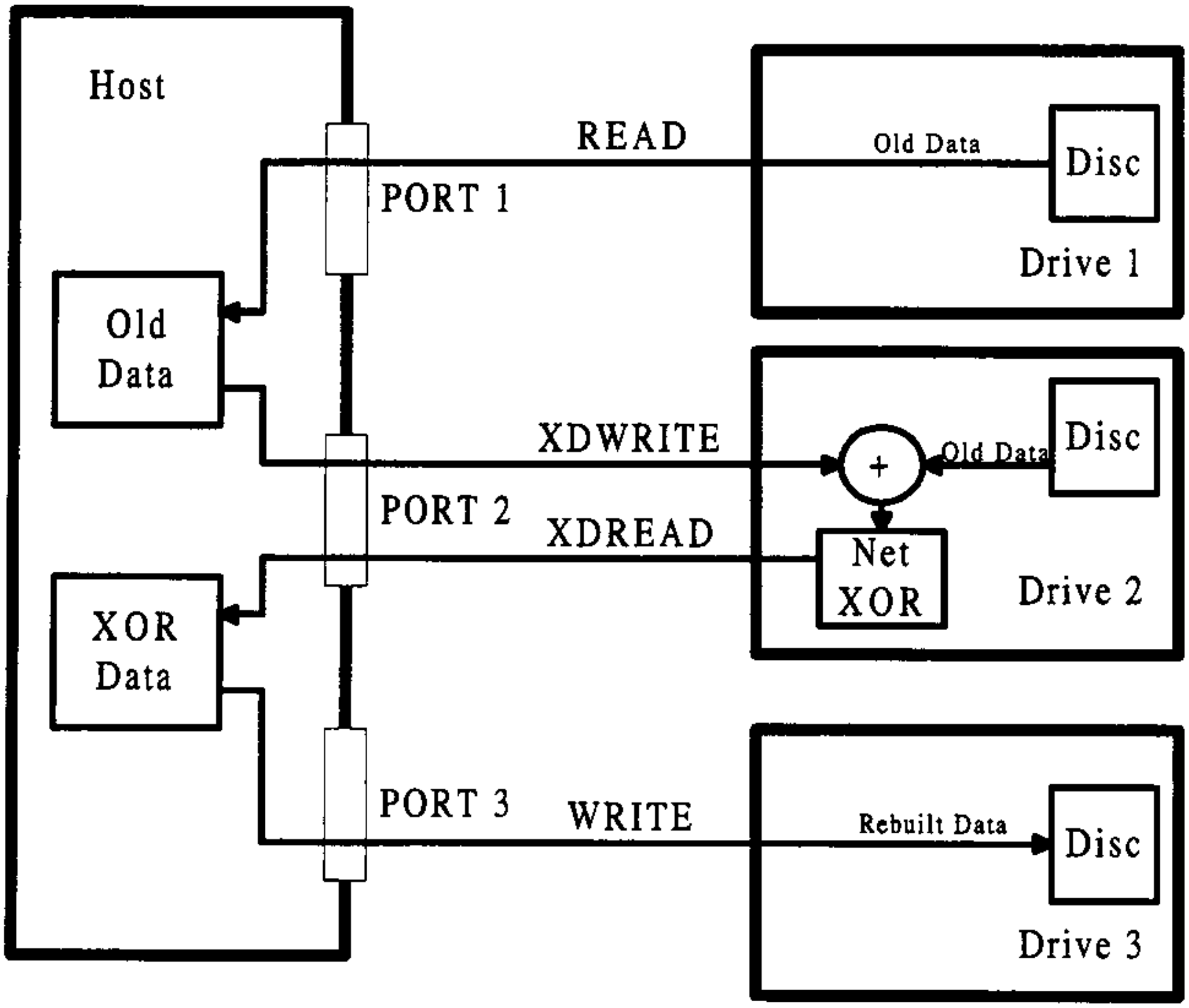
REBUILD Function

(Drives on separate buses or loops)

The host begins by sending the first source drive a READ command. The data received from this drive is sent to the second source drive using an XDWRITE command with the NDISC bit set (to prevent data from being written to disc). The second source drive reads old data from the disc, exclusive ors this old data with the data sent from the host, and stores the result in its buffer. The host retrieves the xor'd data by sending the second drive an XDREAD command with the same logical block address and transfer length as in the associated XDWRITE command. (The second drive is required to retain the xor'd data until successfully completing the XDREAD command.) The host issues the XDWRITE and XDREAD commands in a likewise manner for each successive source drive.

The xor'd data from the last source drive is the "Rebuilt" data, and is sent to the drive being rebuilt using a WRITE command.

Figure 3: PARALLEL REBUILD



054

RAID 5 Support for Single Interface Striping Configurations

As with the Multiple Interface Striping Configuration the Single Interface Striping Configuration needs to support three basic operations to eliminate the XOR engine from the controller. They are, Read-Modify-Write, Regenerate, & Rebuild. The Read-Modify-Write function is used any time a write operation needs to be performed on the array. The Regenerate function is used to Build data from the array when a data drive has malfunctioned. In this situation the Regenerate function takes the place of a normal Read operation. The Rebuild function is used to rebuild a drive that has been added to the array in place of a failed drive. The Rebuilt data is written on the new drive as part of this function.

In Single Interface Striping Configurations the drives have the capability of peer to peer communication. In this environment some of the data passes directly from drive to drive. This technique greatly reduces the amount of work that must be performed in the host as well as reduces the amount of data that must be transferred over the interface. For instance for Read-Modify-Write operations the number of data transfers is reduced from 4 to 2. This section of the document shows how the three basic Functions (described above) are performed using the XOR engine in the drive along with the new SCSI commands.

