American National Standard

ANSI INCITS 131-1994 (R1999)
(formerly ANSI X3.131-1994 (R1999))
Stabilized as
INCITS 131-1994[S2013]

for Information Systems - Small Computer System Interface-2

Developed by

incits SM

Where IT all begins

Approved American National Standard
ANSI

ANSI®
X3.131-1994

American National Standard
for Information Systems –

Small Computer System Interface-2

Secretariat
Computer and Business Equipment Manufacturers Association

Approved January 31, 1994
American National Standards Institute, Inc.

Abstract

The SCSI protocol is designed to provide an efficient peer-to-peer I/O bus with up to 16 devices, including one or more hosts. Data may be transferred asynchronously at rates that only depend on device implementation and cable length. Synchronous data transfers are supported at rates up to 10 mega-transfers per second. With the 32-bit wide data transfer option, data rates of up to 40 megabytes per second are possible.

SCSI-2 includes command sets for magnetic and optical disks, tapes, printers, processors, CD-ROMs, scanners, medium changers, and communications devices.

American National Standard

Approval of an American National Standard requires review by ANSI that the requirements for due process, consensus, and other criteria for approval have been met by the standards developer.

Consensus is established when, in the judgment of the ANSI Board of Standards Review, substantial agreement has been reached by directly and materially affected interests. Substantial agreement means much more than a simple majority, but not necessarily unanimity. Consensus requires that all views and objections be considered, and that a concerted effort be made toward their resolution.

The use of American National Standards is completely voluntary; their existence does not in any respect preclude anyone, whether he has approved the standards or not, from manufacturing, marketing, purchasing, or using products, processes, or procedures not conforming to the standards.

The American National Standards Institute does not develop standards and will in no circumstances give an interpretation of any American National Standard. Moreover, no person shall have the right or authority to issue an interpretation of an American National Standard in the name of the American National Standards Institute. Requests for interpretations should be addressed to the secretariat or sponsor whose name appears on the title page of this standard.

CAUTION NOTICE: This American National Standard may be revised or withdrawn at any time. The procedures of the American National Standards Institute require that action be taken periodically to reaffirm, revise, or withdraw this standard. Purchasers of American National Standards may receive current information on all standards by calling or writing the American National Standards Institute.

CAUTION: The developers of this standard have requested that holders of patents that may be required for the implementation of the standard disclose such patents to the publisher. However, neither the developers nor the publisher have undertaken a patent search in order to identify which, if any, patents may apply to this standard. As of the date of publication of this standard and following calls for the identification of patents that may be required for the implementation of the standard, no such claims have been made. No further patent search is conducted by the developer or publisher in respect to any standard it processes. No representation is made or implied that licenses are not required to avoid infringement in the use of this standard.

Published by

American National Standards Institute
11 West 42nd Street, New York, New York 10036

Printed in the United States of America

Contents

Page

Tables

Page

Tables *(continued)*

Page

Tables *(continued)*

Page

Tables *(continued)*

Page

Tables *(continued)*

Page

Figures

Annexes

Foreword (This foreword is not part of American National Standard X3.131-1994.)

The Small Computer System Interface-2 standard is designed to provide an efficient peer-to-peer I/O bus with up to 16 devices, including one or more hosts. Data may be transferred asynchronously at rates that only depend on device implementation and cable length. Synchronous data transfers are supported at rates up to 10 mega-transfers per second. With the 32-bit wide data transfer option, data rates of up to 40 megabytes per second are possible. This standard includes command sets for magnetic and optical disks, tapes, printers, processors, CD-ROMs, scanners, medium changers, and communications devices.

This standard was developed by Task Group X3T9.2 of Accredited Standards Committee X3 during 1986-90. The standards approval process started in 1991. This document includes five annexes, which are informative and are not considered part of the standard.

Requests for interpretation, suggestions for improvement and addenda, or defect reports are welcome. They should be sent to the X3 Secretariat, Computer and Business Equipment Manufacturers Association, 1250 Eye Street, NW, Suite 200, Washington, DC 20005-3922.

This standard was processed and approved for submittal to ANSI by Accredited Standards Committee on Information Processing Systems, X3. Committee approval of the standard does not necessarily imply that all committee members voted for approval. At the time it approved this standard, the X3 Committee had the following members:

James D. Converse, Chair
Donald C. Loughry, Vice-Chair
Joanne M. Flanagan, Secretary

Organization Represented	*Name of Representative*
American Nuclear Society	Geraldine C. Main
	Sally Hartzell (Alt.)
AMP, Inc.	Edward Kelley
	Charles Brill (Alt.)
Association of the Institute for Certification of Computer Professionals (AICCP)	Kenneth Zemrowski
AT&T/NCR	Thomas W. Kern
	Thomas F. Frost (Alt.)
Apple Computer, Inc.	Karen Higginbottom
Boeing Company	Catherine Howells
	Andrea Vanosdoll (Alt.)
Bull HN Information Systems, Inc.	William George
Compaq Computers	James Barnes
Digital Equipment Corporation	Delbert Shoemaker
	Kevin Lewis (Alt.)
Eastman Kodak Company	James Converse
	Michael Nier (Alt.)
Guide International, Inc.	Frank Kirshenbaum
	Harold Kuneke (Alt.)
Hewlett-Packard	Donald C. Loughry
Hitachi America, Ltd.	John Neumann
	Kei Yamashita (Alt.)
Hughes Aircraft Company	Harold Zebrack
IBM Corporation	Joel Urman
	Mary Ann Lawler (Alt.)
National Communications Systems	Dennis Bodson

Organization Represented	*Name of Representative*
National Institute of Standards and Technology	Robert E. Rountree
	Michael Hogan (Alt.)
Northern Telecom, Inc.	Mel Woinsky
	Subhash Patel (Alt.)
Neville & Associates	Carlton Neville
Recognition Technology Users Association	Herbert P. Schantz
	G. Edwin Hale (Alt.)
Share, Inc.	Gary Ainsworth
	David Thewlis (Alt.)
Sony Corporation of America	Michael Deese
Storage Technology Corporation	Joseph S. Zajaczkowski
	Samuel D. Cheatham (Alt.)
Sun Microsystems, Inc.	Scott Jameson
	Gary S. Robinson (Alt.)
3M Company	Eddie T. Morioka
	Paul D. Jahnke (Alt.)
Unisys Corporation	John Hill
	Stephen P. Oksala (Alt.)
U.S. Department of Defense	William Rinehuls
	C. J. Pasquariello (Alt.)
U.S. Department of Energy	Alton Cox
	Lawrence A. Wasson (Alt.)
U.S. General Services Administration	Douglas Arai
	Larry L. Jackson (Alt.)
Wintergreen Information Services	John Wheeler
Xerox Corporation	Dwight McBain
	Roy Peirce (Alt.)

Subcommittee X3T9 on I/O Interface, which reviewed this standard, had the following members:

Del Shoemaker, Chair	Charles Brill	Doug Morrissey
Bob Fink, Vice-Chair	Bill Burr	Steven Myers
	Jeff Connell	Roger Pandanda
	Steve Cooper	Everett Rigsbee, III
	Roger Cummings	Floyd Ross
	Rhonda Dirvin	Amit Shah
	Jim Hamstra	Jeff Stone
	Dave Husak	Pat Thaler
	Reinhard Knerr	Don Tolmie
	Larry Lamers	Schelto van Doorn
	George Michael	Jim Vogt
	Gene Milligan	Carl Zeitler

Working Group X3T9.2 on Lower Level Interface, which developed this standard, had the following members:

John B. Lohmeyer, Chair
I. Dal Allan, Vice-Chair
Lawrence J. Lamers, Secretary

Bruce Anderson
Dennis Appleyard
Geoff Barton
Robert Bellino
Sudhir Bhatnager
Charles Brill
William E. Burr
Steve Caron
Kurt Chan
Stephen R. Cornaby
Forrest Crowell
Joe Dambach
Jan V. Dedek
Peter Dougherty
William Galloway
Paul Hanmann
Norman H. Harris
David Hess
Steve Kappes
Sam Karunanithi
Robert Kellert
Oscar Kornblum
Richard J. Krass
Robert Liu
Thomas R. Marks
Bob Masterson
David McFadden
James McGrath
Tetsuro Motoyoma
George Penokie
Gary S. Peterson
Heinz Piorunneck
Doug Piper
Chris Pisciotta
Donna Pope
Kenneth Post
D. Michael Robinson
Arnold J. Roccati
Wayne E. Roen
Jeff Rosa
Wayne Sanderson
D. Shaff
Robbie Shergill
Robert L. Simpson
Scott Smyers
Robert N. Snively
Jeff Stai
Pete Tobias
Harvey Waltersdorf
Al Wilhelm
Kurt Witte
Charles Yang
Ed Young
Saied Zangenehpour
Jon Abilay (Alt.)
William F. Alarcon (Alt.)
David C. Andel (Alt.)
Dexter Anderson (Alt.)
Paul Anderson (Alt.)
Harlan Andrews (Alt.)
Percy R. Aria (Alt.)
Joe Chen (Alt.)
Roger Cummings (Alt.)
Lam Dang (Alt.)
Viet Dang (Alt.)
David DesRoches (Alt.)
Brian Fox (Alt.)
John A. Gallant (Alt.)
Edward A. Gardner (Alt.)
Raymond Gilson (Alt.)
Chuck Grant (Alt.)
Douglas Hagerman (Alt.)
George T. Hahn Jr. (Alt.)
Ken Hallam (Alt.)
William Ham (Alt.)
Tom Hanan (Alt.)
Robert C. Herron (Alt.)
Gerald Houlder (Alt.)
Edward Hrvatin (Alt.)
Stephen Huberty (Alt.)
Paul Jackson (Alt.)
Skip Jones (Alt.)
Steven A. Justiss (Alt.)
Chuck Kummeth (Alt.)
Joe Lawlor (Alt.)
Greg Leonhardt (Alt.)
Ernest Luttig (Alt.)
Jim Luttrull (Alt.)
Satwinder S. Mangat (Alt.)
Gerald Marazas (Alt.)
Jim McGrath (Alt.)
Dave McIntyre (Alt.)
Bill Medlinski (Alt.)
Gene Milligan (Alt.)
Hiroshi Minawa (Alt.)
Neil Mitchell (Alt.)
E. J. Mondor (Alt.)
David G. Moore (Alt.)
John Morse (Alt.)
Bob Mortensen (Alt.)
Gary Murdock (Alt.)
Glen Nance (Alt.)
Paul R. Nitza (Alt.)
John F. Osborn (Alt.)
Robert Otis (Alt.)
Dan Paplowski (Alt.)
Doug Pickford (Alt.)
Gregorio Rodriguez (Alt.)
Russell Smith (Alt.)
D. W. Spence (Alt.)
Gary R. Stephens (Alt.)
Peter Stevens (Alt.)
Arlan P. Stone (Alt.)
Sassan Teymouri (Alt.)
David Thayer (Alt.)
Paul Thompson (Alt.)
Adrienne Turenne (Alt.)
Rex Vedder (Alt.)
Mark Veteikis (Alt.)
Bob Whiteman (Alt.)
Jeffrey L. Williams (Alt.)
Natalie Willman (Alt.)
Michael Wingard (Alt.)
Paul Wolf (Alt.)
Fred Yamamoto (Alt.)
Mike Yokoyama (Alt.)

Introduction

ANSI X3.131-1994 replaces ANSI X3.131-1986 *Small Computer System Interface.*

The clauses contain material as described below.

Clause 1	describes the scope.
Clause 2	lists the normative references.
Clause 3	provides a glossary common to the whole document.
Clause 4	provides descriptions and conventions.
Clause 5	describes the physical characteristics.
Clause 6	describes the logical characteristics of the interface.
Clause 7	describes the SCSI command and status structure.
Clause 8	specifies those commands that have a consistent meaning for all device types.
Clause 9	specifies commands for direct-access devices.
Clause 10	specifies commands for sequential-access devices.
Clause 11	specifies commands for printer devices.
Clause 12	specifies commands for processor devices.
Clause 13	specifies commands for write-once devices.
Clause 14	specifies commands for CD-ROM devices.
Clause 15	specifies commands for scanner devices.
Clause 16	specifies commands for optical memory devices.
Clause 17	specifies commands for medium changer devices.
Clause 18	specifies commands for communications devices.
Annex A	illustrates SCSI signal sequence.
Annex B	illustrates fast SCSI skew time.
Annex C	describes other SCSI standardization activities.
Annex D	contains SCSI-2 additional sense codes and operation codes in numeric order.
Annex E	contains the list of SCSI-2 vendor identifications.

The SCSI protocol is designed to provide an efficient peer-to-peer I/O bus with up to 16 devices, including one or more hosts. Data may be transferred asynchronously at rates that only depend on device implementation and cable length. Synchronous data transfers are supported at rates up to 10 mega-transfers per second. With the 32-bit wide data transfer option, data rates of up to 40 megabytes per second are possible.

SCSI-2 includes command sets for magnetic and optical disks, tapes, printers, processors, CD-ROMs, scanners, medium changers, and communications devices.

In 1985, when the first SCSI standard was being finalized, several manufacturers wanted to increase the mandatory requirements of SCSI and to define further features for direct-access devices. Rather than delay the SCSI standard, an ad hoc group was formed to develop a working paper that was eventually called the Common Command Set (CCS). Many disk products were designed using this working paper.

In parallel with the development of the CCS working paper, work began on an enhanced SCSI standard, which was named SCSI-2. SCSI-2 included the results of the CCS working paper and extended them to all device types. It also added caching commands, performance enhancement features, and other worthwhile functions. While SCSI-2 has gone well beyond the original SCSI standard (now referred to as SCSI-1), it retains a high degree of compatibility with SCSI-1 devices.

SCSI-2 has evolved significantly from SCSI-1 with the new document nearly three times larger. Most of the changes are additions, but several obsolete options were removed:

a) Single initiator option;

b) Non-arbitrating systems option;

c) The SCSI-1 alternative 1 shielded connector;

d) Non-extended sense data option;

e) Reservation queuing option;

f) The read-only device command set.

There are several new low-level requirements:

a) Parity is now required;

b) Initiators are required to provide terminator power;

c) The arbitration delay was increased from 2.2 to 2.4 μs;

d) Message support is now required.

Several low-level options were added:

a) Wide SCSI (up to 32 bits wide using a second cable);

b) Fast SCSI (synchronous data transfers of up to 10 mega-transfers per second);

c) Command queuing (up to 256 commands per initiator per logical unit);

d) High-density connector alternatives were added;

e) Asynchronous event notification;

f) Extended contingent allegiance.

New command sets were added including:

a) CD-ROM (replaces read-only device);

b) Scanner device;

c) Optical memory device (provides for write-once, read-only, and erasable media);

d) Medium changer device;

e) Communications device.

All command sets were enhanced:

a) Device models were added;

b) Extended sense was expanded;

c) The INQUIRY data was expanded;

d) The MODE SELECT and MODE SENSE commands were paged for all device types;

e) The CHANGE DEFINITION, LOG SELECT, LOG SENSE, READ BUFFER, and WRITE BUFFER commands were added for all device types;

f) The COPY command definition was expanded to include inexact block size handling and an image copy option;

g) The direct-access device command set was enhanced to add cache management, several new commands and to provide more initiator control over defect management;

h) The sequential-access device command set was enhanced to add a partitioned media concept;

i) The printer device command set was enhanced by adding several mode pages;

j) The write-once (optical) device command set was enhanced by adding several new commands plus extending several command descriptor blocks to twelve bytes to accommodate larger transfer lengths.

American National Standard
for Information Systems –

Information Technology –
Small Computer System Interface-2

1 Scope

This standard defines an input/output bus for interconnecting computers and peripheral devices. It defines extensions to the Small Computer System Interface (ISO 9316: 1989), referred to herein as SCSI-1. It also provides more complete standardization of the previously defined command sets. It includes the necessary specification of the mechanical, electrical, and functional characteristics of the interface to allow interoperability of conforming devices. This standard is referred to herein as SCSI-2. The term, SCSI, is used wherever it is not necessary to distinguish between the two versions.

2 Normative references

The following standards contain provisions which, through reference in this text, constitute provisions of this American National Standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this American National Standard are encouraged to investigate the possibility of applying the most recent editions of the standards and publications listed below.

Members of IEC and ISO maintain registers of currently valid standards.

ANSI X3.170a-1991, *Information systems – Enhanced small device interface* (available only as part of ANSI X3.170-1990)

EIA RS-485-1983, *Standard for electrical characteristics of generators and receivers for use in balanced digital multipoint systems* [1)]

IEC 908: 1987, *Compact disc digital audio system* [2)]

ISO/IEC 10149: 1989, *Information technology – Data interchange on read-only 120 mm optical data disks (CD-ROM)* [2)]

[1)] Available from the Electronic Industries Association, 2001 Pennsylvania Avenue, NW, Washington, DC 20006.

[2)] Available from the American National Standards Institute, 11 West 42nd Street, New York, NY 10036.

3 Definitions, symbols and abbreviations

3.1 Definitions

For the purposes of this standard, the following definitions apply.

3.1.1 active I/O process: An I/O process that is presently in execution (not queued).

3.1.2 byte: Indicates an 8-bit construct.

3.1.3 command descriptor block (CDB): The structure used to communicate commands from an initiator to a target.

3.1.4 command queue: The queue used to store the queued I/O processes (see 7.8).

3.1.5 connect: The initiator function that selects a target to establish a nexus and to start an I/O process. The connection that results is an initial connection.

3.1.6 connection: An initial connection or reconnection. A connection can only occur between one initiator and one target.

3.1.7 contact: The electrically-conductive portion of a connector associated with a single conductor in a cable.

3.1.8 contingent allegiance: A condition typically generated by a CHECK CONDITION status during which a target preserves sense data (see 7.6).

3.1.9 current I/O process: The I/O process that is presently connected on the SCSI bus.

3.1.10 disconnect: The action that occurs when an SCSI device releases control of the SCSI bus, allowing it to go to the BUS FREE phase.

3.1.11 extended contingent allegiance: A condition generated by an INITIATE RECOVERY message to assist in extended error recovery procedures in multi-initiator systems (see 7.7).

3.1.12 field: A group of one or more contiguous bits.

3.1.13 host adapter: A device which connects between a host system and the SCSI bus. The device usually performs the lower layers of the SCSI protocol and normally operates in the initiator role. This function may be integrated into the host system.

3.1.14 initial connection: An initial connection is the result of a connect and it exists from the assertion of the BSY signal in a SELECTION phase until the next BUS FREE phase occurs.

3.1.15 initiator: An SCSI device that requests an I/O process to be performed by another SCSI device (a target).

3.1.16 invalid: An illegal (reserved) or unsupported field or code value.

3.1.17 I/O process: An I/O process consists of one initial connection and zero or more reconnections, all pertaining to a single command or a group of linked commands. More specifically, the connection(s) pertain to a nexus in which zero or more command descriptor blocks are transferred. An I/O process begins with the establishment of a nexus. An I/O process normally ends with the BUS FREE phase following successful transfer of a COMMAND COMPLETE or a RELEASE RECOVERY message. An I/O process also ends with the BUS FREE phase following an ABORT, ABORT TAG, BUS DEVICE RESET, CLEAR QUEUE message, or a hard RESET condition, or an unexpected disconnect occurs.

3.1.18 I_T nexus: A nexus which exists between an initiator and a target.

3.1.19 I_T_L nexus: A nexus which exists between an initiator, a target, and a logical unit. This relationship replaces the prior I_T nexus.

3.1.20 I_T_R nexus: A nexus which exists between an initiator, a target, and a target routine. This relationship replaces the prior I_T nexus.

3.1.21 I_T_x nexus: A nexus which is either an I_T_L or I_T_R nexus.

3.1.22 I_T_L_Q nexus: A nexus between an initiator, a target, a logical unit, and a queue tag following the successful receipt of one of the queue tag messages. This relationship replaces the prior I_T_L nexus.

3.1.23 I_T_x_y nexus: A nexus which is either an I_T_x or I_T_L_Q.

3.1.24 logical block: A unit of data supplied or requested by an initiator.

3.1.25 logical unit: A physical or virtual peripheral device addressable through a target.

3.1.26 logical unit number: An encoded three-bit identifier for the logical unit.

3.1.27 mandatory: The referenced item is required to claim compliance with this standard.

3.1.28 nexus: A relationship that begins with the establishment of an initial connection and ends with the completion of the I/O process. The relationship may be restricted to specify a single logical unit or target routine by the successful transfer of an IDENTIFY message. The relationship may be further restricted by the successful transfer of a queue tag message.

3.1.29 one: A true signal value or a true condition of a variable.

3.1.30 optional: The referenced item is not required to claim compliance with this standard. Implementation of an optional item must be as defined in this standard.

3.1.31 page: Several commands use regular parameter structures that are referred to as pages. These pages are identified with a value known as a page code.

3.1.32 peripheral device: A physical peripheral device that can be attached to an SCSI device, which connects to the SCSI bus. The peripheral device and the SCSI device (peripheral controller) may be physically packaged together. Often there is a one-to-one mapping between peripheral devices and logical units, but this is not required. Examples of peripheral devices are: magnetic disks, printers, optical disks, and magnetic tapes.

3.1.33 queue tag: The value associated with an I/O process that uniquely identifies it from other queued I/O processes in the logical unit for the same initiator.

3.1.34 queued I/O process: An I/O process that is in the command queue.

3.1.35 reconnect: The act of reviving a nexus to continue an I/O process. A target reconnects to an initiator by using the RESELECTION and MESSAGE IN phases after winning arbitration. An initiator reconnects to a target by using the SELECTION and MESSAGE OUT phases after winning arbitration (see 7.5.2).

3.1.36 reconnection: A reconnection is the result of a reconnect and it exists from the assertion of the BSY signal in a SELECTION or RESELECTION phase until the next BUS FREE phase occurs.

3.1.37 reserved: Identifies bits, fields, and code values that are set aside for future standardization.

3.1.38 SCSI address: The hex representation of the unique address (0-15) assigned to an SCSI device. This address would normally be assigned and set in the SCSI device during system installation.

3.1.39 SCSI ID: The bit-significant representation of the SCSI address referring to one of the signal lines DB(7-0).

3.1.40 SCSI device: A host adapter or a target controller that can be attached to the SCSI bus.

3.1.41 signal assertion: The act of driving a signal to the true state.

3.1.42 signal negation: The act of driving a signal to the false state or allowing the cable terminators to bias the signal to the false state (by placing the driver in the high impedance condition).

3.1.43 signal release: The act of allowing the cable terminators to bias the signal to the false state (by placing the driver in the high impedance condition).

3.1.44 status: One byte of information sent from a target to an initiator upon completion of each command.

3.1.45 target: An SCSI device that performs an operation requested by an initiator.

3.1.46 target routine: A target routine is an I/O process directed to a target, and not to a logical unit (see 6.6.7).

3.1.47 third-party: When used in reference to COPY commands, third-party means a COPY command issued to one device to perform a copy operation between two other devices. When used in reference to RESERVE, or RELEASE commands, third-party means a reservation made on behalf of another device (e.g. A processor device requests that a direct-access device reserve itself for exclusive use by a sequential-access device).

3.1.48 unexpected disconnect: A disconnection that occurs as a result of an exception condition (see 6.1.1).

3.1.49 vendor-specific (VS): Something (e.g. a bit, field, code value, etc.) that is not defined by this standard and may be used differently in various implementations.

3.1.50 zero: A false signal value or a false condition of a variable.

3.2 Symbols and abbreviations

AEN Asynchronous event notification (see 7.5.5)
AWG American Wire Gauge
LSB Least significant bit
LUN Logical unit number
MSB Most significant bit
SCSI Either SCSI-1 or SCSI-2
SCSI-1 The Small Computer System Interface (ISO 9316:1989)
SCSI-2 The Small Computer System Interface - 2 (this standard)

4 General

4.1 Overview

SCSI is a local I/O bus that can be operated over a wide range of data rates. The primary objective of the interface is to provide host computers with device independence within a class of devices. Thus, different disk drives, tape drives, printers, optical media drives, and other devices can be added to the host computers without requiring modifications to generic system hardware or software. Provision is made for the addition of special features and functions through the use of vendor unique fields and codes. Reserved fields and codes are provided for future standardization.

A second key objective of SCSI-2 is to provide compatibility with those SCSI-1 devices that support bus parity and that meet conformance level 2 of SCSI-1. While some previously vendor unique commands and parameters have been defined by the SCSI-2 standard, devices meeting SCSI-1 and SCSI-2 can co-exist on the same bus. It is intended that those operating systems providing support for both command sets be able to operate in environments mixing SCSI-1 and SCSI-2 devices. Properly conforming SCSI-1 devices, both initiators and targets, should respond in an acceptable manner to reject SCSI-2 protocol extensions. All SCSI-2 protocol extensions are designed to be permissive of such rejections and to allow the SCSI-1 device to continue operation without requiring the use of the extension.

A third key objective of SCSI-2 is to move device-dependent intelligence out to the SCSI-2 devices. The command set definitions allow a sophisticated operating system to obtain all required initialization information from the attached SCSI-2 devices. The formalized sequence of requests identify the type of attached SCSI-2 device, the characteristics of the device, and all the changeable parameters supported by the device. Further requests can determine the readiness of the device to operate, the types of media supported by the device, and all other pertinent system information. Those parameters not required by the operating system for operation, initialization, or system tuning are not exposed to the SCSI-2 interface, but are managed by the SCSI-2 device itself.

The interface uses logical rather than physical addressing for all data blocks. For direct-access devices, each logical unit may be interrogated to determine how many blocks it contains. A logical unit may coincide with all or part of a peripheral device.

The interface protocol includes provision for the connection of multiple initiators (SCSI devices capable of initiating an operation) and multiple targets (SCSI devices capable of responding to a request to perform an operation). Distributed arbitration (i.e. bus-contention logic) is built into the architecture of SCSI. A priority system awards interface control to the highest priority SCSI device that is contending for use of the bus. The time to complete arbitration is independent of the number of devices that are contending and can be completed in less than 10 μs.

There are two electrical alternatives: single-ended and differential. Single-ended and differential devices are electrically incompatible and can not be mixed on the same physical bus.

Provision is made for cable lengths up to 25 m using differential drivers and receivers. A single-ended driver and receiver configuration is defined for cable lengths of up to 6 m and is primarily intended for applications within a cabinet.

Arbitration is defined to permit multiple initiators and to permit concurrent I/O operations. All SCSI devices are required to be capable of operating with the defined asynchronous transfer protocol. In addition, an optional synchronous transfer protocol is defined. A message protocol for control of the interface is also specified. In most cases, messages are not directly apparent to the host computer software.

Commands are classified as mandatory, optional, or vendor-specific. SCSI devices are required to implement all mandatory commands defined for the appropriate device type and may implement other commands as well. SCSI devices contain commands that facilitate the writing of self-configuring software drivers that can discover all necessary attributes without prior knowledge of specific peripheral characteristics (such as storage capacity). Many commands

also implement a very large logical block address space (2^{32} blocks), although some commands implement a somewhat smaller logical block address space (2^{21} blocks).

Starting with clause 8 and for each clause on a specific device type, the clause is constructed of at least four subclauses. The first subclause is the model for the device type. The model establishes the framework for interpreting the commands for the device type. The attributes and capabilities of the device type are discussed and examples are given. The second subclause defines the commands applicable to the device type. The third subclause defines the parameters applicable to the device type. These are the diagnostic parameters, log parameters, mode parameters and vital product data parameters that are transmitted as part of the appropriate commands. Most of the parameters are formatted into pages. The fourth subclause gives the definition of terms that apply specifically to that device type.

Starting with clause 9 the commands in each of these clauses are unique to the device type, or they have interpretations, fields, or features that are specific for the device type. Thus, for example, although the WRITE command is used for several device types, it has a somewhat different form for each type, with different parameters and meanings. Therefore, it is specified separately for each device type.

4.2 Conventions

Certain words and terms used in this standard have a specific meaning beyond the normal English meaning. These words and terms are defined either in clause 3 or in the text where they first appear. Names of signals, phases, messages, commands, statuses, sense keys, additional sense codes, and additional sense code qualifiers are in all uppercase (e.g. REQUEST SENSE). Lower-case is used for words having the normal English meaning.

Fields containing only one bit are usually referred to as the name bit instead of the name field.

Numbers that are not immediately followed by lower-case b or h are decimal values.

Numbers immediately followed by lower-case b (xxb) are binary values.

Numbers immediately followed by lower-case h (xxh) are hexadecimal values.

5 Physical characteristics

This clause contains the physical definition of SCSI-2. The connectors, cables, signals, terminators, and bus timin values needed to implement the interface are covered.

5.1 Physical description

SCSI devices are daisy-chained together using a common 50-conductor A cable and, optionally, a 68-conductor I cable. Both ends of each cable are terminated. All signals are common between all SCSI devices on the A cable In systems that employ the wide SCSI option, wide SCSI devices additionally connect to the B cable. Various widt SCSI devices may be mixed.

NOTE 1 An alternate 16-bit single-cable solution and an alternate 32-bit solution is being defined and the B cable definition will be removed in a future version of SCSI.

Two driver/receiver alternatives are specified:

a) Single-ended drivers and receivers, which allow a maximum cable length of 6 m (primarily for connection withi an enclosure).
b) Differential drivers and receivers, which allow a maximum cable length of 25 m.

The single-ended and differential alternatives are mutually exclusive on the same physical bus.

NOTE 2 Use of single-ended drivers and receivers with the fast synchronous data transfer option is not recommended.

5.2 Cable requirements

The characteristic impedance of the cable should be no less than 90 Ω and no greater than 140 Ω. The characteristi impedance of the cable used when implementing the fast synchronous data transfer option is defined in 5.2.3.

NOTE 3 There are successful single-ended implementations using cables with less than 90 Ω characteristic impedance. However, system integrity in single-ended implementations is improved when the characteristic impedance of the cable is greater than 90 Ω. Cable parameters other than characteristic impedance are critical to system integrity. Alternative cable parameters are being investigated as a part of a future version of SCSI.

A minimum conductor size of 0,080 42 mm^2 (28 AWG) should be used to minimize noise effects and ensure prope distribution of terminator power. A smaller conductor size may be used for signals other than terminator power.

NOTES
4 To minimize discontinuities and signal reflections, cables of different impedances should not be used in the same bus. Implementations may require trade-offs in shielding effectiveness, cable length, the number of loads, transfer rates, and cost to achieve satisfactory system operation.
5 To minimize discontinuities due to local impedance variation, a flat cable should be spaced at least 1,27 mm (0,050 in) from other cables, any other conductor, or the cable itself when the cable is folded.
6 Regulatory agencies may require use of larger wire size.

5.2.1 Single-ended cable

A 50-conductor flat cable or 25-signal twisted-pair cable shall be used for the A cable. A 68-conductor flat cable o 34-signal twisted-pair cable shall be used for the B cable if the wide SCSI option is implemented. The maximun cumulative cable length shall be 6,0 m. If twisted-pair cables are used, then twisted pairs in the cable shall be wire to physically opposing contacts in the connector.

A stub length of no more than 0,1 m is allowed off the mainline interconnection within any connected equipment o from any connected point.

NOTE 7 Stub clustering should be avoided. Stubs should be spaced at least 0,3 m apart.

SCSI bus termination shall be at each end of the cable and may be internal to the SCSI devices that are at the ends of the cable.

5.2.2 Differential cable

A 50-conductor flat cable or 25-signal twisted-pair cable shall be used for the A cable. A 68-conductor flat cable or 34-signal twisted-pair cable shall be used for the B cable if the wide SCSI option is implemented. The maximum cumulative cable length shall be 25 m. If twisted-pair cables are used, then twisted pairs in the cable shall be wired to physically opposing contacts in the connector.

A stub length of no more than 0,2 m is allowed off the mainline interconnection within any connected equipment or from any connected point.

SCSI bus termination shall be at each end of the cable and may be internal to the SCSI devices that are at the ends of the cable.

NOTE 8 The use of twisted pair cable (either twisted-flat or discrete wire twisted pairs) is strongly recommended. Without twisted pairs, even at slow data rates and very short distances, crosstalk between adjacent signals causes spurious pulses with differential signals.

5.2.3 Cable requirements for fast synchronous data transfer

In systems which use the fast synchronous data transfer option (see 5.8), the A and B cables should meet the conductor size recommendation in 5.2. The cable should have an overall shield suitable for termination in a shielded connector.

In such systems, the cables shall have the following electrical characteristics:

Characteristic impedance:	90 Ω to 132 Ω
Signal attenuation:	0,095 dB maximum per metre at 5 Mhz
Pair-to-pair propagation delay delta:	0,20 ns maximum per metre
DC resistance:	0,230 Ω maximum per metre at 20°C

5.3 Connector requirements

Two types of connectors are defined: non-shielded and shielded. The non-shielded connectors are typically used for in-cabinet applications. Shielded connectors are typically used for external applications where electromagnetic compatibility (EMC) and electrostatic discharge (ESD) protection may be required. Either type of connector may be used with the single-ended or differential drivers.

5.3.1 Non-shielded connector requirements

Two non-shielded connector alternatives are specified for the A cable and one non-shielded connector is specified for the B cable.

5.3.1.1 Non-shielded connector alternative 1 - A cable

The alternative 1 non-shielded high-density SCSI device connector for the A cable (see figure 1) shall be a 50-conductor connector consisting of two rows of 25 female contacts with adjacent contacts 1,27 mm (0,05 in) apart. The non-mating portion of the connector is shown for reference only.

The alternative 1 non-shielded high-density cable connector for the A cable (see figure 2) shall be a 50-conductor connector consisting of two rows of 25 male contacts with adjacent contacts 1,27 mm (0,05 in) apart. The non-mating portion of the connector is shown for reference only.

5.3.1.2 Non-shielded connector alternative 2 - A cable

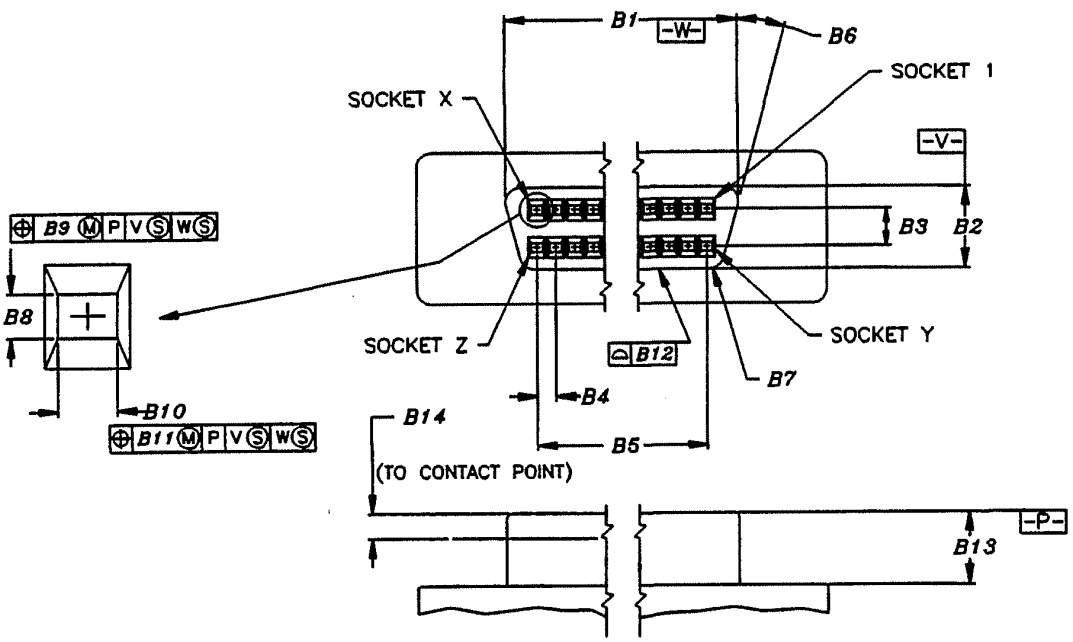
The alternative 2 non-shielded low-density SCSI device connector for the A cable (see figure 3) shall be a 5C conductor connector consisting of two rows of 25 male pins with adjacent pins 2,54 mm (0,1 in) apart. A shroud an header body should be used. The non-mating portion of the connector is shown for reference only.

The alternative 2 non-shielded low-density cable connector for the A cable (see figure 4) shall be a 50-conductc connector consisting of two rows of 25 female contacts with adjacent contacts 2,54 mm (0,1 in) apart. It i recommended that keyed connectors be used.

5.3.1.3 Non-shielded connector - B cable

The non-shielded high-density SCSI device connector for the B cable (see figure 1) shall be a 68-conductor connectc consisting of two rows of 34 female contacts with adjacent contacts 1,27 mm (0,05 in) apart. The non-mating portio of the connector is shown for reference only.

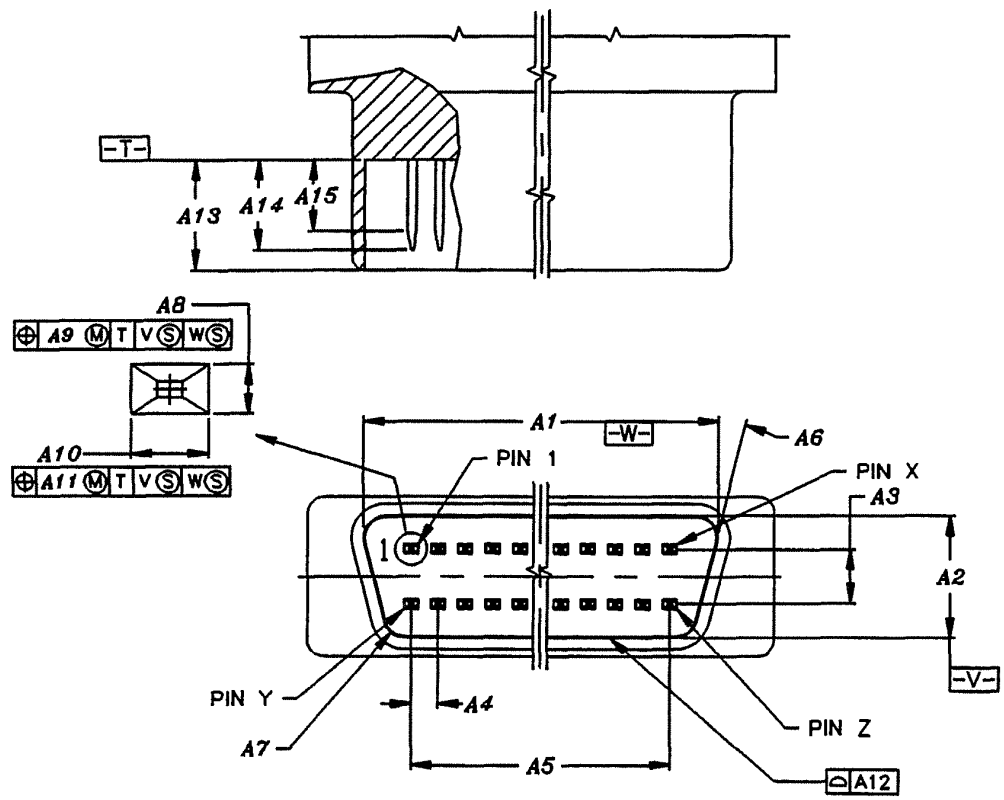
The non-shielded high-density cable connector for the B cable (see figure 2) shall be a 68-conductor connectc consisting of two rows of 34 male contacts with adjacent contacts 1,27 mm (0,05 in) apart. The non-mating portio of the connector is shown for reference only.



Dimensions	50 Position		68 Position	
	mm	in	mm	in
B1	34,70	1,366	46,13	1,816
B2	5,54	0,218	5,54	0,218
B3	2,54	0,100	2,54	0,100
B4	1,27	0,050	1,27	0,050
B5	30,48	1,200	41,91	1,650
B6	15°	15°	15°	15°
B7	1,00 R	0,039 R	1,00 R	0,039 R
B8	0,61±0,05	0,024±0,002	0,61±0,05	0,024±0,002
B9	0,15	0,006	0,15	0,006
B10	0,86±0,10	0,034±0,004	0,86±0,10	0,034±0,004
B11	0,15	0,006	0,15	0,006
B12	0,05	0,002	0,05	0,002
B13	5,00±0,13	0,197±0,005	5,00±0,13	0,197±0,005
B14	1,75 max.	0,069 max.	1,75 max.	0,069 max.
Socket X	25		34	
Socket Y	26		35	
Socket Z	50		68	

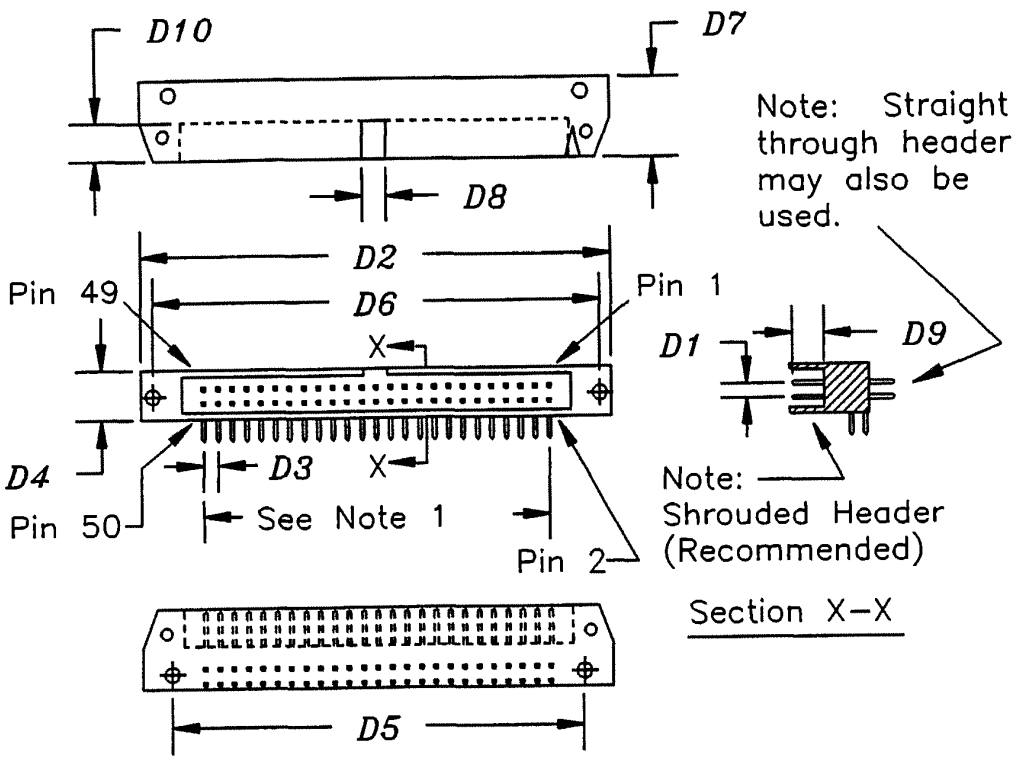
NOTE: Dimensions *B8* and *B10* are the opening in the dielectric. The socket contacts (not shown) fit wthin the opening.

Figure 1 - 50/68-contact non-shielded high-density SCSI device connector (A cable/B cable)



Dimensions	50 Position		68 Position	
	mm	in	mm	in
A1	34,85	1,372	46,28	1,822
A2	5,69	0,224	5,69	0,224
A3	2,54	0,100	2,54	0,100
A4	1,27	0,050	1,27	0,050
A5	30,48	1,200	41,91	1,650
A6	15°	15°	15°	15°
A7	1,04 R	0,041 R	1,04 R	0,041 R
A8	0,40±0,010	0,0156±0,0004	0,40±0,010	0,0156±0,0004
A9	0,23	0,009	0,23	0,009
A10	0,60±0,03	0,024±0,001	0,60±0,03	0,024±0,001
A11	0,23	0,009	0,23	0,009
A12	0,05	0,002	0,05	0,002
A13	5,15±0,15	0,203±0,006	5,15±0,15	0,203±0,006
A14	4,39 max.	0,173 max.	4,39 max.	0,173 max.
A15	3,02 min.	0,119 min.	3,02 min.	0,119 min.
Pin X	25		34	
Pin Y	26		35	
Pin Z	50		68	

Figure 2 - 50/68-contact non-shielded high-density cable connector (A cable/B cable)



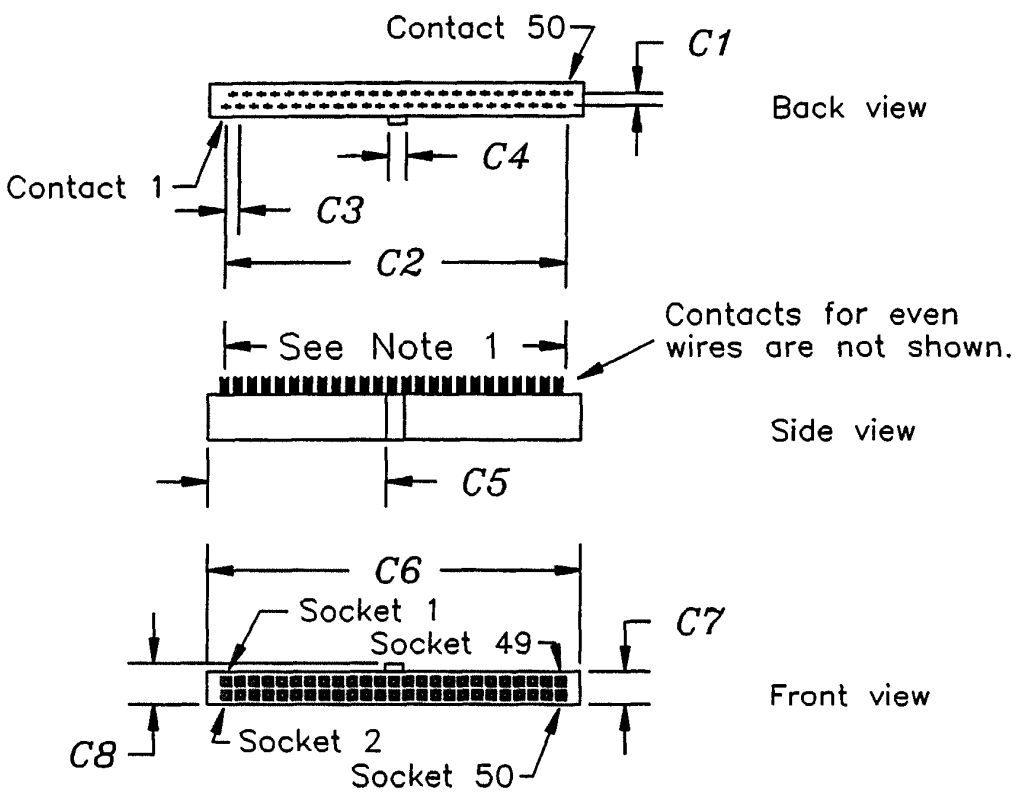
Dimension	mm	in	Comments
D1	2,54	0,100	
D2	82,80	3,260	Reference Only
D3	2,54	0,100	
D4	8,89	0,350	Reference Only
D5	72,64	2,860	Reference Only
D6	78,74	3,100	Reference Only
D7	13,94	0,549	Reference Only
D8	4,19±0,25	0,165±0,010	
D9	6,09	0,240	
D10	6,60	0,260	Reference Only

NOTES

1 Two rows of twenty five contacts on 2,540 mm (0,100 in) spacing = 60,960 mm (2,400 in).

2 Tolerances ±0,127 mm (0,005 in) non–cumulative, unless specified otherwise.

Figure 3 - 50-contact non-shielded low-density SCSI device connector (A cable)



Dimensions	mm	in	Comments
C1	2,54	0,100	
C2	60,96	2,400	
C3	2,54	0,100	
C4	3,30	0,130	
C5	32,38	1,275	
C6	68,07	2,680	
C7	6,10	0,240	
C8	7,62	0,300	Maximum

NOTES

1 Fifty contacts on 1,270 mm (0,050 in) staggered spacing = 62,230 mm (2,450 in) [reference only].

2 Tolerances ±0,127 mm (0,005) non-cumulative, unless specified otherwise.

3 Connector cover and strain relief are optional.

Figure 4 - 50-contact non-shielded low-density cable connector (A cable)

5.3.2 Shielded connector requirements

Two shielded connector alternatives are specified for the A cable and one shielded connector is specified for the B cable. The connector shielding system should provide a d.c. resistance of less than 10 mΩ from the cable shield at its termination point to the SCSI device enclosure.