READ-MODIFY-WRITE Function

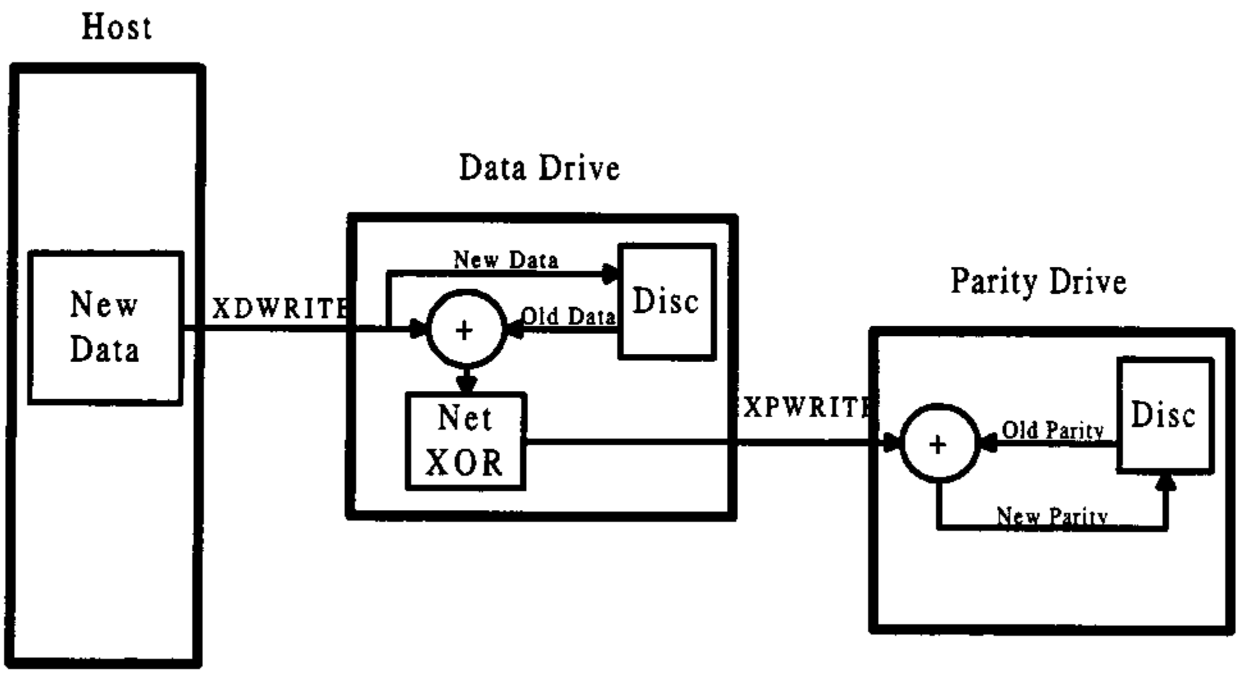
(Drives on same bus or loop)

The host begins by sending the data drive new data using an XDWRITE command, with the Secondary Control bits set to something other than 11 to indicate a secondary target (parity drive) will be involved. Included in the command descriptor block for the XDWRITE command is the address of the parity drive associated with the data block(s) being written. The data drive immediately disconnects, takes on the role of initiator, and sends an XPWRITE command to the parity drive (command phase only at this point - this gets the parity drive started reading old parity from the disc into its buffer). The data drive reads old data from the disc into its buffer, exclusive ors the old data with the new data from the host, sends the xor'd data to the parity drive (data out phase of XPWRITE), and writes the new data from the host to the disc.

The parity drive receives the exclusive or'd data from the data drive, and exclusive ors this data with the old parity data in the buffer. The resulting new parity data is written to the disc.

Upon successful completion of the XPWRITE command good XDWRITE ending status is sent by the data drive to the host. The XPWRITE command is thus "nested" within the XDWRITE command.