In order to support daisy-chain connections, SCSI devices that use shielded connectors should provide two shielded device connectors on the device enclosure. These two connectors may be wired one-to-one with a stub to the SCSI device's drivers and receivers provided the maximum stub length is not violated. Alternatively, two cables may be run from the two shielded connectors to the drivers and receivers so that the maximum stub length is not violated. The length of the cable within the device enclosure is included when calculating the total cable length of the SCSI bus.

NOTE 9 SCSI-1 defined three shielded connector systems in an annex. The alternative 1 shielded connector of SCSI-1 has been replaced by a high-density connector in this standard. The alternative 2 shielded connector remains unchanged. The EUROCARD boxes shielded connector system of SCSI-1 has been deleted in this standard.

5.3.2.1 Shielded connector alternative 1 - A cable

The shielded high-density SCSI device connector for the A cable (see figure 5) is a 50-conductor connector consisting of two rows of 25 female contacts with adjacent contacts 1,27 mm (0,05 in) apart. The non-mating portion of the connector is shown for reference only.

The shielded high-density cable connector for the A cable (see figure 6) is a 50-conductor connector consisting of two rows of 25 male contacts with adjacent contacts 1,27 mm (0,05 in) apart. The non-mating portion of the connector is shown for reference only.

5.3.2.2 Shielded connector alternative 2 - A cable

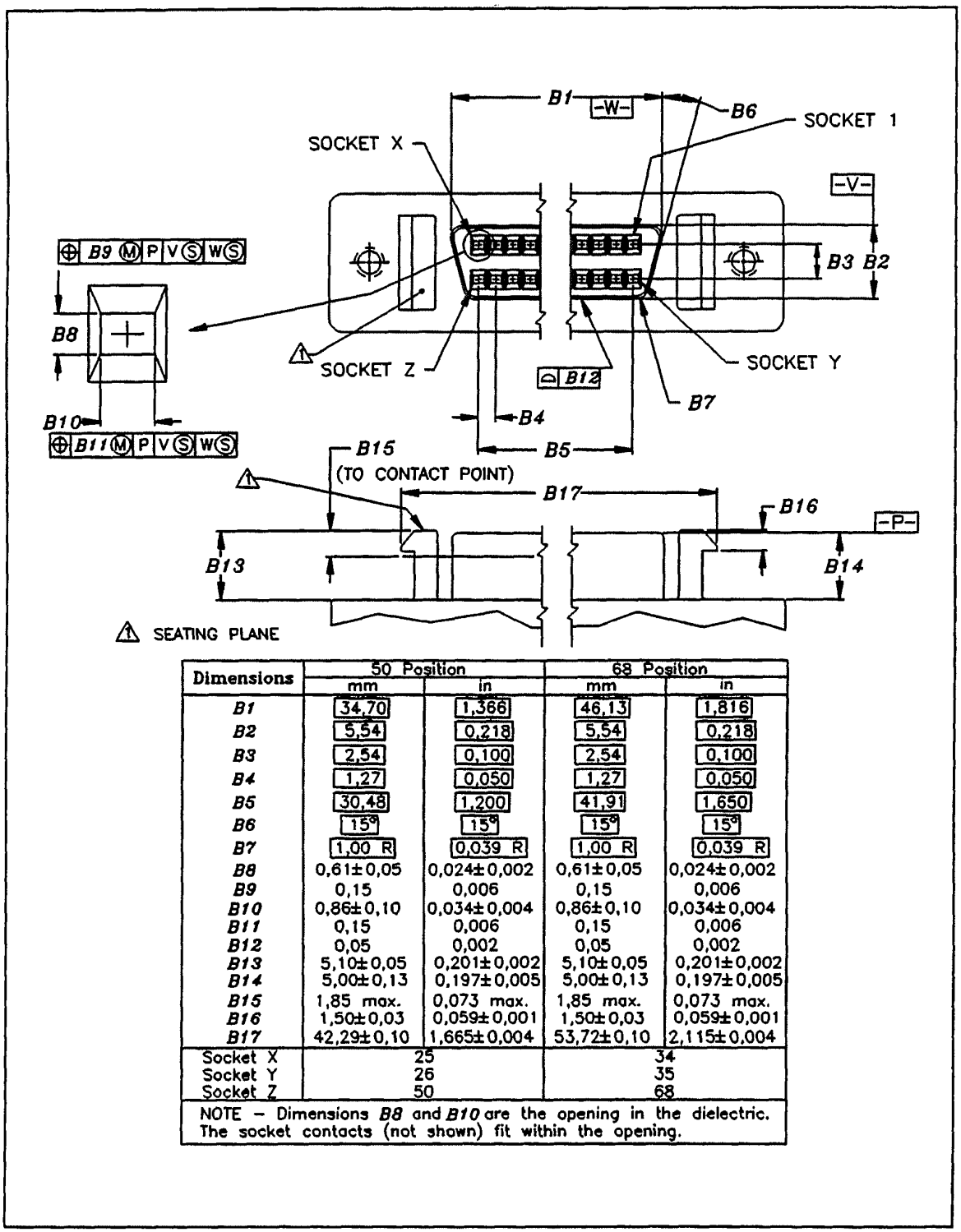
The shielded low-density device connector for the A cable (see figure 7) is a 50-conductor connector consisting of two rows of ribbon contacts spaced 2,16 mm (0,085 in) apart. The non-mating portion of the connector is shown for reference only.

The shielded low-density cable connector for the A cable (see figure 8) is a 50-conductor connector consisting of two rows of ribbon contacts spaced 2,16 mm (0,085 in) apart. The non-mating portion of the connector is shown for reference only.

5.3.2.3 Shielded connector - B cable

The shielded high-density SCSI device connector for the B cable (see figure 5) is a 68-conductor connector consisting of two rows of 34 female contacts with adjacent contacts 1,27 mm (0,05 in) apart. The non-mating portion of the connector is shown for reference only.

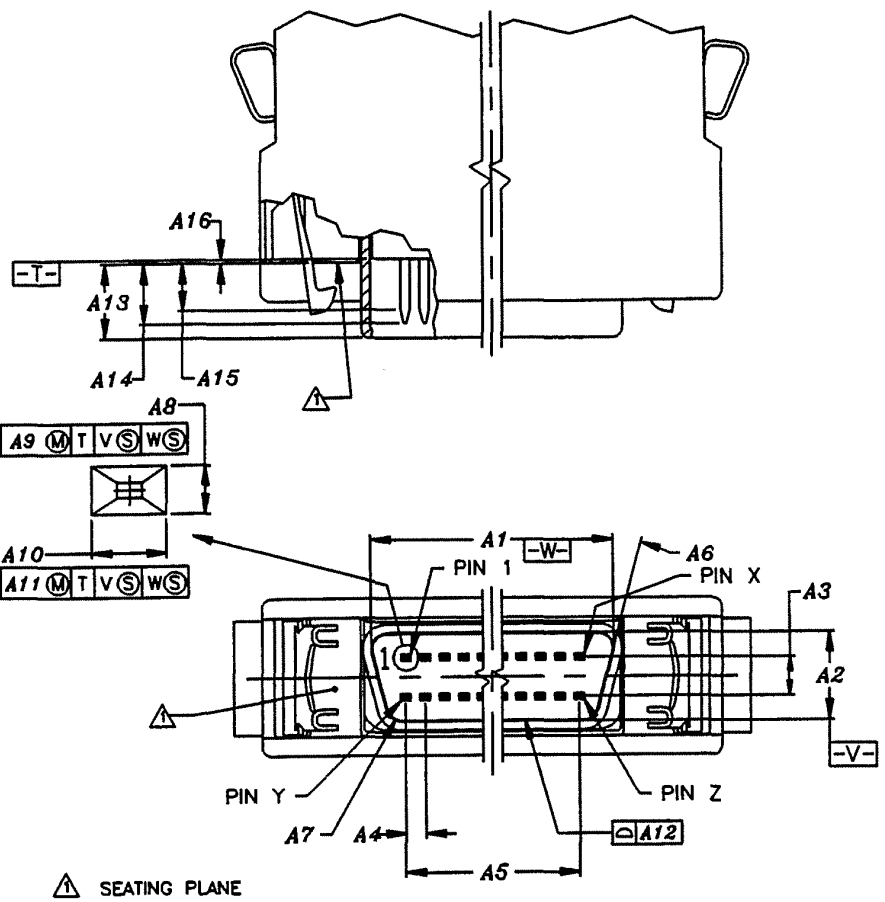
The shielded high-density cable connector for the B cable (see figure 6) is a 68-conductor connector consisting of two rows of 34 male contacts with adjacent contacts 1,27 mm (0,05 in) apart. The non-mating portion of the connector is shown for reference only.



Dimensions	50 Position		68 Position	
	mm	in	mm	in
B1	34,70	1,366	46,13	1,816
B2	5,54	0,218	5,54	0,218
B3	2,54	0,100	2,54	0,100
B4	1,27	0,050	1,27	0,050
B5	30,48	1,200	41,91	1,650
B6	15°	15°	15°	15°
B7	1,00 R	0,039 R	1,00 R	0,039 R
B8	0,61±0,05	0,024±0,002	0,61±0,05	0,024±0,002
B9	0,15	0,006	0,15	0,006
B10	0,86±0,10	0,034±0,004	0,86±0,10	0,034±0,004
B11	0,15	0,006	0,15	0,006
B12	0,05	0,002	0,05	0,002
B13	5,10±0,05	0,201±0,002	5,10±0,05	0,201±0,002
B14	5,00±0,13	0,197±0,005	5,00±0,13	0,197±0,005
B15	1,85 max.	0,073 max.	1,85 max.	0,073 max.
B16	1,50±0,03	0,059±0,001	1,50±0,03	0,059±0,001
B17	42,29±0,10	1,665±0,004	53,72±0,10	2,115±0,004
Socket X	25		34	
Socket Y	26		35	
Socket Z	50		68	

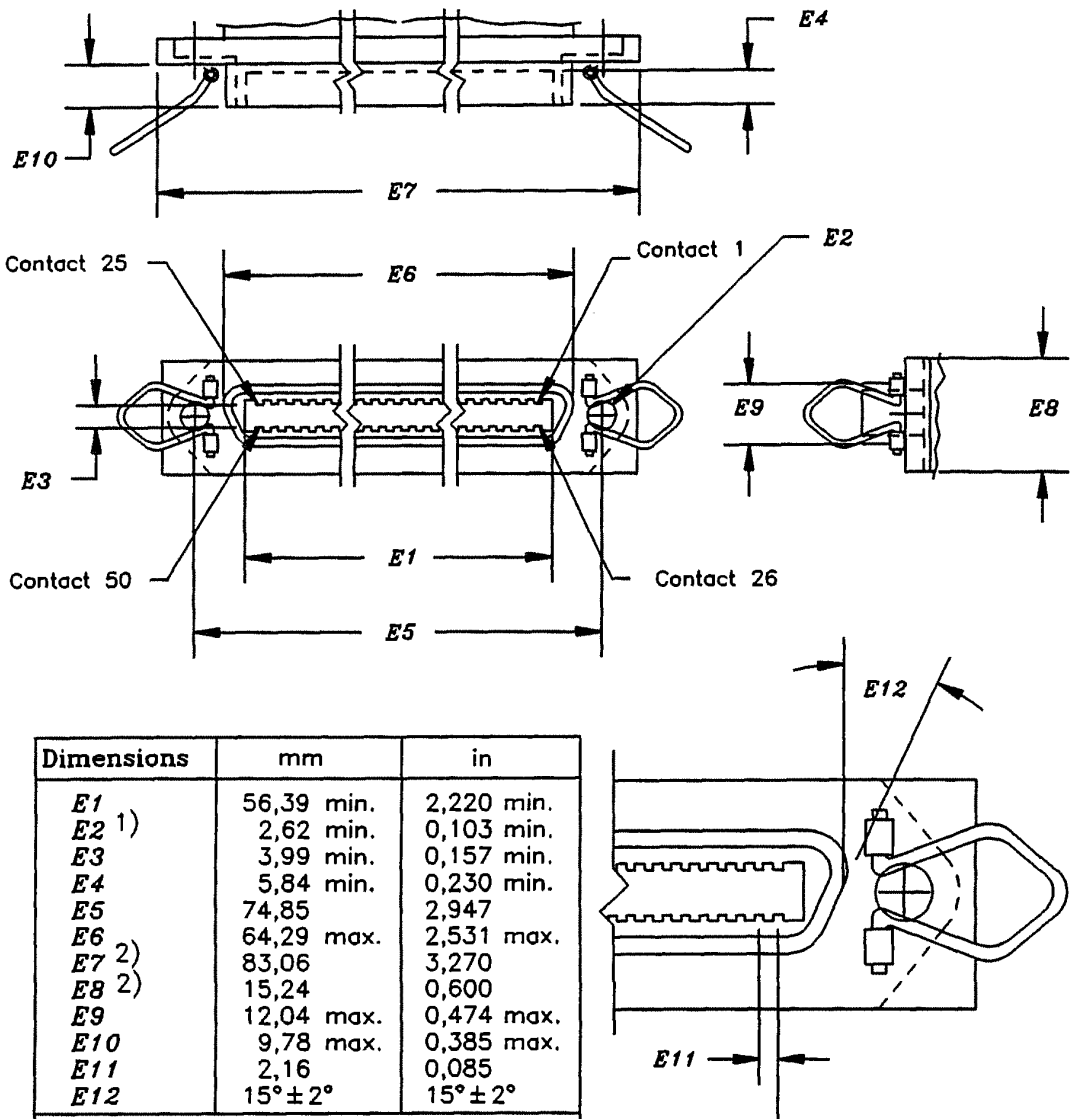
NOTE – Dimensions *B8* and *B10* are the opening in the dielectric. The socket contacts (not shown) fit within the opening.

Figure 5 - 50/68-contact shielded high-density SCSI device connector (A cable/B cable)



Dimensions	50 Position		68 Position	
	mm	in	mm	in
A1	34,85	1,372	46,28	1,822
A2	5,69	0,224	5,69	0,224
A3	2,54	0,100	2,54	0,100
A4	1,27	0,050	1,27	0,050
A5	30,48	1,200	41,91	1,650
A6	15°	15°	15°	15°
A7	1,04 R	0,041 R	1,04 R	0,041 R
A8	0,40±0,010	0,0156±0,0004	0,40±0,010	0,0156±0,0004
A9	0,23	0,009	0,23	0,009
A10	0,60±0,03	0,024±0,001	0,60±0,03	0,024±0,001
A11	0,23	0,009	0,23	0,009
A12	0,05	0,002	0,05	0,002
A13	4,90±0,10	0,193±0,004	4,90±0,10	0,193±0,004
A14	4,27 max.	0,168 max.	4,27 max.	0,168 max.
A15	2,64 min.	0,104 min.	2,64 min.	0,104 min.
A16	0,25±0,13	0,010±0,005	0,25±0,13	0,010±0,005
Pin X	25		34	
Pin Y	26		35	
Pin Z	50		68	

Figure 6 - 50/68-contact shielded high-density cable connector (A cable/B cable)

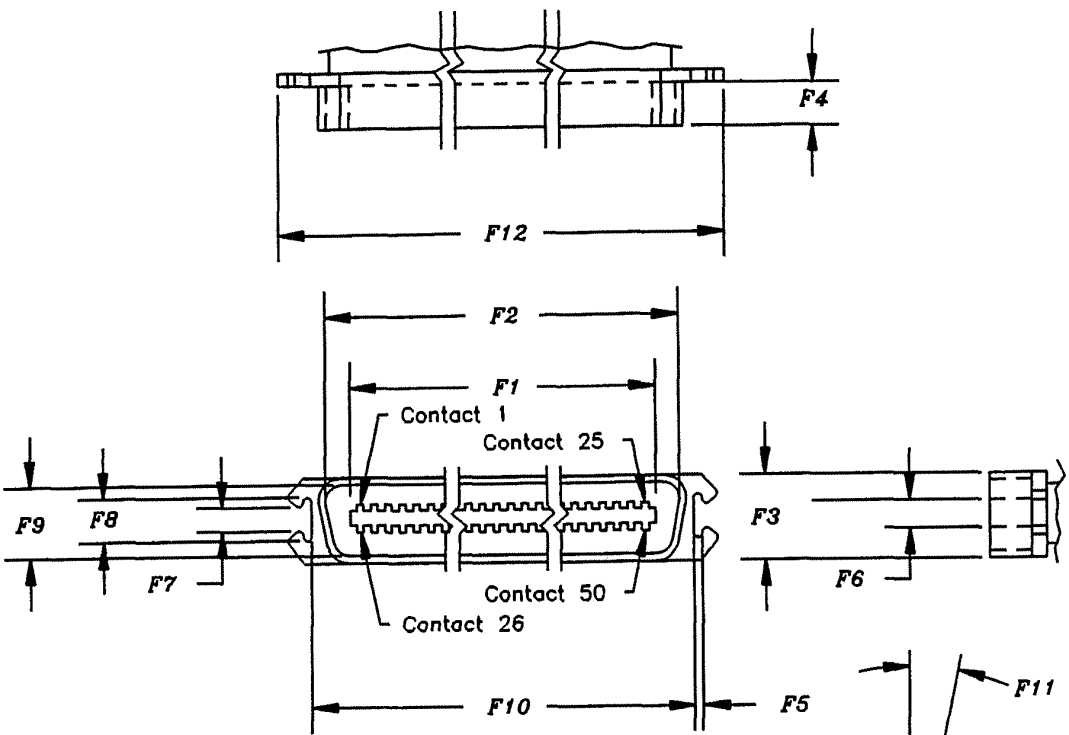


Dimensions	mm	in
E1	56,39 min.	2,220 min.
E2 1)	2,62 min.	0,103 min.
E3	3,99 min.	0,157 min.
E4	5,84 min.	0,230 min.
E5	74,85	2,947
E6	64,29 max.	2,531 max.
E7 2)	83,06	3,270
E8 2)	15,24	0,600
E9	12,04 max.	0,474 max.
E10	9,78 max.	0,385 max.
E11	2,16	0,085
E12	15°±2°	15°±2°

NOTE – Tolerances ± 0,127 mm (0,005 in) non-cumulative, unless specified otherwise.

1) This dimension is selected to accomodate 4–40 or 6–32 threaded screws.
2) These dimensions are shown for reference only.

Figure 7 - 50-contact shielded low-density SCSI device connector



Dimensions	mm	in
F1	56,26 max.	2,215 max.
F2	64,29 min.	2,531 min.
F3 1)	15,24	0,600
F4	7,29 min.	0,287 min.
F5 1)	1,02	0,040
F6	4,09 max.	0,161 max.
F7 1)	5,08	0,200
F8 1)	6,10	0,240
F9	12,04 min.	0,474 min.
F10 1)	68,45	2,695
F11	15° ± 2°	15° ± 2°
F12 1)	76,71	3,020
F13	2,16	0,085

NOTE – Tolerances ± 0,127 mm (0,005 in) non-cumulative, unless specified otherwise.

1) These dimensions are shown for reference only.

Figure 8 - 50-contact shielded low-density cable connector

5.3.3 Connector contact assignments

The connector contact assignments are defined in tables 1 through 5. Table 1 defines which of the other four table to use and which set of contact assignments to use.

Table 1 - Cross-reference to connector contact assignments

Connector type	Driver/ receiver type	Cable	Connector figure	Contact assignment table	Contact set
Non-shielded alternative 1	Single-ended	A	1 & 2	2	2
Non-shielded alternative 1	Single-ended	B	1 & 2	3	
Non-shielded alternative 1	Differential	A	1 & 2	4	2
Non-shielded alternative 1	Differential	B	1 & 2	5	
Non-shielded alternative 2	Single-ended	A	3 & 4	2	1
Non-shielded alternative 2	Differential	A	3 & 4	4	1
Shielded alternative 1	Single-ended	A	5 & 6	2	2
Shielded alternative 1	Single-ended	B	5 & 6	3	
Shielded alternative 1	Differential	A	5 & 6	4	2
Shielded alternative 1	Differential	B	5 & 6	5	
Shielded alternative 2	Single-ended	A	7 & 8	2	2
Shielded alternative 2	Differential	A	7 & 8	4	2

Table 2 - Single-ended contact assignments - A cable

Signal name	Connector contact number		Cable conductor number		Connector contact number		Signal name
	Set 2	Set 1			Set 1	Set 2	
GROUND	1	1	1	2	2	26	-DB(0)
GROUND	2	3	3	4	4	27	-DB(1)
GROUND	3	5	5	6	6	28	-DB(2)
GROUND	4	7	7	8	8	29	-DB(3)
GROUND	5	9	9	10	10	30	-DB(4)
GROUND	6	11	11	12	12	31	-DB(5)
GROUND	7	13	13	14	14	32	-DB(6)
GROUND	8	15	15	16	16	33	-DB(7)
GROUND	9	17	17	18	18	34	-DB(P)
GROUND	10	19	19	20	20	35	GROUND
GROUND	11	21	21	22	22	36	GROUND
RESERVED	12	23	23	24	24	37	RESERVED
OPEN	13	25	25	26	26	38	TERMPWR
RESERVED	14	27	27	28	28	39	RESERVED
GROUND	15	29	29	30	30	40	GROUND
GROUND	16	31	31	32	32	41	-ATN
GROUND	17	33	33	34	34	42	GROUND
GROUND	18	35	35	36	36	43	-BSY
GROUND	19	37	37	38	38	44	-ACK
GROUND	20	39	39	40	40	45	-RST
GROUND	21	41	41	42	42	46	-MSG
GROUND	22	43	43	44	44	47	-SEL
GROUND	23	45	45	46	46	48	-C/D
GROUND	24	47	47	48	48	49	-REQ
GROUND	25	49	49	50	50	50	-I/O

NOTES
1 The minus sign next to a signal indicates active low.
2 The conductor number refers to the conductor position when using 0,050 inch centreline flat ribbon cable with a low-density connector or when using 0,025 inch centreline flat ribbon cable with a high-density connector. Other cable types may be used to implement equivalent contact assignments.
3 Two sets of contact assignments are shown. Refer to table 1 to determine which set of contacts applies to each connector.
4 See 5.4.4 for a definition of the RESERVED lines.

Table 3 - Single-ended contact assignments - B cable

Signal name	Connector contact number	Cable conductor number		Connector contact number	Signal name
GROUND	1	1	2	35	GROUND
GROUND	2	3	4	36	-DB(8)
GROUND	3	5	6	37	-DB(9)
GROUND	4	7	8	38	-DB(10)
GROUND	5	9	10	39	-DB(11)
GROUND	6	11	12	40	-DB(12)
GROUND	7	13	14	41	-DB(13)
GROUND	8	15	16	42	-DB(14)
GROUND	9	17	18	43	-DB(15)
GROUND	10	19	20	44	-DB(P1)
GROUND	11	21	22	45	-ACKB
GROUND	12	23	24	46	GROUND
GROUND	13	25	26	47	-REQB
GROUND	14	27	28	48	-DB(16)
GROUND	15	29	30	49	-DB(17)
GROUND	16	31	32	50	-DB(18)
TERMPWRB	17	33	34	51	TERMPWRB
TERMPWRB	18	35	36	52	TERMPWRB
GROUND	19	37	38	53	-DB(19)
GROUND	20	39	40	54	-DB(20)
GROUND	21	41	42	55	-DB(21)
GROUND	22	43	44	56	-DB(22)
GROUND	23	45	46	57	-DB(23)
GROUND	24	47	48	58	-DB(P2)
GROUND	25	49	50	59	-DB(24)
GROUND	26	51	52	60	-DB(25)
GROUND	27	53	54	61	-DB(26)
GROUND	28	55	56	62	-DB(27)
GROUND	29	57	58	63	-DB(28)
GROUND	30	59	60	64	-DB(29)
GROUND	31	61	62	65	-DB(30)
GROUND	32	63	64	66	-DB(31)
GROUND	33	65	66	67	-DB(P3)
GROUND	34	67	68	68	GROUND

NOTES
1 The minus sign next to a signal indicates active low.
2 The conductor number refers to the conductor position when using 0,025 inch centreline flat ribbon cable. Other cable types may be used to implement contact assignments.

NOTE 10 An alternate 16-bit single-cable solution and an alternate 32-bit solution is being defined and the B cable definition will be removed in a future version of SCSI.

Table 4 - Differential contact assignments - A cable

Signal name	Connector contact number Set 2	Connector contact number Set 1	Cable conductor number		Connector contact number Set 1	Connector contact number Set 2	Signal name
GROUND	1	1	1	2	2	26	GROUND
+DB(0)	2	3	3	4	4	27	-DB(0)
+DB(1)	3	5	5	6	6	28	-DB(1)
+DB(2)	4	7	7	8	8	29	-DB(2)
+DB(3)	5	9	9	10	10	30	-DB(3)
+DB(4)	6	11	11	12	12	31	-DB(4)
+DB(5)	7	13	13	14	14	32	-DB(5)
+DB(6)	8	15	15	16	16	33	-DB(6)
+DB(7)	9	17	17	18	18	34	-DB(7)
+DB(P)	10	19	19	20	20	35	-DB(P)
DIFFSENS	11	21	21	22	22	36	GROUND
RESERVED	12	23	23	24	24	37	RESERVED
TERMPWR	13	25	25	26	26	38	TERMPWR
RESERVED	14	27	27	28	28	39	RESERVED
+ATN	15	29	29	30	30	40	-ATN
GROUND	16	31	31	32	32	41	GROUND
+BSY	17	33	33	34	34	42	-BSY
+ACK	18	35	35	36	36	43	-ACK
+RST	19	37	37	38	38	44	-RST
+ACK	18	35	35	36	36	43	-ACK
+RST	19	37	37	38	38	44	-RST
+MSG	20	39	39	40	40	45	-MSG
+SEL	21	41	41	42	42	46	-SEL
+C/D	22	43	43	44	44	47	-C/D
+REQ	23	45	45	46	46	48	-REQ
+I/O	24	47	47	48	48	49	-I/O
GROUND	25	49	49	50	50	50	GROUND

NOTES
1 The conductor number refers to the conductor position when using 0,050 inch centreline flat ribbon cable with a low-density connector or when using 0,025 inch centreline flat ribbon cable with a high-density connector. Other cable types may be used to implement equivalent contact assignments.
2 Two sets of contact assignments are shown. Refer to table 1 to determine which set of contacts applies to each connector.
3 See 5.4.4 for a definition of the RESERVED lines.

Table 5 - Differential contact assignments - B cable

Signal name	Connector contact number	Cable conductor number		Connector contact number	Signal name
GROUND	1	1	2	35	GROUND
+DB(8)	2	3	4	36	-DB(8)
+DB(9)	3	5	6	37	-DB(9)
+DB(10)	4	7	8	38	-DB(10)
+DB(11)	5	9	10	39	-DB(11)
+DB(12)	6	11	12	40	-DB(12)
+DB(13)	7	13	14	41	-DB(13)
+DB(14)	8	15	16	42	-DB(14)
+DB(15)	9	17	18	43	-DB(15)
+DB(P1)	10	19	20	44	-DB(P1)
+ACKB	11	21	22	45	-ACKB
GROUND	12	23	24	46	DIFFSENS
+REQB	13	25	26	47	-REQB
+DB(16)	14	27	28	48	-DB(16)
+DB(17)	15	29	30	49	-DB(17)
+DB(18)	16	31	32	50	-DB(18)
TERMPWRB	17	33	34	51	TERMPWRB
TERMPWRB	18	35	36	52	TERMPWRB
+DB(19)	19	37	38	53	-DB(19)
+DB(20)	20	39	40	54	-DB(20)
+DB(21)	21	41	42	55	-DB(21)
+DB(22)	22	43	44	56	-DB(22)
+DB(23)	23	45	46	57	-DB(23)
+DB(P2)	24	47	48	58	-DB(P2)
+DB(24)	25	49	50	59	-DB(24)
+DB(25)	26	51	52	60	-DB(25)
+DB(26)	27	53	54	61	-DB(26)
+DB(27)	28	55	56	62	-DB(27)
+DB(28)	29	57	58	63	-DB(28)
+DB(29)	30	59	60	64	-DB(29)
+DB(30)	31	61	62	65	-DB(30)
+DB(31)	32	63	64	66	-DB(31)
+DB(P3)	33	65	66	67	-DB(P3)
GROUND	34	67	68	68	GROUND

NOTE
The conductor number refers to the conductor position when using 0,025 inch centreline flat ribbon cable. Other cable types may be used to implement equivalent contact assignments.

NOTE 11 An alternate 16-bit single-cable solution and an alternate 32-bit solution is being defined and the B cable definition will be removed in a future version of SCSI.

5.4 Electrical description

For the measurements in this subclause, SCSI bus termination is assumed to be external to the SCSI device. See 5.4.4 for the terminating requirements for the RESERVED lines. SCSI devices may have the provision for allowing optional internal termination.

5.4.1 Single-ended alternative

All signals not defined as RESERVED, GROUND, or TERMPWR shall be terminated at both ends of the cable. The implementor may choose one of the following two methods to terminate each end (see figures 9 and 10):

a) The termination of each signal shall consist of 220 Ω (±5%) to the TERMPWR line and 330 Ω (±5%) to ground. Using resistors with ±1% tolerance improves noise margins.
b) The termination of each signal shall meet these requirements:
   1) The terminators shall each supply a characteristic impedance between 100 Ω and 132 Ω.
   2) The terminators shall be powered by the TERMPWR line and may receive additional power from other sources but shall not require such additional power for proper operation (see 5.4.3).
   3) The current available to any signal line driver shall not exceed 48 mA when the driver asserts the line and pulls it to 0,5 V d.c. Only 44,8 mA of this current shall be available from the two terminators.
   4) The voltage on all released signal lines shall be at least 2,5 V d.c. when the TERMPWR line is within specified values (see 5.4.3).
   5) These conditions shall be met with any legal configuration of targets and initiators as long as at least one device is supplying TERMPWR.

The first termination method above is the same as in SCSI-1. The second termination method is recommended for better signal quality.

5.4.1.1 Output characteristics

All signals shall use open-collector or three-state drivers. Each signal driven by an SCSI device shall have the following output characteristics when measured at the SCSI device's connector:

V_{OL} (low-level output voltage) = 0,0 to 0,5 V d.c. at 48 mA sinking (signal assertion)

V_{OH} (high-level output voltage) = 2,5 to 5,25 V d.c. (signal negation)

5.4.1.2 Input characteristics

SCSI devices with power on shall meet the following electrical characteristics on each signal (including both receivers and passive drivers):

V_{IL} (low-level input voltage)	= 0,0 V d.c. to 0,8 V d.c. (signal true)
V_{IH} (high-level input voltage)	= 2,0 V d.c. to 5,25 V d.c. (signal false)
I_{IL} (low-level input current)	= -0,4 mA to 0,0 mA at V_I = 0,5 V d.c.
I_{IH} (high-level input current)	= 0,0 mA to 0,1 mA at V_I = 2,7 V d.c.
Minimum input hysteresis	= 0,2 V d.c.
Maximum input capacitance	= 25 pF (measured at the device connector closest to the stub, if any, within the device)

It is recommended that SCSI devices with power off also meet the above I_{IL} and I_{IH} electrical characteristics on each signal.

To achieve maximum noise immunity and to assure proper operation with complex cable configurations, it is recommended that the nominal switching threshold be approximately 1,4 V.

5.4.2 Differential alternative

All signals consist of two lines denoted +SIGNAL and -SIGNAL. A signal is true when +SIGNAL is more positive tha -SIGNAL, and a signal is false when -SIGNAL is more positive than +SIGNAL. All assigned signals of the A and E cables described in 5.6 shall be terminated at each end of the cable with a terminator network as shown in figure 11 Resistor tolerances in the terminator network shall be $\pm$5 % or less.

The DIFFSENS signal of the connector is used as an active high enable for the differential drivers. If a single-ende device or terminator is inadvertently connected, this signal is grounded, disabling the differential drivers (see figur 12).

The characteristic impedance of differential terminators is 122 Ω.

5.4.2.1 Output characteristics

Each signal driven by an SCSI device shall have the following output characteristics when measured at the SCS device's connector:

V_{OL} (low-level output voltage) = 1,7 V maximum at I_{OL} (low-level output current) = 55 mA.
V_{OH} (high-level output voltage) = 2,7 V minimum at I_{OH} (high-level output current) = -55 mA.
V_{OD} (differential output voltage) = 1,0 V minimum with common-mode voltage ranges from -7 V d.c. to +12 V d.c

V_{OL} and V_{OH} shall be as measured between the output terminal and the SCSI device's logic ground reference.

The output characteristics shall additionally conform to EIA RS-485-1983.

5.4.2.2 Input characteristics

SCSI devices shall meet the following electrical characteristics on each signal (including both receivers and passiv drivers):

I_I (input current on either input) = $\pm$ 2,0 mA maximum.
Maximum input capacitance = 25 pF.

The I_I requirement shall be met with the input voltage varying between -7 V d.c. and +12 V d.c., with power on o off, and with the hysteresis equaling 35 mV, minimum.

The input characteristics shall additionally conform to EIA RS-485-1983.

5.4.3 Terminator power

SCSI initiators shall supply terminator power to the TERMPWR contact(s) and, if it implements the wide SCSI optior to the TERMPWRB contacts. This power shall be supplied through a diode or similar semiconductor that prevent backflow of power to the SCSI device. Targets and SCSI devices that become temporary initiators (e.g. targets whic implement the COPY command or asynchronous event notification) are not required to supply terminator power. An SCSI device may supply terminator power. Interface error rates are lower if the termination voltage is maintained a the extreme ends of the cable.

All terminators independent of location shall be powered from the TERMPWR and TERMPWRB contact(s). The us of keyed connectors is recommended in SCSI devices that provide terminator power to prevent accidental groundin or the incorrect connection of terminator power.

NOTE 12 Regulatory agencies may require limiting maximum (short circuit) current to the terminator power lines. Recommended current limiting is 1,5 A for TERMPWR and 2 A for TERMPWRB. For systems utilizing multiple initiators, the initiators may be configured with option straps or current limiting devices. Maximum available current should not exceed 5 A.

SCSI devices shall sink no more than 1,0 mA from TERMPWR and no more than 1,0 mA from TERMPWRB except to power an optional internal terminator.

Single-ended SCSI devices providing terminator power on cable A shall have the following characteristics:
V_{Term} = 4,25 V d.c. to 5,25 V d.c.
900 mA minimum source drive capability

Differential SCSI devices providing terminator power on cable A shall have the following characteristics:
V_{Term} = 4,0 V d.c. to 5,25 V d.c.
600 mA minimum source drive capability

Single-ended SCSI devices providing terminator power on cable B shall have the following characteristics:
V_{Term} = 4,5 V d.c. to 5,25 V d.c.
1500 mA minimum source drive capability

Differential SCSI devices providing terminator power on cable B shall have the following characteristics:
V_{Term} = 4,0 V d.c. to 5,25 V d.c.
1000 mA minimum source drive capability

NOTE 13 It is recommended that the terminator power lines be decoupled at each terminator with at least a 2,2 μF high-frequency capacitor to improve signal quality.

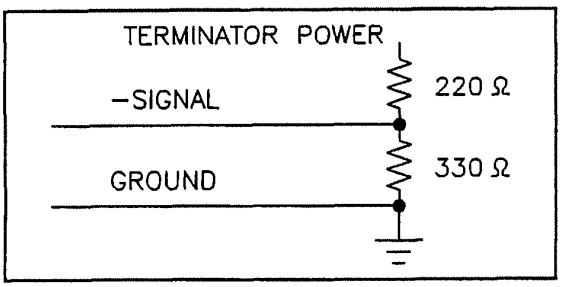
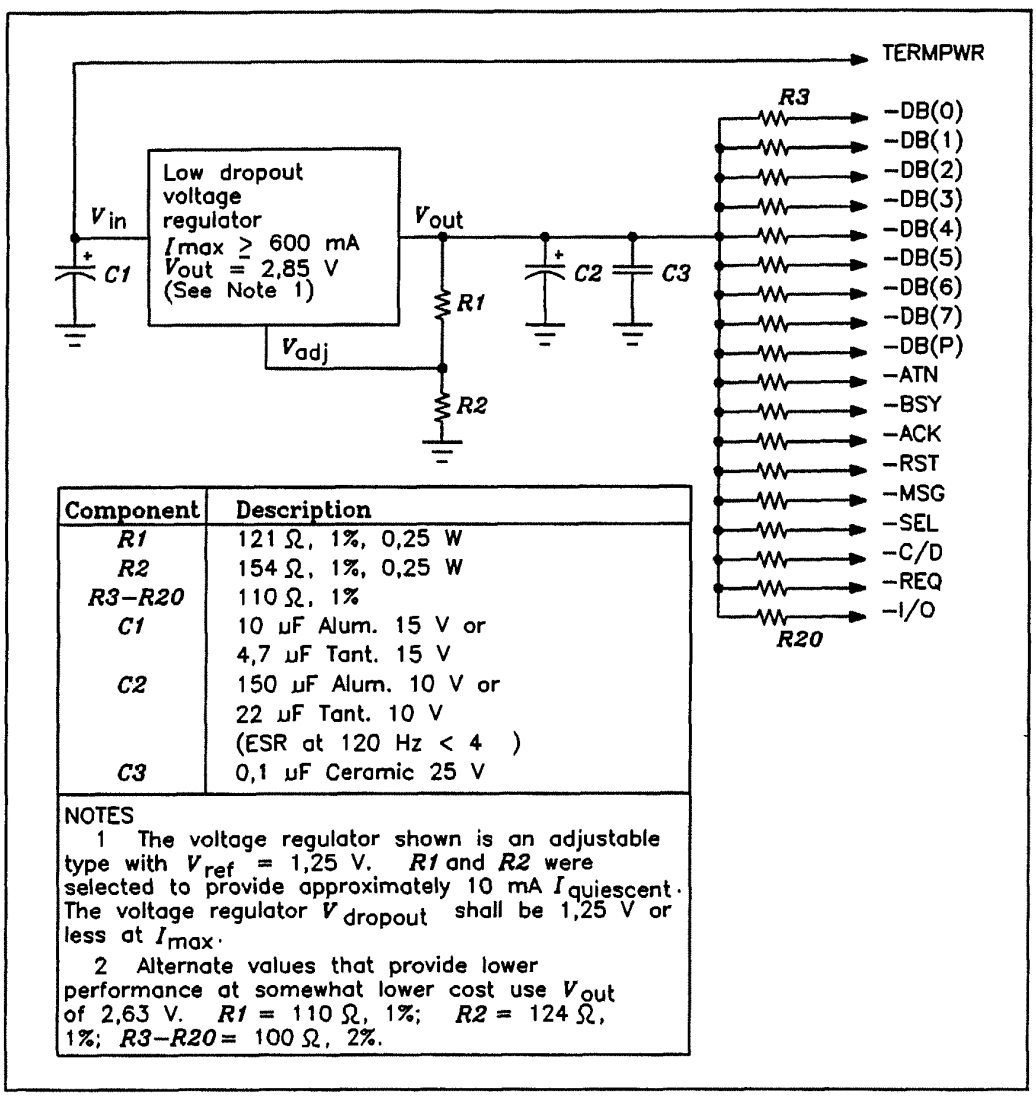


Figure 9 - Alternative 1 termination for single-ended devices



Component	Description
R1	121 Ω, 1%, 0,25 W
R2	154 Ω, 1%, 0,25 W
R3–R20	110 Ω, 1%
C1	10 µF Alum. 15 V or 4,7 µF Tant. 15 V
C2	150 µF Alum. 10 V or 22 µF Tant. 10 V (ESR at 120 Hz < 4)
C3	0,1 µF Ceramic 25 V

NOTES

1 The voltage regulator shown is an adjustable type with V_{ref} = 1,25 V. *R1* and *R2* were selected to provide approximately 10 mA $I_{quiescent}$. The voltage regulator $V_{dropout}$ shall be 1,25 V or less at I_{max}.

2 Alternate values that provide lower performance at somewhat lower cost use V_{out} of 2,63 V. *R1* = 110 Ω, 1%; *R2* = 124 Ω, 1%; *R3–R20* = 100 Ω, 2%.

Figure 10 - Alternative 2 termination for single-ended devices

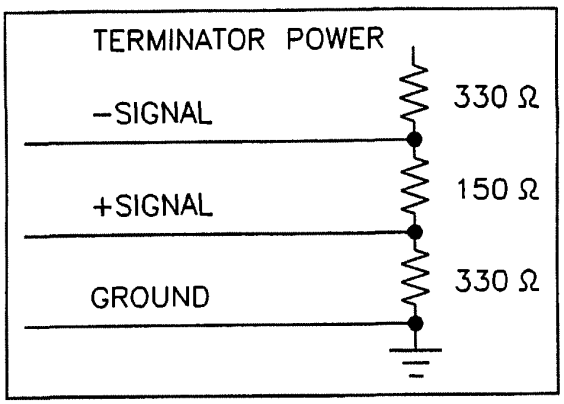


Figure 11 - Termination for differential devices

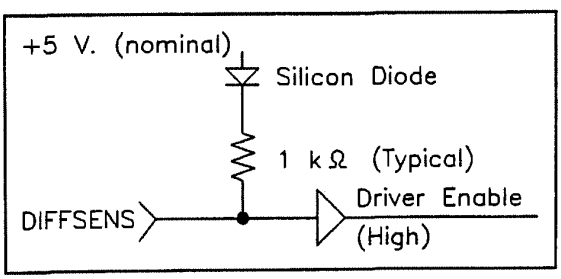


Figure 12 - Differential driver protection circuit

5.4.4 RESERVED lines

The lines labelled RESERVED in the A cable contact assignment tables (table 2 and table 4) shall be connected to ground in the bus terminator assemblies or in the end devices on the SCSI cable. The RESERVED lines should be open in the other SCSI devices, but may be connected to ground.

5.5 SCSI bus

Communication on the SCSI bus is allowed between only two SCSI devices at any given time. There is a maximum of eight SCSI devices. Each SCSI device has an SCSI ID bit assigned as shown in figure 13. Three sample system configurations are shown in figure 14. There can be any combination of initiators and targets provided there is at least one of each.

When two SCSI devices communicate on the SCSI bus, one acts as an initiator and the other acts as a target. The initiator originates an operation and the target performs the operation. An SCSI device usually has a fixed role as an initiator or target, but some devices may be able to assume either role.

An initiator may address up to eight peripheral devices that are connected to a target. The target may be physically housed within the peripheral device in which case the peripheral device is referred to as an embedded SCSI device.

Certain SCSI bus functions are assigned to the initiator and certain SCSI bus functions are assigned to the target. The initiator may arbitrate for the SCSI bus and select a particular target. The target may request the transfer of COMMAND, DATA, STATUS, or other information on the DATA BUS, and in some cases it may arbitrate for the SCSI bus and reselect an initiator for the purpose of continuing an operation.

Information transfers on the DATA BUS are asynchronous and follow a defined REQ/ACK handshake protocol. One byte of information may be transferred with each handshake on the A cable and, if the wide data transfer option is implemented, one or three bytes of information may be transferred with each handshake on the B cable. An option is defined for synchronous data transfer.

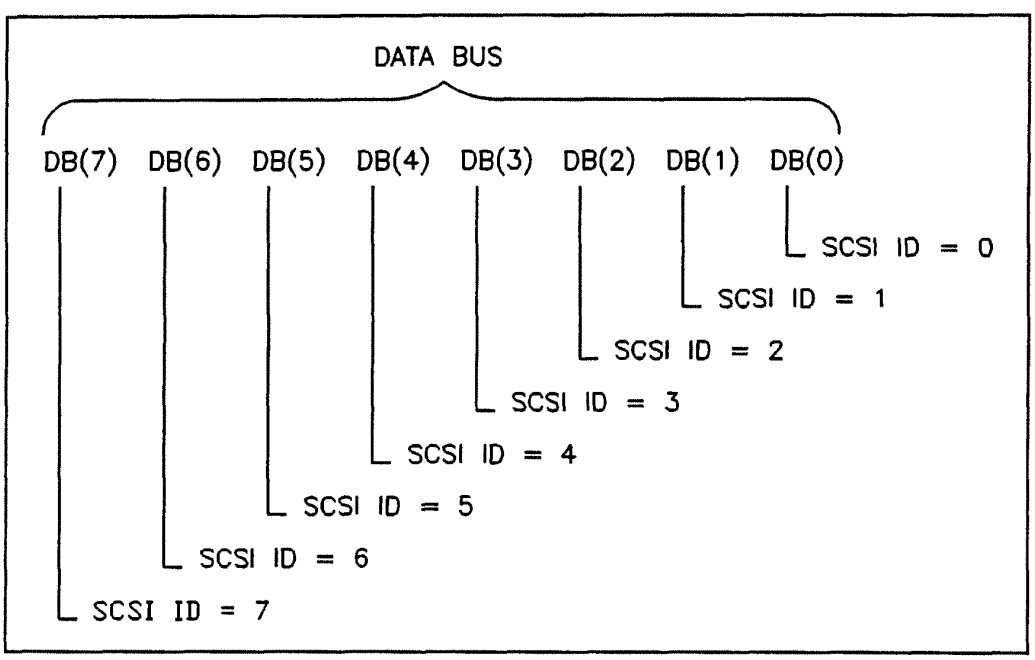


Figure 13 - SCSI ID bits

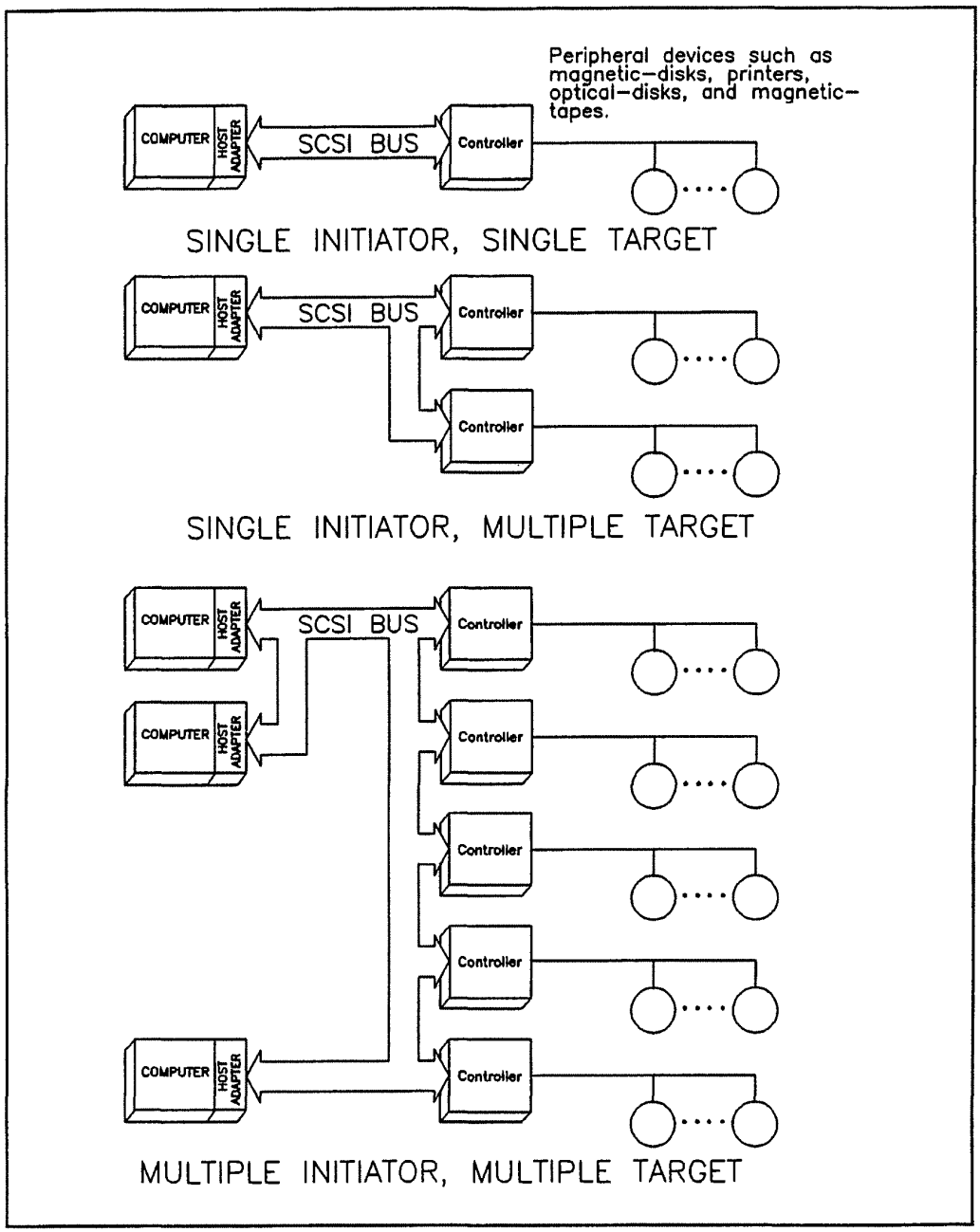


Figure 14 - Sample SCSI configurations

5.6 SCSI bus signals

There are a total of 18 signals on the A cable and 29 signals on the B cable. A total of 11 signals are used for contro and 36 are used for data (messages, commands, status and data), including parity. These signals are described as follows:

a) **BSY (BUSY).** An OR-tied signal that indicates that the bus is being used.

b) **SEL (SELECT).** An OR-tied signal used by an initiator to select a target or by a target to reselect an initiator

NOTE 14 The SEL signal was not defined as OR-tied in SCSI-1. It has been defined as OR-tied in SCSI-2 in anticipation of needing another OR-tied signal for future standardization. This does not cause an operational problem in mixing SCSI-1 and SCSI-2 devices.

c) **C/D (CONTROL/DATA).** A signal driven by a target that indicates whether CONTROL or DATA information is on the DATA BUS. True indicates CONTROL.

d) **I/O (INPUT/OUTPUT).** A signal driven by a target that controls the direction of data movement on the DATA BUS with respect to an initiator. True indicates input to the initiator. This signal is also used to distinguish between SELECTION and RESELECTION phases.

e) **MSG (MESSAGE).** A signal driven by a target during the MESSAGE phase.

f) **REQ (REQUEST).** A signal driven by a target on the A cable to indicate a request for an ACK information transfer handshake.

g) **REQB (REQUEST).** A signal driven by a target on the B cable to indicate a request for an ACKB information transfer handshake.

h) **ACK (ACKNOWLEDGE).** A signal driven by an initiator on the A cable to indicate an acknowledgment for a REQ information transfer handshake.

i) **ACKB (ACKNOWLEDGE).** A signal driven by an initiator on the B cable to indicate an acknowledgment for a REQB information transfer handshake.

j) **ATN (ATTENTION).** A signal driven by an initiator to indicate the ATTENTION condition.

k) **RST (RESET).** An OR-tied signal that indicates the RESET condition.

l) **DB(7-0,P) (DATA BUS).** Eight data-bit signals, plus a parity-bit signal that form a DATA BUS. DB(7) is the most significant bit and has the highest priority during the ARBITRATION phase. Bit number, significance, and priority decrease downward to DB(0). A data bit is defined as one when the signal value is true and is defined as zero when the signal value is false. Data parity DB(P) shall be odd. Parity is undefined during the ARBITRATION phase.

m) **DB(31-8,P1,P2,P3) (DATA BUS).** Twenty-four data-bit signals, plus three parity-bit signals that form an extension to the DATA BUS. DB(P1,P2,P3) are parity bits for DB(15-8), DB(23-16), and DB(31-24) respectively. A data bit is defined as one when the signal value is true and is defined as zero when the signal value is false. Data parity DB(Px) shall be odd.

5.6.1 Signal values

Signals may assume true or false values. There are two methods of driving these signals. In both cases, the signal shall be actively driven true, or asserted. In the case of OR-tied drivers, the driver does not drive the signal to the false state, rather the bias circuitry of the bus terminators pulls the signal false whenever it is released by the drivers at every SCSI device. If any driver is asserted, then the signal is true. In the case of non-OR-tied drivers, the signal may be actively driven false. In this standard, wherever the term negated is used, it means that the signal may be actively driven false, or may be simply released (in which case the bias circuitry pulls it false), at the option of the implementor. The advantage to actively driving signals false during information transfer is that the transition from true to false occurs more quickly and the noise margin is much higher than if the signal is simply released. This facilitates reliable data transfer at high rates, especially at the longer cable lengths used with differential drivers.

5.6.2 OR-tied signals

The BSY, SEL, and RST signals shall be OR-tied only. In the ordinary operation of the bus, the BSY and RST signals may be simultaneously driven true by several drivers. No signals other than BSY, RST, and DB(P) are simultaneously driven by two or more drivers, and any signal other than BSY, SEL, and RST may employ OR-tied or non-OR-tied drivers. DB(P) shall not be driven false during the ARBITRATION phase but may be driven false in other phases. There is no operational problem in mixing OR-tied and non-OR-tied drivers on signals other than BSY and RST.

5.6.3 Signal sources

Table 6 indicates which type of SCSI device is allowed to source each signal. No attempt is made to show if the source is driving asserted, driving negated, or is passive. All SCSI device drivers that are not active sources be in the passive state. The RST signal may be asserted by any SCSI device at any time.

Table 6 - Signal sources

Bus phase	A cable signals					B cable signals		
	BSY	SEL	C/D, I/O, MSG, REQ	ACK, ATN	DB(7-0) DB(P)	REQB	ACKB	DB(31-8) DB(P1) DB(P2) DB(P3)
BUS FREE	None	None	None	None	None	None	None	None
ARBITRATION	All	Win	None	None	S ID	None	None	None
SELECTION	I&T	Init	None	Init	Init	None	None	None
RESELECTION	I&T	Targ	Targ	Init	Targ	None	None	None
COMMAND	Targ	None	Targ	Init	Init	None	None	None
DATA IN	Targ	None	Targ	Init	Targ	Targ	Init	Targ
DATA OUT	Targ	None	Targ	Init	Init	Targ	Init	Init
STATUS	Targ	None	Targ	Init	Targ	None	None	None
MESSAGE IN	Targ	None	Targ	Init	Targ	None	None	None
MESSAGE OUT	Targ	None	Targ	Init	Init	None	None	None

All: The signal shall be driven by all SCSI devices that are actively arbitrating.

S ID: A unique data bit (the SCSI ID) shall be driven by each SCSI device that is actively arbitrating; the other seven data bits shall be released (i.e., not driven) by this SCSI device. The parity bit (DB(P)) may be released or driven to the true state, but shall never be driven to the false state during this phase.

I&T: The signal shall be driven by the initiator, target, or both, as specified in the SELECTION phase and RESELECTION phase.

Init: If driven, this signal shall be driven only by the active initiator.

None: The signal shall be released; that is, not be driven by any SCSI device. The bias circuitry of the bus terminators pulls the signal to the false state.

Win: The signal shall be driven by the one SCSI device that wins arbitration.

Targ: If the signal is driven, it shall be driven only by the active target.

5.7 SCSI bus timing

Unless otherwise indicated, the delay-time measurements for each SCSI device, shown in table 7, shall be calculated from signal conditions existing at that SCSI device's own SCSI bus connection. Thus, these measurements (except cable skew delay) can be made without considering delays in the cable. The timing characteristics of each signal are described in the following paragraphs.

Table 7 - SCSI bus timing values

Timing description	Timing value
Arbitration delay	2,4 μs
Assertion period	90 ns
Bus clear delay	800 ns
Bus free delay	800 ns
Bus set delay	1,8 μs
Bus settle delay	400 ns
Cable skew delay	10 ns
Data release delay	400 ns
Deskew delay	45 ns
Disconnection delay	200 μs
Hold time	45 ns
Negation period	90 ns
Power-on to selection time	10 s recommended
Reset to selection time	250 ms recommended
Reset hold time	25 μs
Selection abort time	200 μs
Selection time-out delay	250 ms recommended
Transfer period	set during an SDTR message
Fast assertion period	30 ns
Fast cable skew delay	5 ns
Fast deskew delay	20 ns
Fast hold time	10 ns
Fast negation period	30 ns

5.7.1 Arbitration delay

The minimum time an SCSI device shall wait from asserting BSY for arbitration until the DATA BUS can be examined to see if arbitration has been won. There is no maximum time.

5.7.2 Assertion period

The minimum time that a target shall assert REQ (or REQB) while using synchronous data transfers. Also, the minimum time that an initiator shall assert ACK (or ACKB) while using synchronous data transfers. REQB and ACKB timings only apply to optional wide data transfers.

5.7.3 Bus clear delay

The maximum time for an SCSI device to stop driving all bus signals after:

a) The BUS FREE phase is detected (see 6.1.1)
b) SEL is received from another SCSI device during the ARBITRATION phase
c) The transition of RST to true.

For the first condition above, the maximum time for an SCSI device to clear the bus is 1200 nanoseconds from BSY and SEL first becoming both false. If an SCSI device requires more than a bus settle delay to detect BUS FREE phase, it shall clear the bus within a bus clear delay minus the excess time.

5.7.4 Bus free delay

The minimum time that an SCSI device shall wait from its detection of the BUS FREE phase (see 6.1.1) until its assertion of BSY when going to the ARBITRATION phase.

5.7.5 Bus set delay

The maximum time for an SCSI device to assert BSY and its SCSI ID bit on the DATA BUS after it detects BUS FREE phase (see 6.1.1) for the purpose of entering the ARBITRATION phase.

5.7.6 Bus settle delay

The minimum time to wait for the bus to settle after changing certain control signals as called out in the protocol definitions.

5.7.7 Cable skew delay

The maximum difference in propagation time allowed between any two SCSI bus signals measured between any two SCSI devices.

5.7.8 Data release delay

The maximum time for an initiator to release the DATA BUS signals following the transition of the I/O signal from false to true.

5.7.9 Deskew delay

The minimum time required for deskew of certain signals.

5.7.10 Disconnection delay

The minimum time that a target shall wait after releasing BSY before participating in an ARBITRATION phase when honouring a DISCONNECT message from the initiator.

5.7.11 Hold time

The minimum time added between the assertion of REQ (or REQB) or ACK (or ACKB) and the changing of the data lines to provide hold time in the initiator or target while using synchronous data transfers. REQB and ACKB timings only apply to optional wide data transfers.

5.7.12 Negation period

The minimum time that a target shall negate REQ (or REQB) while using synchronous data transfers. Also, the minimum time that an initiator shall negate ACK (or ACKB) while using synchronous data transfers. REQB and ACKB timings only apply to optional wide data transfers.

5.7.13 Power-on to selection time

The recommended maximum time from power application until an SCSI target is able to respond with appropriate status and sense data to the TEST UNIT READY, INQUIRY, and REQUEST SENSE commands.

5.7.14 Reset to selection time

The recommended maximum time after a hard RESET condition until an SCSI target is able to respond with appropriate status and sense data to the TEST UNIT READY, INQUIRY, and REQUEST SENSE commands.

5.7.15 Reset hold time

The minimum time for which RST is asserted. There is no maximum time.

5.7.16 Selection abort time

The maximum time that a target (or initiator) shall take from its most recent detection of being selected (or reselected) until asserting a BSY response. This time-out is required to ensure that a target (or initiator) does not assert BSY after a SELECTION (or RESELECTION) phase has been aborted. This is not the selection time-out period; see 6.1.3.1 and 6.1.4.2 for a complete description.

5.7.17 Selection time-out delay

The minimum time that an SCSI device should wait for a BSY response during the SELECTION or RESELECTION phase before starting the time-out procedure.

Note 15 The selection time-out delay is only a recommended time period.

5.7.18 Transfer period

The minimum time allowed between the leading edges of successive REQ pulses or of successive ACK pulses while using synchronous data transfers. (See 6.1.5.2 and 6.6.21.)

5.8 Fast synchronous transfer option

When devices negotiate a synchronous data transfer period of less than 200 ns they are said to be using fast synchronous data transfers. Devices that negotiate a synchronous data transfer period greater than or equal to 200 ns use timing parameters specified in 5.7. When a fast synchronous data transfer period is negotiated, those specific times redefined in this section are used; those not redefined remain the same. The minimum synchronous data transfer period is 100 ns.

5.8.1 Fast assertion period

The minimum time that a target shall assert REQ (or REQB) while using fast synchronous data transfers. It is also the minimum time that an initiator shall assert ACK (or ACKB) while using fast synchronous data transfers. REQB and ACKB timings only apply to optional wide data transfers.

5.8.2 Fast cable skew delay

The maximum difference in propagation time allowed between any two SCSI bus signals measured between any two SCSI devices while using fast synchronous data transfers.

5.8.3 Fast deskew delay

The minimum time required for deskew of certain signals while using fast synchronous data transfers.

5.8.4 Fast hold time

The minimum time added between the assertion of REQ (or REQB) or ACK (or ACKB) and the changing of the data lines to provide hold time in the initiator or target while using fast synchronous data transfers. REQB and ACKB timings only apply to optional wide data transfers.

5.8.5 Fast negation period

The minimum time that a target shall negate REQ (or REQB) while using fast synchronous data transfers. Also, the minimum time that an initiator shall negate ACK (or ACKB) while using fast synchronous data transfers. REQB and ACKB timings only apply to optional wide data transfers.

6 Logical characteristics

6.1 SCSI bus phases

The SCSI architecture includes eight distinct phases:

a) BUS FREE phase
b) ARBITRATION phase
c) SELECTION phase
d) RESELECTION phase
e) COMMAND phase ┐
f) DATA phase ‖ These phases are collectively termed
g) STATUS phase ‖ the information transfer phases.
h) MESSAGE phase ┘

The SCSI bus can never be in more than one phase at any given time. In the following descriptions, signals that are not mentioned shall not be asserted.

6.1.1 BUS FREE phase

The BUS FREE phase indicates that there is no current I/O process and that the SCSI bus is available for a connection.

SCSI devices shall detect the BUS FREE phase after the SEL and BSY signals are both false for at least a bus settle delay.

SCSI devices shall release all SCSI bus signals within a bus clear delay after the BSY and SEL signals become continuously false for a bus settle delay. If an SCSI device requires more than a bus settle delay to detect the BUS FREE phase then it shall release all SCSI bus signals within a bus clear delay minus the excess time to detect the BUS FREE phase. The total time to clear the SCSI bus shall not exceed a bus settle delay plus a bus clear delay.

During normal operation the BUS FREE phase is entered when a target releases the BSY signal. However, the BUS FREE phase may be entered following the release of the SEL signal after a SELECTION or RESELECTION phase time-out.

Initiators normally do not expect BUS FREE phase to begin because of the target's release of the BSY signal except after one of the following occurrences:

a) after a reset condition is detected;
b) after an ABORT message is successfully received by a target;
c) after a BUS DEVICE RESET message is successfully received by a target;
d) after a DISCONNECT message is successfully transmitted from a target (see 6.6.6);
e) after a COMMAND COMPLETE message is successfully transmitted from a target (see 6.6.5);
f) after a RELEASE RECOVERY message is successfully received by a target;
g) after an ABORT TAG message is successfully received by a target;
h) after a CLEAR QUEUE message is successfully received by a target.

If an initiator detects the release of the BSY signal by the target at any other time, the target is indicating an error condition to the initiator. The target may perform this transition to the BUS FREE phase independent of the state of the ATN signal. The initiator shall manage this condition as an unsuccessful I/O process termination. The target terminates the I/O process by clearing all pending data and status information for the affected nexus. The target may optionally prepare sense data that may be retrieved by a REQUEST SENSE command.

6.1.2 ARBITRATION phase

The ARBITRATION phase allows one SCSI device to gain control of the SCSI bus so that it can initiate or resume ar I/O process.

The procedure for an SCSI device to obtain control of the SCSI bus is as follows:

a) The SCSI device shall first wait for the BUS FREE phase to occur. The BUS FREE phase is detected whenever both the BSY and SEL signals are simultaneously and continuously false for a minimum of a bus settle delay

NOTE 16 This bus settle delay is necessary because a transmission line phenomenon known as a wired-OR glitch may cause the BSY signal to briefly appear false, even though it is being driven true.

b) The SCSI device shall wait a minimum of a bus free delay after detection of the BUS FREE phase (i.e. after the BSY and SEL signals are both false for a bus settle delay) before driving any signal.

c) Following the bus free delay in step (b), the SCSI device may arbitrate for the SCSI bus by asserting both the BSY signal and its own SCSI ID, however the SCSI device shall not arbitrate (i.e. assert the BSY signal and its SCSI ID) if more than a bus set delay has passed since the BUS FREE phase was last observed.

NOTE 17 There is no maximum delay before asserting the BSY signal and the SCSI ID following the bus free delay in step (b) as long as the bus remains in the BUS FREE phase. However, SCSI devices that delay longer than a bus settle delay plus a bus set delay from the time when the BSY and SEL signals first become false may fail to participate in arbitration when competing with faster SCSI devices.

d) After waiting at least an arbitration delay (measured from its assertion of the BSY signal) the SCSI device shall examine the DATA BUS. If a higher priority SCSI ID bit is true on the DATA BUS (DB(7) is the highest), then the SCSI device has lost the arbitration and the SCSI device may release its signals and return to step (a). If no higher priority SCSI ID bit is true on the DATA BUS, then the SCSI device has won the arbitration and it shall assert the SEL signal. Any SCSI device other than the winner has lost the arbitration and shall release the BSY signal and its SCSI ID bit within a bus clear delay after the SEL signal becomes true. An SCSI device that loses arbitration may return to step (a).

NOTES
18 Step d) above requires that any device complete the arbitration phase to the point of SEL being asserted if it begins the arbitration phase as stated in step c). This precludes the possibility of the bus being hung.
19 It is recommended that new implementations wait for the SEL signal to become true before releasing the BSY signal and SCSI ID bit when arbitration is lost.

e) The SCSI device that wins arbitration shall wait at least a bus clear delay plus a bus settle delay after asserting the SEL signal before changing any signals.

NOTE 20 The SCSI ID bit is a single bit on the DATA BUS that corresponds to the SCSI device's unique SCSI address. All other DATA BUS bits shall be released by the SCSI device. Parity is not valid during the ARBITRATION phase. During the ARBITRATION phase, DB(P) may be released or asserted, but shall not be actively driven false.

6.1.3 SELECTION phase

The SELECTION phase allows an initiator to select a target for the purpose of initiating some target function (e.g., READ or WRITE command). During the SELECTION phase the I/O signal is negated so that this phase can be distinguished from the RESELECTION phase.

The SCSI device that won the arbitration has both the BSY and SEL signals asserted and has delayed at least a bus clear delay plus a bus settle delay before ending the ARBITRATION phase. The SCSI device that won the arbitration becomes an initiator by not asserting the I/O signal.

The initiator shall set the DATA BUS to a value that is the OR of its SCSI ID bit and the target's SCSI ID bit and it shall assert the ATN signal (indicating that a MESSAGE OUT phase is to follow the SELECTION phase). The initiator shall

then wait at least two deskew delays and release the BSY signal. The initiator shall then wait at least a bus settle delay before looking for a response from the target.

The target shall determine that it is selected when the SEL signal and its SCSI ID bit are true and the BSY and I/O signals are false for at least a bus settle delay. The selected target may examine the DATA BUS in order to determine the SCSI ID of the selecting initiator. The selected target shall then assert the BSY signal within a selection abort time of its most recent detection of being selected; this is required for correct operation of the selection time-out procedure.

The target shall not respond to a selection if bad parity is detected. Also, if more than two SCSI ID bits are on the DATA BUS, the target shall not respond to selection.

NOTE 21 Although an SCSI-2 initiator may not use the single initiator option or the selection without asserting ATN option of SCSI-1, an SCSI-2 target may elect to support these options for compatibility with SCSI-1 initiators. When doing so the SCSI-2 target responds as described in the SCSI-1 standard.

No less than two deskew delays after the initiator detects the BSY signal is true, it shall release the SEL signal and may change the DATA BUS. The target shall wait until the SEL signal is false before asserting the REQ signal to enter an information transfer phase.

6.1.3.1 SELECTION time-out procedure

Two optional selection time-out procedures are specified for clearing the SCSI bus if the initiator waits a minimum of a selection time-out delay and there has been no BSY signal response from the target:

a) Optionally, the initiator shall assert the RST signal (see 6.2.2);
b) Optionally, the initiator shall continue asserting the SEL and ATN signals and shall release the DATA BUS. If the initiator has not detected the BSY signal to be true after at least a selection abort time plus two deskew delays, the initiator shall release the SEL and ATN signals allowing the SCSI bus to go to the BUS FREE phase. SCSI devices shall ensure that when responding to selection that the selection was still valid within a selection abort time of their assertion of the BSY signal. Failure to comply with this requirement could result in an improper selection (two targets connected to the same initiator, wrong target connected to an initiator, or a target connected to no initiator).

6.1.4 RESELECTION phase

RESELECTION is an optional phase that allows a target to reconnect to an initiator for the purpose of continuing some operation that was previously started by the initiator but was suspended by the target, (i.e. the target disconnected by allowing a BUS FREE phase to occur before the operation was complete).

6.1.4.1 RESELECTION

Upon completing the ARBITRATION phase, the winning SCSI device has both the BSY and SEL signals asserted and has delayed at least a bus clear delay plus a bus settle delay. The winning SCSI device becomes a target by asserting the I/O signal. The winning SCSI device shall also set the DATA BUS to a value that is the logical OR of its SCSI ID bit and the initiator's SCSI ID bit. The target shall wait at least two deskew delays and release the BSY signal. The target shall then wait at least a bus settle delay before looking for a response from the initiator.

The initiator shall determine that it is reselected when the SEL and I/O signals and its SCSI ID bit are true and the BSY signal is false for at least a bus settle delay. The reselected initiator may examine the DATA BUS in order to determine the SCSI ID of the reselecting target. The reselected initiator shall then assert the BSY signal within a selection abort time of its most recent detection of being reselected; this is required for correct operation of the time-out procedure. The initiator shall not respond to a RESELECTION phase if bad parity is detected. Also, the initiator shall not respond to a RESELECTION phase if other than two SCSI ID bits are on the DATA BUS.

After the target detects the BSY signal is true, it shall also assert the BSY signal and wait at least two deskew delays and then release the SEL signal. The target may then change the I/O signal and the DATA BUS. After the reselected

initiator detects the SEL signal is false, it shall release the BSY signal. The target shall continue asserting the BS signal until it relinquishes the SCSI bus.

NOTE 22 When the target is asserting the BSY signal, a transmission line phenomenon known as a wired-OR glitch may cause the BSY signal to appear false for up to a round-trip propagation delay following the release of the BSY signal by the initiator. This is the reason why the BUS FREE phase is recognized only after both the BSY and SEL signals are continuously false for a minimum of a bus settle delay. Cables longer than 25 m should not be used even if the chosen driver, receiver, and cable provide adequate noise margins, because they increase the duration of the glitch and could cause SCSI devices to inadvertently detect the BUS FREE phase.

6.1.4.2 RESELECTION time-out procedure

Two optional RESELECTION time-out procedures are specified for clearing the SCSI bus during a RESELECTIO phase if the target waits a minimum of a selection time-out delay and there has been no BSY signal response fror the initiator:

a) Optionally, the target shall assert the RST signal (see 6.2.2);
b) Optionally, the target shall continue asserting the SEL and I/O signals and shall release all DATA BUS signal If the target has not detected the BSY signal to be true after at least a selection abort time plus two deske delays, the target shall release the SEL and I/O signals allowing the SCSI bus to go to the BUS FREE phas SCSI devices that respond to the RESELECTION phase shall ensure that the reselection was still valid withi a selection abort time of their assertion of the BSY signal. Failure to comply with this requirement could resu in an improper reselection (two initiators connected to the same target or the wrong initiator connected to target).

6.1.5 Information transfer phases

NOTE 23 The COMMAND, DATA, STATUS, and MESSAGE phases are all grouped together as the information transfer phases because they are all used to transfer data or control information via the DATA BUS. The actual content of the information is beyond the scope of this section.

The C/D, I/O, and MSG signals are used to distinguish between the different information transfer phases (see tabl 8). The target drives these three signals and therefore controls all changes from one phase to another. The initiato can request a MESSAGE OUT phase by asserting the ATN signal, while the target can cause the BUS FREE phas by releasing the MSG, C/D, I/O, and BSY signals.

The information transfer phases use one or more REQ/ACK handshakes to control the information transfer. Eac REQ/ACK handshake allows the transfer of one byte of information. During the information transfer phases the BS signal shall remain true and the SEL signal shall remain false. Additionally, during the information transfer phases the target shall continuously envelope the REQ/ACK handshake(s) with the C/D, I/O, and MSG signals in such a manner that these control signals are valid for a bus settle delay before the assertion of the REQ signal of the firs handshake and remain valid until after the negation of the ACK signal at the end of the handshake of the last transfe of the phase.

NOTES
24 After the negation of the ACK signal of the last transfer of the phase, the target may prepare for a new phase by asserting or negating the C/D, I/O, and MSG signals. These signals may be changed together or individually. They may be changed in any order and may be changed more than once. It is desirable that each line change only once. A new phase does not begin until the REQ signal is asserted for the first byte of the new phase.
25 A phase is defined as ending when the C/D, I/O, or MSG signals change after the negation of the ACK signal. The time between the end of a phase and the assertion of the REQ signal beginning a new phase is undefined. An initiator is allowed to anticipate a new phase based on the previous phase, the expected new phase, and early information provided by changes in the C/D, I/O, and MSG signals. However, the anticipated phase is not valid until the REQ signal is asserted at the beginning of the next phase.

Table 8 - Information transfer phases

Signal			Phase name	Direction of transfer	Comment
MSG	C/D	I/O			
0	0	0	DATA OUT	Initiator to target \	Data
0	0	1	DATA IN	Initiator from target /	phase
0	1	0	COMMAND	Initiator to target	
0	1	1	STATUS	Initiator from target	
1	0	0	*		
1	0	1	*		
1	1	0	MESSAGE OUT	Initiator to target \	Message
1	1	1	MESSAGE IN	Initiator from target /	phase
Key: 0 = False, 1 = True, * = Reserved for future standardization					

6.1.5.1 Asynchronous information transfer

The target shall control the direction of information transfer by means of the I/O signal. When the I/O signal is true, information shall be transferred from the target to the initiator. When the I/O signal is false, information shall be transferred from the initiator to the target.

If the I/O signal is true (transfer to the initiator), the target shall first drive the DB(7-0,P) signals to their desired values, delay at least one deskew delay plus a cable skew delay, then assert the REQ signal. The DB(7-0,P) signals shall remain valid until the ACK signal is true at the target. The initiator shall read the DB(7-0,P) signals after the REQ signal is true, then indicate its acceptance of the data by asserting the ACK signal. When the ACK signal becomes true at the target, the target may change or release the DB(7-0,P) signals and shall negate the REQ signal. After the REQ signal is false the initiator shall then negate the ACK signal. After the ACK signal is false the target may continue the transfer by driving the DB(7-0,P) signals and asserting the REQ signal, as described above.

If the I/O signal is false (transfer to the target) the target shall request information by asserting the REQ signal. The initiator shall drive the DB(7-0,P) signals to their desired values, delay at least one deskew delay plus a cable skew delay and assert the ACK signal. The initiator shall continue to drive the DB(7-0,P) signals until the REQ signal is false. When the ACK signal becomes true at the target, the target shall read the DB(7-0,P), signals then negate the REQ signal. When the REQ signal becomes false at the initiator, the initiator may change or release the DB(7-0,P) signals and shall negate the ACK signal. After the ACK signal is false the target may continue the transfer by asserting the REQ signal, as described above.

6.1.5.2 Synchronous data transfer

Synchronous data transfer is optional and is only used in data phases. It shall be used in a data phase if a synchronous data transfer agreement has been established (see 6.6.21). The agreement specifies the REQ/ACK offset and the minimum transfer period.

The REQ/ACK offset specifies the maximum number of REQ pulses that can be sent by the target in advance of the number of ACK pulses received from the initiator, establishing a pacing mechanism. If the number of REQ pulses exceeds the number of ACK pulses by the REQ/ACK offset, the target shall not assert the REQ signal until after the leading edge of the next ACK pulse is received. For successful completion of the data phase the number of ACK and REQ pulses shall be equal.

The target shall assert the REQ signal for a minimum of an assertion period. The target shall then wait at least the greater of a transfer period from the last transition of the REQ signal to true or a minimum of a negation period from the last transition of the REQ signal to false before again asserting the REQ signal.

The initiator shall send one pulse on the ACK signal for each REQ pulse received. The ACK signal may be asserted as soon as the leading edge of the corresponding REQ pulse has been received. The initiator shall assert the ACK signal for a minimum of an assertion period. The initiator shall wait at least the greater of a transfer period from the

last transition of the ACK signal to true or for a minimum of a negation period from the last transition of the ACK signa to false before asserting the ACK signal.

If the I/O signal is true (transfer to the initiator), the target shall first drive the DB(7-0,P) signals to their desired values wait at least one deskew delay plus one cable skew delay, then assert the REQ signals. The DB(7-0,P) signals shal be held valid for a minimum of one deskew delay plus one cable skew delay plus one hold time after the assertior of the REQ signal. The target shall assert the REQ signal for a minimum of an assertion period. The target may then negate the REQ signal and change or release the DB(7-0,P) signals. The initiator shall read the value on the DB(7-0,P) signals within one hold time of the transition of the REQ signal to true. The initiator shall then respond with an ACK pulse.

If the I/O signal is false (transfer to the target), the initiator shall transfer one byte for each REQ pulse received. After receiving the leading edge of a REQ pulse, the initiator shall first drive the DB(7-0,P) signals to their desired values, delay at least one deskew delay plus one cable skew delay, then assert the ACK signal. The initiator shall hold the DB(7-0,P) signals valid for at least one deskew delay plus one cable skew delay plus one hold time after the assertion of the ACK signal. The initiator shall assert the ACK signal for a minimum of an assertion period. The initiator may then negate the ACK signal and may change or release the DB(7-0,P) signals. The target shall read the value of the DB(7-0,P) signals within one hold time of the transition of the ACK signal to true.

NOTE 26 The description in SCSI-1 allowed some implementors to presume that the leading edge of the first REQ pulse beyond the REQ/ACK offset agreement would not occur until after the trailing edge of the last ACK pulse within the agreement. Devices implemented with this understanding may be subject to data destruction when in synchronous data transfer mode with devices that issue the leading edge of the next REQ pulse, at the boundary of the agreement, as soon as the leading edge of the last ACK pulse within the agreement is received. Implementors using devices of the former type in initiator designs may ensure data integrity by restricting the synchronous offset agreement to values smaller than the maximum nominally offered by their device.

6.1.5.3 Wide data transfer

Wide data transfer is optional and may be used in the DATA phase only if a non-zero wide data transfer agreement is in effect (see WIDE DATA TRANSFER REQUEST message, 6.6.23). The messages determine the use of wide mode by both SCSI devices and establish a data path width to be used during the DATA phase.

Wide data transfers of 16- or 32-bits may be established. Targets and initiators that support 32-bit wide transfers should also support 16-bit wide transfers. All SCSI devices shall support 8-bit data transfers.

During 16-bit wide data transfers, the first logical data byte for each data phase shall be transferred across the DB(7-0,P) signals on the A cable and the second logical data byte shall be transferred across the DB(15-8,P1) signals on the B cable. Subsequent pairs of data bytes are likewise transferred in parallel across the A and B cables (see figure 15).

NOTE 27 X3T10 is documenting an alternate 16-bit single-cable solution and an alternate 32-bit solution and expects to be able to remove the B cable definition in a future version of SCSI.

During 32-bit wide data transfers, the first logical data byte for each data phase shall be transferred across the DB(7-0,P) signals on the A cable and the second, third, and fourth logical data bytes shall be transferred across the DB(15-8,P1), DB(23-16,P2), and DB(31-24,P3) signals, respectively, on the B cable. Subsequent groups of four data bytes are likewise transferred in parallel across the A and B cables (see figure 15).

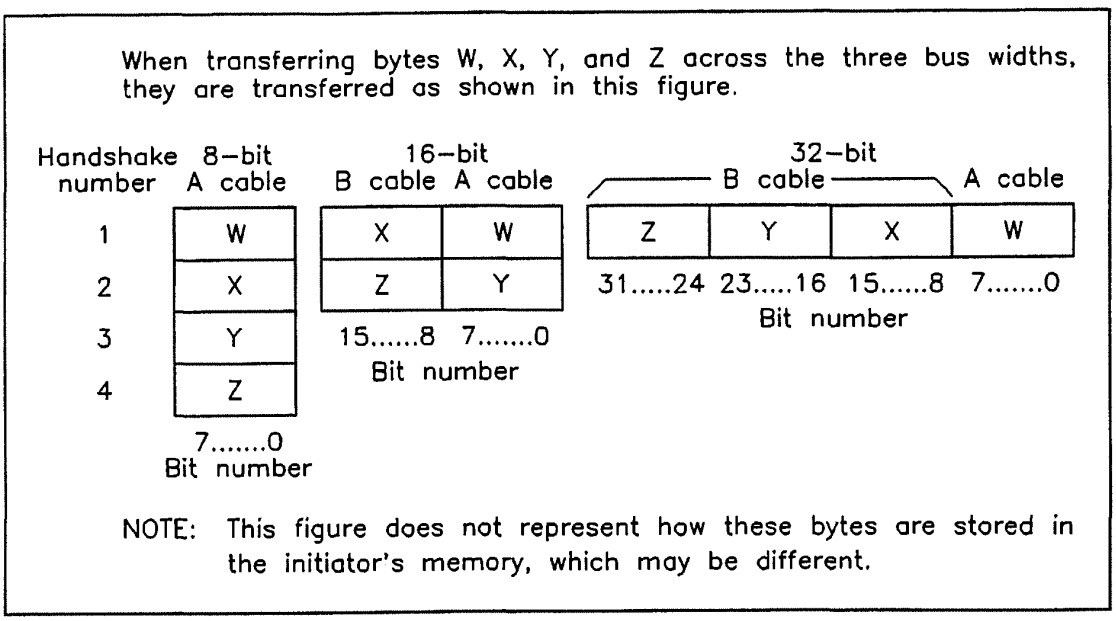


Figure 15 - Wide SCSI byte ordering

If the last data byte transferred does not fall on the DB(15-8,P1) signals for a 16-bit wide transfer or the DB(31-24,P3) signals for a 32-bit wide transfer, then the values of the remaining higher-numbered bits are undefined. However, parity bits for these undefined bytes shall be valid for whatever data is placed on the bus.

To ensure proper data integrity, certain sequence requirements shall be met between the REQ/ACK handshakes on the A cable and the REQB/ACKB handshakes on the B cable:

a) The REQB and ACKB signals shall only be asserted during data phases while a nonzero wide data transfer agreement is in effect. These signals shall not be asserted during other phases.
b) The same information transfer mode (asynchronous or synchronous) shall be used for both the A cable and the B cable. If synchronous data transfer mode is in effect, the same REQ/ACK offset and transfer period shall be used for both cables.
c) The information transfer procedures defined in 6.1.5.1 and 6.1.5.2 for the A cable (the REQ, ACK, and DB(7-0,P) signals) shall also apply to the B cable (the REQB, ACKB, and DB(31-8,P1,P2,P3) signals). The only means available for a target to manage the timing relationship between the signals on the two cables is its management of the REQ and REQB signals. Similarly, the only means for the initiator to manage the timing between the two cables is its management of the ACK and ACKB signals.
d) The target shall ensure that the number of REQ/ACK handshakes and the number of REQB/ACKB handshakes in a data phase are equal before it changes to another phase. The target shall not change the phase until the ACK and ACKB signals have both become false for the last REQ/ACK handshake and the last REQB/ACKB handshake.

NOTE 28 If any violations of these rules are detected by the target, the target may attempt to end the data phase and return CHECK CONDITION status. If it is impossible to correctly terminate the data phase, the target may abnormally terminate the I/O process by an unexpected disconnect. If any violations of these rules are detected by the initiator, the initiator may attempt to send an INITIATOR DETECTED ERROR message to the target. If the initiator is unable to terminate the I/O process normally, it may generate the reset condition.

6.1.6 COMMAND phase

The COMMAND phase allows the target to request command information from the initiator.

The target shall assert the C/D signal and negate the I/O and MSG signals during the REQ/ACK handshake(s) of this phase.

6.1.7 Data phase

The data phase is a term that encompasses both the DATA IN phase and the DATA OUT phase.

6.1.7.1 DATA IN phase

The DATA IN phase allows the target to request that data be sent to the initiator from the target.

The target shall assert the I/O signal and negate the C/D and MSG signals during the REQ/ACK handshake(s) of this phase.

6.1.7.2 DATA OUT phase

The DATA OUT phase allows the target to request that data be sent from the initiator to the target.

The target shall negate the C/D, I/O, and MSG signals during the REQ/ACK handshake(s) of this phase.

6.1.8 STATUS phase

The STATUS phase allows the target to request that status information be sent from the target to the initiator.

The target shall assert the C/D and I/O signals and negate the MSG signal during the REQ/ACK handshake of this phase.

6.1.9 Message phase

The message phase is a term that references either a MESSAGE IN, or a MESSAGE OUT phase. Multiple messages may be sent during either phase. The first byte transferred in either of these phases shall be either a single-byte message or the first byte of a multiple-byte message. Multiple-byte messages shall be wholly contained within a single message phase.

6.1.9.1 MESSAGE IN phase

The MESSAGE IN phase allows the target to request that message(s) be sent to the initiator from the target.

The target shall assert the C/D, I/O, and MSG signals during the REQ/ACK handshake(s) of this phase.

6.1.9.2 MESSAGE OUT phase

The MESSAGE OUT phase allows the target to request that message(s) be sent from the initiator to the target. The target invokes this phase in response to the attention condition created by the initiator (see 6.2.1).

The target shall assert the C/D and MSG signals and negate the I/O signal during the REQ/ACK handshake(s) of this phase. The target shall handshake byte(s) in this phase until the ATN signal is negated, except when rejecting a message.

If the target detects one or more parity error(s) on the message byte(s) received, it may indicate its desire to retry the message(s) by asserting the REQ signal after detecting the ATN signal has gone false and prior to changing to any other phase. The initiator, upon detecting this condition, shall resend all of the previous message byte(s) in the

same order as previously sent during this phase. When resending more than one message byte, the initiator shall assert the ATN signal at least two deskew delays prior to asserting the ACK signal on the first byte and shall maintain the ATN signal asserted until the last byte is sent as described in 6.2.1.

If the target does not retry the MESSAGE OUT phase or it exhausts its retry limit it may

a) return CHECK CONDITION status and set the sense key to ABORTED COMMAND and the additional sense code to MESSAGE ERROR or;
b) indicate an exception condition by performing an unexpected disconnect.

The target may act on messages as received as long as no parity error is detected and may ignore all remaining messages sent under one ATN condition after a parity error is detected. When a sequence of messages is resent by an initiator because of a target detected parity error, the target shall not act on any message which it acted on the first time received.

If the target receives all of the message byte(s) successfully (i.e. no parity errors), it shall indicate that it does not wish to retry by changing to any information transfer phase other than the MESSAGE OUT phase and transfer at least one byte. The target may also indicate that it has successfully received the message byte(s) by changing to the BUS FREE phase (e.g. ABORT or BUS DEVICE RESET messages).

6.1.10 Signal restrictions between phases

When the SCSI bus is between two information transfer phases, the following restrictions shall apply to the SCSI bus signals:

a) The BSY, SEL, REQ, REQB, ACK and ACKB signals shall not change.
b) The C/D, I/O, MSG, and DATA BUS signals may change. When switching the DATA BUS direction from out (initiator driving) to in (target driving), the target shall delay driving the DATA BUS by at least a data release delay plus a bus settle delay after asserting the I/O signal and the initiator shall release the DATA BUS no later than a data release delay after the transition of the I/O signal to true. When switching the DATA BUS direction from in (target driving) to out (initiator driving), the target shall release the DATA BUS no later than a deskew delay after negating the I/O signal.
c) The ATN and RST signals may change as defined under the descriptions for the attention condition (see 6.2.1) and reset condition (see 6.2.2).

6.2 SCSI bus conditions

The SCSI bus has two asynchronous conditions; the attention condition and the reset condition. These conditions cause the SCSI device to perform certain actions and can alter the phase sequence.

Furthermore, SCSI devices may not all be powered-on at the same time. This standard does not address power sequencing issues. However, each SCSI device, as it is powered on, should perform appropriate internal reset operations and internal test operations. Following a power-on to selection time after power-on, SCSI targets should be able to respond with appropriate status and sense data to the TEST UNIT READY, INQUIRY, and REQUEST SENSE commands.

6.2.1 Attention condition

The attention condition allows an initiator to inform a target that the initiator has a message ready. The target may get this message by performing a MESSAGE OUT phase.

The initiator creates the attention condition by asserting ATN at any time except during the ARBITRATION or BUS FREE phases.

The initiator shall negate the ATN signal at least two deskew delays before asserting the ACK signal while transferring the last byte of the messages indicated with a Yes in table 10. If the target detects that the initiator failed to mee this requirement, then the target shall go to BUS FREE phase (see unexpected disconnect, 6.1.1).

The initiator shall assert the ATN signal at least two deskew delays before negating the ACK signal for the last byt transferred in a bus phase for the attention condition to be honoured before transition to a new bus phase. Asserting the ATN signal later might not be honoured until a later bus phase and then may not result in the expected action

A target shall respond with MESSAGE OUT phase as follows:

a) If the ATN signal becomes true during a COMMAND phase, the target shall enter MESSAGE OUT phase afte transferring part or all of the command descriptor block bytes.
b) If the ATN signal becomes true during a DATA phase, the target shall enter MESSAGE OUT phase at th target's earliest convenience (often, but not necessarily on a logical block boundary). The initiator sha continue REQ/ACK handshakes until it detects the phase change.
c) If the ATN signal becomes true during a STATUS phase, the target shall enter MESSAGE OUT phase after th status byte has been acknowledged by the initiator.
d) If the ATN signal becomes true during a MESSAGE IN phase, the target shall enter MESSAGE OUT phas before it sends another message. This permits a MESSAGE PARITY ERROR message from the initiator to b associated with the appropriate message.
e) If the ATN signal becomes true during a SELECTION phase and before the initiator releases the BSY signal the target shall enter MESSAGE OUT phase immediately after that SELECTION phase.
f) If the ATN signal becomes true during a RESELECTION phase, the target shall enter MESSAGE OUT phas after the target has sent its IDENTIFY message for that RESELECTION phase.