Figure 4: SINGLE INTERFACE READ-MODIFY-WRITE

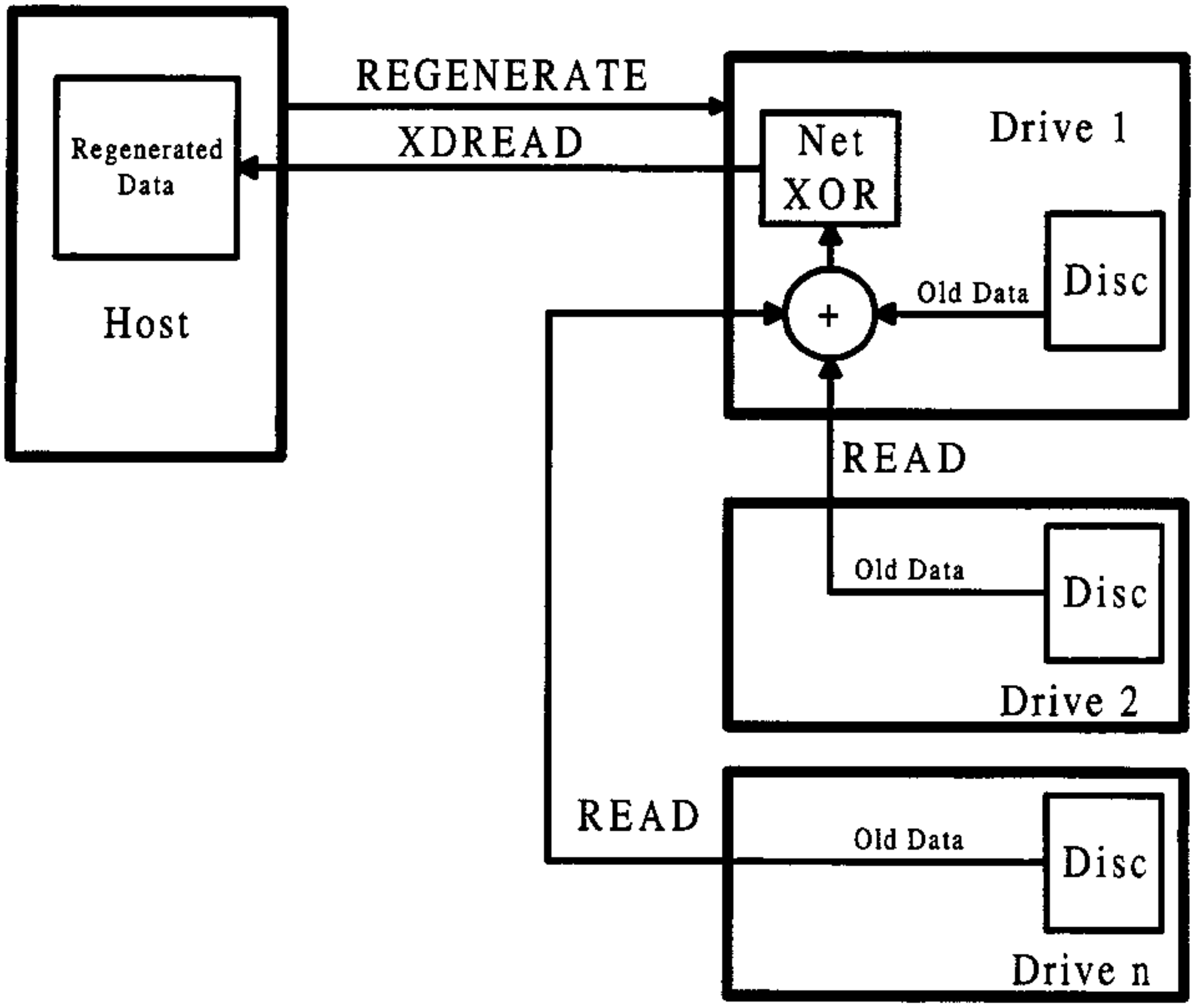


REGENERATE Function

(Drives on <u>same</u> bus or loop)

This function is used in place of a Read command when one data drive in the array has malfunctioned. The host begins by sending a known good drive the REGENERATE command. The addresses of the source drives, as well as the logical block address and regenerate length are passed during the data out phase of the REGENERATE command. The drive then takes on the role of initiator and sends READ commands to all source drives. It also concurrently reads the appropriate data from its own disc. The data from all drives is exclusive or'd and written to the drive's buffer. The host retrieves this built data by sending the drive an XDREAD command with the same logical block address and build length as in the associated REGENERATE command. (The drive is required to retain the xor'd data until successfully completing the associated XDREAD command.)

Figure 5: SINGLE INTERFACE REGENERATE

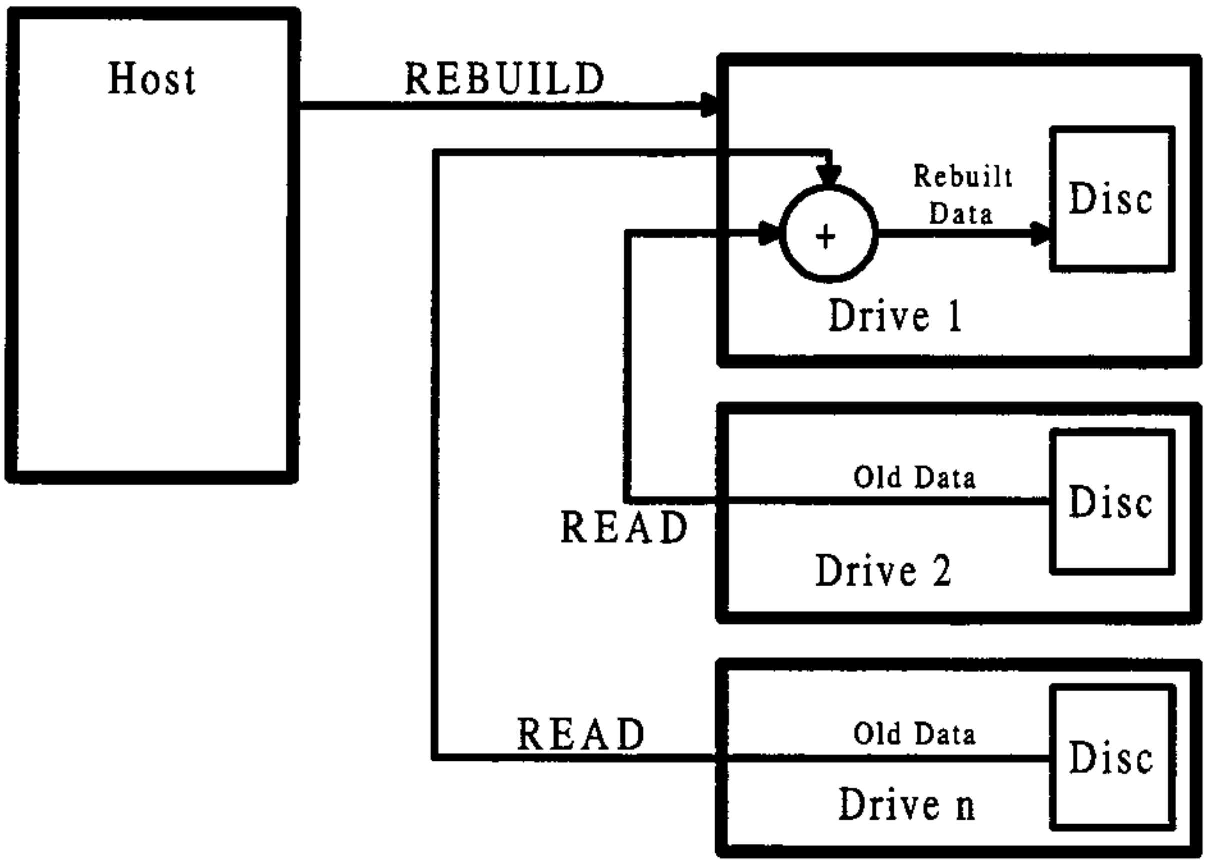


REBUILD Function

(Drives on same bus or loop)

The host begins by sending the REBUILD command to the drive to be rebuilt. The addresses of the source drives, as well as the logical block address and rebuild length are passed during the data out phase of the REBUILD command. The drive then takes on the role of initiator and issues READ commands to all source drives. The data from all source drives is exclusive or'd and written to the disc which is being rebuilt.