NOTE 29 The initiator should only assert the ATN signal during a RESELECTION phase to transmit a BUS DEVICE RESET or DISCONNECT message. Other uses may result in ambiguities concerning the nexus.

The initiator shall keep the ATN signal asserted if more than one byte is to be transferred. The initiator may negat the ATN signal at any time except it shall not negate the ATN signal while the ACK signal is asserted during a MESSAGE OUT phase. Normally, the initiator negates the ATN signal while the REQ signal is true and the ACK signa is false during the last REQ/ACK handshake of the MESSAGE OUT phase.

6.2.2 Reset condition

The reset condition is used to immediately clear all SCSI devices from the bus. This condition shall take precedenc over all other phases and conditions. Any SCSI device may create the reset condition by asserting the RST signa for a minimum of a reset hold time.

All SCSI devices shall release all SCSI bus signals (except the RST signal) within a bus clear delay of the transitior of the RST signal to true. The BUS FREE phase always follows the reset condition.

The effect of the reset condition on I/O processes that have not completed, SCSI device reservations, and SCS device operating modes is determined by whether the SCSI device has implemented the hard reset alternative or the soft reset alternative (one of which shall be implemented) as defined in 6.2.2.1 and 6.2.2.2. The hard and soft rese alternatives are mutually exclusive within a system. A facility for targets to report which reset alternative is implemented is provided in the SftRe bit of the INQUIRY data (see 8.2.5).

NOTE 30 Environmental conditions (e.g. static discharge) may generate brief glitches on the RST signal. It is recommended that SCSI devices not react to these glitches. The manner of rejecting glitches is vendor-specific. The bus clear delay following a RST signal transition to true is measured from the original transition of the RST signal, not from the time that the signal has been confirmed. This limits the time to confirm the RST signal to a maximum of a bus clear delay.

6.2.2.1 Hard reset alternative

SCSI devices that implement the hard reset alternative, upon detection of the reset condition, shall:

a) clear all I/O processes including queued I/O processes.
b) release all SCSI device reservations.
c) return any SCSI device operating modes to their appropriate initial conditions, similar to those conditions that would be found after a normal power-on reset. MODE SELECT conditions shall be restored to their last saved values if saved values have been established. MODE SELECT conditions for which no values have been saved shall be returned to their default values.
d) unit attention condition shall be set (see 7.9).

It is recommended that following a reset to selection time after a hard reset condition ends, SCSI targets be able to respond with appropriate status and sense data to the TEST UNIT READY, INQUIRY, and REQUEST SENSE commands.

6.2.2.2 Soft reset alternative

SCSI devices that implement the soft reset alternative, upon detection of the reset condition, shall:

a) Attempt to complete any I/O processes which have not completed and that were fully identified
b) Preserve all SCSI device reservations
c) Preserve any SCSI device operating modes (MODE SELECT, PREVENT/ALLOW MEDIUM REMOVAL commands, etc.)
d) Preserve all the information required to continue normal dispatching of I/O processes queued prior to the reset condition.

The soft reset alternative allows an initiator to reset the SCSI bus with minimum disruption to the operation of other initiators in a multiple initiator system. To ensure proper operation the following conditions shall be met.

a) An initiator shall not consider an I/O process to be fully identified until the IDENTIFY message (and queue tag message, if any) is sent to the target and the target responds by changing to any other information transfer phase and requests that at least one byte be transferred.
b) A target shall consider an I/O process to be fully identified when it successfully receives the IDENTIFY message and any queue tag message and the initiator negates the ATN signal.
c) If an initiator selects a logical unit for which there already is an active I/O process with the same queue tag (if any) for the same initiator, the target shall clear the original I/O process and perform the new I/O process.
d) If a target reselects an initiator to continue an I/O process for which the initiator has no record, the initiator shall abort that I/O process by sending the ABORT or ABORT TAG message, depending on whether the reselecting I/O process is a tagged I/O process.
e) An initiator shall consider an I/O process to be completed when it negates ACK for a successfully received COMMAND COMPLETE message.
f) A target shall consider an I/O process to be completed when it detects the transition of ACK to false for the COMMAND COMPLETE message with the ATN signal false.
g) An initiator shall not negate the ACK signal for the SAVE DATA POINTER message until it has actually saved the data pointer for the I/O process.
h) A target shall consider the data pointer to be saved when it detects the transition of the ACK signal to false for the SAVE DATA POINTER message with the ATN signal false.
i) If the reset condition occurs between the time that the target asserts the REQ signal for the SAVE DATA POINTER message and it detects the transition of the ACK signal to false, the target shall terminate the I/O process with CHECK CONDITION status. The target shall set the sense key to ABORTED COMMAND. This is necessary because the target cannot determine whether the data pointer has actually been saved.

NOTE 31 If the ATN signal is true in conditions f) or h), the target would normally switch to MESSAGE OUT phase and attempt to transfer a message byte. If the reset condition occurs before the target successfully receives the message byte, it may assume that the initiator has not successfully received the COMMAND COMPLETE message or the SAVE DATA POINTER message. In the case of COMMAND COMPLETE message, the target may reselect the initiator and attempt to send the COMMAND COMPLETE message again. In the case of the SAVE DATA POINTER message, the target may reselect the initiator and terminate the I/O process as described in condition i).

6.3 SCSI bus phase sequences

The order in which phases are used on the SCSI bus follows a prescribed sequence.

The reset condition can abort any phase and is always followed by the BUS FREE phase. Also any other phase can be followed by the BUS FREE phase but many such instances are error conditions (see 6.1.1).

The additional allowable sequences shall be as shown in Figure 16. The normal progression is from the BUS FREE phase to ARBITRATION, from ARBITRATION to SELECTION or RESELECTION, and from SELECTION or RESELECTION to one or more of the information transfer phases (COMMAND, DATA, STATUS, or MESSAGE). The final information transfer phase is normally the MESSAGE IN phase where a DISCONNECT, or COMMAND COMPLETE message is transferred, followed by the BUS FREE phase.

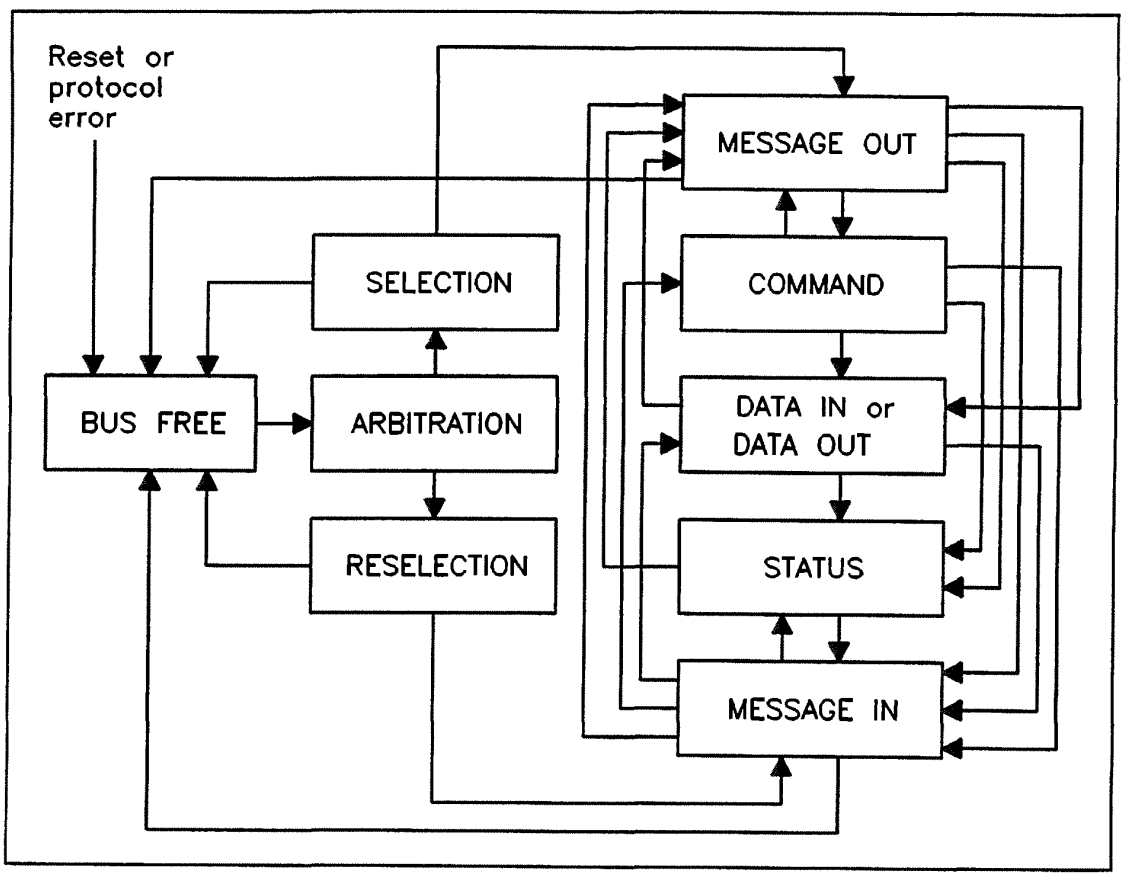


Figure 16 - Phase sequences

6.4 SCSI pointers

Consider the system shown in figure 17 in which an initiator and target communicate on the SCSI bus in order to execute an I/O process.

The SCSI architecture provides for a set of three pointers for each I/O process, called the saved pointers. The set of three pointers consist of one for the command, one for the data, and one for the status. When an I/O process becomes active, its three saved pointers are copied into the initiator's set of three current pointers. There is only one set of current pointers in each initiator. The current pointers point to the next command, data, or status byte to be transferred between the initiator's memory and the target. The saved and current pointers reside in the initiator.

The saved command pointer always points to the start of the command descriptor block (see 7.2) for the I/O process. The saved status pointer always points to the start of the status area for the I/O process. The saved data pointer points to the start of the data area until the target sends a SAVE DATA POINTER message (see 6.6.20) for the I/O process.

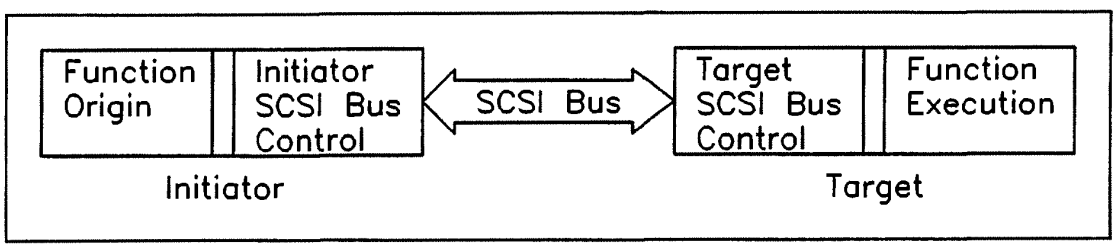


Figure 17 - Simplified SCSI system

In response to the SAVE DATA POINTER message, the initiator stores the value of the current data pointer into the saved data pointer for that I/O process. The target may restore the current pointers to the saved pointer values for the active I/O process by sending a RESTORE POINTERS message (see 6.6.19) to the initiator. The initiator then copies the set of saved pointers into the set of current pointers. Whenever a target disconnects from the bus, only the set of saved pointers are retained. The set of current pointers is restored from the set of saved pointers upon reconnection of the I/O process.

NOTE 32 Since the data pointer value may be modified by the target before the I/O process ends, it should not be used to test for actual transfer length because it is not reliable.

6.5 Message system description

The message system allows communication between an initiator and target for the purpose of interface management. A message may be one, two, or multiple bytes in length. One or more messages may be sent during a single MESSAGE phase, but a message may not be split between multiple MESSAGE phases. The initiator is required to end the MESSAGE OUT phase (by negating ATN) when it sends certain messages identified in table 10.

One-byte, two-byte, and extended message formats are defined. The first byte of the message determines the format as defined in table 9.

Table 9 - Message format

Value	Message format
00h	One-byte message (COMMAND COMPLETE)
01h	Extended messages
02h - 1Fh	One-byte messages
20h - 2Fh	Two-byte messages
30h - 7Fh	Reserved
80h - FFh	One-byte message (IDENTIFY)

One-byte messages consist of a single byte transferred during a MESSAGE phase. The value of the byte determines which message is to be performed as defined in table 10.

Table 10 - Message codes

Code	Support		Message Name	Direction		Negate ATN before last ACK
	Init	Targ				
06h	O	M	ABORT		Out	Yes
0Dh	O	O	ABORT TAG see note 1)		Out	Yes
0Ch	O	M	BUS DEVICE RESET		Out	Yes
0Eh	O	O	CLEAR QUEUE see note 1)		Out	Yes
00h	M	M	COMMAND COMPLETE	In		——
04h	O	O	DISCONNECT	In		——
04h	O	O	DISCONNECT		Out	Yes
80h+	M	O	IDENTIFY	In		——
80h+	M	M	IDENTIFY		Out	No
23h	O	O	IGNORE WIDE RESIDUE (two bytes)	In		——
0Fh	O	O	INITIATE RECOVERY	In		——
0Fh	O	O	INITIATE RECOVERY see note 2)		Out	Yes
05h	M	M	INITIATOR DETECTED ERROR		Out	Yes
0Ah	O	O	LINKED COMMAND COMPLETE	In		——
0Bh	O	O	LINKED COMMAND COMPLETE (WITH FLAG)	In		——
09h	M	M	MESSAGE PARITY ERROR		Out	Yes
07h	M	M	MESSAGE REJECT	In	Out	Yes
***	O	O	MODIFY DATA POINTER	In		——
08h	M	M	NO OPERATION		Out	Yes
			Queue tag messages (two bytes)			
21h	O	O	HEAD OF QUEUE TAG		Out	No
22h	O	O	ORDERED QUEUE TAG		Out	No
20h	O	O	SIMPLE QUEUE TAG	In	Out	No
10h	O	O	RELEASE RECOVERY		Out	Yes
03h	O	O	RESTORE POINTERS	In		——
02h	O	O	SAVE DATA POINTER	In		——
***	O	O	SYNCHRONOUS DATA TRANSFER REQUEST	In	Out	Yes
11h	O	O	TERMINATE I/O PROCESS		Out	Yes
***	O	O	WIDE DATA TRANSFER REQUEST	In	Out	Yes
12h-1Fh			Reserved			
24h-2Fh			Reserved for two-byte messages			
30h-7Fh			Reserved			

Key:
- M = Mandatory support, O = Optional support.
- In = Target to initiator, Out = Initiator to target.
- Yes = Initiator shall negate ATN before last ACK of message.
- No = Initiator may or may not negate ATN before last ACK of message. (see attention condition, 6.2.1.)
- —— = Not applicable
- *** = Extended message (see tables 11 and 12)
- 80h+ = Codes 80h through FFh are used for IDENTIFY messages (see table 13).

NOTES
1 The ABORT TAG and CLEAR QUEUE messages are required if tagged queuing is implemented.
2 Outbound INITIATE RECOVERY messages are only valid during the asynchronous event notification protocol.

Two-byte messages consist of two consecutive bytes transferred during a MESSAGE phase. The value of the first byte determines which message is to be performed as defined in table 10. The second byte is a parameter byte which is used as defined in the message description (see 6.6).

A value of one in the first byte of a message indicates the beginning of a multiple-byte extended message. The minimum number of bytes sent for an extended message is three. The extended message format and the extended message codes are shown in tables 11 and 12, respectively.

Table 11 - Extended message format

Bit / Byte	7	6	5	4	3	2	1	0
0	Extended message (01h)							
1	Extended message length (n)							
2	Extended message code (y)							
3	Extended message arguments							
n+1								

The extended message length specifies the length in bytes of the extended message code plus the extende message arguments to follow. Therefore, the total length of the message is equal to the extended message lengt plus two. A value of zero for the extended message length indicates 256 bytes follow.

The extended message codes are listed in table 12. The extended message arguments are specified within th extended message descriptions.

Table 12 - Extended message codes

Code (y)	Description
02h	Reserved (see note)
00h	MODIFY DATA POINTER
01h	SYNCHRONOUS DATA TRANSFER REQUEST
03h	WIDE DATA TRANSFER REQUEST
04h - 7Fh	Reserved
80h - FFh	Vendor-specific

NOTE - Extended message code 02h was used for the EXTENDED IDENTIFY message in SCSI-1.

The first message sent by the initiator after the SELECTION phase shall be an IDENTIFY, ABORT, or BUS DEVIC RESET message. If a target receives any other message it shall go to BUS FREE phase (see unexpected disconnec 6.1.1).

If the first message is an IDENTIFY message, then it may be immediately followed by other messages, such as th first of a pair of SYNCHRONOUS DATA TRANSFER REQUEST messages. If tagged queuing is used the queue ta message immediately follows the IDENTIFY message (see 6.6.7). The IDENTIFY message establishes a logic connection between the initiator and the specified logical unit or target routine within the target known as an I_T_ nexus or I_T_R nexus. After the RESELECTION phase, the target's first message shall be IDENTIFY. This allows th I_T_L nexus or I_T_R nexus to be reestablished. Only one logical unit or target routine shall be identified for an connection. If a target receives a second IDENTIFY message with a different logical unit number or target routin number during a connection, it shall go to BUS FREE phase (see unexpected disconnect, 6.1.1). The treatment o other logical unit addressing errors is described in 7.5.

All initiators shall implement the mandatory messages tabulated in the Init column of table 10. All targets sha implement the mandatory messages tabulated in the Targ column of table 10.

Whenever an I_T_L nexus or I_T_R nexus is established by an initiator that is allowing disconnection, the initiator sha ensure that the current pointers are equal to the saved pointers for that particular logical unit or target routine. A implied restore pointers operation shall occur as a result of a reconnection.

6.6 Messages

The SCSI messages are defined in this subclause.

6.6.1 ABORT

The ABORT message is sent from the initiator to the target to clear any I/O process for the I_T_x nexus. The target shall go to the BUS FREE phase following successful receipt of this message. The pending data, status, and I/O processes for any other nexus shall not be cleared.

If only an I_T nexus has been established, the target shall go to the BUS FREE phase. No status or message shall be sent for the current I/O process and no other I/O process shall be affected.

NOTES
33 The ABORT message in the case of only an I_T nexus is useful to an initiator that cannot get an IDENTIFY message through to the target due to parity errors and just needs to end the current connection. Any pending data, status, or queued I/O processes for the I_T nexus is not affected.
34 It is not possible to abort an I_T nexus on a reconnection because of item f) in 6.2.1.

It is not an error to issue this message to an I_T_x nexus that does not have an active or queued I/O process.

Previously established conditions, including MODE SELECT parameters, reservations, and extended contingent allegiance shall not be changed by the ABORT message.

NOTE 35 The BUS DEVICE RESET, CLEAR QUEUE, ABORT, and ABORT TAG messages provide a means to clear one or more I/O processes prior to normal termination. The BUS DEVICE RESET message clears all I/O processes for all initiators on all logical units and all target routines of the target. The CLEAR QUEUE message clears all I/O processes for all initiators on the specified logical unit or target routine of the target. The ABORT message clears all I/O processes for the selecting initiator on the specified logical unit or target routine of the target. The ABORT TAG message clears the current I/O process only.

6.6.2 ABORT TAG

The ABORT TAG message shall be implemented if tagged queuing is implemented and may be implemented if untagged queuing is implemented. The target shall go to the BUS FREE phase following successful receipt of this message. The target shall clear the current I/O process. If the target has already started execution of the I/O process, the execution shall be halted. The medium contents may have been modified before the execution was halted. In either case, any pending status or data for the I/O process shall be cleared and no status or ending message shall be sent to the initiator. Pending status, data, and commands for other active or queued I/O processes shall not be affected. Execution of other I/O processes queued for the I_T_x nexus shall not be aborted.

Previously established conditions, including MODE SELECT parameters, reservations, and extended contingent allegiance shall not be changed by the ABORT TAG message.

On a reconnection, the ABORT TAG message aborts the current I/O process if it is fully identified. If the I/O process is not fully identified (i.e. an I_T_L nexus exists, but the target is reconnecting for an I_T_L_Q nexus), then the I/O process is not aborted and the target goes to the BUS FREE phase.

NOTE 36 A nexus is not fully identified on a reconnection if the ATN signal is asserted during or prior to the IDENTIFY message and the target only has tagged I/O processes for that initiator on that logical unit.

6.6.3 BUS DEVICE RESET

The BUS DEVICE RESET message is sent from an initiator to direct a target to clear all I/O processes on that SCSI device. This message forces a hard reset condition to the selected SCSI device. The target shall go to the BUS FREE phase following successful receipt of this message. The target shall create a unit attention condition for all initiators (see 7.9).

6.6.4 CLEAR QUEUE

The CLEAR QUEUE message shall be implemented if tagged queuing is implemented and may be implemented if untagged queuing is implemented. The target shall go to the BUS FREE phase following successful receipt of this message. The target shall perform an action equivalent to receiving a series of ABORT messages from each initiator. All I/O processes, from all initiators, in the queue for the specified logical unit shall be cleared from the queue. All active I/O processes shall be terminated. The medium may have been altered by partially executed commands. All pending status and data for that logical unit or target routine for all initiators shall be cleared. No status or message shall be sent for any of the I/O processes. A unit attention condition shall be generated for all other initiators with I/O processes that either were active or were queued for that logical unit or target routine. When reporting the unit attention condition the additional sense code shall be set to COMMANDS CLEARED BY ANOTHER INITIATOR.

Previously established conditions, including MODE SELECT parameters, reservations, and extended contingent allegiance shall not be changed by the CLEAR QUEUE message.

6.6.5 COMMAND COMPLETE

The COMMAND COMPLETE message is sent from a target to an initiator to indicate that the execution of an I/O process has completed and that valid status has been sent to the initiator. After successfully sending this message, the target shall go to the BUS FREE phase by releasing the BSY signal. The target shall consider the message transmission to be successful when it detects the negation of ACK for the COMMAND COMPLETE message with the ATN signal false.

NOTE 37 The I/O process may have completed successfully or unsuccessfully as indicated in the status.

6.6.6 DISCONNECT

The DISCONNECT message is sent from a target to inform an initiator that the present connection is going to be broken (the target plans to disconnect by releasing the BSY signal), but that a later reconnect will be required in order to complete the current I/O process. This message shall not cause the initiator to save the data pointer. After successfully sending this message, the target shall go to the BUS FREE phase by releasing the BSY signal. The target shall consider the message transmission to be successful when it detects the negation of the ACK signal for the DISCONNECT message with the ATN signal false.

Targets that break data transfers into multiple connections shall end each successful connection (except possibly the last) with a SAVE DATA POINTER - DISCONNECT message sequence.

This message may also be sent from an initiator to a target to instruct the target to disconnect from the SCSI bus. If this option is supported, and after the DISCONNECT message is received, the target shall switch to MESSAGE IN phase, send the DISCONNECT message to the initiator (possibly preceded by SAVE DATA POINTER message), and then disconnect by releasing BSY. After releasing the BSY signal, the target shall not participate in another ARBITRATION phase for at least a disconnection delay or the time limit specified in the disconnect time limit mode parameter (see 8.3.3.2) whichever is greater. If this option is not supported or the target cannot disconnect at the time when it receives the DISCONNECT message from the initiator, the target shall respond by sending a MESSAGE REJECT message to the initiator.

6.6.7 IDENTIFY

The IDENTIFY message (see table 13) is sent by either the initiator or the target to establish an I_T_L nexus or an I_T_R nexus.

NOTE 38 Use of the IDENTIFY message to establish an I_T_R nexus allows connection to one of up to eight target routines or functions in the target. These target routines are expected to be used for maintenance and diagnostic purposes.

Table 13 - IDENTIFY message format

Bit	7	6	5	4	3	2	1	0
	Identify	DiscPriv	LUNTAR	Reserved	Reserved	LUNTRN		

The identify bit shall be set to one to specify that this is an IDENTIFY message.

A disconnect privilege (DiscPriv) bit of one specifies that the initiator has granted the target the privilege of disconnecting. A DiscPriv bit of zero specifies that the target shall not disconnect. This bit is not defined and shall be set to zero when an IDENTIFY message is sent by a target.

A logical unit target (LUNTAR) bit of zero specifies that the I/O process is directed to or from a logical unit. A LUNTAR bit of one specifies that the I/O process is directed to or from a target routine.

The logical unit number target routine number (LUNTRN) field specifies a logical unit number if the LUNTAR bit is zero. The LUNTRN field specifies a target routine number if the LUNTAR bit is one. The response to an invalid value in the LUNTRN field is described in 7.5.3. Only the INQUIRY and REQUEST SENSE commands are valid for target routines. If a target receives any other command for a target routine, it shall return CHECK CONDITION status and shall set the sense key to ILLEGAL REQUEST.

An IDENTIFY message is invalid if a reserved bit is set to one or if the LUNTAR bit is set to one and the target does not implement target routines. A device may respond to an invalid IDENTIFY message by immediately sending a MESSAGE REJECT message or by returning CHECK CONDITION status. If a CHECK CONDITION status is returned, the sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID BITS IN IDENTIFY MESSAGE FIELD.

Only one logical unit number or target routine number shall be identified per I/O process. The initiator may send one or more IDENTIFY messages during a connection. A second IDENTIFY message with a different value in either the LUNTAR bit or LUNTRN field shall not be issued before a BUS FREE phase has occurred; if a target receives a second IDENTIFY message with a different value in either of these fields, it shall go to BUS FREE phase (see unexpected disconnect, 6.1.1). Thus an initiator may change the DiscPriv bit, but may not attempt to switch to another I/O process. (See the DTDC field of the disconnect-reconnect page (8.3.3.2) for additional controls over disconnection.)

An implied RESTORE POINTERS message shall be performed by the initiator prior to the assertion of the ACK signal on the next phase for an inbound IDENTIFY message sent during reconnection.

An implied RESTORE POINTERS message shall be performed by the initiator following successful identification of the nexus during the MESSAGE IN phase of a reconnection and before the negation of the ACK signal for the next transfer following the successful identification.

Identification is considered successful during an initial connection or an initiator reconnect when the target detects no error during the transfer of the IDENTIFY message and an optional queue tag message in the MESSAGE OUT phase immediately following the SELECTION phase. See 6.5 for the ordering of the IDENTIFY and queue tag messages. See 6.1.9.2 for handling target detected errors during the MESSAGE OUT phase.

Identification is considered successful during a target reconnect when the ATN signal is not asserted during the transfer of either the IDENTIFY message or the SIMPLE QUEUE TAG message for an I_T_L_Q nexus in the MESSAGE IN phase immediately following the RESELECTION phase. See clause 6.5 for the ordering of the IDENTIFY and queue tag messages. See 6.2.1, item d), for handling target detected errors during the MESSAGE IN phase.

6.6.8 IGNORE WIDE RESIDUE

Table 14 - IGNORE WIDE RESIDUE message format

Bit / Byte	7	6	5	4	3	2	1	0
0	Message code (23h)							
1	Ignore (01h, 02h, 03h)							

The IGNORE WIDE RESIDUE message (see table 14) shall be sent from a target to indicate that the number of valid bytes sent during the last REQ/ACK handshake and REQB/ACKB handshake of a DATA IN phase is less than the negotiated transfer width. The ignore field indicates the number of invalid data bytes transferred. This message shall be sent immediately following that DATA IN phase and prior to any other messages. The ignore field is defined in table 15.

NOTE 39 More than one IGNORE WIDE RESIDUE message may occur during an I/O process.

Table 15 - Ignore field definition

Ignore	Invalid data bits	
	32-bit transfers	16-bit transfers
00h	Reserved	Reserved
01h	DB(31-24)	DB(15-8)
02h	DB(31-16)	Reserved
03h	DB(31-8)	Reserved
04h - FFh	Reserved	Reserved

Even though a byte is invalid its corresponding parity bit shall be valid for the value transferred. For 16-bit transfers DB(31-16) are always invalid and the corresponding parity bits are also invalid.

6.6.9 INITIATE RECOVERY

A target that supports extended contingent allegiance shall inform the initiator it is entering this condition by sending an INITIATE RECOVERY message immediately following a CHECK CONDITION or COMMAND TERMINATED status. The extended contingent allegiance condition remains in effect until terminated as described in 7.7.

If an asynchronous event occurs, the target may enter an extended contingent allegiance condition by becoming a temporary initiator and sending the INITIATE RECOVERY message following the IDENTIFY message and any queue tag message and before the COMMAND phase of the SEND command that is used to perform the asynchronous event notification (see 7.5.5). The successful transmission of this message establishes the extended contingent allegiance condition which remains in effect until terminated as described in 7.7.

NOTE 40 If the target notifies multiple initiators of the asynchronous event, it should include the INITIATE RECOVERY message in only one of the notifications.

A MESSAGE REJECT response to an INITIATE RECOVERY message indicates that an extended contingent allegiance condition shall not be established. The enabled or disabled state of an extended contingent allegiance (see 8.3.3.1) is not changed by the rejection of an INITIATE RECOVERY message.

6.6.10 INITIATOR DETECTED ERROR

The INITIATOR DETECTED ERROR message is sent from an initiator to inform a target that an error has occurred that does not preclude the target from retrying the operation. The source of the error may either be related to previous activities on the SCSI bus or may be internal to the initiator and unrelated to any previous SCSI bus activity. Although present pointer integrity is not assured, a RESTORE POINTERS message or a disconnect followed by a reconnect, shall cause the pointers to be restored to their defined prior state.

6.6.11 LINKED COMMAND COMPLETE

The LINKED COMMAND COMPLETE message is sent from a target to an initiator to indicate that the execution of a linked command has completed and that status has been sent. The initiator shall then set the pointers to the initial state for the next linked command.

6.6.12 LINKED COMMAND COMPLETE (WITH FLAG)

The LINKED COMMAND COMPLETE (WITH FLAG) message is sent from a target to an initiator to indicate that the execution of a linked command (with the flag bit set to one) has completed and that status has been sent. The initiator shall then set the pointers to the initial state of the next linked command. Typically this message would be used to cause an interrupt in the initiator between two linked commands.

6.6.13 MESSAGE PARITY ERROR

The MESSAGE PARITY ERROR message is sent from the initiator to the target to indicate that it received a message byte with a parity error (see 6.1.9.2).

In order to indicate its intentions of sending this message, the initiator shall assert the ATN signal prior to its release of the ACK signal for the REQ/ACK handshake of the message byte that has the parity error. This provides an interlock so that the target can determine which message byte has the parity error. If the target receives this message under any other circumstance, it shall signal a catastrophic error condition by releasing the BSY signal without any further information transfer attempt (see 6.1.1).

If after receiving the MESSAGE PARITY ERROR message the target returns to the MESSAGE IN phase before switching to some other phase, the target shall resend the entire message that had the parity error.

6.6.14 MESSAGE REJECT

The MESSAGE REJECT message is sent from either the initiator or target to indicate that the last message or message byte it received was inappropriate or has not been implemented.

In order to indicate its intentions of sending this message, the initiator shall assert the ATN signal prior to its release of the ACK signal for the REQ/ACK handshake of the message byte that is to be rejected. If the target receives this message under any other circumstance, it shall reject this message.

When a target sends this message, it shall change to MESSAGE IN phase and send this message prior to requesting additional message bytes from the initiator. This provides an interlock so that the initiator can determine which message byte is rejected.

After a target sends a MESSAGE REJECT message and if the ATN signal is still asserted, then it shall return to the MESSAGE OUT phase. The subsequent MESSAGE OUT phase shall begin with the first byte of a message.

6.6.15 MODIFY DATA POINTER Message

Table 16 - MODIFY DATA POINTER

Bit / Byte	7	6	5	4	3	2	1	0
0	Extended message (01h)							
1	Extended message length (05h)							
2	MODIFY DATA POINTER (00h)							
3	(MSB)			Argument				
6								(LSB)

The MODIFY DATA POINTER message (see table 16) is sent from the target to the initiator and requests that the signed argument be added (two's complement) to the value of the current data pointer.

6.6.16 NO OPERATION

The NO OPERATION message is sent from an initiator in response to a target's request for a message when the initiator does not currently have any other valid message to send.

For example, if the target does not respond to the attention condition until a later phase and at that time the origina message is no longer valid the initiator may send the NO OPERATION message when the target enters the MESSAGE OUT phase.

6.6.17 Queue tag messages

Table 17 - Queue tag message format

Bit / Byte	7	6	5	4	3	2	1	0
0	Message code (20h, 21h, 22h)							
1	Queue tag (00h - FFh)							

Table 17 defines the format for the queue tag messages. If the target implements tagged queuing, all of the queue tag messages are mandatory: HEAD OF QUEUE TAG, ORDERED QUEUE TAG, and SIMPLE QUEUE TAG. Tagged queuing is only defined for logical units, not for target routines.

If a target does not implement tagged queuing and a queue tag message is received or if a queue tag message is received for a target routine, it shall respond with a MESSAGE REJECT message and accept the I/O process as i it were untagged.

The queue tag messages are used to specify an identifier, called a queue tag, for the I/O process that establishes the I_T_L_Q nexus. The queue tag field is an 8-bit unsigned integer assigned by the initiator during an initia connection. The queue tag for every I/O process for each I_T_L nexus should be unique. If the target receives a queue tag that is currently in use for the I_T_L nexus, then it shall respond as defined in 7.5.2. A queue tag becomes available for reassignment when the I/O process ends. The numeric value of a queue tag has no effect on the orde of execution.

NOTE 41 For each logical unit on each target, each initiator has up to 256 queue tags to assign to I/O processes. Thus a target with eight logical units could have up to 14 336 I/O processes concurrently in existence if there were seven initiators on the bus.

Whenever an initiator connects to a target, the appropriate queue tag message shall be sent immediately following the IDENTIFY message and within the same MESSAGE OUT phase to establish the I_T_L_Q nexus for the I/O process. Only one I_T_L_Q nexus may be established during a connection. If a queue tag message is not sent, then only an I_T_x nexus is established for the I/O process (untagged command).

Whenever a target reconnects to an initiator to continue a tagged I/O process, the SIMPLE QUEUE TAG message shall be sent immediately following the IDENTIFY message and within the same MESSAGE IN phase to revive the I_T_L_Q nexus for the I/O process. Only one I_T_L_Q nexus may be revived during a reconnection. If the SIMPLE QUEUE TAG message is not sent, then only an I_T_x nexus is revived for the I/O process (untagged command).

If a target attempts to reconnect using an invalid queue tag, then the initiator should respond with an ABORT TAG message.

6.6.17.1 HEAD OF QUEUE TAG

The HEAD OF QUEUE TAG message specifies that the I/O process be placed first in that logical unit's command queue. An I/O process already being executed by the target shall not be preempted. A subsequent I/O process received with a HEAD OF QUEUE TAG message shall be placed at the head of the command queue for execution in last-in, first-out order.

6.6.17.2 ORDERED QUEUE TAG

The ORDERED QUEUE TAG message specifies that the I/O process be placed in that logical unit's command queue for execution in the order received. All queued I/O processes for the logical unit received prior to this I/O process shall be executed before this I/O process is executed. All queued I/O processes received after this I/O process shall be executed after this I/O process, except for I/O processes received with a HEAD OF QUEUE TAG message.

6.6.17.3 SIMPLE QUEUE TAG

The SIMPLE QUEUE TAG message specifies that the I/O process be placed in that logical unit's command queue. The order of execution is described in 7.8.

6.6.18 RELEASE RECOVERY

The RELEASE RECOVERY message is sent from an initiator to a target to terminate an extended contingent allegiance condition previously established by an INITIATE RECOVERY message. This message shall be sent immediately following the IDENTIFY message in the same MESSAGE OUT phase. The extended contingent allegiance condition ends upon successful receipt of the RELEASE RECOVERY message. The target shall go to the BUS FREE phase following successful receipt of this message.

If a RELEASE RECOVERY message is received by a target that implements extended contingent allegiance when an extended contingent allegiance condition does not exist, the message shall not be rejected and the target shall go to the BUS FREE phase.

6.6.19 RESTORE POINTERS

The RESTORE POINTERS message is sent from a target to direct the initiator to copy the most recently saved command, data, and status pointers for the I/O process to the corresponding current pointers. The command and status pointers shall be restored to the beginning of the present command and status areas. The data pointer shall be restored to the value at the beginning of the data area in the absence of a SAVE DATA POINTER message or to the value at the point at which the last SAVE DATA POINTER message occurred for that nexus.

6.6.20 SAVE DATA POINTER

The SAVE DATA POINTER message is sent from a target to direct the initiator to copy the current data pointer to the saved data pointer for the current I/O process. (See 6.4 for a definition of pointers.)

6.6.21 SYNCHRONOUS DATA TRANSFER REQUEST

Table 18 - SYNCHRONOUS DATA TRANSFER REQUEST

Bit / Byte	7	6	5	4	3	2	1	0
0	Extended message (01h)							
1	Extended message length (03h)							
2	SYNCHRONOUS DATA TRANSFER REQUEST code (01h)							
3	Transfer period factor							
4	REQ/ACK offset							

A SYNCHRONOUS DATA TRANSFER REQUEST (SDTR) message (see table 18) exchange shall be initiated by an SCSI device whenever a previously-arranged data transfer agreement may have become invalid. The agreement becomes invalid after any condition which may leave the data transfer agreement in an indeterminate state such as:

a) after a hard reset condition;
b) after a BUS DEVICE RESET message and;
c) after a power cycle.

In addition, an SCSI device may initiate an SDTR message exchange whenever it is appropriate to negotiate a new data transfer agreement (either synchronous or asynchronous). SCSI devices that are capable of synchronous data transfers shall not respond to an SDTR message with a MESSAGE REJECT message.

NOTES
42 Renegotiation at every selection is not recommended, since a significant performance impact is likely.
43 Due to historical problems with early host adapters that could not accept an SDTR message, some targets may not initiate synchronous negotiation after a power cycle as required by this standard. Host adapters that support synchronous mode may avoid the ensuing failure modes when the target is independently power cycled by initiating a synchronous negotiation on each REQUEST SENSE and INQUIRY command.

The SDTR message exchange establishes the permissible transfer periods and the REQ/ACK offsets for all logical units and target routines on the two devices. This agreement only applies to data phases.

The transfer period factor times four is the value of the transfer period. The transfer period is the minimum time allowed between leading edges of successive REQ pulses and of successive ACK pulses to meet the device requirements for successful reception of data.

The REQ/ACK offset is the maximum number of REQ pulses allowed to be outstanding before the leading edge of its corresponding ACK pulse is received at the target. This value is chosen to prevent overflow conditions in the device's reception buffer and offset counter. A REQ/ACK offset value of zero shall indicate asynchronous data transfer mode; a value of FFh shall indicate unlimited REQ/ACK offset.

The originating device (the device that sends the first of the pair of SDTR messages) sets its values according to the rules above to permit it to receive data successfully. If the responding device can also receive data successfully with these values (or smaller transfer periods or larger REQ/ACK offsets or both), it returns the same values in its SDTR message. If it requires a larger transfer period, a smaller REQ/ACK offset, or both in order to receive data successfully, it substitutes values in its SDTR message as required, returning unchanged any value not required to be changed. Each device when transmitting data shall respect the limits set by the other's SDTR message, but it is

permitted to transfer data with larger transfer periods, smaller REQ/ACK offsets, or both than specified in the other's SDTR message. The successful completion of an exchange of SDTR messages implies an agreement as follows:

Responding device SDTR response	Implied agreement
a) Non-zero REQ/ACK offset	Each device transmits data with a transfer period equal to or greater than and a REQ/ACK offset equal to or less than the values received in the other device's SDTR message.
b) REQ/ACK offset equal to zero	Asynchronous transfer
c) MESSAGE REJECT message	Asynchronous transfer

If the initiator recognizes that negotiation is required, it asserts the ATN signal and sends a SDTR message to begin the negotiating process. After successfully completing the MESSAGE OUT phase, the target shall respond with the proper SDTR message. If an abnormal condition prevents the target from returning an appropriate response, both devices shall go to asynchronous data transfer mode for data transfers between the two devices.

Following target response (1) above, the implied agreement for synchronous operation shall be considered to be negated by both the initiator and the target if the initiator asserts the ATN signal and the first message out is either MESSAGE PARITY ERROR or MESSAGE REJECT. In this case, both devices shall go to asynchronous data transfer mode for data transfers between the two devices. For the MESSAGE PARITY ERROR case, the implied agreement shall be reinstated if a retransmittal of the second of the pair of messages is successfully accomplished. After a vendor-specific number of retry attempts (greater than zero), if the target receives a MESSAGE PARITY ERROR message, it shall terminate the retry activity. This may be done either by changing to any other information transfer phase and transferring at least one byte of information or by going to the BUS FREE phase (see 6.1.1). The initiator shall accept such action as aborting the negotiation, and both devices shall go to asynchronous data transfer mode for data transfers between the two devices.

If the target recognizes that negotiation is required, it sends an SDTR message to the initiator. Prior to releasing the ACK signal on the last byte of the SDTR message from the target, the initiator shall assert the ATN signal and respond with its SDTR message or with a MESSAGE REJECT message. If an abnormal condition prevents the initiator from returning an appropriate response, both devices shall go to asynchronous data transfer mode for data transfers between the two devices.

Following an initiator's responding SDTR message, an implied agreement for synchronous operation shall not be considered to exist until the target leaves the MESSAGE OUT phase, indicating that the target has accepted the negotiation. After a vendor-specific number of retry attempts (greater than zero), if the target has not received the initiator's responding SDTR message, it shall go to the BUS FREE phase without any further information transfer attempt (see 6.1.1). This indicates that a catastrophic error condition has occurred. Both devices shall go to asynchronous data transfer mode for data transfers between the two devices.

If, following an initiator's responding SDTR message, the target shifts to MESSAGE IN phase and the first message in is MESSAGE REJECT, the implied agreement shall be considered to be negated and both devices shall go to asynchronous data transfer mode for data transfers between the two devices.

The implied synchronous agreement shall remain in effect until a BUS DEVICE RESET message is received, until a hard reset condition occurs, or until one of the two SCSI devices elects to modify the agreement. The default data transfer mode is asynchronous data transfer mode. The default data transfer mode is entered at power on, after a BUS DEVICE RESET message, or after a hard reset condition.

6.6.22 TERMINATE I/O PROCESS

The TERMINATE I/O PROCESS message is sent from the initiator to the target to terminate the current I/O process without corrupting the medium.

With the following exceptions, the target shall terminate the current I/O process and return COMMAND TERMINATED status. The sense key shall be set to NO SENSE. The additional sense code and qualifier are set to I/O PROCESS TERMINATED.

If the associated I/O process involves a data phase, the target shall set the valid bit in the sense data to one and set the information field as follows:

a) If the command descriptor block specifies an allocation length or parameter list length, the information field shall be set to the difference (residue) between the number of bytes successfully transferred and the requested length.
b) If the command descriptor block specifies a transfer length field, the information field shall be set as defined in the REQUEST SENSE command (see 8.2.14).

If an error is detected for the associated I/O process the target shall ignore the TERMINATE I/O PROCESS message.

If the operation requested for the associated I/O process has been completed but status has not been returned, the target shall ignore the TERMINATE I/O PROCESS message.

If the target does not support this message or is unable to stop the current I/O process, it shall send a MESSAGE REJECT message to the initiator and continue the I/O process in a normal manner.

The effect of a TERMINATE I/O PROCESS message on the command queue depends on the queue error recovery option specified in the control mode page (see 8.3.3.1) and on whether or not a contingent allegiance condition is generated.

NOTE 44 The TERMINATE I/O PROCESS message provides a means for the initiator to request the target to reduce the transfer length of the current command to the amount that has already been transferred. The initiator can use the sense data to determine the actual number of bytes or blocks that have been transferred. This message is normally used by the initiator to stop a lengthy read, write, or verify operation when a higher-priority command is available to be executed. It is up to the initiator to complete the terminated command at a later time, if required.

6.6.23 WIDE DATA TRANSFER REQUEST

Table 19 - WIDE DATA TRANSFER MESSAGE

Bit / Byte	7	6	5	4	3	2	1	0
0	Extended message (01h)							
1	Extended message length (02h)							
2	WIDE DATA TRANSFER REQUEST code (03h)							
3	Transfer width exponent							

A WIDE DATA TRANSFER REQUEST (WDTR) message (see table 19) exchange shall be initiated by an SCSI device whenever a previously-arranged transfer width agreement may have become invalid. The agreement becomes invalid after any condition which may leave the data transfer agreement in an indeterminate state such as:

a) after a hard reset condition;
b) after a BUS DEVICE RESET message and;
c) after a power cycle.

In addition, an SCSI device may initiate a WDTR message exchange whenever it is appropriate to negotiate a new transfer width agreement. SCSI devices that are capable of wide data transfers (greater than eight bits) shall not respond to a WDTR message with a MESSAGE REJECT message.

NOTE 45 Renegotiation at every selection is not recommended, since a significant performance impact is likely.

The WDTR message exchange establishes an agreement between two SCSI devices on the width of the data path to be used for DATA phase transfers between the two devices. This agreement applies to DATA IN and DATA OUT phases only. All other information transfer phases shall use an eight-bit data path.

If an SCSI device implements both wide data transfer option and synchronous data transfer option, then it shall negotiate the wide data transfer agreement prior to negotiating the synchronous data transfer agreement. If a synchronous data transfer agreement is in effect, then an SCSI device that accepts a WDTR message shall reset the synchronous agreement to asynchronous mode.

The transfer width is two to the transfer width exponent bytes wide. The transfer width that is established applies to all logical units on both SCSI devices. Valid transfer widths are 8 bits (m = 00h), 16 bits (m = 01h), and 32 bits (m = 02h). Values of m greater than 02h are reserved.

The originating SCSI device (the SCSI device that sends the first of the pair of WDTR messages) sets its transfer width value to the maximum data path width it elects to accommodate. If the responding SCSI device can also accommodate this transfer width, it returns the same value in its WDTR message. If it requires a smaller transfer width, it substitutes the smaller value in its WDTR message. The successful completion of an exchange of WDTR messages implies an agreement as follows:

Responding device WDTR response	Implied agreement
a) Non-zero transfer width	Each device transmits and receives data with a transfer width equal to the responding SCSI device's transfer width.
b) Transfer width equal to zero	Eight-bit data transfer
c) MESSAGE REJECT message	Eight-bit data transfer

If the initiator recognizes that negotiation is required, it asserts the ATN signal and sends a WDTR message to begin the negotiating process. After successfully completing the MESSAGE OUT phase, the target shall respond with the proper WDTR message. If an abnormal condition prevents the target from returning an appropriate response, both devices shall go to eight-bit data transfer mode for data transfers between the two devices.