Figure 6: SINGLE INTERFACE REBUILD



New RAID Specific SCSI Commands and Mode Pages

XDWRITE Command

(Exclusive Or Data Write)

Table 1: XDWRITE COMMAND

BIT BYTE(S)	7	6	5	4	3	2	1	0
0	OPERATION CODE (XXh)							
1	Reserved	Mirror	NDisc	DPO	FUA	Sec Ctl		Reserved
2 - 5	LOGICAL BLOCK ADDRESS							
6 - 7	TRANSFER LENGTH							
8 - 10	SECONDARY ADDRESS							
11 - 14	SECONDARY LOGICAL BLOCK ADDRESS							
15	CONTROL						Flag	Link

2345

The EXCLUSIVE-OR DATA WRITE command (Table 1) requests that the target read old data from the medium, write to the medium the data transferred by the initiator, exclusive-or the old data with the data transferred by the initiator, and send the xor'd data to the destination specified by the initiator. The destination shall be either 1) another target (secondary target), or 2) the target's buffer (so it can be retrieved with a subsequent command). The exclusive-or operation shall be bit for bit; byte 0, bit 0 of the transferred data shall be xor'd with byte 0, bit 0 of the old data, etc.

IMPLEMENTOR'S NOTE: The XDWRITE command is well suited for the read-modify-write operation in a RAID 5 (Redundant Array of Inexpensive Discs, level 5) environment. The XDWRITE is typically used on the data drive, and allows the xor function to be handled in the device rather than in the controller.

If the Mirror bit is set to one, the target shall not perform the exclusive-or operation. Instead, the data transferred by the initiator shall be sent by the target directly to the destination specified by the initiator. If the specified destination is a secondary target, a non-xor WRITE command shall be used for the secondary transfer. If the Mirror bit is set to zero the exclusive-or operation shall be performed, and the resulting data sent to the destination specified by the initiator.

IMPLEMENTOR'S NOTE: The Mirror bit is typically set to one in a RAID 1 environment, where all data written to the device is duplicated on a second "mirror" device.

If the NDisc bit is set to zero, the data transferred by the initiator shall be written to the medium. If the NDisc bit is set to one, the data shall not be written to the medium. If the NDisc bit is set to one, the Secondary Control bits shall be set to 11. If the NDisc bit is set to one and the Secondary Control bits are not set to 11 the target shall return CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and an additional sense code of INVALID FIELD IN CDB. **The Force Unit Access (FUA) bit is invalid and shall be ignored if the NDisc bit is set to one.**

IMPLEMENTOR'S NOTE: The NDisc bit is typically set to one during an initiator supervised regenerate or rebuild operation in the RAID configuration. The data transferred by the initiator is xor'd with data stored on the medium. The result is then retrieved by the initiator (without being written to the medium) to be similarly processed by successive targets in the RAID configuration.

See ______ for a definition of the cache control bits (DPO and FUA).
(Note to editor: the current SCSI standard refers to the READ (10) command for the cache control bit definitions.)

The Secondary Control bits specify how the xor'd data will be handled by the target. The bits are defined as follows:

00	The target shall perform a secondary transfer of the xor'd data using the same port that received the XDWRITE command.
01	The target shall perform a secondary transfer of the xor'd data using a port other than that which received the XDWRITE command.
10	The target shall perform a secondary transfer of the xor'd data using a port chosen by the target.
11	The target shall retain the xor'd data in its buffer to be retrieved by the initiator.

If a secondary transfer has been specified (Secondary Control bits equal 00, 01, or 10) then the xor'd data shall be sent to a secondary target whose address is specified in the Secondary Address field. Disconnection is required in this case. The target takes on the initiator role and sends an

XPWRITE command to the secondary target. This secondary target operation is "nested" within the XDWRITE command - i.e. the ending status for the XDWRITE command is not sent to the initiator until the secondary operation has completed. **The XDWRITE command shall be atomic with respect to the specified extent of the device - i.e. once execution of the XDWRITE command has begun, no other commands shall be allowed to read or write the portion of the medium specified in the XDWRITE command until the XDWRITE command and any nested commands have completed.** If the XPWRITE command ends with CHECK CONDITION status and the sense bytes indicate an unrecoverable error, the XDWRITE command shall end with CHECK CONDITION status. Targets that are capable of accepting the XDWRITE command with a secondary transfer specified shall also be capable of sending the XPWRITE command (i.e. assume an initiator role).

IMPLEMENTOR'S NOTE: A secondary transfer is typically specified if the devices in the RAID configuration are on the same bus or loop and can directly access one another. The secondary target is normally the parity device, to which the xor'd data is transferred.

IMPLEMENTOR'S NOTE: If the Secondary Control bits specify that the secondary transfer shall take place on a port chosen by the target (bits set to 01 on a target with more than two ports, or bits set to 10), it is assumed that the address of the specified secondary target is the same across all ports. If the secondary target address is not the same across all ports then the initiator should not allow the XDWRITE target to choose the secondary transfer port.

IMPLEMENTOR'S NOTE: If the XPWRITE command is sent to the secondary target before XPWRITE data is available (for efficiency, this is normal), the "Target Transfer Disable" and "Continue I/O Process" messages should be used to adjust the timing of the secondary data transfer to a time when data is available.

If the Secondary Control bits equal 11 (no secondary transfer specified) then the xor'd data shall be retained in the target's buffer until it is successfully transferred via a received associated XDREAD command. The xor'd data shall not be sent to a secondary target. The associated XDREAD command "satisfies" the XDWRITE command; the XDWRITE command remains "unsatisfied" until an associated XDREAD command is successfully completed. Good XDWRITE status implies, at a minimum, that the xor'd data is available in the buffer. Depending on the caching configuration of the target and the state of the NDisc bit, good XDWRITE status may also imply that data has been written to the medium (see NDisc and FUA bits above, and WCE bit of Mode Sense page 8). Targets that are capable of accepting the XDWRITE command with the Secondary Control bits set to 11 shall also be capable of accepting the XDREAD command.

IMPLEMENTOR'S NOTE: The Secondary Control bits are typically set to 11 if the devices in the RAID configuration are on isolated busses or loops. In this case, the RAID controller (initiator) must supervise the read-modify-write operation, since the devices cannot directly access one another. The controller is normally the initiator of the subsequent associated XDREAD command.