Following target response 1) above, the implied agreement for wide data transfers shall be considered to be negated by both the initiator and the target if the initiator asserts ATN and the first message out is either MESSAGE PARITY ERROR or MESSAGE REJECT. In this case, both devices shall go to eight-bit data transfer mode for data transfers between the two devices. For the MESSAGE PARITY ERROR case, the implied agreement shall be reinstated if a retransmittal of the second of the pair of messages is successfully accomplished. After a vendor-specific number of retry attempts (greater than zero), if the target receives a MESSAGE PARITY ERROR message, it shall terminate the retry activity. This may be done either by changing to any other information transfer phase and transferring at least one byte of information or by going to the BUS FREE phase (see 6.1.1). The initiator shall accept such action as aborting the negotiation, and both devices shall go to eight-bit data transfer mode for data transfers between the two devices.

If the target recognizes that negotiation is required, it sends a WDTR message to the initiator. Prior to releasing the ACK signal on the last byte of the WDTR message from the target, the initiator shall assert the ATN signal and respond with its WDTR message or with a MESSAGE REJECT message. If an abnormal condition prevents the initiator from returning an appropriate response, both devices shall go to eight-bit data transfer mode for data transfers between the two devices.

Following an initiator's responding WDTR message, an implied agreement for wide data transfer operation shall not be considered to exist until the target leaves the MESSAGE OUT phase, indicating that the target has accepted the negotiation. After a vendor-specific number of retry attempts (greater than zero), if the target has not received the initiator's responding WDTR message, it shall go to the BUS FREE phase without any further information transfer attempt (see 6.1.1). This indicates that a catastrophic error condition has occurred. Both devices shall go to eight-bit data transfer mode for data transfers between the two devices.

If, following an initiator's responding WDTR message, the target shifts to MESSAGE IN phase and the first message in is MESSAGE REJECT, the implied agreement shall be considered to be negated and both devices shall go to eight-bit data transfer mode for data transfers between the two devices.

The implied transfer width agreement shall remain in effect until a BUS DEVICE RESET message is received, until a hard reset condition occurs, or until one of the two SCSI devices elects to modify the agreement. The default data transfer width is eight-bit data transfer mode. The default data transfer mode is entered at power on, after a BUS DEVICE RESET message, or after a hard reset condition.

7 SCSI commands and status

This clause defines the SCSI command and status structures and gives several examples.

By keeping to a minimum the functions essential to communicate via this protocol, a wide range of peripheral devices of varying capability can operate in the same environment. Because subsets of the full architecture may be implemented, optional functions are noted.

7.1 Command implementation requirements

The first byte of all SCSI commands shall contain an operation code as defined in this standard. Targets shall implement all commands with a mandatory operation code (see 7.1.2) both in clause 8 and in the appropriate clause for their device type.

7.1.1 Reserved

Reserved bits, fields, bytes, and code values are set aside for future standardization. Their use and interpretation may be specified by future extensions to this standard. A reserved bit, field, or byte shall be set to zero, or in accordance with a future extension to this standard. A target that receives a reserved bit, field, or byte that is not zero or receives a reserved code value shall terminate the command with CHECK CONDITION status and the sense key shall be set to ILLEGAL REQUEST. It shall also be acceptable for a target to interpret a bit, field, byte, or code value in accordance with a future extension to this standard.

7.1.2 Operation code types

The operation code types are defined in table 20.

Table 20 - Operation code type

Operation code type	Description
M	Mandatory - Commands so designated shall be implemented in order to meet the minimum requirement of this standard.
O	Optional - Commands so designated, if implemented, shall be implemented as defined in this standard.
V	Vendor-specific - Operation codes so designated are available for vendor defined commands. See the vendor specifications where compatibility is desired.
R	Reserved - Operation codes so designated shall not be used. They are reserved for future extensions to this standard.

7.2 Command descriptor block

A command is communicated by sending a command descriptor block to the target. For several commands, the command descriptor block is accompanied by a list of parameters sent during the DATA OUT phase. See the specific commands for detailed information.

The command descriptor block always has an operation code as its first byte and a control byte as its last byte.

For all commands, if there is an invalid parameter in the command descriptor block, then the target shall terminate the command without altering the medium.

Table 21 - Typical command descriptor block for six-byte commands

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code							
1	Logical unit number			(MSB)				
2	Logical block address (if required)							
3								(LSB)
4	Transfer length (if required) Parameter list length (if required) Allocation length (if required)							
5	Control							

Table 22 - Typical command descriptor block for ten-byte commands

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code							
1	Logical unit number			Reserved				
2	(MSB)							
3	Logical block address (if required)							
4								
5								(LSB)
6	Reserved							
7	(MSB)			Transfer length (if required) Parameter list length (if required)				
8				Allocation length (if required)				(LSB)
9	Control							

Table 23 - Typical command descriptor block for twelve-byte commands

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code							
1	Logical unit number			Reserved				
2	(MSB)							
3				Logical block address (if required)				
4								
5								(LSB)
6	(MSB)							
7				Transfer length (if required) Parameter list length (if required) Allocation length (if required)				
8								
9								(LSB)
10	Reserved							
11	Control							

7.2.1 Operation code

The operation code (see table 24) of the command descriptor block has a group code field and a command code field. The three-bit group code field provides for eight groups of command codes. The five-bit command code field provides for thirty-two command codes in each group. Thus, a total of 256 possible operation codes exist. Operation codes are defined in the subsequent subclauses.

The group code specifies one of the following groups:

a) Group 0 - six-byte commands (see table 21)
b) Group 1 - ten-byte commands (see table 22)
c) Group 2 - ten-byte commands (see table 22)
d) Group 3 - reserved
e) Group 4 - reserved
f) Group 5 - twelve-byte commands (see table 23)
g) Group 6 - vendor-specific
h) Group 7 - vendor-specific

Table 24 - Operation code

Bit	7	6	5	4	3	2	1	0
	Group code			Command code				

7.2.2 Logical unit number

The logical unit number is defined in the IDENTIFY message (6.6.7). The target shall ignore the logical unit numbe specified within the command descriptor block if an IDENTIFY message was received. It is recommended that th logical unit number in the command descriptor block be set to zero.

NOTICE: The logical unit number field is included in the command descriptor block for compatibility with some SCSI-1 devices. This field may be reclaimed in SCSI-3. New implementations should use the outbound IDENTIFY message, which is mandatory in SCSI-2, to establish the I_T_L nexus.

7.2.3 Logical block address

The logical block address on logical units or within a partition on device volumes shall begin with block zero and b contiguous up to the last logical block on that logical unit or within that partition.

A six-byte command descriptor block contains a 21-bit logical block address. The ten-byte and the twelve-byt command descriptor blocks contain 32-bit logical block addresses. Logical block addresses in additional paramete data have their length specified for each occurrence. See the specific command descriptions.

7.2.4 Transfer length

The transfer length field specifies the amount of data to be transferred, usually the number of blocks. For severa commands the transfer length indicates the requested number of bytes to be sent as defined in the commanc description. For these commands the transfer length field may be identified by a different name. See the followin descriptions and the individual command descriptions for further information.

Commands that use one byte for the transfer length allow up to 256 blocks of data to be transferred by on command. A transfer length value of 1 to 255 indicates the number of blocks that shall be transferred. A value o zero indicates 256 blocks.

In commands that use multiple bytes for the transfer length, a transfer length of zero indicates that no data transfe shall take place. A value of one or greater indicates the number of blocks that shall be transferred.

Refer to the specific command description for further information.

7.2.5 Parameter list length

The parameter list length is used to specify the number of bytes sent during the DATA OUT phase. This field i typically used in command descriptor blocks for parameters that are sent to a target (e.g. mode parameters diagnostic parameters, log parameters, etc.). A parameter length of zero indicates that no data shall be transferred This condition shall not be considered as an error.

7.2.6 Allocation length

The allocation length field specifies the maximum number of bytes that an initiator has allocated for returned data An allocation length of zero indicates that no data shall be transferred. This condition shall not be considered as ar error. The target shall terminate the DATA IN phase when allocation length bytes have been transferred or when a available data have been transferred to the initiator, whichever is less. The allocation length is used to limit th maximum amount of data (e.g. sense data, mode data, log data, diagnostic data, etc.) returned to an initiator.

7.2.7 Control field

The control field is the last byte of every command descriptor block. The control field is defined in table 25.

Table 25 - Control field

Bit	7	6	5	4	3	2	1	0
	Vendor-specific		Reserved				Flag	Link

The flag bit specifies which message the target shall return to the initiator if the link bit is one and the command completes without error. Implementation of the flag bit is optional.

The flag bit should be set to zero if the link bit is zero. If link bit is zero and the flag bit is one, the target shall return CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST.

If the flag bit is zero and the link bit is one, and if the command completes successfully, the target shall send the LINKED COMMAND COMPLETE message. If the flag bit is one and the link bit is one, and if the command completes successfully, the target shall send the LINKED COMMAND COMPLETE (WITH FLAG) message.

NOTE 46 The flag bit is typically used to cause an interrupt in the initiator between linked commands.

The link bit is used to continue the I/O process across multiple commands. Implementation of the link bit is optional.

A link bit of one indicates that the initiator requests a continuation of the I/O process and that the target should enter the command phase upon successful completion of the current command.

If the link bit is one, and if the command completes successfully, the target shall return INTERMEDIATE or INTERMEDIATE-CONDITION MET status and shall then send one of the two messages defined by the flag bit.

If either of the link and flag bits are set to one, and the target does not implement linked commands, it shall return CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST.

7.3 Status

The status byte and status byte code are defined in tables 26 and 27. A status byte shall be sent from the target to the initiator during the STATUS phase at the completion of each command unless the command is terminated by one of the following events:

a) an ABORT message;
b) an ABORT TAG message;
c) a BUS DEVICE RESET message;
d) a CLEAR QUEUE message;
e) a hard reset condition;
f) an unexpected disconnect (see 6.1.1).

The STATUS phase normally occurs at the end of a command but in some cases may occur prior to transferring the command descriptor block.

Table 26 - Status byte

Bit	7	6	5	4	3	2	1	0
	Reserved		Status byte code					Reserved

Table 27 - Status byte code

Bits of status byte								Status
7	6	5	4	3	2	1	0	
R	R	0	0	0	0	0	R	GOOD
R	R	0	0	0	0	1	R	CHECK CONDITION
R	R	0	0	0	1	0	R	CONDITION MET
R	R	0	0	1	0	0	R	BUSY
R	R	0	1	0	0	0	R	INTERMEDIATE
R	R	0	1	0	1	0	R	INTERMEDIATE-CONDITION MET
R	R	0	1	1	0	0	R	RESERVATION CONFLICT
R	R	1	0	0	0	1	R	COMMAND TERMINATED
R	R	1	0	1	0	0	R	QUEUE FULL
All other codes								Reserved
Key: R = Reserved bit								

A definition of the status byte codes is given below.

7.3.1 GOOD: This status indicates that the target has successfully completed the command.

7.3.2 CHECK CONDITION: This status indicates that a contingent allegiance condition has occurred (see 7.6).

7.3.3 CONDITION MET: This status or INTERMEDIATE-CONDITION MET is returned whenever the requested operation is satisfied (see the SEARCH DATA and PREFETCH commands).

7.3.4 BUSY: This status indicates that the target is busy. This status shall be returned whenever a target is unable to accept a command from an otherwise acceptable initiator (i.e. no reservation conflicts). The recommended initiator recovery action is to issue the command again at a later time.

7.3.5 INTERMEDIATE: This status or INTERMEDIATE-CONDITION MET shall be returned for every successfully completed command in a series of linked commands (except the last command), unless the command is terminated with CHECK CONDITION, RESERVATION CONFLICT, or COMMAND TERMINATED status. If INTERMEDIATE or INTERMEDIATE-CONDITION MET status is not returned, the series of linked commands is terminated and the I/O process is ended.

7.3.6 INTERMEDIATE-CONDITION MET: This status is the combination of the CONDITION MET and INTERMEDIATE statuses.

7.3.7 RESERVATION CONFLICT: This status shall be returned whenever an initiator attempts to access a logical unit or an extent within a logical unit that is reserved with a conflicting reservation type for another SCSI device (see the RESERVE and RESERVE UNIT commands). The recommended initiator recovery action is to issue the command again at a later time.

7.3.8 COMMAND TERMINATED: This status shall be returned whenever the target terminates the current I/O process after receiving a TERMINATE I/O PROCESS message (see 6.6.22). This status also indicates that a contingent allegiance condition has occurred (see 7.6).

7.3.9 QUEUE FULL: This status shall be implemented if tagged queuing is implemented. This status is returned when a SIMPLE QUEUE TAG, ORDERED QUEUE TAG, or HEAD OF QUEUE TAG message is received and the command queue is full. The I/O process is not placed in the command queue.

7.4 Command examples

The following subclauses give examples of typical command processing in the SCSI environment.

7.4.1 Single command example

An I/O process containing one untagged READ command is used in this clause to illustrate a simple I/O process on the SCSI bus. This example does not include error or exception conditions.

The initiator has one set of active pointers that includes a command pointer, a data pointer, and a status pointer. In addition, the initiator has one set of saved pointers for each I/O process that it is able to concurrently manage. The initiator sets up the saved pointers to point to the appropriate bytes for the I/O process and copies the saved pointers to the active pointers. It then arbitrates for the SCSI bus, and upon winning arbitration, selects the target. Once the target is selected, the target assumes control of the I/O process.

During the SELECTION phase, the initiator asserts the ATN signal to inform the target that the initiator wishes to send a message. The target enters the MESSAGE OUT phase and transfers the IDENTIFY message from the initiator. This message informs the target of which logical unit is to be used. At this point, an I_T_L nexus has been established for the I/O process. This nexus associates the initiator's pointers with the I/O process.

The target switches to the COMMAND phase and transfers the command descriptor block from the initiator. In this case, the command descriptor block contains a READ command. The target interprets the command and switches to the DATA IN phase, transfers the data, switches to STATUS phase, sends GOOD status, switches to MESSAGE IN phase, and transfers a COMMAND COMPLETE message. After successfully sending the COMMAND COMPLETE message, the target goes to the BUS FREE phase by releasing the BSY signal and the I/O process ends.

7.4.2 Disconnect example

In the above single command example, the length of time necessary to obtain the data may require a time-consuming physical positioning operation. In order to improve system throughput, the target may disconnect from the initiator, thereby freeing the SCSI bus to allow other I/O process to occur.

After the target has received the READ command (and has determined that there will be a delay), it disconnects from the SCSI bus by sending a DISCONNECT message and by going to the BUS FREE phase.

After the target retrieves the requested data from the peripheral device it arbitrates for the SCSI bus. Upon winning arbitration, it reselects the initiator and sends an IDENTIFY message to the initiator via the MESSAGE IN phase. This revives the I_T_L nexus so that the initiator can retrieve the correct set of pointers for the I/O process. The initiator restores the active pointers to their most recent saved values (which, in this case, are the initial values) and the target continues (as in the single command example) to finish the I/O process.

If target wishes to disconnect after transferring part of the data (e.g. while crossing a cylinder boundary), it may do so by sending a SAVE DATA POINTER message and a DISCONNECT message to the initiator and then disconnecting. When reconnection is completed, the current data pointer is restored to its value immediately prior to the SAVE DATA POINTER message.

On those occasions when an error or exception condition occurs and the target elects to repeat the information transfer, the target may repeat the transfer by either issuing a RESTORE POINTERS message or by disconnecting without issuing a SAVE DATA POINTER message. When reconnection is completed, the most recent saved pointer values are restored.

7.4.3 Linked command example

An I/O process may contain multiple commands linked together. Upon completing a linked command successfull the target automatically proceeds to the next linked command for the I/O process. All commands in a series of linke commands are addressed to the same nexus and are part of a single I/O process.

The commands are not entirely independent. When using the relative address bit (see 9.1.11), the address of th last logical block accessed by one of the commands is available to the next command. Thus one can search for particular data pattern using a SEARCH DATA command and then read the logical block containing the data patter with a READ command linked to the SEARCH DATA command. One can also read a logical block at a specifie displacement from the block containing the data pattern.

A LINKED COMMAND COMPLETE or LINKED COMMAND COMPLETE (WITH FLAG) message is sent from the targ to the initiator to indicate that a linked command completed. The initiator then updates the saved pointers for th nexus so that subsequent transfers from the target reference the next command of the series. Command processin of linked and single commands is similar except that relative addressing is permitted in linked commands.

For example, a successful completion of a SEARCH DATA EQUAL command causes the target to continue with th linked READ command from the initiator. If the relative address bit in the READ command has been set to one, an the address field of the READ command is set to zero, the target transfers the successfully searched block to th initiator.

7.5 Command processing considerations and exception conditions

The following subclauses describe some exception conditions and errors associated with command processing an the sequencing of commands.

7.5.1 Programmable operating definition

Some applications require that the operating definition of a logical unit be modified to meet the special requirement of a particular initiator. The program-controlled modification of the operating definition is typically provided to allo operating systems to change the operating definition of a more recently developed targets to one that is mor compatible with the operating system. This ability requires that the system comply with the low-level hardwar definitions of SCSI-2.

The parameters that can be changed by modifying the operating definition of a logical unit include the vend identification, the device type, the device model, the SCSI compliance level, the SCSI specification level, th command set, and other parameters. The low-level hardware parameters including signal timing and parity definition cannot be changed by modifying the operating definition. The present operating definition of a logical unit wit respect to an initiator can be determined at any time by execution of an INQUIRY command. In some vendor-specifi cases, it may also be necessary to perform other commands including MODE SENSE and READ CAPACITY.

Each logical unit begins with a particular operating definition. If the logical unit supports the CHANGE DEFINITIO command, the present operating definition can be changed to any other operating definition supported by the logic unit. The actual details of the operating definition of a logical unit are vendor-specific. If the operating definition changed to one that does not include the CHANGE DEFINITION command, the target should continue to accept th CHANGE DEFINITION command.

If an error occurs during execution of a CHANGE DEFINITION command, the original operating definition remair in effect after the command is executed. The new operating definition becomes active only after successful executio of the CHANGE DEFINITION command.

Since new operating definitions may preclude the execution of I/O processes that are already in progress, the targ may disconnect to allow completion of any I/O processes that are in progress. Operating definition changes that ma cause conflicts with the normal operation from other initiators shall be indicated to those initiators by generating

unit attention condition for each other initiator. The additional sense code shall be set to CHANGED OPERATING DEFINITION.

An initiator may request a list of the operating definitions that the target supports and descriptive text for each operating definition using the INQUIRY command.

7.5.2 Incorrect initiator connection

An incorrect initiator connection occurs on a reconnection if:

a) an initiator attempts to reconnect to an I/O process, and
b) a soft reset condition has not occurred, and
c) the initiator does not send an ABORT, ABORT TAG, BUS DEVICE RESET, CLEAR QUEUE, or TERMINATE I/O PROCESS message during the same MESSAGE OUT phase as the IDENTIFY message.

An incorrect initiator connection also occurs on an initial connection when an initiator:

a) attempts to establish an I_T_L_Q nexus when an I_T_L nexus already exists from a previous connection, or
b) attempts to establish an I_T_L nexus when an I_T_L_Q nexus already exists, unless there is a contingent allegiance or extended contingent allegiance condition present for the logical unit.

A target that detects an incorrect initiator connection shall abort all I/O processes for the initiator on the logical unit or target routine and shall return CHECK CONDITION status. The sense key shall be set to ABORTED COMMAND and the additional sense code shall be set to OVERLAPPED COMMANDS ATTEMPTED.

If an initiator reconnects to an I/O process and a soft reset condition has occurred, the target shall meet the requirements of 6.2.2.2.

NOTES
47 An incorrect initiator connection may be indicative of a serious error and, if not detected, could result in an I/O process operating with a wrong set of pointers. This is considered a catastrophic failure on the part of the initiator. Therefore, vendor-specific error recovery procedures may be required to guarantee the data integrity on the medium. The target may return additional sense data to aid in this error recovery procedure (e.g. sequential-access devices may return the residue of blocks remaining to be written or read at the time the second command was received).
48 Some targets may not detect an incorrect initiator connection until after the command descriptor block has been received.

7.5.3 Selection of an invalid logical unit

The target's response to selection of a logical unit that is not valid is described in the following paragraphs.

The logical unit may not be valid because:

a) the target does not support the logical unit (e.g. some targets support only one peripheral device). In response to an INQUIRY command, the target shall return the INQUIRY data with the peripheral qualifier set to the value required in 8.2.5.1. In response to any other command except REQUEST SENSE, the target shall terminate the command with CHECK CONDITION status. In response to a REQUEST SENSE command, the target shall return sense data. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to LOGICAL UNIT NOT SUPPORTED.
b) the target supports the logical unit, but the peripheral device is not currently attached to the target. In response to an INQUIRY command, the target shall return the INQUIRY data with the peripheral qualifier set to the value required in 8.2.5.1. In response to any other command except REQUEST SENSE, the target shall terminate the command with CHECK CONDITION status. In response to a REQUEST SENSE command, the target shall return sense data. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to LOGICAL UNIT NOT SUPPORTED.
c) the target supports the logical unit and the peripheral device is attached, but not operational. In response to an INQUIRY command, the target shall return the INQUIRY data with the peripheral qualifier set to the value required in 8.2.5.1. The target's response to any command other than INQUIRY and REQUEST SENSE is vendor-specific.

d) the target supports the logical unit but is incapable of determining if the peripheral device is attached or is no operational when it is not ready. In response to an INQUIRY command,the target shall return the INQUIRY data with the peripheral qualifier set to the value required in 8.2.5.1. In response to a REQUEST SENSE command the target shall return the REQUEST SENSE data with a sense key of NO SENSE unless a contingen allegiance exists. The target's response to any other command is vendor-specific.

7.5.4 Parameter rounding

Certain parameters sent to a target with various commands contain a range of values. Targets may choose to implement only selected values from this range. When the target receives a value that it does not support, it eithe rejects the command (CHECK CONDITION status with ILLEGAL REQUEST sense key) or it rounds the value received to a supported value. The target shall reject unsupported values unless rounding is permitted in the description o the parameter.

Rounding of parameter values, when permitted, shall be performed as follows - A target that receives a paramete value that is not an exact supported value shall adjust the value to one that it supports and shall return CHECK CONDITION status with a sense key of RECOVERED ERROR. The additional sense code shall be set to ROUNDED PARAMETER. The initiator is responsible to issue an appropriate command to learn what value the target has selected.

NOTE 49 Generally, the target should adjust maximum-value fields down to the next lower supported value than the one specified by the initiator. Minimum-value fields should be rounded up to the next higher supported value than the one specified by the initiator. In some cases, the type of rounding (up or down) is explicitly specified in the description of the parameter.

7.5.5 Asynchronous event notification

Implementation of asynchronous event notification is optional. This protocol can be used to inform processor devices that an asynchronous event has occurred. A SEND command with an AEN bit of one is issued to a processor device with a subsequent data phase that includes the REQUEST SENSE information. SCSI devices that respond to an INQUIRY command as a processor device type with asynchronous event notification capability may be notified o asynchronous events using this protocol. An SCSI device has to be capable of acting as an initiator in order to perform asynchronous event notification.

NOTE 50 Asynchronous event notification cannot be used with a device that acts as a temporary initiator (e.g. devices executing COPY commands) since they are not identified as a processor device.

Parameters affecting the use of asynchronous event notification are contained in the control mode page (see 8.3.3.1)

The four uses of asynchronous event notification are:

a) informing a processor of an error condition encountered after command completion;
b) informing all processor devices that a newly initialized device is available;
c) informing all processor devices of other unit attention conditions;
d) informing all processor devices of other asynchronous events.

An example of the first case above is a device that implements a write cache. If the target is unable to write cached data to the medium, it may use an asynchronous event notification to inform the initiator of the failure. An extended contingent allegiance condition may also be established during the same I_T_L nexus used for the asynchronous event notification (see 6.6.9).

An example of the third case above is a device that supports removable media. Asynchronous event notification may be used to inform an initiator of a not-ready-to-ready transition (medium changed) or of an operator initiated even (e.g. activating a write protect switch or activating a start or stop switch).

An example of the fourth case above is a sequential-access device performing a REWIND command with the immediate bit set to one. Asynchronous event notification may be used to inform an initiator that the beginning c

medium has been reached. Completion of a CD-ROM AUDIO PLAY command started in the immediate mode is another example of this case.

Notification of an asynchronous event is performed using the SEND command with the AEN bit set to one. The information identifying the condition being reported shall be returned during the DATA OUT phase of the SEND command. See 12.2.2 for further information on the format of the data sent.

An error condition or unit attention condition shall be reported once per occurrence of the event causing it. The target may choose to use an asynchronous event notification or to return CHECK CONDITION status on a subsequent command, but not both. Notification of command-related error conditions shall be sent only to the processor that initiated the I/O process.

The asynchronous event notification protocol can be used to notify processor devices that a system resource has become available. If a target chooses to use this method, the sense key in the sense data sent to the processor device shall be set to UNIT ATTENTION.

The asynchronous event notification protocol shall be sent only to SCSI devices that return processor device type with an AENC bit of one in response to an INQUIRY command. The INQUIRY command should be issued to logical unit zero of each SCSI device responding to selection. This procedure shall be conducted prior to the first asynchronous event notification and shall be repeated whenever the device deems it appropriate or when an event occurs that may invalidate the current information. (See 6.6.21, SYNCHRONOUS DATA TRANSFER REQUEST message, for examples of these events.)

Each SCSI device that returns processor device type with an AENC bit of one shall be issued a TEST UNIT READY command to determine that the SCSI device is ready to receive an asynchronous event notification. An SCSI device returning CHECK CONDITION status is issued a REQUEST SENSE command. This clears any pending unit attention condition. An SCSI device that returns processor device type with an AENC bit of one and returns GOOD status when issued a TEST UNIT READY command shall accept a SEND command with an AEN bit of one.

NOTE 51 An SCSI device which can use asynchronous event notification at initialization time should provide means to defeat these notifications. This can be done with a switch or jumper wire. Devices that implement saved parameters may alternatively save the asynchronous event notification permissions either on a per SCSI device basis or as a system-wide option. In any case, a device conducts a survey with INQUIRY commands to be sure that the devices on the SCSI bus are appropriate destinations for SEND commands with an AEN bit of one. (The devices on the bus or the SCSI ID assignments may have changed.)

7.5.6 Unexpected reselection

An unexpected reselection occurs if an SCSI device attempts to reconnect to an I/O process for which a nexus does not exist. An SCSI device should respond to an unexpected reselection by sending an ABORT message.

7.6 Contingent allegiance condition

The contingent allegiance condition shall exist following the return of CHECK CONDITION or COMMAND TERMINATED status. The contingent allegiance condition shall be preserved for the I_T_x nexus until it is cleared. The contingent allegiance condition shall be cleared upon the generation of a hard reset condition, or by an ABORT message, a BUS DEVICE RESET message, or any subsequent command for the I_T_x nexus. While the contingent allegiance condition exists the target shall preserve the sense data for the initiator.

While the contingent allegiance condition exists, if the target is unable to maintain separate sense data, the target shall respond to any other requests for access to the logical unit or target routine from another initiator with a BUSY status. Execution of all tagged I/O processes for the I_T_L nexus for which the contingent allegiance condition exists shall be suspended until the contingent allegiance condition is cleared.

7.7 Extended contingent allegiance condition

Implementation of extended contingent allegiance is optional. The extended contingent allegiance condition extends the contingent allegiance condition for an I_T_x nexus. This condition is generated by the target sending an INITIATE RECOVERY message to the initiator following CHECK CONDITION or COMMAND TERMINATED status and prior to the COMMAND COMPLETE message. This condition shall be preserved until it is cleared by a BUS DEVICE RESET message, a RELEASE RECOVERY message, or a hard reset condition.

While the extended contingent allegiance condition exists, the target shall respond to any other request for access to the logical unit from another initiator with BUSY status. Execution of all I/O processes for the logical unit for which the extended contingent allegiance condition exists shall be suspended until the RELEASE RECOVERY message is received by the target.

NOTES
52 It is not required to generate an extended contingent allegiance condition for every CHECK CONDITION or COMMAND TERMINATED status that occurs. Simple errors not requiring an extended recovery may be dealt with by using the contingent allegiance protocol.
53 During the existence of the extended contingent allegiance condition, appropriate error recovery sequences may be executed. Such commands can correct data, modify or delete queued commands, perform LOG SENSE commands and obtain diagnostic information. Extended contingent allegiance is recommended for error conditions that may require execution of multiple-step error-recovery protocols without interference from other initiators.

An extended contingent allegiance condition may also be generated using an asynchronous event notification protocol. When the event is detected, the bus device that detected the event assumes the initiator role and transmits a SEND command with an AEN bit of one to the appropriate device(s) (see 12.2.2).

If the device wishes to generate an extended contingent allegiance condition during an asynchronous event notification, it shall send an INITIATE RECOVERY message following the IDENTIFY message (and following any queue tag message) and prior to the COMMAND phase of the SEND command. An extended contingent allegiance condition can be generated for only one I_T_L nexus at a time. The extended contingent allegiance condition is cleared when a RELEASE RECOVERY message is received from the device to which the INITIATE RECOVERY message was sent. The generation of a hard reset condition, or receipt of a BUS DEVICE RESET message, shall also clear the extended contingent allegiance condition.

During an extended contingent allegiance, only untagged I/O processes from the SCSI device to which the INITIATE RECOVERY message was sent shall be executed by the target for the logical unit. If the initiator sends a queue tag message, the target shall respond with QUEUE FULL status. After the extended contingent allegiance condition is cleared, any commands remaining in the command queue shall be executed.

7.8 Queued I/O processes

The implementation of queuing for I/O processes is optional. Queuing of I/O processes allows a target to accept multiple I/O processes.

There are two methods for implementation of queuing, tagged and untagged. Tagged queuing allows a target to accept multiple I/O processes from each initiator for each logical unit. Untagged queuing allows a target to accept one I/O process from each initiator for each logical unit or target routine. Untagged queuing may be supported by SCSI-1 or SCSI-2 devices. Tagged queuing is new in SCSI-2.

A target may support both tagged and untagged queuing. An initiator may not mix the use of tagged and untagged queuing for I/O processes to a logical unit, except during a contingent allegiance or extended contingent allegiance condition when only untagged initial connections are allowed. An initiator that elects to use tagged queuing does not preclude another initiator on the same SCSI bus from using untagged queuing.

7.8.1 Untagged queuing

Untagged queuing allows a target to accept a command from an initiator for a logical unit or target routine while I/O processes from other initiators are being executed. Only one command for each I_T_x nexus shall be accepted at a time.

An I/O process may be initiated any time the BUS FREE phase exists, even if an I/O process from a different initiator is active. If the disconnect privilege is not granted in the IDENTIFY message of the current I/O process, the target may either suspend all other I/O processes or it may return BUSY status to the current I/O process.

The I_T_x nexus sufficiently specifies the relationship so that the target can reconnect to the initiator to restore the pointers for I/O process as long as only one I/O process per I_T_x nexus is issued. It is the responsibility of the initiator to assure that only one such I/O process is issued at any time (see 7.5.2).

7.8.2 Tagged queuing

Tagged queuing allows a target to accept multiple I/O processes from the same or different initiators until the logical unit's command queue is full. If an I/O process is received and the command queue is full, the target shall terminate the I/O process with QUEUE FULL status.

The command queue is setup by the target for each supported logical unit. Initiators may add or delete I/O processes to the queue. When adding an I/O process, the initiator may specify fixed order of execution, allow the target to define the order of execution, or specify that the I/O process is to be executed next.

If the disconnect privilege is not granted in the IDENTIFY message for a tagged I/O process, the target shall return BUSY status.

The queue tag messages (see 6.6.17) allow the initiator to establish a unique I_T_L_Q nexus to identify each I/O process. The I_T_L_Q nexus allows the target to reconnect to a specific I/O process, and allows the initiator to restore the set of pointers for that I/O process. An initiator may have several I/O processes ongoing to the same or different logical units as long as each has a unique nexus.

If only SIMPLE QUEUE TAG messages are used, the target may execute the commands in any order that is deemed desirable within the constraints of the queue management algorithm specified in the control mode page (see 8.3.3.1).

If ORDERED QUEUE TAG messages are used, the target shall execute the commands in the order received with respect to other commands received with ORDERED QUEUE TAG messages. All commands received with a SIMPLE QUEUE TAG message prior to a command received with an ORDERED QUEUE TAG message, regardless of initiator, shall be executed before that command with the ORDERED QUEUE TAG message. All commands received with a SIMPLE QUEUE TAG message after a command received with an ORDERED QUEUE TAG message, regardless of initiator, shall be executed after that command with the ORDERED QUEUE TAG message.

A command received with a HEAD OF QUEUE TAG message is placed first in the queue, to be executed next. A command received with a HEAD OF QUEUE TAG message shall be executed prior to any queued I/O process Consecutive commands received with HEAD OF QUEUE TAG messages are executed in a last-in-first-out order.

An I/O process received without a queue tag message, while there are any tagged I/O commands in the command queue from the same initiator, is an incorrect initiator connection (see 7.5.2), unless there is a contingent allegiance or extended contingent allegiance condition.

A series of linked commands constitute a single I/O process. These commands are assigned the queue tag established in the initial connection. A command received with a HEAD OF QUEUE TAG message shall not suspend a series of linked commands for which the target has begun execution.

The RESERVE and RELEASE commands should be sent with an ORDERED QUEUE TAG message. Use of the HEAD OF QUEUE TAG message with these commands could result in reservation conflicts with previously issued commands.

NOTE 54 Initiators should not issue another I/O processes following a reservation request until that request is honoured by the target. This prevents possible sequencing errors or file system corruption.

There are two methods of dealing with the command queue following a contingent allegiance condition. The method used is specified in the control mode page by the queue error management bit (see 8.3.3.1).

The first method allows the execution of tagged I/O processes to resume when the contingent allegiance or extended contingent allegiance condition is cleared. The target returns BUSY status to other initiators while the contingent allegiance or extended contingent allegiance condition exists. During this time, all tagged I/O processes are suspended. All I/O processes used for recovery operations shall be untagged. The queue may be modified by removing all or selected I/O processes from the queue as part of the recovery procedure.

If commands are combined by the queuing algorithm such that the exception condition affects more than one command, then a contingent allegiance condition shall be generated for all affected commands.

The second method aborts all I/O processes when the contingent allegiance or extended contingent allegiance condition is cleared. A unit attention condition shall be generated for all other initiators and the additional sense code shall be set to COMMANDS CLEARED BY ANOTHER INITIATOR.

A device that does not support tagged I/O processes (e.g. not implemented, disabled by the Dque bit in the control mode page, or switched to an operating definition that does not include tagged I/O processes) it shall respond to any queue tag message with a MESSAGE REJECT message. The target shall continue the I/O process as if it were an untagged I/O process.

Tagged queuing may also be temporarily disabled during certain initialization periods or to control internal resource utilization. During these periods the target may return QUEUE FULL status or it may respond to any queue tag message with a MESSAGE REJECT message.

Several messages may be used to clear part or all of the command queue. Please refer to the ABORT, ABORT TAG, BUS DEVICE RESET, and CLEAR QUEUE messages in clause 6 for details.

7.8.3 Example of queued I/O process

This example of queuing I/O processes illustrates the execution of a number of commands. After each command, the state of the queue kept in the target is shown to indicate the function actually performed by the queuing.

7.8.3.1 Typical sequences for tagged queuing

An I/O process using tagged queuing uses the following sequences for normal execution. The initiator first arbitrates for the SCSI bus, and after successfully obtaining the SCSI bus, selects the appropriate SCSI device. The ATN signal is asserted during the SELECTION phase to indicate that a MESSAGE OUT phase is requested by the initiator. The first message byte transferred is an IDENTIFY message. The ATN signal continues to be asserted during the MESSAGE OUT phase to indicate that the initiator has another message. The second message byte transferred is the first byte of the appropriate queue tag message, in this case a SIMPLE QUEUE TAG message. The third and last message byte is transmitted containing the second byte of the queue tag message, the queue tag. As it is transferred, the ATN signal is negated to indicate that no more message bytes are available. The target then transfers the command descriptor block. Assuming the command requires disconnection, the target transmits a DISCONNECT message to the initiator and then enters the BUS FREE phase. The target places the command, identified by the I_T_L_Q nexus, at the appropriate place in the command queue.

When the target removes I/O processes from the queue for execution, a physical latency period may occur. At the end of this period, when the target is prepared to transfer the appropriate data, the target begins an ARBITRATION phase and, upon winning, enters a RESELECTION phase. After a successful reselection, the target sends the IDENTIFY message followed by a SIMPLE QUEUE TAG message with the queue tag value originally sent by the initiator. The initiator uses the I_T_L_Q nexus to identify the correct set of pointers and control blocks associated with the I/O process and to establish the necessary conditions for data transfer. The target begins data transfer. When the data transfer is successfully completed, the target returns GOOD status and terminates the I/O process with a COMMAND COMPLETE message.

7.8.3.2 Example of tagged queuing

An example of the execution of five queued I/O processes is described to demonstrate how tagged queuing operates. All tagged I/O processes are from one initiator to a single logical unit of a single target. The five I/O processes are defined in table 28. The target is a direct-access device. At the time the I/O processes are first being executed, it is assumed that the actuator is in position to access logical block 10 000.

Table 28 - Commands in order received by target

Command	Queue tag message	Queue tag value	Logical block address	Transfer length	Status
READ	SIMPLE	01h	10 000	1 000	Queued
READ	SIMPLE	02h	100	1	Queued
READ	ORDERED	03h	1 000	1 000	Queued
READ	SIMPLE	04h	10 000	1	Queued
READ	SIMPLE	05h	2 000	1 000	Queued

The optimum order would require that those blocks close to the actuator position be the first blocks accessed, followed by those increasingly far from the actuator position. However, the command with queue tag 03h is an ordered I/O process, so that all simple I/O processes transferred previously must be executed before, while all simple I/O processes transferred after the ordered I/O process must be executed after the ordered I/O process.

If a target supports an optimizing algorithm the actual order in which the I/O processes are executed could be as shown in table 29.

Table 29 - Commands in order of execution

Command	Queue tag message	Queue tag value	Logical block address	Transfer length	Status
READ	SIMPLE	01h	10 000	1 000	Queued
READ	SIMPLE	02h	100	1	Queued
READ	ORDERED	03h	1 000	1 000	Queued
READ	SIMPLE	05h	2 000	1 000	Queued
READ	SIMPLE	04h	10 000	1	Queued

I/O processes with queue tag values 01h and 02h are executed in the order received since the actuator is already in position to execute I/O process 01h. I/O process 02h must be executed before I/O process 04h or 05h because the ordered I/O process 03h was transmitted after I/O processes 01h and 02h but before I/O processes 04h and 05h. I/O process 03h is then executed after I/O process 02h. The I/O processes 04h and 05h are executed after the ordered I/O process 03h. I/O process 05h is executed before I/O process 04h because the actuator is in position to access block 2 000 after executing I/O process 03h. I/O process 04h is executed last.

As an example of the operation of the HEAD OF QUEUE TAG I/O process, consider that a new I/O process, identified by a HEAD OF QUEUE TAG message with a queue tag of 08h, is transmitted to the target while the ordered I/O process 03h is being executed. The I/O process 03h continues execution, but the new HEAD OF QUEUE TAG I/O process is placed in the queue for execution before all subsequent I/O processes. In this case, the queue for execution after the ordered I/O process 03h was executed would appear as shown in table 30.

Table 30 - Modified by HEAD OF QUEUE TAG message

Command	Queue tag message	Queue tag value	Logical block address	Transfer length	Status
READ	ORDERED	03h	1 000	1 000	Executing
READ	HEAD OF QUEUE	08h	0	8	Queued
READ	SIMPLE	05h	2 000	1 000	Queued
READ	SIMPLE	04h	10 000	1	Queued

To obtain maximum performance gains using tagged queuing requires careful implementation of the queuing algorithms in the target. In addition, initiators should allow a maximum number of simple I/O processes to be executed with a minimum number of ordered I/O processes. RESERVE and RELEASE commands, SET LIMITS commands, and appropriate software locking conventions should be used to guarantee the proper relationship between the commands executed and the data stored on the peripheral devices. These conventions are not defined by this standard.

7.9 Unit attention condition

The target shall generate a unit attention condition for each initiator on each valid logical unit whenever the target has been reset by a BUS DEVICE RESET message, a hard reset condition, or by a power-on reset. The target shall also generate a unit attention condition on the affected logical unit(s) for each initiator whenever one of the following events occurs:

a) A removable medium may have been changed;
b) The mode parameters in effect for this initiator have been changed by another initiator;
c) The version or level of microcode has been changed;
d) Tagged commands queued for this initiator were cleared by another initiator;
e) INQUIRY data has been changed;
f) The mode parameters in effect for the initiator have been restored from non-volatile memory;
g) A change in the condition of a synchronized spindle;
h) Any other event occurs that requires the attention of the initiator.

NOTES
56 Targets may queue unit attention conditions on logical units. After the first unit attention condition is cleared, another unit attention condition may exist (e.g. a power on condition followed by a microcode change condition).
57 See 7.5.3 for requirements concerning selection of an invalid logical unit.

The unit attention condition shall persist on the logical unit for each initiator until that initiator clears the condition as described in the following paragraphs.

If an INQUIRY command is received from an initiator to a logical unit with a pending unit attention condition (before the target generates the contingent allegiance condition), the target shall perform the INQUIRY command and shall not clear the unit attention condition. If the INQUIRY command is received after the target has generated the contingent allegiance condition for a pending unit attention condition, then the unit attention condition on the logical unit shall be cleared, and the target shall perform the INQUIRY command.

If any other command is received after the target has generated the contingent allegiance condition for a pending unit attention condition, the unit attention condition on the logical unit shall be cleared, and if no other unit attention condition is pending the target shall perform the command. If another unit attention condition is pending, the target shall not perform the command and shall generate another contingent allegiance condition.

If a REQUEST SENSE command is received from an initiator with a pending unit attention condition (before the target generates the contingent allegiance condition), then the target shall either:

a) report any pending sense data and preserve the unit attention condition on the logical unit, or,
b) report the unit attention condition, may discard any pending sense data, and clear the unit attention condition on the logical unit for that initiator.

If the target has already generated the contingent allegiance condition for the unit attention condition, the target shall perform the second action listed above.

If an initiator issues a command other than INQUIRY or REQUEST SENSE while a unit attention condition exists for that initiator (prior to generating the contingent allegiance condition for the unit attention condition), the target shall not perform the command and shall report CHECK CONDITION status unless a higher priority status as defined by the target is also pending (e.g. BUSY or RESERVATION CONFLICT).

If, after generating the contingent allegiance condition for a pending unit attention condition, the next command received from that initiator on the logical unit is not REQUEST SENSE, then that command shall be performed and the unit attention condition shall be cleared for that initiator on the logical unit and the sense data is lost (see 7.6).

If a target becomes a temporary initiator to issue a SEND command with an AEN bit of one, which informs the initiator (temporary target) of the unit attention condition, and the SEND command completes with GOOD status, then the target shall clear the unit attention condition for that initiator on the logical unit (see 7.5.5).

8 All device types

8.1 Model for all device types

This model describes some of the general characteristics expected of most SCSI devices. It is not intended to define any requirements nor is it intended to alter any requirements defined elsewhere in this standard. Clause 7 also contains model information pertaining to all device types.

8.1.1 SCSI addresses

There are two levels of addresses within the SCSI architecture: the SCSI device address and the logical unit number or target routine number.

8.1.1.1 SCSI device address

SCSI devices occupy (i.e. respond to) one address on the SCSI bus. Generally, the SCSI device provides a means (e.g. switches, jumpers) to select one of the eight available addresses (0 through 7). This address is used during bus arbitration and selection or reselection of SCSI devices. Each device on the SCSI bus is assigned a unique address.

Normally, the SCSI device address is set when the system is configured and it remains static thereafter. Some systems and devices provide vendor-specific means to alter this address at other times.

8.1.1.2 Logical units

Each target has one or more logical units, beginning with logical unit zero. There is a maximum of eight logical units. These logical units are usually mapped directly to peripheral devices, but they may be a portion of a peripheral device or may comprise multiple peripheral devices.

An initiator can determine whether a target implements a logical unit by issuing an INQUIRY command and examining the returned peripheral qualifier and peripheral device type.

The concept of a logical unit is not defined for an initiator. (An SCSI device may implement both the initiator role and the target role. In this case logical unit(s) are defined only for the target role.)

8.1.1.3 Target routines

An optional feature of the SCSI architecture permits each target to have one or more target routines, beginning with target routine number zero. There is a maximum of eight target routines. These target routines are processes that execute directly on the target and are not associated with a particular logical unit or peripheral device. Target routines are addressed using the LUNTAR bit of the IDENTIFY message (see 6.6.7).

Target routines are principally intended to return information about the target and the only valid commands are INQUIRY and REQUEST SENSE.

8.1.2 Commands implemented by all SCSI devices

This standard defines four commands that all SCSI-2 targets implement - INQUIRY, REQUEST SENSE, SEND DIAGNOSTIC, and TEST UNIT READY. These commands are used to configure the system, to test targets, and to return important information concerning errors and exception conditions.

8.1.2.1 Using the INQUIRY command

The INQUIRY command may be used by a system to determine the configuration of the SCSI bus. Target devices respond with information that includes their type and standard level and may include the vendor's identification, model number and other useful information. It is recommended that SCSI targets be capable of returning this information (or whatever part of it that is available) upon completing power-on initialization. An SCSI device may take longer to get certain portions of this information, especially if it retrieves the information from the medium.

8.1.2.2 Using the REQUEST SENSE command

Whenever a contingent allegiance condition (see 7.6) is established, the initiator that received the error should issue a REQUEST SENSE command to receive the sense data describing what caused the contingent allegiance condition. If the initiator issues some other command, the sense data is lost.

8.1.2.3 Using the SEND DIAGNOSTIC command

The SEND DIAGNOSTIC command provides a means to request the target to perform a self test. While the test is target specific, the means of requesting the test is standardized and the response is simply GOOD status if all is well or CHECK CONDITION status if the test fails.

The SEND DIAGNOSTIC command also provides other powerful features when used in conjunction with the RECEIVE DIAGNOSTIC RESULTS command, but this capability is optional.

8.1.2.4 Using the TEST UNIT READY command

The TEST UNIT READY command is useful in that it allows an initiator to poll a logical unit until it is ready without the need to allocate space for returned data. It is especially useful to check cartridge status of logical units with removable media. Targets are expected to respond promptly to indicate the current status of the device (i.e. a target should avoid lengthy disconnections in an attempt to respond with GOOD status).

8.2 Commands for all device types

The operation codes for commands that apply to all device types are listed in table 31.

Table 31 - Commands for all device types

Command name	Operation code	Type	Subclause
CHANGE DEFINITION	40h	O	8.2.1
COMPARE	39h	O	8.2.2
COPY	18h	O	8.2.3
COPY AND VERIFY	3Ah	O	8.2.4
INQUIRY	12h	M	8.2.5
LOG SELECT	4Ch	O	8.2.6
LOG SENSE	4Dh	O	8.2.7
MODE SELECT(6)	15h	Z	8.2.8
MODE SELECT(10)	55h	Z	8.2.9
MODE SENSE(6)	1Ah	Z	8.2.10
MODE SENSE(10)	5Ah	Z	8.2.11
READ BUFFER	3Ch	O	8.2.12
RECEIVE DIAGNOSTIC RESULTS	1Ch	O	8.2.13
REQUEST SENSE	03h	M	8.2.14
SEND DIAGNOSTIC	1Dh	M	8.2.15
TEST UNIT READY	00h	M	8.2.16
WRITE BUFFER	3Bh	O	8.2.17

Key: M = Command implementation is mandatory.
O = Command implementation is optional.
Z = Command implementation is device type specific.

8.2.1 CHANGE DEFINITION command

The CHANGE DEFINITION command (see table 32) modifies the operating definition of the selected logical unit o target with respect to commands from the selecting initiator or for all initiators.

Table 32 - CHANGE DEFINITION command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (40h)							
1	Logical unit number			Reserved				
2	Reserved							Save
3	Reserved	Definition parameter						
4	Reserved							
5	Reserved							
6	Reserved							
7	Reserved							
8	Parameter data length							
9	Control							

A save control bit (Save) of zero indicates that the target shall not save the operating definition. A Save bit of on indicates that the target shall save the operating definition to non-volatile memory.

The definition parameter field is defined in table 33.

Table 33 - Definition parameter field

Value	Meaning of definition parameter
00h	Use current operating definition
01h	SCSI-1 operating definition
02h	CCS operating definition
03h	SCSI-2 operating definition
04 - 3Fh	Reserved
40 - 7Fh	Vendor-specific

NOTE 57 The current operating definition parameter values establish operating definitions compatible with the appropriate SCSI specification. Vendor-specific values are available for those applications where more complex operation definition changes are required. Definitions supported by a device are returned in the implemented operating definition page (see 8.3.4.3).

The parameter data length field specifies the length in bytes of the parameter data that shall be transferred from th initiator to the target. A parameter data length of zero indicates that no data shall be transferred. This condition sha not be considered as an error. Parameter data lengths greater than zero indicate the number of bytes of paramete data that shall be transferred.

The parameter data is vendor-specific.

NOTE 58 The parameter data may be used to specify a password to validate an operating definition change.

The CHANGE DEFINITION command causes one of the operating definition modifications listed below:

a) Change the operating definition of a logical unit relative to the initiator that issued the command. In this case, the target is capable of maintaining an unique operating definition for each logical unit relative to each initiator in the system.
b) Change the operating definition of the target relative to the initiator that issued the command. In this case, the target is capable of maintaining a unique operating definition, for each initiator in the system, that applies to all logical units of the target.
c) The operating definition of a logical unit relative to all initiators in the system. In this case, the target is capable of maintaining an unique operating definition for each logical unit relative to all initiators in the system.
d) The operating definition of the target relative to all initiators in the system. In this case, the target is capable of maintaining only one operating definition.

NOTES
59 This standard does not provide a direct means to determine which of the above four methods has been implemented by the target. An indirect means of determining which method is implemented exists in that the target is required to inform affected initiators of operating definition changes via the unit attention condition.
60 The final two modifications listed above may result in incompatibilities if other initiators are operated below the SCSI-2 level.

The operating definition is modified after successful completion of the command. A target shall consider the command successfully completed when it detects the assertion of the ACK signal for the COMMAND COMPLETE message. The initiator should verify the new operating definition by issuing an INQUIRY command requesting the implemented operating definition page (see 8.3.4.1).

It is permissible for an SCSI-2 device that has its definition changed to an SCSI-1 device to accept a CHANGE DEFINITION command.

NOTE 61 The method of changing the operating definition is implementation dependent. Some implementations may require that the target's operating mode be reinitialized as if a power-up or hard-reset had occurred. Other implementations may modify only those operating definitions that are affected by the CHANGE DEFINITION command.

If the CHANGE DEFINITION command is not executed successfully for any reason, the operating definition shall remain the same as it was before the CHANGE DEFINITION command was attempted. If it is impossible to return to the previous operating definition, a unit attention condition shall be generated by the target.

NOTE 62 The present operating definition of the target may always be interrogated through the INQUIRY command. When an SCSI-2 target has its operating definition changed to CCS or SCSI-1, certain changes are needed to promote compatibility with preexisting SCSI-1 initiators. The recommended changes are as follows:
a) The target should not initiate selections to other SCSI devices to determine if any initiators support AEN. The target should assume that none are capable of receiving AEN and not issue an AEN.
b) The target should not generate extended contingent allegiance conditions by issuing an INITIATE RECOVERY message.
c) If a REQUEST SENSE command with an allocation length of zero is received, the target should return four bytes of sense data.
d) If an INQUIRY command is received, the returned data should have appropriate values in the ANSI-approved version field and the response data format field. The features supported bits should be zero.
e) A change in the operating definition may change the vendor identifier, the device type, the device model, the SCSI implementation level, the command set, mode pages, and any other operating characteristics.

After a power-on condition or a hard RESET condition, the target shall set its initial operating definition to the last saved value, if saving is implemented, or its default value, if saving is not implemented.

8.2.2 COMPARE command

The COMPARE command (see table 34) provides the means to compare data from one logical unit with another o the same logical unit in a manner similar to the COPY command.

Table 34 - COMPARE command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (39h)							
1	Logical unit number			Reserved				Pad
2	Reserved							
3	(MSB)							
4	Parameter list length							
5								(LSB)
6	Reserved							
7	Reserved							
8	Reserved							
9	Control							

This command functions in the same manner as the COPY command, except that the data from the source is compared on a byte-by-byte basis with the data from the destination. The parameter list transferred to the target is the same as for the COPY command. This parameter list contains the information to identify the logical units involvec in the comparison and the length of the comparison. (See 8.2.3 for additional information about the COPY command.)

If the comparison is unsuccessful, the command shall be terminated with CHECK CONDITION status and the sense key shall be set to MISCOMPARE. The remaining fields in the sense data shall be set as documented in the COPY command.

8.2.3 COPY command

The COPY command (see table 35) provides a means to copy data from one logical unit to another or the same logical unit. The logical unit that receives and performs the COPY command is called the copy manager. The copy manager is responsible for copying data from a logical unit (source device) to a logical unit (destination device). These logical units may reside on different SCSI devices or the same SCSI device (in fact all three may be the same logical unit). Some SCSI devices that implement this command may not support copies to or from another SCSI device, or may not support third party copies (i.e. both the source and the destination logical units reside on other SCSI devices).

Table 35 - COPY command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (18h)							
1	Logical unit number			Reserved				Pad
2	(MSB)							
3	Parameter list length							
4								(LSB)
5	Control							

The Pad bit (8.2.3.7) is used in conjunction with the Cat bit (8.2.3.7) in the segment descriptors to define what action should be taken when a segment of the copy does not fit exactly into an integer number of destination blocks.

The parameter list length field specifies the length in bytes of the parameters 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 as an error.

The COPY parameter list (see table 36) begins with a four-byte header that contains the COPY function code and priority. Following the header is one or more segment descriptors.

Table 36 - COPY parameter list

Bit / Byte	7	6	5	4	3	2	1	0
0	COPY function code					Priority		
1	Vendor-specific							
2	Reserved							
3	Reserved							
	Segment descriptor(s)							
0	Segment descriptor 0							
n	(See specific table for length.)							
	:							
0	Segment descriptor x							
n	(See specific table for length.)							

The COPY function code field defines a specific format for the segment descriptors. The COPY function codes ar defined in table 37. A target need not support all function codes for its device type.

Table 37 - COPY function codes

Peripheral device type		COPY function code	Segment descriptor table	Comments
Source	Destination			
Block devices (Device types 0,4,5,7)	Stream devices (Device types 1,2,3,9)	0	38	
Stream devices (Device types 1,3,9)	Block devices (Device types 0,4,5,7)	1	38	(Note 3)
Block devices (Device types 0,4,5,7)	Block devices (Device types 0,4,5,7)	2	39	(Note 3)
Stream devices (Device types 1,3,9)	Stream devices (Device types 1,2,3,9)	3	40	
Sequential-access (Device type 1)	Sequential-access (Device type 1)	4	41	Image copy

NOTES
1 COPY function codes 05h - 0Fh are reserved.
2 COPY function codes 10h - 1Fh are vendor-specific.
3 When using the COMPARE command the destination block device may be a CD-ROM device or an optical-memory device that uses read-only media.
4 See 8.2.5.1 for peripheral device type definitions.

The priority field of the COPY parameter list establishes the relative priority of this COPY command to othe commands being executed by the same target. All other commands are assumed to have a priority of 1. Priority (is the highest priority, with increasing values indicating lower priorities.

The segment descriptor formats are determined by the COPY function code. The segment descriptor format usec for block devices (i.e. write-once, CD-ROM, optical-memory, and direct-access devices) shall be the same. The segment descriptor format used for stream devices (i.e. printer, processor, communications, and sequential-acces devices)
shall be the same. Thus a copy operation from a write-once device to a printer device uses the same segmen descriptor format as a copy operation from a direct-access device to a sequential-access device (see table 37). The segment descriptor formats are described in 8.2.3.3 through 8.2.3.6. A maximum of 256 segment descriptors ar permitted. The segment descriptors are identified by ascending numbers beginning with zero.

8.2.3.1 Errors detected by the managing SCSI device

Two classes of exception conditions may occur during execution of a COPY command. The first class consists o those exception conditions detected by the SCSI device that received the COPY command and is managing the execution of the command. These conditions include parity errors while transferring the COPY command and statu byte, invalid parameters in the COPY command, invalid segment descriptors, and inability of the SCSI devic controlling the COPY functions to continue operating. In the event of such an exception condition, the SCSI devic managing the COPY shall:

a) terminate the COPY command with CHECK CONDITION status.
b) set the valid bit in the sense data to one. The segment number shall contain the number of the segmen descriptor being processed at the time the exception condition is detected. The sense key shall contain th sense key code describing the exception condition (i.e. not COPY ABORTED). The information field sha contain the difference between the number of blocks field in the segment descriptor being processed at th time of the failure and the number of blocks successfully copied. This number is the residue of unprocesse blocks remaining for the segment descriptor.

8.2.3.2 Errors detected by a target

The second class of errors consists of exception conditions detected by the SCSI device transferring data at the request of the SCSI device managing the transfer. The SCSI device managing the COPY command detects exception conditions by receiving CHECK CONDITION status from one of the SCSI devices it is managing. It then shall recover the sense data associated with the exception condition.

The SCSI device managing the COPY command may also be the source or destination SCSI device (or both). It shall distinguish between a failure of the management of the COPY and a failure of the data transfer being requested. It shall then create the appropriate sense data internally.

After recovering the sense data associated with the detected error, the SCSI device managing the COPY command shall:

a) terminate the COPY command with CHECK CONDITION status.
b) set the valid bit in the sense data to one. The segment number shall contain the number of the segment descriptor being processed at the time the exception condition is detected. The sense key shall be set to COPY ABORTED. The information field shall contain the difference between the number of blocks field in the segment descriptor being processed at the time of the failure and the number of blocks successfully copied. This number is the residue of unprocessed blocks remaining for the segment descriptor.

The first byte of the command-specific information field shall specify the starting byte number, relative to the first byte of sense data, of an area that contains (unchanged) the source logical unit's status byte and sense data. A zero value indicates that no status byte or sense data is being returned for the source logical unit.

The second byte of the command-specific information field shall specify the starting byte number, relative to the first byte of sense data, of an area that contains (unchanged) the destination logical unit's status byte and sense data. A zero value indicates that no status byte or sense data is being returned for the destination logical unit.

8.2.3.3 COPY function code 00h and 01h

The format for the segment descriptors for COPY transfers between block and stream devices is specified in table 38. This format is required for COPY function codes 00h or 01h. The segment descriptor may be repeated up to 256 times within the parameter list length specified in the command descriptor block.

Table 38 - Segment descriptor for COPY function codes 00h and 01h

Bit / Byte	7	6	5	4	3	2	1	0
0	Source address			Reserved	Cat	Source LUN		
1	Destination address			Reserved		Destination LUN		
2	(MSB)							
3				Stream device block length				(LSB)
4	(MSB)							
7				Block device number of blocks				(LSB)
8	(MSB)							
11				Block device logical block address				(LSB)

The source address and source LUN fields specify the SCSI bus ID and logical unit of the device to copy the data from for this segment of the COPY command. The destination address and destination LUN fields specify the SCSI bus ID and logical unit to copy the data to for this segment of the COPY command. Some SCSI devices may not support third-party COPY in which the copying SCSI device is not the source or destination device. Some SCSI

devices only support COPY within the SCSI device and not to other SCSI devices. If an unsupported COPY operatior is requested, the command shall be terminated with CHECK CONDITION status and the sense key shall be set tc ILLEGAL REQUEST with an additional sense code of INVALID FIELD IN PARAMETER LIST (see 8.2.3.1).

A catenate (Cat) bit (optional) of one indicates that the COPY manager shall catenate the last source block of ε segment with the first source block of the next segment if the last source block does not end exactly at the end o the destination block. The definition of a Cat bit of zero depends on the setting of the Pad bit in the commanc descriptor block (see 8.2.3.7).

The stream device block-length field specifies the block length to be used on the stream device logical unit during this segment of the COPY command. If the SCSI device managing the COPY knows this block length is no supported, the command shall be terminated with CHECK CONDITION status and the sense key shall be set tc ILLEGAL REQUEST with an additional sense code of INVALID FIELD IN PARAMETER LIST. If the block length is found to be invalid while executing a read or write operation to the stream device, the command shall be terminatec with CHECK CONDITION status and the sense key shall be set to COPY ABORTED (see 8.2.3.2).

The block device number of blocks field specifies the number of blocks in the current segment to be copied. A value of zero indicates that no blocks shall be transferred in this segment.

The block device logical block address field specifies the starting logical block address on the logical unit for this segment.

8.2.3.4 COPY function code 02h

The format for the segment descriptors for COPY transfers among block devices is specified in table 39. This forma is required for COPY function code 02h. The segment descriptor may be repeated up to 256 times within the parameter list length specified in the command descriptor block.

Table 39 - Segment descriptor for COPY function code 02h

Bit / Byte	7	6	5	4	3	2	1	0
0	Source address			DC	Cat	Source LUN		
1	Destination address			Reserved		Destination LUN		
2	Reserved							
3	Reserved							
4	(MSB)							
7	Number of blocks							(LSB)
8	(MSB)							
11	Source logical block address							(LSB)
12	(MSB)							
15	Destination logical block address							(LSB)

See 8.2.3.3 for definitions of the source address, the source LUN, the destination address, the destination LUN, and Cat fields.

A destination count (DC) bit of zero indicates that the number of blocks field refers to the source logical unit. A DC bit of one indicates that the number of blocks field refers to the destination logical unit.

The number of blocks field specifies the number of blocks to be transferred to or from (depending on the DC bit) the block device during this segment. A value of zero indicates that no blocks shall be transferred.

The source logical block address field specifies the starting logical block address on the source block device.

The destination logical block address field specifies the starting logical block address on the destination block device.

8.2.3.5 COPY function code 03h

The format for the segment descriptors for COPY transfers among stream devices is specified by table 40. This format is required for COPY function code 03h. The segment descriptor may be repeated up to 256 times within the parameter list length specified in the command descriptor block.

Table 40 - Segment descriptor for COPY function code 03h

Bit / Byte	7	6	5	4	3	2	1	0
0	Source address			DC	Cat	Source LUN		
1	Destination address			Reserved		Destination LUN		
2	Reserved							
3	Reserved							
4	(MSB)							
5	Source block length							(LSB)
6	(MSB)							
7	Destination block length							(LSB)
8	(MSB)							
11	Number of blocks							(LSB)

See 8.2.3.3 for definitions of the source address, the source LUN, the destination address, the destination LUN, and Cat fields.

A destination count (DC) bit of zero indicates that the number of blocks field refers to the source logical unit. A DC bit of one indicates that the number of blocks field refers to the destination logical unit.

The source block length field specifies the block-length of the source device for this segment of the COPY. A zero in this field indicates variable block-length. For non-zero values, this field shall match the logical unit's actual block-length.

If block-length mismatches are detected prior to the beginning of the read operation by the SCSI device managing the COPY, the command shall be terminated with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN PARAMETER LIST (see 8.2.3.1).

If the mismatches are detected during the read operation by the COPY manager, the command shall be terminated with CHECK CONDITION status. The sense key shall be set to COPY ABORTED (see 8.2.3.2). and the additional sense code shall be set to INVALID FIELD IN PARAMETER LIST.

The destination block-length field specifies the block length to be used on the destination logical unit during the COPY. Destination block length mismatches are handled in an analogous manner as source block length mismatches.

The number of blocks field specifies the number of blocks to be transferred to or from (depending on the DC bit) the device during this segment. A value of zero indicates that no blocks shall be transferred.

8.2.3.6 COPY function code 04h

The format for the segment descriptors for image COPY transfers between sequential-access devices is specified in table 41. This format is required for COPY function code 04h. The segment descriptor may be repeated up to 256 times within the parameter list length specified in the command descriptor block.

Table 41 - Segment descriptor for COPY function code 04h

Bit / Byte	7	6	5	4	3	2	1	0
0	Source address			Reserved		Source LUN		
1	Destination address			Reserved		Destination LUN		
2	Count							
3 — 7	Reserved							
8 — 11	Vendor-specific							

See 8.2.3.3 for definitions of the source address, the source LUN, the destination address, the destination LUN, and Cat fields.

The image mode COPY command copies an exact image of the source device medium to the destination device medium, beginning at their current positions. The copy function terminates when the source device:

a) encounters an end-of-partition as defined by the source device;
b) encounters an end-of-data as defined by the source device (i.e. BLANK CHECK sense key);
c) has copied the number of consecutive filemarks specified in the count field from the source device to the destination device;
d) has copied the number of consecutive setmarks specified in the count field from the source device to the destination device, if the RSmk bit in the device configuration page (see 10.3.3.1) is one.

A count field of zero indicates that the COPY command shall not terminate due to any number of consecutive filemarks or setmarks. Other error or exception conditions (e.g. early-warning end-of-partition on the destination device) may cause the COPY command to terminate prior to completion. In such cases, it is not possible to calculate a residue, so the information field in the sense data shall be set to zero.

8.2.3.7 Copies with unequal block lengths

When copying data between two devices with unequal block lengths, it is possible for the last source block to not completely fill the last destination block for one or more segments in the COPY command. Two optional bits are defined to assist in controlling the copy manager's actions in this circumstance. The Pad bit (in the command descriptor block) and the Cat bit (in each applicable segment descriptor) are defined in table 42.

Table 42 - Pad and Cat bit definition

Pad	Cat	COPY manager's action
0	0	On inexact segments, it is device specific whether the COPY manager rejects the COPY command with CHECK CONDITION status and ILLEGAL REQUEST sense key, the COPY manager writes or accepts short blocks (variable-block mode on sequential-access devices), or the COPY manager adds pad characters (00h) to the destination block or strips pad characters from the source block.
1	0	On inexact segments, the COPY manager shall add pad characters (00h) to the destination block to completely fill the block, or it shall strip pad characters from the source block, always stopping at the end of a complete block.
X	1	The COPY manager shall always write or read complete blocks. On inexact segments, the remainder of the block contains data from the next segment. This code is not valid in the last segment of the COPY command.

NOTE 63 Use of pad characters is intended to assist in managing COPY commands between devices of different block lengths where partial-block residues may occur. The initiator who issued the COPY command is responsible for management of these pad areas (i.e. remembering where they are). One possible method is to write the COPY parameter list information to the destination medium prior to issuing the COPY command for backup and to read this information prior to issuing the COPY command for restore.

8.2.4 COPY AND VERIFY command

The COPY AND VERIFY command (see table 43) performs the same function as the COPY command, except that a verification of the data written to the destination logical unit is performed after the data is written. The parameter list transferred to the target is the same as for the COPY command. This parameter list contains the information to identify the logical units involved in the copy and the length of the copy. See 8.2.3 for additional information about the COPY command.

Table 43 - COPY AND VERIFY command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (3Ah)							
1	Logical unit number			Reserved			BytChk	Pad
2	Reserved							
3	(MSB)							
4	Parameter list length							
5								(LSB)
6	Reserved							
7	Reserved							
8	Reserved							
9	Control							

A byte check (BytChk) bit of zero causes a medium verification to be performed with no data comparison. A BytChk bit of one causes a byte-by-byte comparison of data written on the destination medium and the data transferred from the source medium. If the comparison is unsuccessful for any reason, the copy manager shall return CHECK

CONDITION status with the sense key set to MISCOMPARE. The remaining fields in the sense data shall be set as documented in the COPY command.

8.2.5 INQUIRY command

The INQUIRY command (see table 44) requests that information regarding parameters of the target and its attached peripheral device(s) be sent to the initiator. An option allows the initiator to request additional information about the target or logical unit (see 8.2.5.2).

Table 44 - INQUIRY command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (12h)							
1	Logical unit number			Reserved				EVPD
2	Page code							
3	Reserved							
4	Allocation length							
5	Control							

An enable vital product data (EVPD) bit of one specifies that the target shall return the optional vital product data specified by the page code field. If the target does not support vital product data and this bit is set to one, 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.

An EVPD bit of zero specifies that the target shall return the standard INQUIRY data. If the page code field is not zero, 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 page code field specifies which page of vital product data information the target shall return (see 8.3.4).

The INQUIRY command shall return CHECK CONDITION status only when the target cannot return the requested INQUIRY data.

NOTE 64 The INQUIRY data should be returned even though the peripheral device may not be ready for other commands.

If an INQUIRY command is received from an initiator with a pending unit attention condition (i.e. before the target reports CHECK CONDITION status), the target shall perform the INQUIRY command and shall not clear the unit attention condition (see 7.9).

NOTES
65 The INQUIRY command is typically used by the initiator after a reset or power-up condition to determine the device types for system configuration. To minimize delays after a reset or power-up condition, the standard INQUIRY data should be available without incurring any media access delays. If the target does store some of the INQUIRY data on the device, it may return zeros or ASCII spaces (20h) in those fields until the data is available from the device.
66 The INQUIRY data may change as the target executes its initialization sequence or in response to a CHANGE DEFINITION command. For example, the target may contain a minimum command set in its non-volatile memory and may load its final firmware from the device when it becomes ready. After it has loaded the firmware, it may support more options and therefore return different supported options information in the INQUIRY data.

8.2.5.1 Standard INQUIRY data

The standard INQUIRY data (see table 45) contains 36 required bytes, followed by a variable number of vendor-specific parameters. Bytes 56 through 95, if returned, are reserved for future standardization.

Table 45 - Standard INQUIRY data format

Bit / Byte	7	6	5	4	3	2	1	0
0	Peripheral qualifier			Peripheral device type				
1	RMB	Device-type modifier						
2	ISO version			ECMA version		ANSI-approved version		
3	AENC	TrmIOP	Reserved		Response data format			
4	Additional length (n-4)							
5	Reserved							
6	Reserved							
7	RelAdr	WBus32	WBus16	Sync	Linked	Reserved	CmdQue	SftRe
8 – 15	(MSB) Vendor identification (LSB)							
16 – 31	(MSB) Product identification (LSB)							
32 – 35	(MSB) Product revision level (LSB)							
36 – 55	Vendor-specific							
56 – 95	Reserved							
	Vendor-specific parameters							
96 – n	Vendor-specific							

The peripheral qualifier and peripheral device-type fields identify the device currently connected to the logical unit. If the target is not capable of supporting a device on this logical unit, this field shall be set to 7Fh (peripheral qualifier set to 011b and peripheral device type set to 1Fh). The peripheral qualifier is defined in table 46 and the peripheral device type is defined in table 47.

Table 46 - Peripheral qualifier

Qualifier	Description
000b	The specified peripheral device type is currently connected to this logical unit. If the target cannot determine whether or not a physical device is currently connected, it shall also use this peripheral qualifier when returning the INQUIRY data. This peripheral qualifier does not mean that the device is ready for access by the initiator.
001b	The target is capable of supporting the specified peripheral device type on this logical unit; however, the physical device is not currently connected to this logical unit.
010b	Reserved
011b	The target is not capable of supporting a physical device on this logical unit. For this peripheral qualifier the peripheral device type shall be set to 1Fh to provide compatibility with previous versions of SCSI. All other peripheral device type values are reserved for this peripheral qualifier.
1XXb	Vendor-specific

Table 47 - Peripheral device type

Code	Description
00h	Direct-access device (e.g. magnetic disk)
01h	Sequential-access device (e.g. magnetic tape)
02h	Printer device
03h	Processor device
04h	Write-once device (e.g. some optical disks)
05h	CD-ROM device
06h	Scanner device
07h	Optical memory device (e.g. some optical disks)
08h	Medium changer device (e.g. jukeboxes)
09h	Communications device
0Ah - 0Bh	Defined by ASC IT8 (Graphic arts pre-press devices)
0Ch - 1Eh	Reserved
1Fh	Unknown or no device type

A removable medium (RMB) bit of zero indicates that the medium is not removable. A RMB bit of one indicates that the medium is removable.

The device-type modifier field was defined in SCSI-1 to permit vendor-specific qualification codes of the device type. This field is retained for compatibility with SCSI-1. Targets that do not support this field should return a value of zero.

The usage of non-zero code values in the ISO version and ECMA version fields are defined by the International Organization for Standardization and the European Computer Manufacturers Association, respectively. A zero code value in these fields shall indicate that the target does not claim compliance to the ISO version of SCSI (ISO 9316) or the ECMA version of SCSI (ECMA-111). It is possible to claim compliance to more than one of these SCSI standards.

The ANSI-approved version field indicates the implemented version of this standard and is defined in table 48.

Table 48 - ANSI-approved version

Code	Description
0h	The device might or might not comply to an ANSI-approved standard.
1h	The device complies to ANSI X3.131-1986 (SCSI-1).
2h	The device complies to this version of SCSI. This code is reserved to designate this standard upon approval by ANSI.
3h - 7h	Reserved

The asynchronous event notification capability (AENC) bit indicates that the device supports the asynchronous event notification capability as defined in 7.5.5.

a) Processor device-type definition: An AENC bit of one indicates that the processor device is capable of accepting asynchronous event notifications. An AENC bit of zero indicates that the processor device does not support asynchronous event notifications.
b) All other device-types: This bit is reserved.

A terminate I/O process (TrmIOP) bit of one indicates that the device supports the TERMINATE I/O PROCESS message as defined in 6.6.22. A value of zero indicates that the device does not support the TERMINATE I/O PROCESS message.

A response data format value of zero indicates the INQUIRY data format is as specified in SCSI-1. A response data format value of one indicates compatibility with some products that were designed prior to the development of this standard (i.e. CCS). A response data format value of two indicates that the data shall be in the format specified in this standard. Response data format values greater than two are reserved.

The additional length field shall specify the length in bytes of the parameters. If the allocation length of the command descriptor block is too small to transfer all of the parameters, the additional length shall not be adjusted to reflect the truncation.

A relative addressing (RelAdr) bit of one indicates that the device supports the relative addressing mode for this logical unit. If this bit is set to one, the linked command (Linked) bit shall also be set to one; since relative addressing can only be used with linked commands. A RelAdr bit of zero indicates the device does not support relative addressing for this logical unit.

A wide bus 32 (Wbus32) bit of one indicates that the device supports 32-bit wide data transfers. A value of zero indicates that the device does not support 32-bit wide data transfers.

A wide bus 16 (Wbus16) bit of one indicates that the device supports 16-bit wide data transfers. A value of zero indicates that the device does not support 16-bit wide data transfers.

NOTE 67 If the values of both the Wbus16 and Wbus32 bits are zero, the device only supports 8-bit wide data transfers.

A synchronous transfer (Sync) bit of one indicates that the device supports synchronous data transfer. A value of zero indicates the device does not support synchronous data transfer.

A linked command (Linked) bit of one indicates that the device supports linked commands for this logical unit. A value of zero indicates the device does not support linked commands for this logical unit.

A command queuing (CmdQue) bit of one indicates that the device supports tagged command queuing for this logical unit. A value of zero indicates the device does not support tagged command queuing for this logical unit.

A soft reset (SftRe) bit of zero indicates that the device responds to the RESET condition with the hard RESET alternative (see 6.2.2.1). A SftRe bit of one indicates that the device responds to the RESET condition with the soft RESET alternative (see 6.2.2.2).

ASCII data fields shall contain only graphic codes (i.e. code values 20h through 7Eh). Left-aligned fields shall place any unused bytes at the end of the field (highest offset) and the unused bytes shall be filled with space characters (20h). Right-aligned fields shall place any unused bytes at the start of the field (lowest offset) and the unused bytes shall be filled with space characters (20h).

The vendor identification field contains eight bytes of ASCII data identifying the vendor of the product. The data shall be left aligned within this field.

NOTE 68 It is intended that this field provide a unique vendor identification of the manufacturer of the SCSI device. In the absence of a formal registration procedure, X3T10 maintains a list of vendor identification codes in use. Vendors are requested to voluntarily submit their identification codes to X3T10 to prevent duplication of codes (see annex E).

The product identification field contains sixteen bytes of ASCII data as defined by the vendor. The data shall be left-aligned within this field.

The product revision level field contains four bytes of ASCII data as defined by the vendor. The data shall be left-aligned within this field.

8.2.5.2 Vital product data

Implementation of vital product data is optional. The information returned consists of configuration data (e.g. vendor identification, product identification, model, serial number), manufacturing data (e.g. plant and date of manufacture), field replaceable unit data and other vendor- or device-specific data.

The initiator requests the vital product data information by setting the EVPD bit to one and specifying the page code of the desired vital product data (see 8.3.4). If the target does not implement the requested page it shall return CHECK CONDITION status. The a sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN CDB.

NOTES
69 The target should have the ability to execute the INQUIRY command even when a device error occurs that prohibits normal command execution. In such a case, CHECK CONDITION status should be returned for commands other than INQUIRY or REQUEST SENSE. The sense data returned may contain the field replaceable unit code. The vital product data should be obtained for the failing device using the INQUIRY command.
70 This standard defines a format that allows device-independent initiator software to display the vital product data returned by the INQUIRY command. For example, the initiator may display the data associated for the field replaceable unit returned in the sense data. The contents of the data may be vendor-specific; therefore, it may not be usable without detailed information about the device.
71 This standard does not define the location or method of storing the vital product data. The retrieval of the data may require completion of initialization operations within the device that may induce delays before the data is available to the initiator. Time-critical requirements are an implementation consideration and are not addressed in this standard.

8.2.6 LOG SELECT command

The LOG SELECT command (see table 49) provides a means for the initiator to manage statistical information maintained by the device about the device or its logical units. Targets that implement the LOG SELECT command shall also implement the LOG SENSE command. Structures in the form of log parameters within log pages are defined as a way to manage the log data. The LOG SELECT command provides for sending zero or more log pages during a DATA OUT phase. This standard defines the format of the log pages, but does not define the exact conditions and events that are logged.

Table 49 - LOG SELECT command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (4Ch)							
1	Logical unit number			Reserved			PCR	SP
2	PC			Reserved				
3	Reserved							
4	Reserved							
5	Reserved							
6	Reserved							
7	(MSB)			Parameter list length				
8								(LSB)
9	Control							

A parameter code reset (PCR) bit of one and a parameter list length of zero shall cause all implemented parameters to be set to the target-defined default values (e.g. zero). If the PCR bit is one and the parameter list length is greater than zero, the command is terminated with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN CDB. A PCR bit of zero specifies that the log parameters shall not be reset.

A save parameters (SP) bit of one indicates that after performing the specified LOG SELECT operation the target shall save to non-volatile memory all parameters identified as savable by the DS bit in the log page (see 8.3.2). A SP bit of zero specifies that parameters shall not be saved.

Saving of log parameters is optional and indicated for each log parameter by the DS bit in the page. Log parameters may be saved at vendor-specific times subject to the TSD bit (see 8.3.2) in the log parameter. If the target does not implement saved parameters for any log parameter and the SP bit is set to one, the command shall be terminated with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

It is not an error to set the SP bit to one and to set the DS bit of a log parameter to one. In this case, the parameter value for that log parameter is not saved.

The page control (PC) field defines the type of parameter values to be selected. The page control field is defined in table 50.

Table 50 - Page control field

Type	LOG SELECT parameter values	LOG SENSE parameter values
00b	Current threshold values	Threshold values
01b	Current cumulative values	Cumulative values
10b	Default threshold values	Default threshold values
11b	Default cumulative values	Default cumulative values

The current cumulative values may be updated by the target or by the initiator using the LOG SELECT command to reflect the cumulative number of events experienced by the target. Fields in the parameter control byte (8.3.2) of each log parameter control the updating and saving of the current cumulative parameters.

The target shall set the current threshold parameters to the default threshold values in response to a LOG SELECT command with the PC field set to 10b and the parameter list length field set to zero.

The target shall set all cumulative parameters to their default values in response to a LOG SELECT command with the PC field set to 11b and the parameter list length field set to zero.

The current threshold value can only be modified by the initiator via the LOG SELECT command. If the initiator attempts to change current threshold values that are not available or not implemented for that log parameter, then the target shall terminate the LOG SELECT command with CHECK CONDITION status, the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST. The saving of current threshold parameters and the criteria for the current threshold being met are controlled by bits in the parameter control byte (8.3.2).

NOTE 72 Pages or log parameters that are not available may become available at some later time (e.g. after the device has become ready).

The parameter list length field specifies the length in bytes of the parameter list that shall be transferred from the initiator to the target during the DATA OUT phase. A parameter list length of zero indicates that no pages shall be transferred. This condition shall not be considered an error. If the initiator sends page codes or parameter codes within the parameter list that are reserved or not implemented by the target, the target shall terminate the LOG SELECT command with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

If a parameter list length results in the truncation of any log parameter, the target shall terminate the command with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN CDB.

The initiator should send pages in ascending order by page code value if multiple pages are sent during a DATA OUT phase. If multiple log parameters within a page are sent during the DATA OUT phase, they should be sent in ascending order by parameter code value. The target shall return CHECK CONDITION status if the initiator sends pages out of order or parameter codes out of order. The sense key shall be set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

NOTE 73 Initiators should issue LOG SENSE commands prior to issuing LOG SELECT commands to determine supported pages and page lengths.

The target may provide independent sets of log parameters for each logical unit or for each combination of logical units and initiators. If the target does not support independent sets of log parameters and any log parameters are changed that affect other initiators, then the target shall generate a unit attention condition for all initiators except the one that issued the LOG SELECT command (see 7.9). This unit attention condition is returned with an additional sense code of LOG PARAMETERS CHANGED.

If the initiator sends a log parameter that is not supported by the target, the target shall terminate the command with CHECK CONDITION status, set the sense key to ILLEGAL REQUEST, and set the additional sense code to INVALID FIELD IN PARAMETER LIST.

8.2.7 LOG SENSE command

The LOG SENSE command (see table 51) provides a means for the initiator to retrieve statistical information maintained by the device about the device or its logical units. It is a complementary command to the LOG SELECT command.

Table 51 - LOG SENSE command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (4Dh)							
1	Logical unit number			Reserved			PPC	SP
2	PC			Page code				
3	Reserved							
4	Reserved							
5	(MSB)			Parameter pointer				
6								(LSB)
7	(MSB)			Allocation length				
8								(LSB)
9	Control							

The parameter pointer control (PPC) bit controls the type of parameters requested from the target:

a) A PPC bit of one indicates that the target shall return a log page with parameter code values that have changed since the last LOG SELECT or LOG SENSE command. The target shall return only those parameter codes following the parameter pointer field.

b) A PPC bit of zero indicates that the log parameter requested from the target shall begin with the parameter code specified in the parameter pointer field and return the number of bytes specified by the allocation length field in ascending order of parameter codes from the specified log page. A PPC bit of zero and a parameter pointer field of zero shall cause all available log parameters for the specified log page to be returned to the initiator subject to the specified allocation length.

Saving parameters is an optional function of the LOG SENSE command. If the target does not implement saving log parameters and if the save parameters (SP) bit is one, then the target shall return CHECK CONDITION status, set the sense key to ILLEGAL REQUEST, and set the additional sense code to INVALID FIELD IN CDB.

An SP bit of zero indicates the target shall perform the specified LOG SENSE command and shall not save any log parameters. If saving log parameters is implemented, an SP bit of one indicates that the target shall perform the specified LOG SENSE command and shall save all log parameters identified as savable by the DS bit (8.3.2) to a non-volatile, vendor-specific location.

The page control (PC) field defines the type of parameter values to be selected (see 8.2.6 for the definition of the page control field). The parameter values returned by a LOG SENSE command are determined as follows:

a) The specified parameter values at the last update (in response to a LOG SELECT or LOG SENSE command or done automatically by the target for cumulative values).

b) The saved values, if an update has not occurred since the last power-on, hard RESET condition, or BUS DEVICE RESET message and saved parameters are implemented.

c) The default values, if an update has not occurred since the last power-on, hard RESET condition, or BUS DEVICE RESET message and saved values are not available or not implemented.

The page code field identifies which page of data is being requested (see 8.3.2). If the page code is reserved or not implemented, the target shall terminate the command with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST with the additional sense code set to INVALID FIELD IN CDB.

The parameter pointer field allows the initiator to request parameter data beginning from a specific parameter code to the maximum allocation length or the maximum parameter code supported by the target, whichever is less. If the value of the parameter pointer field is larger than the largest available parameter code that can be returned by the target on the specified page, the target shall terminate the command with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN CDB.

Log parameters within the specified log page shall be transferred in ascending order according to parameter code.

8.2.8 MODE SELECT(6) command

The MODE SELECT(6) command (see table 52) provides a means for the initiator to specify medium, logical unit, or peripheral device parameters to the target. Targets that implement the MODE SELECT command shall also implement the MODE SENSE command. Initiators should issue MODE SENSE prior to each MODE SELECT to determine supported pages, page lengths, and other parameters.

Table 52 - MODE SELECT(6) command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (15h)							
1	Logical unit number			PF	Reserved			SP
2	Reserved							
3	Reserved							
4	Parameter list length							
5	Control							

If a target supports saved pages, it may save only one copy of the page for each logical unit and have it apply to all initiators, or it may save separate copies for each initiator for each logical unit. If separate copies are saved, the target shall maintain separate current values for each I_T_L nexus. Pages that are common to all initiators are not required to have multiple copies.

If an initiator sends a MODE SELECT command that changes any parameters applying to other initiators, the target shall generate a unit attention condition for all initiators except the one that issued the MODE SELECT command (see 7.9). The target shall set the additional sense code to MODE PARAMETERS CHANGED.

The target may provide for independent sets of parameters for each attached logical unit or for each combination of logical unit and initiator. If independent sets of parameters are implemented, and a third party reservation is requested, the target transfers the set of parameters in effect for the initiator of the RESERVE command to the parameters used for commands from the third party device (see 9.2.12.3 and 10.2.10.1).

A page format (PF) bit of zero indicates that the MODE SELECT parameters are as specified in SCSI-1, (i.e. all parameters after the block descriptors are vendor-specific). A PF bit of one indicates that the MODE SELECT parameters following the header and block descriptor(s) are structured as pages of related parameters and are as specified in this standard.

A save pages (SP) bit of zero indicates the target shall perform the specified MODE SELECT operation, and shall not save any pages. An SP bit of one indicates that the target shall perform the specified MODE SELECT operation, and shall save to a non-volatile vendor-specific location all the savable pages including any sent during the DATA OUT phase. The SP bit is optional, even when mode pages are supported by the target. Pages that are saved are identified by the parameter savable bit that is returned in the page header by the MODE SENSE command (see 8.3.3). If the PS bit is set in the MODE SENSE data then the page shall be savable by issuing a MODE SELECT command with the SP bit set. If the target does not implement saved pages and the SP bit is set to one, the command shall be terminated with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

The parameter list length field specifies the length in bytes of the mode parameter list that shall be transferred from the initiator to the target during the DATA OUT phase. A parameter list length of zero indicates that no data shall be transferred. This condition shall not be considered as an error.

The target shall terminate the command with CHECK CONDITION status if the parameter list length results in the truncation of any mode parameter header, mode parameter block descriptor(s), or mode page. The sense key shall be set to ILLEGAL REQUEST, and the additional sense code shall be set to PARAMETER LIST LENGTH ERROR.

The mode parameter list for the MODE SELECT and MODE SENSE commands is defined in 8.3.3. Parts of each mode parameter list are uniquely defined for each device-type.

The target shall terminate the MODE SELECT command with CHECK CONDITION status, set the sense key to ILLEGAL REQUEST, set the additional sense code to INVALID FIELD IN PARAMETER LIST, and shall not change any mode parameters for the following conditions:

a) If the initiator sets any field that is reported as not changeable by the target to a value other than its current value.
b) If the initiator sets any field in the mode parameter header or block descriptor(s) to an unsupported value.
c) If an initiator sends a mode page with a page length not equal to the page length returned by the MODE SENSE command for that page.
d) If the initiator sends a unsupported value for a mode parameter and rounding is not implemented for that mode parameter.
e) If the initiator sets any reserved field in the mode parameter list to a non-zero value.

If the initiator sends a value for a mode parameter that is outside the range supported by the target and rounding is implemented for that mode parameter, the target may either:

a) round the parameter to an acceptable value and terminate the command as described in 7.5.4;
b) terminate the command with CHECK CONDITION status, the sense key set to ILLEGAL REQUEST, and set the additional sense code to INVALID FIELD IN PARAMETER LIST.

A target may alter any mode parameter in any mode page (even those reported as non-changeable) as a result of changes to other mode parameters.

The target validates the non-changeable mode parameters against the current values that existed for those mode parameters prior to the MODE SELECT command.

NOTE 74 If the current values calculated by the target affect the initiator's operation, the initiator should issue a MODE SENSE command after each MODE SELECT command.

8.2.9 MODE SELECT(10) command

The MODE SELECT(10) command (see table 53) provides a means for the initiator to specify medium, logical unit or peripheral device parameters to the target. See the MODE SELECT(6) command (8.2.8) for a description of the fields in this command. Initiators should issue MODE SENSE prior to each MODE SELECT to determine supported mode pages, mode page lengths, and other parameters.

Table 53 - MODE SELECT(10) command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (55h)							
1	Logical unit number			PF	Reserved			SP
2	Reserved							
3	Reserved							
4	Reserved							
5	Reserved							
6	Reserved							
7	(MSB)			Parameter list length				
8								(LSB)
9	Control							

Targets that implement the MODE SELECT(10) command shall also implement the MODE SENSE(10) command.

8.2.10 MODE SENSE(6) command

The MODE SENSE(6) command (see table 54) provides a means for a target to report parameters to the initiator It is a complementary command to the MODE SELECT(6) command.

Table 54 - MODE SENSE(6) command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (1Ah)							
1	Logical unit number			Reserved	DBD	Reserved		
2	PC		Page code					
3	Reserved							
4	Allocation length							
5	Control							

A disable block descriptors (DBD) bit of zero indicates that the target may return zero or more block descriptors in the returned MODE SENSE data (see 8.3.3), at the target's discretion. A DBD bit of one specifies that the target shall not return any block descriptors in the returned MODE SENSE data.

The page control (PC) field defines the type of mode parameter values to be returned in the mode pages. The page control field is defined in table 55.

Table 55 - Page control field

Code	Type of parameter	Subclause
00b	Current values	8.2.10.1
01b	Changeable values	8.2.10.2
10b	Default values	8.2.10.3
11b	Saved values	8.2.10.4

NOTE 75 The page control field only affects the mode parameters within the mode pages, however the PS bit, page code and page length fields should return current values since they have no meaning when used with other types. The mode parameter header and mode parameter block descriptor should return current values.

The page code specifies which mode page(s) to return. Mode page code usage is defined in table 56.

Table 56 - Mode page code usage for all devices

Page code	Description	Subclause
00h	Vendor-specific (does not require page format)	
01h - 1Fh	See specific device-types	
20h - 3Eh	Vendor-specific (page format required)	
3Fh	Return all mode pages	

An initiator may request any one or all of the supported mode pages from a target. If an initiator issues a MODE SENSE command with a page code value not implemented by the target, the target shall return CHECK CONDITION status and shall set the sense key to ILLEGAL REQUEST and the additional sense code to INVALID FIELD IN CDB.

A page code of 3Fh indicates that all mode pages implemented by the target shall be returned to the initiator. If the mode parameter list exceeds 256 bytes for a MODE SENSE(6) command or 65 536 bytes for a MODE SENSE(10) command, the target shall return CHECK CONDITION status and the sense key shall be set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN CDB.

Mode page 00h, if implemented, shall be returned after all other mode pages.

NOTES
76 Mode pages should be returned in ascending page code order except for mode page 00h.
77 If the PC field and the page code field are both set to zero the target should return a mode parameter header and block descriptor (if applicable). This provides for compatibility with existing SCSI-1 initiators.

The mode parameter list for all device types for MODE SELECT and MODE SENSE is defined in 8.3.3. Parts of the mode parameter list are specifically defined for each device type. See subclause three of each peripheral device type for further information.

8.2.10.1 Current values

A PC field value of 0h requests that the target return the current values of the mode parameters. The current values returned are:

a) the current values of the mode parameters established by last successful MODE SELECT command;
b) the saved values of the mode parameters if a MODE SELECT command has not successfully completed since the last power-on, hard RESET condition, or BUS DEVICE RESET message;
c) the default values of the mode parameters, if saved values, are not available or not supported.