The Secondary Control bits shall not be set to 01 if the target is not a multiple port device. If the Secondary Control bits are set to 01 and the target is not a multiple port device the target shall return CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and an additional sense code of INVALID FIELD IN CDB.

The logical block address field specifies the first logical block of the range of logical blocks to be operated on at the target.

The transfer length field specifies the number of contiguous logical blocks of data that shall be transferred. A transfer length of zero indicates that no logical blocks shall be transferred. This condition shall not be considered an error and no data shall be written. Any other value indicates the number of logical blocks that shall be transferred. The value contained in this field shall be used in the transfer length field of any associated XPWRITE command sent by the target.

The secondary address field specifies the address of the secondary target. This field is not valid and shall be ignored if the Secondary Control bits are set to 11.

> IMPLEMENTOR'S NOTE: The secondary address may be the address of the initiator of the XDWRITE command. In this case, the initiator must be capable of becoming a target and accepting the nested XPWRITE command that will be sent by the XDWRITE target. This provides an alternate approach to using the XDREAD command in the case where the RAID devices are isolated from one another.

The secondary logical block address field specifies the value which shall be used in the logical block address field of any associated XPWRITE command sent by the target. This field is not valid and shall be ignored if the Secondary Control bits are set to 11.

Errors During the XDWRITE Command

Two classes of exception conditions may occur during execution of an XDWRITE command. Either or both of these classes of exception may occur during command execution:

Primary Errors
The first class consists of those exception conditions which are detected by the device that received the XDWRITE command and are not due to the failure of a secondary command. These conditions include invalid parameters in the XDWRITE command, inability of the device to continue operating, and parity errors while transferring the XDWRITE command, data, or status byte. In the event of such an exception condition, the target of the XDWRITE command shall:

(1) Terminate the XDWRITE command with CHECK CONDITION status.
(2) Build sense data according to the exception condition.

Secondary Errors
The second class of errors consists of exception conditions resulting from a failure of a secondary (nested) command (XPWRITE for example). The exception may be detected by either the initiator of the nested command (XDWRITE target), the target of the nested command, or both. The sense data for secondary errors shall be passed to the XDWRITE initiator in the Additional Sense Bytes area of the sense data.

If the nested command initiator detects the exception it shall:

(1) Terminate the XDWRITE command with CHECK CONDITION status.
(2) Set the sense key to ABORTED COMMAND if there are no primary errors to report. Otherwise, the sense key shall be set according to the primary error.
(3) Set the first byte of the command specific information field of the sense data to the starting byte number, relative to the first byte of sense data, of an area that contains (unchanged) the nested command initiator's sense data for the secondary error. A zero value in this byte indicates no secondary error has been detected by the nested command initiator. The secondary sense data shall be built in the normal sense data format as defined for the REQUEST SENSE command.

If the nested command target detects the exception, the initiator of the nested command receives CHECK CONDITION status from the target of the nested command. The nested command initiator shall recover the sense data associated with the exception condition and shall:

(1) Terminate the XDWRITE command with CHECK CONDITION status.
(2) Set the sense key to ABORTED COMMAND if there are no primary errors to report. Otherwise, the sense key shall be set according to the primary error.
(3) Set the second byte of the command specific information field of the sense data to the starting byte number, relative to the first byte of sense data, of an area that contains (unchanged) the nested command target's status byte followed by its sense data. A zero value in this byte indicates no secondary error has been detected by the nested command target.

Notes on Secondary Error Recovery and ACA Handling

(Note: This section assumes that a device which has had a third party reserve performed on it's behalf has the ability to clear that reserve. As of the date of this document, this has not yet been proposed to ANSI.)

Upon the occurrence of an error in the target of an XPWRITE command, that target will have an ACA condition with allegiance to the XPWRITE initiator. In a RAID configuration with peer to peer device communication, the XPWRITE initiator is most likely the target of an associated XDWRITE command from a RAID controller. Since it may be desirable to allow error handling to be performed by that RAID controller, the ACA condition between the XPWRITE initiator and target must be cleared in order to allow the RAID controller access to the XPWRITE target. At the same time, other initiators must be prevented from accessing the XPWRITE target. The XDWRITE initiator can accomplish this by performing a 3rd party reserve of the XPWRITE target, on behalf of the RAID controller, then clearing the ACA condition. The RAID controller then has exclusive access to the failing XPWRITE target. When the RAID controller has completed its error handling, it can release the XPWRITE target.

6XPWRITE Command

(Exclusive Or Parity Write)

Table 7: XPWRITE COMMAND

BIT BYTE(S)	7	6	5	4	3	2	1	0
0	OPERATION CODE (XXh)							
1	RESERVED			DPO	FUA	Reserved		
2 - 5	LOGICAL BLOCK ADDRESS							
6	RESERVED							
7 - 8	TRANSFER LENGTH							
9	CONTROL						Flag	Link

The EXCLUSIVE-OR PARITY WRITE command (Table 2) requests that the target read old data from the medium, and write to the medium the old data xor'd with the data transferred by the initiator. The exclusive-or operation shall be bit for bit; byte 0, bit 0 of the transferred data shall be xor'd with byte 0, bit 0 of the old data, etc.

> IMPLEMENTOR'S NOTE: The XPWRITE command is well suited for the read-modify-write operation in a RAID 5 (Redundant Array of Inexpensive Discs, level 5) environment. The XPWRITE is typically used on the parity drive, and allows the xor function to be handled in the device rather than in the controller. The data transferred by

the initiator is typically the result of the previous associated XDWRITE command issued to a data drive.

See ______ for a definition of the cache control bits (DPO and FUA).
(Note to editor: the current SCSI standard refers to the READ (10) command for the cache control bit definitions.)

The logical block address field specifies the first logical block of the range of logical blocks to be operated on by this command.

The transfer length field specifies the number of contiguous logical blocks of data that shall be transferred. A transfer length of zero indicates that no logical blocks shall be transferred. This condition shall not be considered an error and no data shall be written. Any other value indicates the number of logical blocks that shall be transferred.