8.2.10.2 Changeable values

A PC field value of 1h requests that the target return a mask denoting those mode parameters that are changeable In the mask, the fields of the mode parameters that are changeable shall be set to all one bits and the fields of the mode parameters that are non-changeable (i.e. defined by the target) shall be set to all zero bits.

NOTES
78 An attempt to change a non-changeable mode parameter (via MODE SELECT) results in an error condition (see 8.2.8).
79 The initiator should issue a MODE SENSE command with the PC field set to 1h and the page code field set to 3Fh to determine which mode pages are supported, which mode parameters within the mode pages are changeable, and the supported length of each mode page prior to issuing any MODE SELECT commands.

8.2.10.3 Default values

A PC field value of 2h requests that the target return the default values of the mode parameters. Parameters not supported by the target shall be set to zero. Default values are accessible even if the device is not ready.

8.2.10.4 Saved values

A PC field value of 3h requests that the target return the saved values of the mode parameters. Implementation of saved page parameters is optional. Mode parameters not supported by the target shall be set to zero. If saved values are not implemented, the command shall be terminated with CHECK CONDITION status, the sense key set to ILLEGAL REQUEST and the additional sense code set to SAVING PARAMETERS NOT SUPPORTED.

NOTE 80 The method of saving parameters is vendor-specific. The parameters are preserved in such a manner that they are retained when the target is powered down. All savable pages can be considered saved when a MODE SELECT command issued with the SP bit set to one has returned a GOOD status or after the successful completion of a FORMAT UNIT command.

8.2.10.5 Initial responses

After a power-up condition or hard reset condition, the target shall respond in the following manner:

a) If default values are requested, report the default values.
b) If saved values are requested, report valid restored mode parameters, or restore the mode parameters and report them. If the saved values of the mode parameters are not able to be accessed from the non-volatile vendor-specific location, terminate the command with CHECK CONDITION status and set the sense key set to NOT READY. If saved parameters are not implemented respond as defined in 8.2.10.4.
c) If current values are requested and the current values of the mode parameters have not been sent by the initiator (via a MODE SELECT command), the target may return either the default or saved values, as defined above. If current values have been sent, the current values shall be reported.

8.2.11 MODE SENSE(10) command

The MODE SENSE(10) command (see table 57) provides a means for a target to report parameters to the initiator. It is a complementary command to the MODE SELECT(10) command. If the MODE SELECT(10) command is implemented the MODE SENSE(10) command shall be implemented. See the MODE SENSE(6) command for a description of the fields in this command.

Table 57 - MODE SENSE(10) command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (5Ah)							
1	Logical unit number			Reserved	DBD	Reserved		
2	PC		Page code					
3	Reserved							
4	Reserved							
5	Reserved							
6	Reserved							
7	(MSB) Allocation length							
8								(LSB)
9	Control							

8.2.12 READ BUFFER command

The READ BUFFER command (see table 58) is used in conjunction with the WRITE BUFFER command as a diagnostic function for testing target memory and the SCSI bus integrity. This command shall not alter the medium.

Table 58 - READ BUFFER command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (3Ch)							
1	Logical unit number			Reserved		Mode		
2	Buffer ID							
3	(MSB) Buffer offset							
4								
5								(LSB)
6	(MSB) Allocation length							
7								
8								(LSB)
9	Control							

The function of this command and the meaning of fields within the command descriptor block depend on the contents of the mode field. The mode field is defined in table 59.

Table 59 - READ BUFFER mode field

Mode	Description	Type
000b	Combined header and data	Optional
001b	Vendor-specific	Vendor-specific
010b	Data	Optional
011b	Descriptor	Optional
100b	Reserved	Reserved
101b	Reserved	Reserved
110b	Reserved	Reserved
111b	Reserved	Reserved

NOTE 81 Modes 000b and 001b are included for compatibility with products that were designed prior to the generation of this standard. Some products that were designed prior to the generation of this standard restrict the available length to 65 535 bytes.

8.2.12.1 Combined header and data mode (000b)

In this mode, a four-byte header followed by data bytes is returned to the initiator during the DATA IN phase. The buffer ID and the buffer offset fields are reserved.

The four-byte READ BUFFER header (see table 60) is followed by data bytes from the target's data buffer.

Table 60 - READ BUFFER header

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved							
1	(MSB)							
—	Buffer capacity							
3								(LSB)

The buffer capacity field specifies the total number of data bytes available in the target's data buffer. This number is not reduced to reflect the allocation length; nor is it reduced to reflect the actual number of bytes written using the WRITE BUFFER command. Following the READ BUFFER header, the target shall transfer data from its data buffer. The target terminates the DATA IN phase when allocation length bytes of header plus data have been transferred or when all available header and buffer data have been transferred to the initiator, whichever is less.

8.2.12.2 Vendor-specific mode (001b)

In this mode, the meanings of the buffer ID, buffer offset, and allocation length fields are not specified by this standard.

8.2.12.3 Data mode (010b)

In this mode, the DATA IN phase contains buffer data. The buffer ID field identifies a specific buffer within the target from which data shall be transferred. The vendor assigns buffer ID codes to buffers within the target. Buffer ID zero shall be supported. If more than one buffer is supported, additional buffer ID codes shall be assigned contiguously, beginning with one. Buffer ID code assignments for the READ BUFFER command shall be the same as for the WRITE BUFFER command. If an unsupported buffer ID code is selected, the target shall return CHECK CONDITION status, shall set the sense key to ILLEGAL REQUEST, and set the additional sense code to ILLEGAL FIELD IN CDB. The target terminates the DATA IN phase when allocation length bytes have been transferred or when all the available data from the buffer has been transferred to the initiator, whichever amount is less.

The buffer offset field contains the byte offset within the specified buffer from which data shall be transferred from. The initiator should conform to the offset boundary requirements returned in the READ BUFFER descriptor (see

8.2.12.4). If the target is unable to accept the specified buffer offset, it shall return CHECK CONDITION status, shall set the sense key to ILLEGAL REQUEST, and set the additional sense code to ILLEGAL FIELD IN CDB.

8.2.12.4 Descriptor mode (011b)

In this mode, a maximum of four bytes of READ BUFFER descriptor information is returned. The target shall return the descriptor information for the buffer specified by the buffer ID (see the description of the buffer ID in 8.2.12.3). If there is no buffer associated with the specified buffer ID, the target shall return all zeros in the READ BUFFER descriptor. The buffer offset field is reserved in this mode. The allocation length should be set to four or greater. The target shall transfer the lesser of the allocation length or four bytes of READ BUFFER descriptor. The READ BUFFER descriptor is defined as shown in table 61.

Table 61 - READ BUFFER descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	Offset boundary							
1	(MSB)							
				Buffer capacity				
3								(LSB)

The offset boundary field returns the boundary alignment within the selected buffer for subsequent WRITE BUFFER and READ BUFFER commands. The value contained in the offset boundary field shall be interpreted as a power of two.

The value contained in the buffer offset field of subsequent WRITE BUFFER and READ BUFFER commands should be a multiple of $2^{\text{offset boundary}}$ as shown in table 62.

Table 62 - Buffer offset boundary

Offset boundary	2**Offset boundary	Buffer offsets
0	2**0 = 1	Byte boundaries
1	2**1 = 2	Even-byte boundaries
2	2**2 = 4	Four-byte boundaries
3	2**3 = 8	Eight-byte boundaries
4	2**4 = 16	16-byte boundaries
.	.	.
.	.	.
FFh	Not applicable	0 is the only supported buffer offset.

The buffer capacity field shall return the size of the selected buffer in bytes.

NOTE 82 In a multi-tasking system, a buffer may be altered between the WRITE BUFFER and READ BUFFER commands by another task. Buffer testing applications may wish to ensure that only a single task is active. Use of reservations (to all logical units on the device) or linked commands may also be helpful in avoiding buffer alteration between these two commands.

8.2.13 RECEIVE DIAGNOSTIC RESULTS command

The RECEIVE DIAGNOSTIC RESULTS command (see table 63) requests analysis data be sent to the initiator afte completion of a SEND DIAGNOSTIC command (see 8.2.15). If the target supports the optional page format, the page code field sent in the previous SEND DIAGNOSTIC command specifies the format of the returned data.

Table 63 - RECEIVE DIAGNOSTIC RESULTS command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (1Ch)							
1	Logical unit number			Reserved				
2	Reserved							
3	(MSB) Allocation length							
4								(LSB)
5	Control							

NOTES

83 To ensure that the diagnostic command information is not destroyed by a command sent from another initiator, either the SEND DIAGNOSTIC command should either be linked to the RECEIVE DIAGNOSTIC RESULTS command or the logical unit should be reserved.

84 Although diagnostic software is generally device-specific, this command and the SEND DIAGNOSTIC command provide a means to isolate the operating system software from the device-specific diagnostic software. Hence, the operating system can remain device-independent. This also allows diagnostic software to be transferred more easily to other operating systems.

See 8.3.1 for RECEIVE DIAGNOSTIC RESULTS page format definitions.

8.2.14 REQUEST SENSE command

The REQUEST SENSE command (see table 64) requests that the target transfer sense data to the initiator.

Table 64 - REQUEST SENSE command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (03h)							
1	Logical unit number			Reserved				
2	Reserved							
3	Reserved							
4	Allocation length							
5	Control							

The sense data:

a) shall be available if a contingent allegiance condition exists for the I_T_x nexus;
b) shall be available if other information (e.g. medium position) is available in any field;
c) may be available if an unexpected disconnect occurred.

If the target has no other sense data available to return, it shall return a sense key of NO SENSE and an additional sense code of NO ADDITIONAL SENSE INFORMATION.

The sense data shall be preserved by the target fo the initiator until retrieved by a REQUEST SENSE command or until the receipt of any other I/O process for the same I_T_x nexus. Sense data shall be cleared upon receipt of any subsequent I/O process (including REQUEST SENSE) to the same I_T_x nexus.

NOTE 85 Some target implementations do not update sense data except on commands that return CHECK CONDITION or COMMAND TERMINATED status. Thus when polling for a logical unit to become ready, the initiator should issue TEST UNIT READY commands until GOOD status is returned. If desired, the initiator may issue REQUEST SENSE commands after the TEST UNIT READY commands that return CHECK CONDITION or COMMAND TERMINATED status to obtain the sense data.

The target shall return CHECK CONDITION status for a REQUEST SENSE command only to report exception conditions specific to the command itself. For example:

a) A non-zero reserved bit is detected in the command descriptor block;
b) An unrecovered parity error is detected on the data bus;
c) A target malfunction prevents return of the sense data.

If a recovered error occurs during the execution of the REQUEST SENSE command, the target shall return the sense data with GOOD status. If a target returns CHECK CONDITION status for a REQUEST SENSE command, the sense data may be invalid.

NOTE 86 The sense data appropriate to the selection of an invalid logical unit is defined in 7.5.3.

Targets shall be capable of returning eighteen bytes of data in response to a REQUEST SENSE command. If the allocation length is eighteen or greater, and a target returns less than eighteen bytes of data, the initiator should assume that the bytes not transferred would have been zeros had the target returned those bytes. Initiators can determine how much sense data has been returned by examining the allocation length parameter in the command descriptor block and the additional sense length in the sense data. Targets shall not adjust the additional sense length to reflect truncation if the allocation length is less than the sense data available.

The sense data format for error codes 70h (current errors) and 71h (deferred errors) are defined in table 65. Error code values of 72h to 7Eh are reserved. Error code 7Fh is for a vendor-specific sense data format. Targets shall implement error code 70h; implementation of error code 71h is optional. Error code values of 00h to 6Fh are not defined by this standard and their use is not recommended.

Table 65 - Error codes 70h and 71h sense data format

Bit / Byte	7	6	5	4	3	2	1	0
0	Valid	Error code (70h or 71h)						
1	Segment number							
2	Filemark	EOM	ILI	Reserved	Sense key			
3 – 6	(MSB) Information (LSB)							
7	Additional sense length (n-7)							
8 – 11	(MSB) Command-specific information (LSB)							
12	Additional sense code							
13	Additional sense code qualifier							
14	Field replaceable unit code							
15 – 17	SKSV	Sense-key specific						
18 – n	Additional sense bytes							

A valid bit of zero indicates that the information field is not as defined in this standard. A valid bit of one indicates the information field contains valid information as defined in this standard. Targets shall implement the valid bit.

The segment number field contains the number of the current segment descriptor if the REQUEST SENSE command is in response to a COPY, COMPARE, or COPY AND VERIFY command. Up to 256 segments are supported, beginning with segment zero.

The filemark bit is mandatory for sequential-access devices, and this bit is reserved for all other device types. A filemark bit of one indicates that the current command has read a filemark or setmark. The additional sense code field may be used to indicate whether a filemark or setmark was read. Reporting of setmarks is optional and indicated by the Rsmk bit for sequential-access devices in the configuration parameters page (see 10.3.3.1).

The end-of-medium (EOM) bit is mandatory for sequential-access and printer devices, and this bit is reserved for all other device types. An EOM bit of one indicates that an end-of-medium condition (end-of-partition, beginning-of-partition, out-of-paper, etc.) exists. For sequential-access devices, this bit indicates that the unit is at or past the early-warning if the direction was forward, or that the command could not be completed because beginning-of-partition was encountered if the direction was reverse.

An incorrect length indicator (ILI) bit of one usually indicates that the requested logical block length did not match the logical block length of the data on the medium.

The sense key, additional sense code and additional sense code qualifier provide a hierarchy of information. The intention of the hierarchy is to provide a top-down approach for an initiator to determine information relating to the error and exception conditions. The sense key provides generic categories in which error and exception conditions can be reported. Initiators would typically use sense keys for high level error recovery procedures. Additional sense codes provide further detail describing the sense key. Additional sense code qualifiers add further detail to the

additional sense code. The additional sense code and additional sense code qualifier can be used by initiators where sophisticated error recovery procedures require detailed information describing the error and exception conditions.

The sense key field is mandatory and indicates generic information describing an error or exception condition. The sense keys are defined in 8.2.14.3.

The contents of the information field is device-type or command specific and is defined within the appropriate clause for the device type or command of interest. Targets shall implement the information field. Unless specified otherwise, this field contains:

a) the unsigned logical block address associated with the sense key, for direct-access devices (device type 0), write-once devices (device type 4), CD-ROM devices (device type 5), and optical memory devices (device type 7).
b) the difference (residue) of the requested length minus the actual length in either bytes or blocks, as determined by the command, for sequential-access devices (device type 1), printer devices (device type 2), processor devices (device type 3) and some direct access device commands, except as defined for d) below. (Negative values are indicated by two's complement notation.)
c) the difference (residue) of the requested number of blocks minus the actual number of blocks copied or compared for the current segment descriptor of a COPY, COMPARE, or COPY AND VERIFY command.
d) For sequential-access devices operating in buffered modes 1h or 2h that detect an unrecoverable write error when unwritten data blocks, filemarks, or setmarks remain in the buffer, the value of the information field for all commands shall be:
    1) the total number of data blocks, filemarks, and setmarks in the buffer if the device is in fixed block mode (block length field of the MODE SENSE block descriptor is non-zero and the fixed bit of the WRITE command is one).
    2) the number of bytes in the buffer, including filemarks and setmarks, if the device is in variable mode (the fixed bit of the WRITE command is zero).

The additional sense length field indicates the number of additional sense bytes to follow. If the allocation length of the command descriptor block is too small to transfer all of the additional sense bytes, the additional sense length is not adjusted to reflect the truncation.

The command-specific information field contains information that depends on the command that was executed. Further meaning for this field is defined within the command description. The command-specific information field is mandatory if the target supports any of the following commands: COPY, COMPARE, COPY AND VERIFY, SEARCH DATA, and REASSIGN BLOCKS.

The additional sense code (ASC) field indicates further information related to the error or exception condition reported in the sense key field. Targets shall support the additional sense code field. Support of the additional sense codes not explicitly required by this standard is optional. A list of additional sense codes is in 8.2.14.3. If the target does not have further information related to the error or exception condition, the additional sense code is set to NO ADDITIONAL SENSE INFORMATION.

The additional sense code qualifier (ASCQ) indicates detailed information related to the additional sense code. The additional sense code qualifier is optional. If the error or exception condition is reportable by the device, the value returned shall be as specified in 8.2.14.3. If the target does not have detailed information related to the error or exception condition, the additional sense code qualifier is set to zero.

Non-zero values in the field replaceable unit code field are used to define a device-specific mechanism or unit that has failed. A value of zero in this field shall indicate that no specific mechanism or unit has been identified to have failed or that the data is not available. The field replaceable unit code field is optional. The format of this information is not specified by this standard. Additional information about the field replaceable unit may be available in the ASCII information page (see 8.3.4.2), if supported by the target.

The sense-key specific bytes are described in 8.2.14.1, below.

The additional sense bytes field may contain command specific data, peripheral device specific data, or vendor-specific data that further defines the nature of the CHECK CONDITION status.

8.2.14.1 Sense-key specific

The sense-key specific field as defined by this standard when the value of the sense-key specific valid (SKSV) bit is one. The sense-key specific valid bit and sense-key specific field are optional. The definition of this field is determined by the value of the sense key field. This field is reserved for sense keys not described below. An SKSV value of zero indicates that this field is not as defined by this standard.

If the sense key field is set to ILLEGAL REQUEST and the SKSV bit is set to one, the sense-key specific field shall be as defined as shown in table 66. The field pointer field indicates which illegal parameters in the command descriptor block or the data parameters are in error.

Table 66 - Field pointer bytes

Bit / Byte	7	6	5	4	3	2	1	0
15	SKSV	C/D	Reserved	Reserved	BPV	Bit pointer		
16	(MSB)	Field pointer						
17								(LSB)

A command data (C/D) bit of one indicates that the illegal parameter is in the command descriptor block. A C/D bit of zero indicates that the illegal parameter is in the data parameters sent by the initiator during the DATA OUT phase.

A bit pointer valid (BPV) bit of zero indicates that the value in the bit pointer field is not valid. A BPV bit of one indicates that the bit pointer field specifies which bit of the byte designated by the field pointer field is in error. When a multiple-bit field is in error, the bit pointer field shall point to the most-significant (left-most) bit of the field.

The field pointer field indicates which byte of the command descriptor block or of the parameter data was in error. Bytes are numbered starting from zero, as shown in the tables describing the commands and parameters. When a multiple-byte field is in error, the pointer shall point to the most- significant (left-most) byte of the field.

NOTE 87 Bytes identified as being in error are not necessarily the place that has to be changed to correct the problem.

If the sense key is RECOVERED ERROR, HARDWARE ERROR or MEDIUM ERROR and if the SKSV bit is one, the sense-key specific field shall be as shown in table 67.

Table 67 - Actual retry count bytes

Bit / Byte	7	6	5	4	3	2	1	0
15	SKSV	Reserved						
16	(MSB)	Actual retry count						
17								(LSB)

The actual retry count field returns implementation-specific information on the actual number of retries of the recovery algorithm used in attempting to recover an error or exception condition.

NOTE 88 This field should relate to the retry count fields within the error recovery page of the MODE SELECT command.

If the sense key is NOT READY and the SKSV bit is one, the sense-key specific field shall be as shown in table 68. These fields only apply to the FORMAT UNIT command with the Immed bit set to one.

Table 68 - Format progress indication bytes

Bit / Byte	7	6	5	4	3	2	1	0
15	SKSV	Reserved						
16	(MSB)	Progress indication						
17								(LSB)

The progress indication field is a percent complete indication in which the returned value is the numerator that has 65 536 (10000h) as its denominator. The progress indication shall based upon the total format operation including any certification or initialization operations.

NOTE 89 It is intended that the progress indication be time related. However, since format time varies with the number of defects encountered, etc., it is reasonable for the target to assign values to various steps within the process. The granularity of these steps should be small enough to provide reasonable assurances to the initiator that progress is being made.

8.2.14.2 Deferred errors

Error code 70h indicates that the CHECK CONDITION or COMMAND TERMINATED status returned is the result of an error or exception condition on the I/O process that returned the CHECK CONDITION or COMMAND TERMINATED status or an unexpected disconnect. This includes errors generated during execution of the command by the actual execution process. It also includes errors not related to any command that are first observed during execution of a command. Examples of this latter type of error include disk servo-mechanism, off-track errors, and power-up test errors.

Error code 71h (deferred error) indicates that the CHECK CONDITION status returned is the result of an error or exception condition that occurred during execution of a previous command for which GOOD status has already been returned. Such commands are associated with use of the immediate bit, with some forms of caching, and with multiple command buffering. Targets that implement these features are required to implement deferred error reporting.

The deferred error indication may be sent at a time selected by the target through the asynchronous event notification process (see 7.5.5) if AEN is supported by both the initiator and target.

If AEN is not supported, the deferred error may be indicated by returning CHECK CONDITION status to the appropriate initiator as described below. The subsequent execution of a REQUEST SENSE command shall return the deferred error sense information.

If an I/O process terminates with CHECK CONDITION status and the subsequent sense data returns a deferred error that I/O process shall not have been executed. After the target detects a deferred error condition on a logical unit, it shall return a deferred error according to the rules described below:

a) If a deferred error can be recovered with no external system intervention, a deferred error indication shall not be posted unless required by the error handling parameters of the MODE SELECT command. The occurrence of the error may be logged if statistical or error logging is supported.
b) If a deferred error can be associated with a causing initiator and with a particular function or a particular subset of data, and the error is either unrecovered or required to be reported by the mode parameters, a deferred error indication shall be returned to the causing initiator. If an initiator other than the causing initiator attempts access to the particular function or subset of data associated with the deferred error, a BUSY status shall be returned to that initiator in response to the command attempting the access.

NOTE 90 Not all devices may be sufficiently sophisticated to identify the function or data that failed. Those that cannot should treat the error in the following manner.

c) If a deferred error cannot be associated with a causing initiator or with a particular subset of data, a deferrec error indication shall be returned on behalf of the failing logical unit to each initiator. If multiple deferred errors have accumulated for some initiators, only the last error shall be returned.
d) If a deferred error cannot be associated with a particular logical unit, it shall be returned to the appropriate initiator for all logical units supported by the target.
e) If a current command has not yet started executing, and a deferred error occurs, the command shall be terminated with CHECK CONDITION status and deferred error information posted in the sense data. By convention, the current command is considered to have started execution if the target has changed phase from the COMMAND phase to the next normal phase of the command sequence. If a deferred error occurs while a current command is executing and the current command has been affected by the error, the command shall be terminated by CHECK CONDITION status and current error information shall be returned in the sense data. In this case, if the current error information does not adequately define the deferred error condition, a deferrec error may be returned after the current error information has been recovered. If a deferred error occurs while a current command is executing and the current command completes successfully, the target may choose to return the deferred error information after the completion of the current command.

NOTE 91 Deferred errors may indicate that an operation was unsuccessful long after the command performing the data transfer returned GOOD status. If data that cannot be replicated or recovered from other sources is being stored using buffered write operations, synchronization commands should be performed before the critical data is destroyed in the host initiator. This is necessary to be sure that recovery actions can be taken if deferred errors do occur in the storing of the data. If AEN is not implemented, the synchronizing process should provide the necessary commands to allow returning CHECK CONDITION status and subsequent returning of deferred error sense information after all buffered operations are guaranteed to be complete.

8.2.14.3 Sense key and sense code definitions

The sense keys are defined in tables 69 and 70.

Table 69 - Sense key (0h-7h) descriptions

Sense key	Description
0h	NO SENSE. Indicates that there is no specific sense key information to be reported for the designated logical unit. This would be the case for a successful command or a command that received CHECK CONDITION or COMMAND TERMINATED status because one of the filemark, EOM, or ILI bits is set to one.
1h	RECOVERED ERROR. Indicates that the last command completed successfully with some recovery action performed by the target. Details may be determinable by examining the additional sense bytes and the information field. When multiple recovered errors occur during one command, the choice of which error to report (first, last, most severe, etc.) is device specific.
2h	NOT READY. Indicates that the logical unit addressed cannot be accessed. Operator intervention may be required to correct this condition.
3h	MEDIUM ERROR. Indicates that the command terminated with a non-recovered error condition that was probably caused by a flaw in the medium or an error in the recorded data. This sense key may also be returned if the target is unable to distinguish between a flaw in the medium and a specific hardware failure (sense key 4h).
4h	HARDWARE ERROR. Indicates that the target detected a non-recoverable hardware failure (for example, controller failure, device failure, parity error, etc.) while performing the command or during a self test.
5h	ILLEGAL REQUEST. Indicates that there was an illegal parameter in the command descriptor block or in the additional parameters supplied as data for some commands (FORMAT UNIT, SEARCH DATA, etc.). If the target detects an invalid parameter in the command descriptor block, then it shall terminate the command without altering the medium. If the target detects an invalid parameter in the additional parameters supplied as data, then the target may have already altered the medium. This sense key may also indicate that an invalid IDENTIFY message was received (6.6.7).
6h	UNIT ATTENTION. Indicates that the removable medium may have been changed or the target has been reset. See 7.9 for more detailed information about the unit attention condition.
7h	DATA PROTECT. Indicates that a command that reads or writes the medium was attempted on a block that is protected from this operation. The read or write operation is not performed.

Table 70 - Sense key (8h-Fh) descriptions

Sense key	Description
8h	BLANK CHECK. Indicates that a write-once device or a sequential-access device encountered blank medium or format-defined end-of-data indication while reading or a write-once device encountered a non-blank medium while writing.
9h	VENDOR-SPECIFIC. This sense key is available for reporting vendor specific conditions.
Ah	COPY ABORTED. Indicates a COPY, COMPARE, or COPY AND VERIFY command was aborted due to an error condition on the source device, the destination device, or both. (See 8.2.3.2 for additional information about this sense key.)
Bh	ABORTED COMMAND. Indicates that the target aborted the command. The initiator may be able to recover by trying the command again.
Ch	EQUAL. Indicates a SEARCH DATA command has satisfied an equal comparison.
Dh	VOLUME OVERFLOW. Indicates that a buffered peripheral device has reached the end-of-partition and data may remain in the buffer that has not been written to the medium. A RECOVER BUFFERED DATA command(s) may be issued to read the unwritten data from the buffer.
Eh	MISCOMPARE. Indicates that the source data did not match the data read from the medium.
Fh	RESERVED.

The additional sense codes and additional sense code qualifiers are defined in table 71.

Table 71 - ASC and ASCQ assignments

```
                D - DIRECT ACCESS DEVICE
                .T - SEQUENTIAL ACCESS DEVICE
                . L - PRINTER DEVICE
                .  P - PROCESSOR DEVICE
                .  .W - WRITE ONCE READ MULTIPLE DEVICE
                .  . R - READ ONLY (CD-ROM) DEVICE
                .  .  S - SCANNER DEVICE
                .  .  .O - OPTICAL MEMORY DEVICE
                .  .  . M - MEDIA CHANGER DEVICE
                .  .  .  C - COMMUNICATION DEVICE
                .  .  .  .
ASC  ASCQ       DTLPWRSOMC  DESCRIPTION
---  ---        ----------  -----------
13h  00h        D   W  O    ADDRESS MARK NOT FOUND FOR DATA FIELD
12h  00h        D   W  O    ADDRESS MARK NOT FOUND FOR ID FIELD
00h  11h           R        AUDIO PLAY OPERATION IN PROGRESS
00h  12h           R        AUDIO PLAY OPERATION PAUSED
00h  14h           R        AUDIO PLAY OPERATION STOPPED DUE TO ERROR
00h  13h           R        AUDIO PLAY OPERATION SUCCESSFULLY COMPLETED
00h  04h    T     S         BEGINNING-OF-PARTITION/MEDIUM DETECTED
14h  04h    T               BLOCK SEQUENCE ERROR
30h  02h    DT  WR O        CANNOT READ MEDIUM - INCOMPATIBLE FORMAT
30h  01h    DT  WR O        CANNOT READ MEDIUM - UNKNOWN FORMAT
52h  00h    T               CARTRIDGE FAULT
3Fh  02h    DTLPWRSOMC      CHANGED OPERATING DEFINITION
11h  06h        WR O        CIRC UNRECOVERED ERROR
30h  03h    DT              CLEANING CARTRIDGE INSTALLED
4Ah  00h    DTLPWRSOMC      COMMAND PHASE ERROR
2Ch  00h    DTLPWRSOMC      COMMAND SEQUENCE ERROR
2Fh  00h    DTLPWRSOMC      COMMANDS CLEARED BY ANOTHER INITIATOR
2Bh  00h    DTLPWRSO C      COPY CANNOT EXECUTE SINCE HOST CANNOT DISCONNECT
41h  00h    D               DATA PATH FAILURE (SHOULD USE 40 NN)
4Bh  00h    DTLPWRSOMC      DATA PHASE ERROR
11h  07h        W  O        DATA RESYNCHRONIZATION ERROR
16h  00h    D   W  O        DATA SYNCHRONIZATION MARK ERROR
19h  00h    D      O        DEFECT LIST ERROR
19h  03h    D      O        DEFECT LIST ERROR IN GROWN LIST
19h  02h    D      O        DEFECT LIST ERROR IN PRIMARY LIST
19h  01h    D      O        DEFECT LIST NOT AVAILABLE
1Ch  00h    D      O        DEFECT LIST NOT FOUND
32h  01h    D   W  O        DEFECT LIST UPDATE FAILURE
40h  NNh    DTLPWRSOMC      DIAGNOSTIC FAILURE ON COMPONENT NN (80H-FFH)
63h  00h           R        END OF USER AREA ENCOUNTERED ON THIS TRACK
00h  05h    T     S         END-OF-DATA DETECTED
14h  03h    T               END-OF-DATA NOT FOUND
00h  02h    T     S         END-OF-PARTITION/MEDIUM DETECTED
51h  00h    T      O        ERASE FAILURE
0Ah  00h    DTLPWRSOMC      ERROR LOG OVERFLOW
11h  02h    DT  W SO        ERROR TOO LONG TO CORRECT
03h  02h    T               EXCESSIVE WRITE ERRORS
3Bh  07h     L              FAILED TO SENSE BOTTOM-OF-FORM
3Bh  06h     L              FAILED TO SENSE TOP-OF-FORM
00h  01h    T               FILEMARK DETECTED
14h  02h    T               FILEMARK OR SETMARK NOT FOUND
09h  02h        WR O        FOCUS SERVO FAILURE
31h  01h    D L     O       FORMAT COMMAND FAILED
58h  00h            O       GENERATION DOES NOT EXIST
```

Table 71 (continued)

ASC	ASCQ	DTLPWRSOMC	DESCRIPTION
1Ch	02h	D O	GROWN DEFECT LIST NOT FOUND
00h	06h	DTLPWRSOMC	I/O PROCESS TERMINATED
10h	00h	D W O	ID CRC OR ECC ERROR
22h	00h	D	ILLEGAL FUNCTION (SHOULD USE 20 00, 24 00, OR 26 00)
64h	00h	R	ILLEGAL MODE FOR THIS TRACK
28h	01h	M	IMPORT OR EXPORT ELEMENT ACCESSED
30h	00h	DT WR OM	INCOMPATIBLE MEDIUM INSTALLED
11h	08h	T	INCOMPLETE BLOCK READ
48h	00h	DTLPWRSOMC	INITIATOR DETECTED ERROR MESSAGE RECEIVED
3Fh	03h	DTLPWRSOMC	INQUIRY DATA HAS CHANGED
44h	00h	DTLPWRSOMC	INTERNAL TARGET FAILURE
3Dh	00h	DTLPWRSOMC	INVALID BITS IN IDENTIFY MESSAGE
2Ch	02h	S	INVALID COMBINATION OF WINDOWS SPECIFIED
20h	00h	DTLPWRSOMC	INVALID COMMAND OPERATION CODE
21h	01h	M	INVALID ELEMENT ADDRESS
24h	00h	DTLPWRSOMC	INVALID FIELD IN CDB
26h	00h	DTLPWRSOMC	INVALID FIELD IN PARAMETER LIST
49h	00h	DTLPWRSOMC	INVALID MESSAGE ERROR
11h	05h	WR O	L-EC UNCORRECTABLE ERROR
60h	00h	S	LAMP FAILURE
5Bh	02h	DTLPWRSOM	LOG COUNTER AT MAXIMUM
5Bh	00h	DTLPWRSOM	LOG EXCEPTION
5Bh	03h	DTLPWRSOM	LOG LIST CODES EXHAUSTED
2Ah	02h	DTL WRSOMC	LOG PARAMETERS CHANGED
21h	00h	DT WR OM	LOGICAL BLOCK ADDRESS OUT OF RANGE
08h	00h	DTL WRSOMC	LOGICAL UNIT COMMUNICATION FAILURE
08h	02h	DTL WRSOMC	LOGICAL UNIT COMMUNICATION PARITY ERROR
08h	01h	DTL WRSOMC	LOGICAL UNIT COMMUNICATION TIME-OUT
05h	00h	DTLPWRSOMC	LOGICAL UNIT DOES NOT RESPOND TO SELECTION
4Ch	00h	DTLPWRSOMC	LOGICAL UNIT FAILED SELF-CONFIGURATION
3Eh	00h	DTLPWRSOMC	LOGICAL UNIT HAS NOT SELF-CONFIGURED YET
04h	01h	DTLPWRSOMC	LOGICAL UNIT IS IN PROCESS OF BECOMING READY
04h	00h	DTLPWRSOMC	LOGICAL UNIT NOT READY, CAUSE NOT REPORTABLE
04h	04h	DTL O	LOGICAL UNIT NOT READY, FORMAT IN PROGRESS
04h	02h	DTLPWRSOMC	LOGICAL UNIT NOT READY, INITIALIZING COMMAND REQUIRED
04h	03h	DTLPWRSOMC	LOGICAL UNIT NOT READY, MANUAL INTERVENTION REQUIRED
25h	00h	DTLPWRSOMC	LOGICAL UNIT NOT SUPPORTED
15h	01h	DTL WRSOM	MECHANICAL POSITIONING ERROR
53h	00h	DTL WRSOM	MEDIA LOAD OR EJECT FAILED
3Bh	0Dh	M	MEDIUM DESTINATION ELEMENT FULL
31h	00h	DT W O	MEDIUM FORMAT CORRUPTED
3Ah	00h	DTL WRSOM	MEDIUM NOT PRESENT
53h	02h	DT WR OM	MEDIUM REMOVAL PREVENTED
3Bh	0Eh	M	MEDIUM SOURCE ELEMENT EMPTY
43h	00h	DTLPWRSOMC	MESSAGE ERROR
3Fh	01h	DTLPWRSOMC	MICROCODE HAS BEEN CHANGED
1Dh	00h	D W O	MISCOMPARE DURING VERIFY OPERATION
11h	0Ah	DT O	MISCORRECTED ERROR
2Ah	01h	DTL WRSOMC	MODE PARAMETERS CHANGED
07h	00h	DTL WRSOM	MULTIPLE PERIPHERAL DEVICES SELECTED
11h	03h	DT W SO	MULTIPLE READ ERRORS
00h	00h	DTLPWRSOMC	NO ADDITIONAL SENSE INFORMATION
00h	15h	R	NO CURRENT AUDIO STATUS TO RETURN
32h	00h	D W O	NO DEFECT SPARE LOCATION AVAILABLE
11h	09h	T	NO GAP FOUND
01h	00h	D W O	NO INDEX/SECTOR SIGNAL
06h	00h	D WR OM	NO REFERENCE POSITION FOUND

Table 71 (continued)

ASC	ASCQ	DTLPWRSOMC	DESCRIPTION
02h	00h	D WR OM	NO SEEK COMPLETE
03h	01h	T	NO WRITE CURRENT
28h	00h	DTLPWRSOMC	NOT READY TO READY TRANSITION, MEDIUM MAY HAVE CHANGED
5Ah	01h	DT WR OM	OPERATOR MEDIUM REMOVAL REQUEST
5Ah	00h	DTLPWRSOM	OPERATOR REQUEST OR STATE CHANGE INPUT (UNSPECIFIED)
5Ah	03h	DT W O	OPERATOR SELECTED WRITE PERMIT
5Ah	02h	DT W O	OPERATOR SELECTED WRITE PROTECT
61h	02h	S	OUT OF FOCUS
4Eh	00h	DTLPWRSOMC	OVERLAPPED COMMANDS ATTEMPTED
2Dh	00h	T	OVERWRITE ERROR ON UPDATE IN PLACE
3Bh	05h	L	PAPER JAM
1Ah	00h	DTLPWRSOMC	PARAMETER LIST LENGTH ERROR
26h	01h	DTLPWRSOMC	PARAMETER NOT SUPPORTED
26h	02h	DTLPWRSOMC	PARAMETER VALUE INVALID
2Ah	00h	DTL WRSOMC	PARAMETERS CHANGED
03h	00h	DTL W SO	PERIPHERAL DEVICE WRITE FAULT
50h	02h	T	POSITION ERROR RELATED TO TIMING
3Bh	0Ch	S	POSITION PAST BEGINNING OF MEDIUM
3Bh	0Bh	S	POSITION PAST END OF MEDIUM
15h	02h	DT WR O	POSITIONING ERROR DETECTED BY READ OF MEDIUM
29h	00h	DTLPWRSOMC	POWER ON, RESET, OR BUS DEVICE RESET OCCURRED
42h	00h	D	POWER-ON OR SELF-TEST FAILURE (SHOULD USE 40 NN)
1Ch	01h	D O	PRIMARY DEFECT LIST NOT FOUND
40h	00h	D	RAM FAILURE (SHOULD USE 40 NN)
15h	00h	DTL WRSOM	RANDOM POSITIONING ERROR
3Bh	0Ah	S	READ PAST BEGINNING OF MEDIUM
3Bh	09h	S	READ PAST END OF MEDIUM
11h	01h	DT W SO	READ RETRIES EXHAUSTED
14h	01h	DT WR O	RECORD NOT FOUND
14h	00h	DTL WRSO	RECORDED ENTITY NOT FOUND
18h	02h	D WR O	RECOVERED DATA - DATA AUTO-REALLOCATED
18h	05h	D WR O	RECOVERED DATA - RECOMMEND REASSIGNMENT
18h	06h	D WR O	RECOVERED DATA - RECOMMEND REWRITE
17h	05h	D WR O	RECOVERED DATA USING PREVIOUS SECTOR ID
18h	03h	R	RECOVERED DATA WITH CIRC
18h	01h	D WR O	RECOVERED DATA WITH ERROR CORRECTION & RETRIES APPLIED
18h	00h	DT WR O	RECOVERED DATA WITH ERROR CORRECTION APPLIED
18h	04h	R	RECOVERED DATA WITH L-EC
17h	03h	DT WR O	RECOVERED DATA WITH NEGATIVE HEAD OFFSET
17h	00h	DT WRSO	RECOVERED DATA WITH NO ERROR CORRECTION APPLIED
17h	02h	DT WR O	RECOVERED DATA WITH POSITIVE HEAD OFFSET
17h	01h	DT WRSO	RECOVERED DATA WITH RETRIES
17h	04h	WR O	RECOVERED DATA WITH RETRIES AND/OR CIRC APPLIED
17h	06h	D W O	RECOVERED DATA WITHOUT ECC - DATA AUTO-REALLOCATED
17h	07h	D W O	RECOVERED DATA WITHOUT ECC - RECOMMEND REASSIGNMENT
17h	08h	D W O	RECOVERED DATA WITHOUT ECC - RECOMMEND REWRITE
1Eh	00h	D W O	RECOVERED ID WITH ECC CORRECTION
3Bh	08h	T	REPOSITION ERROR
36h	00h	L	RIBBON, INK, OR TONER FAILURE
37h	00h	DTL WRSOMC	ROUNDED PARAMETER
5Ch	00h	D O	RPL STATUS CHANGE
39h	00h	DTL WRSOMC	SAVING PARAMETERS NOT SUPPORTED
62h	00h	S	SCAN HEAD POSITIONING ERROR
47h	00h	DTLPWRSOMC	SCSI PARITY ERROR
54h	00h	P	SCSI TO HOST SYSTEM INTERFACE FAILURE
45h	00h	DTLPWRSOMC	SELECT OR RESELECT FAILURE

Table 71 (concluded)

ASC	ASCQ	DTLPWRSOMC	DESCRIPTION
3Bh	00h	TL	SEQUENTIAL POSITIONING ERROR
00h	03h	T	SETMARK DETECTED
3Bh	04h	L	SLEW FAILURE
09h	03h	WR O	SPINDLE SERVO FAILURE
5Ch	02h	D O	SPINDLES NOT SYNCHRONIZED
5Ch	01h	D O	SPINDLES SYNCHRONIZED
1Bh	00h	DTLPWRSOMC	SYNCHRONOUS DATA TRANSFER ERROR
55h	00h	P	SYSTEM RESOURCE FAILURE
33h	00h	T	TAPE LENGTH ERROR
3Bh	03h	L	TAPE OR ELECTRONIC VERTICAL FORMS UNIT NOT READY
3Bh	01h	T	TAPE POSITION ERROR AT BEGINNING-OF-MEDIUM
3Bh	02h	T	TAPE POSITION ERROR AT END-OF-MEDIUM
3Fh	00h	DTLPWRSOMC	TARGET OPERATING CONDITIONS HAVE CHANGED
5Bh	01h	DTLPWRSOM	THRESHOLD CONDITION MET
26h	03h	DTLPWRSOMC	THRESHOLD PARAMETERS NOT SUPPORTED
2Ch	01h	S	TOO MANY WINDOWS SPECIFIED
09h	00h	DT WR O	TRACK FOLLOWING ERROR
09h	01h	WR O	TRACKING SERVO FAILURE
61h	01h	S	UNABLE TO ACQUIRE VIDEO
57h	00h	R	UNABLE TO RECOVER TABLE-OF-CONTENTS
53h	01h	T	UNLOAD TAPE FAILURE
11h	00h	DT WRSO	UNRECOVERED READ ERROR
11h	04h	D W O	UNRECOVERED READ ERROR - AUTO REALLOCATE FAILED
11h	0Bh	D W O	UNRECOVERED READ ERROR - RECOMMEND REASSIGNMENT
11h	0Ch	D W O	UNRECOVERED READ ERROR - RECOMMEND REWRITE THE DATA
46h	00h	DTLPWRSOMC	UNSUCCESSFUL SOFT RESET
59h	00h	O	UPDATED BLOCK READ
61h	00h	S	VIDEO ACQUISITION ERROR
50h	00h	T	WRITE APPEND ERROR
50h	01h	T	WRITE APPEND POSITION ERROR
0Ch	00h	T S	WRITE ERROR
0Ch	02h	D W O	WRITE ERROR - AUTO REALLOCATION FAILED
0Ch	01h	D W O	WRITE ERROR RECOVERED WITH AUTO REALLOCATION
27h	00h	DT W O	WRITE PROTECTED
80h THROUGH FFh	XXh XX		Vendor-specific.
XXh THROUGH XXh	80h FFh		Vendor-specific QUALIFICATION OF STANDARD ASC.
			ALL CODES NOT SHOWN ARE RESERVED.

NOTE - Annex D contains the ASC and ASCQ assignments in numeric order.

8.2.15 SEND DIAGNOSTIC command

The SEND DIAGNOSTIC command (see table 72) requests the target to perform diagnostic operations on itself, on the logical unit, or on both. The only mandatory implementation of this command is the self-test feature with the parameter list length of zero. Except when the self-test bit is one, this command is usually followed by a RECEIVE DIAGNOSTIC RESULTS command.

Table 72 - SEND DIAGNOSTIC command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (1Dh)							
1	Logical unit number			PF	Reserved	SelfTest	DevOfL	UnitOfL
2	Reserved							
3	(MSB)			Parameter list length				
4								(LSB)
5	Control							

A page format (PF) bit of one specifies that the SEND DIAGNOSTIC parameters conform to the page structure as specified in this standard. The implementation of the PF bit is optional. See 8.3.1 for the definition of diagnostic pages. A PF bit of zero indicates that the SEND DIAGNOSTIC parameters are as specified in SCSI-1 (i.e. all parameters are vendor-specific).

A self-test (SelfTest) bit of one directs the target to complete its default self-test. If the self-test successfully passes, the command shall be terminated with GOOD status; otherwise, the command shall be terminated with CHECK CONDITION status and the sense key shall be set to HARDWARE ERROR.

A self-test bit of zero requests that the target perform the diagnostic operation specified in the parameter list. The diagnostic operation might or might not require a target to return data that contains diagnostic results. If the return of data is not required, the return of GOOD status indicates successful completion of the diagnostic operation. If the return of data is required, the target shall either:

a) perform the requested diagnostic operation, prepare the data to be returned and indicate completion by returning GOOD status. The initiator issues a RECEIVE DIAGNOSTIC RESULTS command to recover the data;
b) accept the parameter list, and if no errors are detected in the parameter list, return GOOD status. The requested diagnostic operation and the preparation of the data to be returned are performed upon receipt of a RECEIVE DIAGNOSTIC RESULTS command.

NOTE 92 To insure that the diagnostic command information is not destroyed by a command sent from another initiator, either the SEND DIAGNOSTIC command should be linked to the RECEIVE DIAGNOSTIC RESULTS command or the logical unit should be reserved.

The device off-line (DevOfL) and unit off-line (UnitOfL) bits are generally set by operating system software, while the parameter list is prepared by diagnostic application software. These bits grant permission to perform vendor-specific diagnostic operations on the target that may be visible to attached initiators. Thus, by preventing operations that are not enabled by these bits, the target assists the operating system in protecting its resources.

A UnitOfL bit of one grants permission to the target to perform diagnostic operations that may affect the user accessible medium on the logical unit, e.g. write operations to the user accessible medium, or repositioning of the medium on sequential access devices. The implementation of the UnitOfL bit is optional. A UnitOfL bit of zero prohibits any diagnostic operations that may be detected by subsequent I/O processes.

A DevOfL bit of one grants permission to the target to perform diagnostic operations that may affect all the logical units on a target, e.g. alteration of reservations, log parameters, or sense data. The implementation of the DevOfL bit is optional. A DevOfL bit of zero prohibits diagnostic operations that may be detected by subsequent I/O processes.

The parameter list length field specifies the length in bytes of the parameter list that shall be transferred from the initiator to the target. A parameter list length of zero indicates that no data shall be transferred. This condition shall not be considered an error. If the specified parameter list length results in the truncation of one or more pages (PF bit set to one) the target shall return CHECK CONDITION status with a sense key of ILLEGAL REQUEST and an additional sense code of INVALID FIELD IN CDB.