IMPLEMENTOR'S NOTE: The logical block address and transfer length fields typically contain values that were passed in the associated XDWRITE **command.**

XDREAD Command

(Exclusive Or Data Read)

Table 8: XDREAD COMMAND

BIT / BYTE(S)	7	6	5	4	3	2	1	0
0	OPERATION CODE (XXh)							
1	RESERVED							
2 - 5	LOGICAL BLOCK ADDRESS							
6	RESERVED							
7 - 8	TRANSFER LENGTH							
9	CONTROL						Flag	Link

The EXCLUSIVE-OR DATA READ command (Table 3) requests that the target transfer to the initiator data stored in the targets buffer as a result of a previous associated XDWRITE command with the Secondary Control bits set to 11, or a previous associated REGENERATE command. The successful completion of this command "satisfies" the associated command, relieving the target of having to retain the data in its buffer. Until the target receives this subsequent associated XDREAD command, an XDWRITE or REGENERATE command remains "unsatisfied" (see XDWRITE and REGENERATE commands).

The logical block address field shall contain the same value as was used in the associated command, and is used by the target to associate the two commands. If this field contains a value which does not match any unsatisfied command's logical block address, the target shall return CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and an additional sense code of INVALID FIELD IN CDB.

The transfer length field specifies the number of contiguous logical blocks of data that shall be transferred, and shall contain the same value that was used in the transfer length field of the associated command. If this field contains a value which is different from the transfer length used in the associated command, the target shall return CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and an additional sense code of INVALID FIELD IN CDB.

REBUILD Command

Table 9: REBUILD COMMAND

BIT BYTE(S)	7	6	5	4	3	2	1	0
0	OPERATION CODE (XXh)							
1	RESERVED					Int Data	Sec Ctl	
2 - 5	REBUILD LENGTH (blocks)							
6	RESERVED							
7 - 8	PARAMETER LIST LENGTH (bytes)							
9	CONTROL						Flag	Link

The REBUILD command (Table 4) requests that the target write to the medium data which has been built by xor-ing data from the source(s) specified by the initiator (the target of the REBUILD command is not one of the sources, as it is with the REGENERATE command). The target assumes the role of initiator and reads the specified data from each source specified.

IMPLEMENTOR'S NOTE: The REBUILD command is used in a RAID configuration to restore data which has been previously lost. Because of the redundancy nature of RAID, the lost data can be built by xor-ing data from the other devices in the configuration.

The exclusive-or operation shall be bit for bit; byte 0, bit 0 of the data from one specified source shall be xor'd with byte 0, bit 0 of the data from the next specified source, etc., until all specified data from the two sources has been xor'd. The data resulting from this operation shall be xor'd in a likewise manner with the next specified source (note: since xor operands are commutable, the source order is irrelevant). This shall continue until the data from all specified sources has been xor'd. If only one source is specified, no exclusive-or operation shall take place.

IMPLEMENTOR'S NOTE: A typical instance of specifying only one source is in a RAID 1 configuration, where data on a mirror device is duplicated on the device being rebuilt.

The rebuild operation shall be performed in ascending logical block order, beginning with the specified logical block address.

IMPLEMENTOR'S NOTE: In the case of a rebuild error, the sense data contains the logical block address of the failing block. Since blocks are rebuilt in ascending order, all blocks with a lower logical block address can be assumed to be uncorrupted.

The Secondary Control bits specify how the source data will be handled by the target. The bits are defined as follows:

00	The target shall perform the read(s) of the source data using the same port that received the REBUILD command.
01	The target shall perform the read(s) of the source data using a port other than that which received the REBUILD command.
10	The target shall perform the read(s) of the source data using a port chosen by the target.
11	Reserved

IMPLEMENTOR'S NOTE: If the Secondary Control bits specify that the secondary transfer shall take place on a port chosen by the target (bits set to 01 on a target with more than two ports, or bits set to 10), it is assumed that the addresses of the specified source targets are the same across all ports. If the source target addresses are not the same across all ports then the initiator should not allow the REBUILD target to choose the secondary transfer port.

IMPLEMENTOR'S NOTE: If the READ command is sent to a source target before the REBUILD target has the resources to accept the read data (for efficiency, this is normal), the "Target Transfer Disable" and "Continue I/O Process" messages should be used to adjust the timing of the secondary data transfer to a time when resources are available.

The Secondary Control bits shall not be set to 01 if the target is not a multiple port device. If the Secondary Control bits are set to 01 and the target is not a multiple port device the target shall return CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and an additional sense code of INVALID FIELD IN CDB.

If the Intermediate Data bit is set to one, the rebuild command parameters shall include intermediate data. **The rebuild length indicates the length in blocks of the intermediate data.** This data shall be treated as an additional source, and xor'd in the same manner as the data from the specified sources. If the Intermediate Data bit is set to zero, then no intermediate data shall be included in the command parameters.

IMPLEMENTOR'S NOTE: Intermediate data is typically used in a configuration where more than one string or loop of devices make up a RAID group. e.g. In a two string configuration, the intermediate data can be obtained by sending a REGENERATE command to a device on the first string. The data obtained from the regenerate operation can then be passed as intermediate data to the device to be rebuilt on the second string.

The rebuild length field indicates the length in logical blocks of the rebuild operation. This corresponds to the length in logical blocks of the transfers from the specified sources. All F's in the rebuild length field indicates that the target shall rebuild through the last existing logical block on the medium.

The parameter list length field specifies the length in bytes of the parameters (**not including intermediate data**) that shall be sent during the DATA OUT phase of the command. A parameter list length of zero indicates that no data shall be transferred. This condition shall not be considered an error, and no rebuild or write operation shall take place.

The REBUILD parameter list is described in Table 5.

Table 10: REBUILD COMMAND PARAMETERS

BIT BYTE(S)	7	6	5	4	3	2	1	0
0	NUMBER OF SOURCES							
1	SOURCE ENTRY TYPE							
2 - 3	RESERVED							
4 - 7	LOGICAL BLOCK ADDRESS							

SOURCE ADDRESS ENTRIES

0 - 3	SOURCE ADDRESS 0
4 - 7	SOURCE 0 LOGICAL BLOCK ADDRESS

⋮

0 - 3	SOURCE ADDRESS n
4 - 7	SOURCE n LOGICAL BLOCK ADDRESS

INTERMEDIATE DATA

0 - n	INTERMEDIATE DATA

The number of sources field indicates the number of source devices specified for use in the rebuild operation, and indicates the number of source address entries in the command parameters. Note: intermediate data is not considered a source for purposes of this field.

The Source Entry Type field defines the size and layout of the source address entries. Currently, the only valid source entry type is 01.

The logical block address field specifies the logical block address at which the target is to begin the data rebuild.

The source address field indicates the address of the source of the data to be xor'd during the data reconstruction. This field is repeated for each source.

The source logical block address field indicates the logical block address at which to begin transferring data from the source. This field is repeated for each source.

The intermediate data field contains any data included **after** the command parameters to be xor'd with the data from the specified sources (see Intermediate Data bit in the command descriptor block). The length of this field shall be consistent with the rebuild length - i.e. if the rebuild length field specifies two blocks, and the block length of the device is 1024 bytes, then the intermediate data field shall be 2048 bytes in length.

REGENERATE Command

Table 11: REGENERATE COMMAND

BIT BYTE(S)	7	6	5	4	3	2	1	0
0	OPERATION CODE (XXh)							
1	RESERVED					Int Data	Sec Ctl	
2 - 5	REGENERATE LENGTH (blocks)							
6	RESERVED							
7 - 8	PARAMETER LIST LENGTH (bytes)							
9	CONTROL						Flag	Link

The REGENERATE command (Table 6) requests that the target write to its buffer data which has been built by xor-ing data from the source(s) specified by the initiator (the target of the REGENERATE command is inherently one of the sources). The target assumes the role of initiator and reads the specified data from each source specified. The result of the exclusive-or operation is placed in the buffer so it can be retrieved by a subsequent command.

IMPLEMENTOR'S NOTE: The REGENERATE command is typically used for read operations in a RAID configuration when one data device has malfunctioned. Because of the redundancy nature of RAID, the data from the malfunctioning device can be regenerated by xor-ing data from the other devices in the configuration.

The exclusive-or operation shall be bit for bit; byte 0, bit 0 of the data from one specified source shall be xor'd with byte 0, bit 0 of the data from the next specified source, etc., until all specified data from the two sources has been xor'd. The data resulting from this operation shall be xor'd in a likewise manner with the next specified source (note: since xor operands are commutable, the source order is irrelevant). This shall continue until the data from all specified sources has been xor'd. The result is written to the target's buffer.

The xor'd data shall be retained in the target's buffer until it is successfully transferred via a received associated XDREAD command. The subsequent XDREAD command "satisfies" the REGENERATE command. The REGENERATE command remains "unsatisfied" until an associated XDREAD command is successfully completed. Good REGENERATE status is not returned until the xor'd data is available in the buffer. Targets that are capable of accepting the REGENERATE command shall also be capable of accepting the XDREAD command.

IMPLEMENTOR'S NOTE: The XDREAD command is typically sent by the initiator that sent the REGENERATE command.

The Secondary Control bits specify how the source data will be handled by the target. The bits are defined as follows:

00	The target shall perform the read(s) of the source data using the same port that received the REGENERATE command.
01	The target shall perform the read(s) of the source data using a port other than that which received the REGENERATE command.
10	The target shall perform the read(s) of the source data using a port chosen by the target.
11	Reserved

IMPLEMENTOR'S NOTE: If the Secondary Control bits specify that the secondary transfer shall take place on a port chosen by the target (bits set to 01 on a target with more than two ports, or bits set to 10), it is assumed that the addresses of the specified source targets are the same across all ports. If the source target addresses are not the same across all ports then the initiator should not allow the REGENERATE target to choose the secondary transfer port.

IMPLEMENTOR'S NOTE: If the READ command is sent to a source target before the REBUILD target has the resources to accept the read data (for efficiency, this is normal), the "Target Transfer Disable" and "Continue I/O Process" messages should be used to adjust the timing of the secondary data transfer to a time when resources are available.

The Secondary Control bits shall not be set to 01 if the target is not a multiple port device. If the Secondary Control bits are set to 01 and the target is not a multiple port device the target shall return CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and an additional sense code of INVALID FIELD IN CDB.

If the Intermediate Data bit is set to one, the regenerate command parameters shall include intermediate data. **The regenerate length indicates the length in blocks of the intermediate data.** This data shall be treated as source data, and shall be xor'd in addition to, and in the same manner as, the data from the specified sources. If the Intermediate Data bit is set to zero, then no intermediate data shall be included in the command parameters.

IMPLEMENTOR'S NOTE: Intermediate data is typically used in a configuration where more than one string or loop of devices make up a RAID group. e.g. In a two string configuration, the intermediate data can be obtained by sending a REGENERATE command to a device on the first string. The data obtained from the regenerate operation can then be passed as intermediate data to a device on the second string.

The regenerate length field indicates the length in logical blocks of the regenerate operation. This corresponds to the length in logical blocks of the transfers from the specified sources.

The parameter list length field specifies the length in bytes of the parameters (**not including intermediate data**) that shall be sent during the DATA OUT phase of the command. A parameter list length of zero indicates that no data shall be transferred. This condition shall not be considered an error.

The REGENERATE parameter list layout is shown in Table 7.

Table 12: REGENERATE COMMAND PARAMETERS

BIT / BYTE(S)	7	6	5	4	3	2	1	0
0	NUMBER OF SOURCES							
1	SOURCE ENTRY TYPE							
2 - 3	RESERVED							
4 - 7	LOGICAL BLOCK ADDRESS							

SOURCE ADDRESS ENTRIES

0 - 3	SOURCE ADDRESS 0
4 - 7	SOURCE 0 LOGICAL BLOCK ADDRESS

⋮

0 - 3	SOURCE ADDRESS n
4 - 7	SOURCE n LOGICAL BLOCK ADDRESS

INTERMEDIATE DATA

0 - n	INTERMEDIATE DATA

The number of sources field indicates the number of source devices specified for use in the regenerate operation, and indicates the number of source address entries in the command parameters. Note: intermediate data is not considered a source for purposes of this field.

The Source Entry Type field defines the size and layout of the source address entries. Currently, the only valid source entry type is 01.

The logical block address field specifies the target's own logical block address at which to begin the regenerate operation.

The source address field indicates the address of the source of the data to be xor'd during the regenerate operation. This field is repeated for each source.

The source logical block address field indicates the logical block address at which to begin transferring data from the source. This field is repeated for each source.

The intermediate data field contains any data included **after** the command parameters to be xor'd with the data from the specified sources (see Intermediate Data bit in the command descriptor block). The length of this field shall be consistent with the regenerate length - i.e. if the regenerate length field specifies two blocks, and the block length of the device is 1024 bytes, then the intermediate data field shall be 2048 bytes in length.

Resets During XOR Commands

The effect of a reset during an XOR command (XDWRITE, XPWRITE, etc.) depends on which reset alternative has been implemented within the target.

For those targets which implement the hard reset alternative, a reset shall cause any XOR command for which status has not been sent, and any nested command, to be aborted. If status has been sent and the XOR command is still in process (e.g. a write operation with write caching enabled), then the command shall be completed prior to reset execution. (Whether status is returned prior to command completion is a function of the caching configuration of the target.) Any unsatisfied commands shall be satisfied by the reset - e.g. if the Secondary Control bits were set to 11 in the XDWRITE command, a reset relieves the target of having to save the resulting data in its buffer.

For those targets which implement the soft reset alternative, an XOR command in process, and any nested command, shall not be affected by a reset. The command shall be completed to the best of the target's ability, and any unsatisfied XOR commands shall remain unsatisfied until satisfied in the normal manner.

RAID CONTROL mode page

The RAID Control mode page (Table 8) provides the initiator with the means to obtain/modify certain RAID operating parameters of the target.

Table 13: RAID CONTROL PAGE

BIT / BYTE(S)	7	6	5	4	3	2	1	0
0			PAGE CODE (XXh)					
1	PAGE LENGTH (02)							
2	RESERVED							LOG MODE
3	RESERVED							
4 - 7	MAX XDWRITE SIZE							
8 - B	MAX XPWRITE SIZE							
C - F	MAX REGEN SIZE							
10 - 13	MAX REBUILD READ SIZE							
14 - 15	REBUILD DELAY							

A Log Mode bit of zero specifies that the target is not enabled as a logging device. A Log Mode bit of one specifies that the target is enabled as a logging device (see "Additional Performance Enhancements" section).

The Maximum XDWRITE Size field specifies the maximum transfer length in bytes that the target will accept for the XDWRITE command.

The Maximum XPWRITE Size field specifies the maximum transfer length in bytes that the target will accept for the XPWRITE command.

The Maximum Regenerate Size field specifies the maximum regenerate length in bytes that the target will accept for the REGENERATE command.

The Maximum Rebuild Read Size field specifies the maximum transfer length in bytes that the target will use for nested READ commands during a rebuild operation.

The Rebuild Delay field specifies a minimum time in milliseconds which the target will delay, following the issuing of any nested READ command during a rebuild operation, before issuing another nested READ command during the same rebuild operation.

Additional Performance Enhancements

Notice to Reader: This section of the document is still being worked on and does not yet fully describe how the Logging Function works.

The single biggest impact of a RAID 5 architecture is the performance degradation that occurs as a result of the Read-Modify-Write operation required for all writes. Typically the only way to improve on this situation has been to use a non-volatile write cache so that completion status can be returned to the OS before the Read-Modify-Write is complete on the disc drives. One way to improve write performance without a non-volatile write cache is to implement a special Logging mode in a spare disc drive. Many RAID 5 systems include a hot spare for system availability reasons. If this spare drive were used as a logging drive (when it isn't needed to replace a downed drive) system write performance could be significantly improved.

To implement this functionality the drive needs to be placed in a special mode called Logging Mode. This is done using the RAID Control mode page described earlier. Once the drive is in logging mode the host writes data to the drive using standard write commands.

Logging Mode

The Log Mode bit in the RAID Control page (Table 8) provides the initiator the means to enable the target as a high speed "logging" device, which causes normal write and read commands to be executed as logged commands. In this mode write data is written to the next "available" location on the medium. Once written, a location becomes "unavailable" until the successful completion of a subsequent LCLEAR command for that location. When enabled as a logging device, the target shall be capable of accepting the LCLEAR command.

> IMPLEMENTOR'S NOTE: Configuring a target as a logging device causes an improvement in the average time for command complete status to be returned following a write operation. The logging device is normally used for temporary write data storage while the same data is being written to a more permanent location. Once the more permanent write is complete the LCLEAR command is normally issued to the logging device, causing the occupied location to once again become available for writes. Typically, data is not read from the logging device except for data recovery purposes - e.g. there was a failure during the write to the more permanent location.
>
> The logging mode target will need to maintain a non-volatile record which links the logical block address of received commands to the physical location on the medium, since there is otherwise no correlation between the two. For example, if an initiator issues a write command specifying "n" as the logical block address, the next available block on the medium becomes logical block "n", regardless of its physical location.

LCLEAR Command

(Log Clear)

The Log Clear command requests that the logging mode target (see RAID Control mode sense page) make the specified blocks of data available for future write commands.

The logical block address field specifies the first logical block of the range of logical blocks to be operated on by this command.

The length field specifies the number of contiguous logical blocks of data that shall be cleared. A transfer length of zero indicates that no logical blocks shall be cleared. This condition shall not be considered an error.

In order to accept this command the target shall have been placed in logging mode via the RAID Control page of the Mode Select command. If this command is received by a target which is not in logging mode, the target shall return CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and an additional sense code of INVALID COMMAND OPERATION CODE.

> IMPLEMENTOR'S NOTE: This command is typically sent when the data stored in the specified blocks has been written to a more permanent location, and the need to retain it on the logging device no longer exists.