See note 83 under the RECEIVE DIAGNOSTIC RESULTS command in 8.2.13.

8.2.16 TEST UNIT READY command

The TEST UNIT READY command (see table 73) provides a means to check if the logical unit is ready. This is not a request for a self-test. If the logical unit would accept an appropriate medium-access command without returning CHECK CONDITION status, this command shall return a GOOD status. If the logical unit cannot become operational or is in a state such that an initiator action (e.g. START UNIT command) is required to make the unit ready, the target shall return CHECK CONDITION status with a sense key of NOT READY.

Table 73 - TEST UNIT READY command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (00h)							
1	Logical unit number			Reserved				
2	Reserved							
3	Reserved							
4	Reserved							
5	Control							

Table 74 defines the preferred responses to the TEST UNIT READY command. Higher-priority responses (e.g. BUSY or RESERVATION CONFLICT) are also permitted.

Table 74 - Preferred TEST UNIT READY responses

Status	Sense key	ASC and ASCQ
GOOD	NO SENSE	NO ADDITIONAL SENSE INFORMATION or other valid additional sense code.
CHECK CONDITION	ILLEGAL REQUEST	LOGICAL UNIT NOT SUPPORTED
CHECK CONDITION	NOT READY	LOGICAL UNIT DOES NOT RESPOND TO SELECTION
CHECK CONDITION	NOT READY	MEDIUM NOT PRESENT
CHECK CONDITION	NOT READY	LOGICAL UNIT NOT READY, CAUSE NOT REPORTABLE
CHECK CONDITION	NOT READY	LOGICAL UNIT IS IN PROCESS OF BECOMING READY
CHECK CONDITION	NOT READY	LOGICAL UNIT NOT READY, INITIALIZING COMMAND REQUIRED
CHECK CONDITION	NOT READY	LOGICAL UNIT NOT READY, MANUAL INTERVENTION REQUIRED
CHECK CONDITION	NOT READY	LOGICAL UNIT NOT READY, FORMAT IN PROGRESS

8.2.17 WRITE BUFFER command

The WRITE BUFFER command (see table 75) is used in conjunction with the READ BUFFER command as a diagnostic for testing target memory and the SCSI bus integrity. Additional modes are provided for downloading microcode and for downloading and saving microcode.

Table 75 - WRITE BUFFER command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (3Bh)							
1	Logical unit number			Reserved		Mode		
2	Buffer ID							
3	(MSB)							
4	Buffer offset							
5								(LSB)
6	(MSB)							
7	Parameter list length							
8								(LSB)
9	Control							

This command shall not alter any medium of the target when the data mode or the combined header and data mode is specified.

The function of this command and the meaning of fields within the command descriptor block depend on the contents of the mode field. The mode field is defined in table 76.

Table 76 - WRITE BUFFER mode field

Mode	Description	Implementation requirements
000b	Write combined header and data	Optional
001b	Vendor-specific	Vendor-specific
010b	Write data	Optional
011b	Reserved	Reserved
100b	Download microcode	Optional
101b	Download microcode and save	Optional
110b	Reserved	Reserved
111b	Reserved	Reserved

NOTE 93 Modes 000b and 001b are included for compatibility with CCS products that were designed prior to the generation of this standard. These products restrict the maximum transfer length to 65 535 bytes.

8.2.17.1 Combined header and data mode (000b)

In this mode, data to be transferred is preceded by a four-byte header. The four-byte header consists of all reserved bytes. The buffer ID and the buffer offset fields shall be zero. The parameter list length field specifies the maximum number of bytes that shall be transferred during the DATA OUT phase. This number includes four bytes of header, so the data length to be stored in the target's buffer is parameter list length minus four. The initiator should attempt to ensure that the parameter list length is not greater than four plus the buffer capacity (see 8.2.12.1) that is returned in the header of the READ BUFFER command (mode 000b). If the parameter list length exceeds the buffer capacity target shall return CHECK CONDITION status and shall set the sense key to ILLEGAL REQUEST.

8.2.17.2 Vendor-specific mode (001b)

In this mode, the meaning of the buffer ID, buffer offset, and parameter list length fields are not specified by this standard.

8.2.17.3 Data mode (010b)

In this mode, the DATA OUT phase contains buffer data. The buffer ID field identifies a specific buffer within the target. The vendor assigns buffer ID codes to buffers within the target. Buffer ID zero shall be supported. If more than one buffer is supported, additional buffer ID codes shall be assigned contiguously, beginning with one. If an unsupported buffer ID code is selected, the target shall return CHECK CONDITION status and shall set the sense key to ILLEGAL REQUEST with an additional sense code of INVALID FIELD IN CDB.

Data are written to the target buffer starting at the location specified by the buffer offset. The initiator should conform to the offset boundary requirements returned in the READ BUFFER descriptor. If the target is unable to accept the specified buffer offset, it shall return CHECK CONDITION status and it shall set the sense key to ILLEGAL REQUEST with an additional sense code of INVALID FIELD IN CDB.

The parameter list length specifies the maximum number of bytes that shall be transferred during the DATA OUT phase to be stored in the specified buffer beginning at the buffer offset. The initiator should attempt to ensure that the parameter list length plus the buffer offset does not exceed the capacity of the specified buffer. (The capacity of the buffer can be determined by the buffer capacity field in the READ BUFFER descriptor.) If the buffer offset and parameter list length fields specify a transfer that would exceed the buffer capacity, the target shall return CHECK CONDITION status and shall set the sense key to ILLEGAL REQUEST with an additional sense code of INVALID FIELD IN CDB.

8.2.17.4 Download microcode mode (100b)

In this mode, vendor-specific microcode or control information shall be transferred to the control memory space of the target. After a power-cycle or reset, the device operation shall revert to a vendor-specific condition. The meanings of the buffer ID, buffer offset, and parameter list length fields are not specified by this standard and are not required to be zero-filled. When the microcode download has completed successfully the target shall generate a unit attention condition for all initiators except the one that issued the WRITE BUFFER command (see 7.9). The additional sense code shall be set to MICROCODE HAS BEEN CHANGED.

8.2.17.5 Download microcode and save mode (101b)

In this mode, vendor-specific microcode or control information shall be transferred to the target and, if the WRITE BUFFER command is completed successfully, also shall be saved in a non-volatile memory space (semiconductor, disk, or other). The downloaded code shall then be effective after each power-cycle and reset until it is supplanted in another download microcode and save operation. The meanings of the buffer ID, buffer offset, and parameter list length fields are not specified by this standard and are not required to be zero-filled. When the download microcode and save command has completed successfully the target shall generate a unit attention condition (see 7.9) for all initiators except the one that issued the WRITE BUFFER command. When reporting the unit attention condition, the target shall set the additional sense code to MICROCODE HAS BEEN CHANGED.

8.3 Parameters for all device types

8.3.1 Diagnostic parameters

This subclause describes the diagnostic page structure and the diagnostic pages that are applicable to all SCSI devices. Pages specific to each device type are described in the third subclause of each device-type clause (i.e. 9.3, 10.3, etc.).

A SEND DIAGNOSTIC command with a PF bit of one specifies that the SEND DIAGNOSTIC parameter list consists of zero or more diagnostic pages and that the data returned by the subsequent RECEIVE DIAGNOSTIC RESULTS command shall use the diagnostic page format (see table 77) described in this International Standard.

Table 77 - Diagnostic page format

Bit / Byte	7	6	5	4	3	2	1	0
0	Page code							
1	Reserved							
2	(MSB)			Page length (n-3)				
3								(LSB)
4	Diagnostic parameters							
n								

Each diagnostic page defines a function or operation that the target shall perform. The page contains a page header followed by the analysis data that is formatted according to the page code specified in the previous SEND DIAGNOSTIC command.

Targets that implement diagnostic pages are only required to accept a single diagnostic page per command.

The page code field identifies which diagnostic page is being sent or returned. The page codes are defined in table 78.

Table 78 - Diagnostic page codes

Page code	Description	Subclause
00h 01h - 3Fh 40h - 7Fh 80h - FFh	Supported diagnostics pages Reserved (for all device type pages) See specific device type for definition Vendor-specific pages	8.3.1.1

The page length field specifies the length in bytes of the diagnostic parameters that follow this field. If the initiator sends a page length that results in the truncation of any parameter, the target shall terminate the command with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST with the additional sense code set to INVALID FIELD IN PARAMETER LIST.

The diagnostic parameters are defined for each page code. The diagnostic parameters within a page may be defined differently in a SEND DIAGNOSTIC command than in a RECEIVE DIAGNOSTIC RESULTS command.

8.3.1.1 Supported diagnostic pages

The supported diagnostics page (see table 79) returns the list of diagnostic pages implemented by the target. This page shall be implemented if the target implements the page format option of the SEND DIAGNOSTIC and RECEIVE DIAGNOSTIC RESULTS commands.

Table 79 - Supported diagnostic pages

Bit Byte	7	6	5	4	3	2	1	0
0	Page code (00h)							
1	Reserved							
2	(MSB)			Page length (n-3)				
3								(LSB)
4	Supported page list							
n								

The definition of this page for the SEND DIAGNOSTIC command includes only the first four bytes. If the page length field is not zero, the target shall terminate the SEND DIAGNOSTIC command with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST with an additional sense code of INVALID FIELD IN PARAMETER LIST. This page instructs the target to make available the list of all supported diagnostic pages to be returned by a subsequent RECEIVE DIAGNOSTIC RESULTS command.

The definition of this page for the RECEIVE DIAGNOSTIC RESULTS command includes the list of diagnostic pages supported by the target.

The page length field specifies the length in bytes of the following supported page list.

The supported page list field shall contain a list of all diagnostic page codes implemented by the target in ascending order beginning with page code 00h.

8.3.2 Log parameters

This subclause describes the log page structure and the log pages that are applicable to all SCSI devices. Pages specific to each device type are described in the third subclause of each device-type clause (i.e. 9.3.2, 10.3.2, etc.). The LOG SELECT command supports the ability to send zero or more log pages. The LOG SENSE command returns a single log page specified in the page code field of the command descriptor block (see 8.2.7).

Each log page begins with a four-byte page header followed by zero or more variable-length log parameters defined for that page. The log page format is defined in table 80.

Table 80 - Log page format

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved			Page code				
1	Reserved							
2	(MSB) Page length (n-3)							
3								(LSB)
	Log parameters(s)							
4	Log parameter (First) (Length x)							
x+3								
	. .							
n-y+1	Log parameter (Last) (Length y)							
n								

The page code field identifies which log page is being transferred.

The page length field specifies the length in bytes of the following log parameters. If the initiator sends a page length that results in the truncation of any parameter, the target shall terminate the command with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST with the additional sense code set to INVALID FIELD IN PARAMETER LIST.

Most log pages contain one or more special data structures called log parameters (see table 81). Log parameters may be data counters that record a count of a particular event (or events) or log parameters may be list parameters (strings) which contain a description of a particular event.

Table 81 - Log parameter

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1	Parameter code							(LSB)
2	DU	DS	TSD	ETC	TMC		Reserved	LP
3	Parameter length (n-3)							
4	Parameter value							
n								

Each log parameter begins with a four-byte parameter header followed by one or more bytes of parameter value data

The parameter code field identifies the log parameter is being transferred for that log page.

The DU, DS, TSD, ETC, TMC, and LP fields are collectively referred to as the parameter control byte. These fields are described below.

For cumulative log parameter values (indicated by the PC field of the LOG SELECT and LOG SENSE command descriptor block), the disable update (DU) bit is defined as follows:

a) A zero value indicates that the target shall update the log parameter value to reflect all events that should be noted by that parameter.
b) A one value indicates that the target shall not update the log parameter value except in response to a LOG SELECT command that specifies a new value for the parameter.

NOTE 94 When updating cumulative log parameter values, a target may use volatile memory to hold these values until a LOG SELECT or LOG SENSE command is received with an SP bit of one (or a target-defined event occurs). Thus the updated cumulative log parameter values may be lost if a power cycle occurs.

The DU bit is not defined for threshold values (indicated by the PC field of the LOG SENSE command descriptor block) nor for list parameters (indicated by the LP bit). The target shall ignore the value of any DU bits in a LOG SELECT command.

A disable save (DS) bit of zero indicates that the target supports saving for that log parameter. The target shall save the current cumulative or the current threshold parameter value (depending on the value in the PC field of the command descriptor block) in response to a LOG SELECT or LOG SENSE command with an SP bit of one. A DS bit of one indicates that the target does not support saving that log parameter in response to a LOG SELECT or LOG SENSE command with an SP bit of one.

A target save disable (TSD) bit of zero indicates that the target provides a target-defined method for saving log parameters. This implicit saving operation shall be done frequently enough to insure that the cumulative parameter values retain statistical significance (i.e. across power cycles). A TSD bit of one indicates that either the target does not provide a target-defined method for saving log parameters or the target-defined method has been disabled by the initiator.

An enable threshold comparison (ETC) bit of one indicates that a comparison to the threshold value is performed whenever the cumulative value is updated. An ETC bit of zero indicates that a comparison is not performed. The value of the ETC bit is the same for cumulative and threshold parameters.

The threshold met criteria (TMC) field (see table 82) defines the basis for comparison of the cumulative and threshold values. The TMC field is valid only if the ETC bit is one. The value of the TMC field is the same for cumulative and threshold parameters.

Table 82 - Threshold met criteria

Code	Basis for comparison
00b	Every update of the cumulative value
01b	Cumulative value equal threshold value
10b	Cumulative value not equal threshold value
11b	Cumulative value greater than threshold value

If the ETC bit is one and the result of the comparison is true, a unit attention condition shall be generated for all initiators. When reporting the unit attention condition, the target shall set the sense key to UNIT ATTENTION and set the additional sense code to THRESHOLD CONDITION MET.

The list parameter (LP) bit indicates the format of the log parameter. If an initiator attempts to set the value of the LP bit to a value other than the one returned for the same parameter in the LOG SENSE command, the target shall terminate the command with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST with the additional sense code set to INVALID FIELD IN PARAMETER LIST.

An LP bit of zero indicates that the parameter is a data counter. Data counters are associated with one of more events; the data counter is updated whenever one of these events occurs by incrementing of the counter value. If each data counter has associated with it a target-defined maximum value. Upon reaching this maximum value, the data counter shall not be incremented (i.e. it does not wrap). When a data counter reaches its maximum value, the target shall set the associated DU bit to one. If the data counter is at or reaches its maximum value during the execution of a command, the target shall complete the command. If the command completes correctly (except for the data counter being at its maximum value) and if the RLEC bit of the control mode page (8.3.3.1) is set to one; then the target shall terminate the command with CHECK CONDITION status and set the sense key to RECOVERED ERROR with the additional sense code set to LOG COUNTER AT MAXIMUM.

An LP bit of one indicates that the parameter is a list parameter. List parameters are not counters and thus the ETC and TMC fields shall be set to zero. A list parameter is a string of ASCII graphic codes (i.e. code values 20h through 7Eh).

If more than one list parameter is defined in a single log page, the following rules apply to assigning parameter codes:

a) The parameter updated last shall have a higher parameter code than the previous parameter, except as defined in rule b).
b) When the maximum parameter code value supported by the target is reached, the target shall assign the lowest parameter code value to the next log parameter (i.e. wrap-around parameter codes). If the associated command completes correctly (except for the parameter code being at its maximum value) and if the RLEC bit of the control mode page (8.3.3.1) is set to one; then the target shall terminate the command with CHECK CONDITION status and set the sense key to RECOVERED ERROR with the additional sense code set to LOG LIST CODES EXHAUSTED.

NOTE 95 List parameters can be used to store the locations of defective blocks in the following manner. When a defective block is identified, a list parameter is updated to reflect the location and cause of the defect. When the next defect is encountered, the list parameter with the next higher parameter code is updated to record this defect. The size of the page can be made target specific to accommodate memory limitations. It is recommended that one or more data counter parameters be defined for the page to keep track of the number of valid list parameters and the parameter code of the parameter with the oldest recorded defect. This technique can be adapted to record other types of information.

The parameter length field specifies the length in bytes of the following parameter value. If the initiator sends a parameter length value that results in the truncation of the parameter value, the target shall terminate the command with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST with the additional sense code set to INVALID FIELD IN PARAMETER LIST.

If the initiator sends a log parameter value that is outside the range supported by the target, and rounding is implemented for that parameter, the target may either:

a) round to an acceptable value and terminate the command as described in 7.5.4; or
b) terminate the command with CHECK CONDITION status and set the sense key to ILLEGAL REQUEST with the additional sense code set to INVALID FIELD IN PARAMETER LIST.

When any counter in a log page reaches its maximum value, incrementing of all counters in that log page shall cease until reinitialized by the initiator via a LOG SELECT command. If the RLEC bit of the control mode page is one, ther the target shall report the exception condition.

The page code assignments for the log pages are listed in table 83.

Table 83 - Log page codes

Page code	Description	Subclause
01h	Buffer over-run/under-run page	8.3.2.1
03h	Error counter page (read) page	8.3.2.2
04h	Error counter page (read reverse) page	8.3.2.2
05h	Error counter page (verify) page	8.3.2.2
02h	Error counter page (write) page	8.3.2.2
07h	Last n error events page	8.3.2.3
06h	Non-medium error page	8.3.2.4
00h	Supported log pages	8.3.2.5
08h - 2Fh	Reserved	
3Fh	Reserved	
30h - 3Eh	Vendor-specific pages	

8.3.2.1 Buffer over-run/under-run page

The buffer over-run/under-run page (page code 01h) defines 24 data counters that may be used to record the number of buffer over-runs or under-runs for the logical unit. A target that implements this page may implement one or more of the defined data counters.

A buffer over-run or under-run can occur when an initiator does not transmit data to or from the target's buffer fast enough to keep up with reading or writing the media. This can be caused by a slow transfer rate across the SCSI bus or by a high SCSI bus utilization that prevents reconnection by the target. A buffer over-run condition can occur during a read operation when a buffer full condition prevents continued transfer of data from the media to the buffer. A buffer under-run condition can occur during a write operation when a buffer empty condition prevents continued transfer of data to the media from the buffer. Most devices incur a delay at this point while the media is repositioned

Table 84 defines the parameter code field for the buffer over-run/under-run counters.

Table 84 - Parameter code field for buffer over-run/under-run counters

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved							
1	Count basis			Cause				Type

The parameter code field for buffer over-run/under-run counters is a 16-bit value comprised of eight reserved bits, a three-bit count basis field (see table 85), a four-bit cause field (see table 86), and a one-bit type field. These are concatenated to determine the value of the parameter code for that log parameter. For example, a counter for parameter code value of 0023h specifies a count basis of 001b; a cause of 0001b; and a type of 1b; this counter is incremented once per command that experiences an over-run due to the SCSI bus being busy.

Table 85 - Count basis definition

Count basis	Description
000b	Undefined
001b	Per command
010b	Per failed reconnect
011b	Per unit of time
100b - 111b	Reserved

The count basis field defines the criteria for incrementing the counter. The following criteria are defined:

NOTE 96 The per unit of time count basis is device type specific. Direct-access devices typically use a latency period (i.e. one revolution of the medium) as the unit of time.

The cause field indicates the reason that the over-run or under-run occurred. The following causes are defined in table 86.

Table 86 - Cause field definition

Cause	Description
0h	Undefined
1h	SCSI bus busy
2h	Transfer rate too slow
3h - Fh	Reserved

The type field indicates whether the counter records under-runs or over-runs. A value of zero specifies a buffer under-run condition and a value of one specifies a buffer over-run condition.

The counters contain the total number of times buffer over-run or under-run conditions have occurred since the last time the counter was cleared. The counter shall be incremented for each occurrence of an under-run or over-run condition and can be incremented more than once for multiple occurrences during the execution of a single command.

8.3.2.2 Error counter pages

This clause defines the optional error counter pages for write errors (page code 02h), read errors (page code 03h), read reverse errors (page code 04h) and verify errors (page code 05h). The log page format is defined near the beginning of 8.3.2. A page can return one or more log parameters this record events defined by the parameter codes.

Table 87 defines the parameter codes for the error counter pages. Support of each log parameter is optional.

Table 87 - Parameter codes for error counter pages

Parameter code	Description
0000h	Errors corrected without substantial delay
0001h	Errors corrected with possible delays
0002h	Total (e.g. rewrites or rereads)
0003h	Total errors corrected
0004h	Total times correction algorithm processed
0005h	Total bytes processed
0006h	Total uncorrected errors
0007h - 7FFFh	Reserved
8000h - FFFFh	Vendor-specific

NOTE 97 The exact definition of the error counters is not part of this standard. These counters should not be used to compare products because the products may define errors differently.

8.3.2.3 Last *n* error events page

Log page (07h) provides for a number of error-event records using the list parameter format of the log page. The number of these error-event records supported, *n*, is device-specific. Each error-event record contains device-specific diagnostic information for a single error encountered by the device. The parameter code associated with error-event record indicates the relative time at which the error occurred. A higher parameter code indicates that the error event occurred later in time.

The content of the parameter value field of each log parameter is an ASCII character string which may describe the error event. The exact contents of the character string is not defined by this standard.

When the last supported parameter code is used by an error-event record, the recording on this page of all subsequent error information shall cease until one or more of the list parameters with the highest parameter codes have been reinitialized. If the RLEC bit of the control mode page (8.3.3.1) is set to one, the target shall return CHECK CONDITION status with the sense key set to RECOVERED ERROR and the additional sense code set to LOG LIST CODES EXHAUSTED. Alternatively, the target may report this condition via asynchronous event notification (see 7.5.5).

8.3.2.4 Non-medium error page

This page (page code 06h) provides for summing the occurrences of recoverable error events other than write, read, or verify failures. No discrimination among the various types of events is provided by parameter code (see table 88). Vendor-specific discrimination may be provided through the vendor-specific parameter codes.

Table 88 - Non-medium error event parameter codes

Parameter code	Description
0000h	Non-medium error count
0001h - 7FFFh	Reserved
8000h - FFFFh	Vendor-specific error counts

8.3.2.5 Supported log pages

The supported log page (see table 89) returns the list of log pages implemented by the target. Targets that implement the LOG SENSE command shall implement this log page.

Table 89 - Supported log pages

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved		Page code (00h)					
1	Reserved							
2	(MSB)							
3	Page length (n-3)							(LSB)
4	Supported page list							
n								

This page is not defined for the LOG SELECT command. This log page returns the list of supported log pages for the specified logical unit.

The page length field specifies the length in bytes of the following supported page list.

The supported page list field shall contain a list of all log page codes implemented by the target in ascending order beginning with page code 00h.

8.3.3 Mode parameters

This subclause describes the block descriptors and the pages used with MODE SELECT and MODE SENSE commands that are applicable to all SCSI devices. Pages specific to each device type are described in the third subclause of each device-type clause (i.e. 9.3, 10.3, etc.).

The mode parameter list shown in table 90 contains a header, followed by zero or more block descriptors, followed by zero or more variable-length pages. Parameter lists are defined for each device type.

Table 90 - Mode parameter list

Bit Byte	7	6	5	4	3	2	1	0
0 - n	Mode parameter header							
0 - n	Block descriptor(s)							
0 - n	Page(s)							

The six-byte command descriptor block parameter header is defined in table 91.

Table 91 - Mode parameter header(6)

Bit Byte	7	6	5	4	3	2	1	0
0	Mode data length							
1	Medium type							
2	Device-specific parameter							
3	Block descriptor length							

The ten-byte command descriptor block parameter header is defined in table 92.

Table 92 - Mode parameter header(10)

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1				Mode data length				(LSB)
2				Medium type				
3				Device-specific parameter				
4				Reserved				
5				Reserved				
6	(MSB)							
7				Block descriptor length				(LSB)

When using the MODE SENSE command, the mode data length field specifies the length in bytes of the following data that is available to be transferred. The mode data length does not include itself. When using the MODE SELECT command, this field is reserved.

NOTE 98 Targets that support more than 256 bytes of block descriptors and pages may need to implement ten-byte mode commands. The mode data length field in the six-byte command descriptor block header limits the returned data to 256 bytes.

Medium types are unique for each device type. Refer to the mode parameters clause of the specific device type for definition of these values. Some device types reserve this field.

The device specific parameter is unique for each device type. Refer to the mode parameters clause of the specific device type for definition of this field. Some device types reserve all or part of this field.

The block descriptor length specifies the length in bytes of all the block descriptors. It is equal to the number of block descriptors times eight, and does not include pages or vendor-specific parameters, if any, that may follow the last block descriptor. A block descriptor length of zero indicates that no block descriptors are included in the mode parameter list. This condition shall not be considered an error.

The mode parameter block descriptor is shown in table 93.

Table 93 - Mode parameter block descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0				Density code				
1	(MSB)							
2				Number of blocks				
3								(LSB)
4				Reserved				
5	(MSB)							
6				Block length				
7								(LSB)

Block descriptors specify some of the medium characteristics for all or part of a logical unit. Support for block descriptors is optional. Each block descriptor contains a density code field, a number of blocks field, and a block length field. Block descriptor values are always current (i.e. saving is not supported). A unit attention condition (see 7.9) shall be generated when any block descriptor values are changed.

The density code field is unique for each device type. Refer to the mode parameters clause of the specific device type for definition of this field. Some device types reserve all or part of this field.

The number of blocks field specifies the number of logical blocks on the medium to which the density code and block length fields apply. A value of zero indicates that all of the remaining logical blocks of the logical unit shall have the medium characteristics specified.

NOTE 99 There may be implicit association between parameters defined in the pages and block descriptors. For direct-access devices, the block length affects the optimum values (the values that achieve best performance) for the sectors per track, bytes per physical sector, track skew factor, and cylinder skew factor fields in the format parameters page. In this case, the target may change parameters not explicitly sent with the MODE SELECT command. A subsequent MODE SENSE command would reflect these changes.

The number of remaining logical blocks may be unknown for some device types.

The block length specifies the length in bytes of each logical block described by the block descriptor. For sequential-access devices, a block length of zero indicates that the logical block size written to the medium is specified by the transfer length field in the command descriptor block (see 10.2.4 and 10.2.14)

The mode page format is defined in table 94.

Table 94 - Mode page format

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	Reserved	Page code					
1	Page length (n-1))							
2	Mode parameters							
n								

Each mode page contains a page code, a page length, and a set of mode parameters. The page codes are defined in this subclause and in the mode parameter subclauses of the specific device type.

When using the MODE SENSE command, a parameters savable (PS) bit of one indicates that the mode page can be saved by the target in a non-volatile, vendor-specific location. A PS bit of zero indicates that the supported parameters cannot be saved. When using the MODE SELECT command, the PS bit is reserved.

The page code field identifies the format and parameters defined for that mode page. Some page codes are defined as applying to all device types and other page codes are defined for the specific device type.

When using the MODE SENSE command, if page code 00h (vendor-specific page) is implemented, the target shall return that page last in response to a request to return all pages (page code 3Fh). When using the MODE SELECT command, this page should be sent last.

The page length field specifies the length in bytes of the mode parameters that follow. If the initiator does not set this value to the value that is returned for the page by the MODE SENSE command, the target shall terminate the command with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST with the additional sense code set to INVALID FIELD IN PARAMETER LIST. The target is permitted to implement a mode page that is

less than the full page length defined in this standard, provided no field is truncated and the page length fiel correctly specifies the actual length implemented.

The mode parameters for each page are defined in the following subclauses, or in the mode parameters subclaus for the specific device type. Mode parameters not implemented by the target shall be set to zero.

Table 95 defines the mode pages that are applicable to all device types that include the MODE SELECT and MOD SENSE commands.

Table 95 - Mode page codes

Page code	Description	Subclause
0Ah	Control mode page	8.3.3.1
02h	Disconnect-reconnect page	8.3.3.2
09h	Peripheral device page	8.3.3.3
01h	(See specific device type)	
03h - 08h	(See specific device type)	
0Bh - 1Fh	(See specific device type)	
00h	Vendor-specific (does not require page format)	
20h - 3Eh	Vendor-specific (page format required)	
3Fh	Return all pages (valid only for the MODE SENSE command)	

8.3.3.1 Control mode page

The control mode page (see table 96) provides controls over several SCSI-2 features that are applicable to all devic types such as tagged queuing, extended contingent allegiance, asynchronous event notification, and error loggin

Table 96 - Control mode page

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	Reserved	Page code (0Ah)					
1	Page length (06h)							
2	Reserved							RLEC
3	Queue algorithm modifier				Reserved		QErr	DQue
4	EECA	Reserved				RAENP	UAAENP	EAENP
5	Reserved							
6	(MSB) Ready AEN holdoff period							
7								(LSB)

A report log exception condition (RLEC) bit of one specifies that the target shall report log exception conditions a described in 8.3.2. A RLEC bit of zero specifies that the target shall not report log exception conditions.

The queue algorithm modifier field (see table 97) specifies restrictions on the algorithm used for reordering command that are tagged with the SIMPLE QUEUE TAG message.

Table 97 - Queue algorithm modifier

Value	Definition
0h	Restricted reordering
1h	Unrestricted reordering allowed
2h - 7h	Reserved
8h - Fh	Vendor-specific

A value of zero in this field specifies that the target shall order the actual execution sequence of the commands with a SIMPLE QUEUE tag such that data integrity is maintained for that initiator. This means that, if the transmission of new commands is halted at any time, the final value of all data observable on the medium shall have exactly the same value as it would have if the commands had been executed in the same received sequence without tagged queuing. The restricted reordering value shall be the default value.

A value of one in this field specifies that the target may reorder the actual execution sequence of the commands with a SIMPLE QUEUE tag in any manner. Any data integrity exposures related to command sequence order are explicitly handled by the initiator through the selection of appropriate commands and queue tag messages.

A queue error management (QErr) bit of zero specifies that remaining suspended I/O process shall resume after the contingent allegiance condition or extended contingent allegiance condition (see 7.8).

A QErr bit of one specifies all remaining suspended I/O processes shall be aborted after the contingent allegiance condition or extended contingent allegiance condition (see 7.8). A unit attention condition (see 7.9) shall be generated for each initiator that had a suspended I/O process aborted except for the initiator that had the contingent allegiance condition or extended contingent allegiance condition. The target shall set the additional sense code to TAGGED COMMANDS CLEARED BY ANOTHER INITIATOR.

A disable queuing (DQue) bit of zero specifies that tagged queuing shall be enabled if the target supports tagged queuing. A DQue bit of one specifies that tagged queuing shall be disabled. Any queued commands for that I_T_x nexus shall be aborted. Any subsequent queue tag message received shall be rejected with a MESSAGE REJECT message and the I/O process shall be executed as an untagged command (see 7.8.1).

An enable extended contingent allegiance (EECA) bit of one specifies that extended contingent allegiance is enabled (see 7.7). An EECA bit of zero specifies that extended contingent allegiance is disabled.

The RAENP, UAAENP, and EAENP bits enable specific events to be reported via the asynchronous event notification protocol. When all three bits are zero, the target shall not create asynchronous event notifications.

A ready AEN permission (RAENP) bit of one specifies that the target may issue an asynchronous event notification upon completing its initialization sequence instead of generating a unit attention condition. A RAENP bit of zero specifies that the target shall not issue an asynchronous event notification upon completing its initialization sequence.

NOTE 100 If the target's default value for the RAENP bit is one and it does not implement saved parameters or include a hardware switch, then it may not be possible to disable the initialization sequence asynchronous event notification.

A unit attention AEN permission (UAAENP) bit of one specifies that the target may issue an asynchronous event notification instead of creating a unit attention condition upon detecting an event that would cause a unit attention condition (other than upon completing an initialization sequence). A UAAENP bit of zero specifies that the target shall not issue an asynchronous event notification instead of creating a unit attention condition.

An error AEN permission (EAENP) bit of one specifies that the target may issue an asynchronous event notification upon detecting a deferred error condition instead of waiting to report the deferred error on the next command. An EAENP bit of zero specifies that the target shall not report deferred error conditions via an asynchronous event notification.

The ready AEN holdoff period field specifies the minimum time in milliseconds after the target starts its initialization sequence that it shall delay before attempting to issue an asynchronous event notification. This value may be rounded up as defined in 7.5.4.

8.3.3.2 Disconnect-reconnect page

The disconnect-reconnect page (see table 98) provides the initiator the means to tune the performance of the SCSI bus.

Table 98 - Disconnect-reconnect page

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	Reserved	Page code (02h)					
1	Page length (0Eh)							
2	Buffer full ratio							
3	Buffer empty ratio							
4	(MSB)							
5	Bus inactivity limit							(LSB)
6	(MSB)							
7	Disconnect time limit							(LSB)
8	(MSB)							
9	Connect time limit							(LSB)
10	(MSB)							
11	Maximum burst size							(LSB)
12	Reserved							DTDC
13	Reserved							
14	Reserved							
15	Reserved							

The buffer full ratio field indicates to the target, on read operations, how full the buffer should be prior to attempting a reselection. Targets that do not implement the requested ratio should round down to the nearest implemented ratio as defined in 7.5.4.

The buffer empty ratio field indicates to the target, on write operations, how empty the buffer should be prior to attempting a reselection. Targets that do not implement the requested ratio should round down to the nearest implemented ratio as defined in 7.5.4.

The buffer full and buffer empty ratios are numerators of a fractional multiplier that has 256 as its denominator. A value of zero indicates that the target determines when to initiate reselection consistent with the disconnect time limit parameter. These parameters are advisory to the target.

NOTE 101 As an example, consider a target with ten 512-byte buffers and a specified buffer full ratio of 3Fh. The formula is: INTEGER((ratio/256)*number of buffers). Thus INTEGER((3Fh/256)*10) = 2. The target should attempt to reselect the initiator on read operations whenever two or more buffers are full.

The bus inactivity limit field indicates the maximum time in 100 μs increments that the target is permitted to assert the BSY signal without a REQ/ACK handshake. If the bus inactivity limit is exceeded the target shall attempt to disconnect if the initiator has granted the disconnect privilege (see 6.6.7) and it is not restricted by DTDC. This value may be rounded as defined in 7.5.4. A value of zero indicates that there is no bus inactivity limit.

The disconnect time limit field indicates the minimum time in 100 μs increments that the target shall wait after releasing the SCSI bus before attempting reselection. This value may be rounded as defined in 7.5.4. A value of zero indicates that there is no disconnect time limit.

The connect time limit field indicates the maximum time in 100 μs increments that the target is allowed to use the SCSI bus before disconnecting, if the initiator has granted the disconnect privilege (see 6.6.7) and it is not restricted by DTDC. This value may be rounded as defined in 7.5.4. A value of zero indicates that there is no connect time limit.

The maximum burst size field indicates the maximum amount of data that the target shall transfer during a data phase before disconnecting if the initiator has granted the disconnect privilege. This value is expressed in increments of 512 bytes (e.g. a value of one means 512 bytes, two means 1024 bytes, etc.). A value of zero indicates there is no limit on the amount of data transferred per connection.

The data transfer disconnect control (DTDC) field (see table 99) defines further restrictions on when a disconnect is permitted.

Table 99 - Data transfer disconnect control

DTDC	Description
00b	Data transfer disconnect control is not used. Disconnect is controlled by the other fields in this page.
01b	A target shall not attempt to disconnect once the data transfer of a command has started until all data the command is to transfer has been transferred. The connect time limit and bus inactivity limit are ignored during the data transfer.
10b	Reserved
11b	A target shall not attempt to disconnect once the data transfer of a command has started, until the command is complete. The connect time limit and bus inactivity limit are ignored once data transfer has started.

If DTDC is non-zero and the maximum burst size is non-zero, the target shall return CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code set to ILLEGAL FIELD IN PARAMETER LIST.

8.3.3.3 Peripheral device page

The peripheral device page (see table 100) is used to pass vendor-specific information between an initiator and a peripheral interface below the target (i.e. between the target and the peripheral device). This standard does not define the format of this data, except to provide a standard header.

Table 100 - Peripheral device page

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	Reserved	Page code (09h)					
1	Page length (n-1)							
2	(MSB)							
3	Interface identifier							(LSB)
4	Reserved							
5	Reserved							
6	Reserved							
7	Reserved							
8	Vendor-specific							
n								

Interface identifier codes are defined in the table 101.

Table 101 - Interface identifier codes

Code value	Interface	ANSI/ISO Reference standard
0000h	Small computer system interface	X3.131-1994
0001h	Storage module interface	X3.91M-1987
0002h	Enhanced small device interface	X3.170A-1991
0003h	Intelligent peripheral interface-2	9318-2-1990
0004h	Intelligent peripheral interface-3	9318-3-1990 9318-4-1992
0005h - 7FFFh	Reserved	
8000h - FFFFh	Vendor-specific	

8.3.4 Vital product data parameters

This clause describes the optional vital product data page structure and the vital product data pages (see table 102) that are applicable to all SCSI devices. These pages are optionally returned by the INQUIRY command (8.2.5) and contain vendor-specific product information about a target or logical unit. The vital product data may include vendor identification, product identification, unit serial numbers, device operating definitions, manufacturing data, field replaceable unit information, and other vendor-specific information. This standard defines the structure of the vital product data, but not the contents.

Table 102 - Vital product data page codes

Page code	Description	Subclause
82h	ASCII implemented operating definition page	8.3.4.1
01h - 7Fh	ASCII information page	8.3.4.2
81h	Implemented operating definition page	8.3.4.3
00h	Supported vital product data pages	8.3.4.4
80h	Unit serial number page	8.3.4.5
83h - BFh	Reserved	
C0h - FFh	Vendor-specific	

8.3.4.1 ASCII implemented operating definition page

The ASCII implemented operation definition page (see table 103) contains operating definition description data for all operating definitions implemented by the target. The contents of this data is not defined by this standard.

Table 103 - ASCII implemented operating definition

Bit / Byte	7	6	5	4	3	2	1	0
0	Peripheral qualifier			Peripheral device type				
1	Page code (82h)							
2	Reserved							
3	Page length (n-3)							
4	ASCII operating definition description length (m-4)							
5 — m	ASCII operating definition description data							
m+1 — n	Vendor-specific description data							

The peripheral qualifier field and the peripheral device type field are as defined in 8.2.5.1.

The page length field specifies the length of the following page data. If the allocation length is less than the length of the data to be returned, the page length shall not be adjusted to reflect the truncation.

The ASCII operating definition description length field specifies the length in bytes of the ASCII operating definition description data that follows. If the allocation length is less than the length of data to be returned, the ASCII operating definition description length shall not be adjusted to reflect the truncation. A value of zero in this field indicates that no ASCII operating definition description data is available.

The ASCII operating definition description data field contains the ASCII operating definition description data for the target or logical unit. The data in this field shall be formatted in lines (or character strings). Each line shall contain only graphic codes (i.e. code values 20h through 7Eh) and shall be terminated with a NULL (00h) character.

8.3.4.2 ASCII information page

The ASCII information page (see table 104) returns information for the field replaceable unit code returned in the REQUEST SENSE data (see 8.2.14).

Table 104 - ASCII information page

Bit / Byte	7	6	5	4	3	2	1	0
0	Peripheral qualifier			Peripheral device type				
1	Page code (01h - 7Fh)							
2	Reserved							
3	Page length (n-3)							
4	ASCII length (m-4)							
5 - m	ASCII information							
m+1 - n	Vendor-specific information							

The peripheral qualifier field and the peripheral device type field are defined in 8.2.5.1.

The page code field contains the same value as in the page code field of the INQUIRY command descriptor block (see 8.2.5) and is associated with the field replaceable unit code returned by the REQUEST SENSE command.

NOTE 102 The field replaceable unit field in the sense data provides for 255 possible codes, while the page code field provides for only 127 possible codes. Thus it is not possible to return ASCII information pages for the upper code values.

The page length field specifies the length of the following page data. If the allocation length of the command descriptor block is too small to transfer all of the page, the page length shall not be adjusted to reflect the truncation.

The ASCII length field specifies the length in bytes of the ASCII information that follows. If the allocation length is less than the length of the data to be returned, the ASCII length shall not be adjusted to reflect the truncation. A value of zero in this field indicates that no ASCII information is available for the specified page code.

The ASCII information field contains ASCII information concerning the field replaceable unit identified by the page code. The data in this field shall be formatted in one or more lines (or character strings). Each line shall contain only graphic codes (i.e. code values 20h through 7Eh) and shall be terminated with a NULL (00h) character.

The contents of the vendor-specific information field is not defined in this standard.

8.3.4.3 Implemented operating definition page

The implemented operating definition page (see table 105) defines the current operating definition, the default operating definition, and the operating definitions implemented by the target. These operating definition values are specified in the CHANGE DEFINITION command (see 8.2.1).

Table 105 - Implemented operating definition page

Bit / Byte	7	6	5	4	3	2	1	0
0	Peripheral qualifier			Peripheral device type				
1	Page code (81h)							
2	Reserved							
3	Page length (n-3)							
4	Reserved	Current operating definition						
5	SavImp	Default operating definition						
6	SavImp	Supported operating definition list						
n	SavImp							

The peripheral qualifier field and the peripheral device type field are defined in 8.2.5.1.

The page length field specifies the length of the following operating definitions. If the allocation length of the command descriptor block is too small to transfer all of the page, the page length shall not be adjusted to reflect the truncation.

For each operating definition, there is an associated save implemented (SavImp) bit. A SavImp bit of zero indicates that the corresponding operating definition parameter cannot be saved. A SavImp bit of one indicates that the corresponding operating definition parameter can be saved.

All returned operating definitions use the codes defined in table 33. The current operating definition field returns the value of the present operating definition. If no operating definition is saved, the default operating definition field returns the value of the operating definition the target uses when power is applied. The supported operating definition list returns one or more operating definitions implemented by the target.

8.3.4.4 Supported vital product data pages

The supported vital product data pages are shown in table 106.

Table 106 - Supported vital product data pages

Bit / Byte	7	6	5	4	3	2	1	0
0	Peripheral qualifier			Peripheral device type				
1	Page code (00h)							
2	Reserved							
3	Page length (n-3)							
4	Supported page list							
n								

The peripheral qualifier field and the peripheral device type field are defined in 8.2.5.1.

The page code field shall be set to the value of the page code field in the INQUIRY command descriptor block (see 8.2.5).

The page length field specifies the length of the supported page list. If the allocation length is too small to transfer all of the page, the page length shall not be adjusted to reflect the truncation.

The supported page list field shall contain a list of all vital product data page codes implemented for the target or logical unit in ascending order beginning with page code 00h.

8.3.4.5 Unit serial number page

This page (see table 107) provides a product serial number for the target or logical unit.

Table 107 - Unit serial number page

Bit / Byte	7	6	5	4	3	2	1	0
0	Peripheral qualifier			Peripheral device type				
1	Page code (80h)							
2	Reserved							
3	Page length (n-3)							
4	Product serial number							
n								

The peripheral qualifier field and the peripheral device type field are defined in 8.2.5.1.

The page length field specifies the length of the product serial number. If the allocation length is too small to transfer all of the page, the page length shall not be adjusted to reflect the truncation.

The product serial number field contains ASCII data that is vendor-specific. The least significant ASCII character of the serial number shall appear as the last byte of a successful data transfer. If the product serial number is not available, the target shall return ASCII spaces (20h) in this field.

9 Direct-access devices

9.1 Direct-access device model

Direct-access devices store blocks of data for later retrieval. Each block of data is stored at a unique logical bloc address. An initiator issues WRITE commands to store the blocks of data (write operations) and READ command to retrieve the blocks of data (read operations). Other commands issued by the initiator may also cause write an read operations to occur. A write operation causes a block of data to be written on the medium. A read operatio causes a block of data to be read from the medium. A verify operation confirms that a block of data can be rea without error from the medium.

Blocks of data are stored by a process that causes localized changes or transitions within the medium. The change made to the medium to store the blocks of data may be volatile (i.e. not retained through a power cycle) or non volatile (retained through a power cycle). The medium may be divided in parts that are used for data blocks, part that are reserved for defect management, and parts that are reserved for use by the controller for the managemen of the device.

9.1.1 Removable medium

The medium may be removable (e.g. used in a floppy disk drive) or non-removable (e.g. used in a hard disk drive) Removable medium is contained within a cartridge (or jacket) to prevent damage to the recording surfaces. The combination of medium and cartridge is often called a volume.

A volume has an attribute of being mounted or de-mounted on a suitable transport mechanism. A volume is mounte when the direct-access device is capable of performing write or read operations to the medium. A mounted volum may not be accessible by an initiator if it is reserved by another initiator. A volume is de-mounted at any other time (e.g. during loading, unloading, or storage).

An initiator may check whether a volume is mounted by issuing a TEST UNIT READY command. A volume that is loaded may need a START STOP UNIT command issued to become accessible for write or read operations.

The PREVENT ALLOW MEDIUM REMOVAL command allows an initiator to restrict the demounting of the volume This is useful in maintaining system integrity. If the direct-access device implements cache memory, it must ensure that all logical blocks of the medium contain the most recent data prior to permitting demounting of the volume. I the initiator issues a START STOP UNIT command to eject the cartridge, and the direct-access device is prevente from demounting by the PREVENT ALLOW MEDIUM REMOVAL command, the START STOP unit command is rejected by the direct-access device.

9.1.2 Logical blocks

Blocks of data are stored on the medium along with additional information that the controller uses to manage the storage and retrieval. The format of the additional information is unique and is hidden from the initiator during norma read or write operations. This additional information is often used to identify the physical location of the blocks o data and the address of the logical block, and to provide protection against the loss of the user data.

The address of the first logical block is zero. The address of the last logical block is [*n*-1], where [*n*] is the numbe of logical blocks available on the medium. A READ CAPACITY command may be issued to determine the value o [*n*-1]. If a command is issued that requests access to a logical block not within the capacity of the medium, the command is terminated with CHECK CONDITION.

The number of bytes of data contained in a logical block is known as the block length. Each logical block has a block length associated with it. The block length may be different for each logical block on the medium. However in a typical device only one block length is used at a time. The block descriptor in the MODE SENSE data describes the block lengths that are used on the medium. A MODE SELECT command can be used to set up extents. Ar

extent is a specified number of logical blocks that have the specified block length. The FORMAT UNIT command is typically required to change the block length of devices that support variable block lengths and make the extents that were setup active.

The location of a logical block on the medium does not have a relationship to the location of any other logical block. However, in a typical device the logical blocks are located in an ascending order. The time to access the logical block at address [x] and then the logical block at address [x+1] need not be less than time to access [x] and then [x+100]. The READ CAPACITY with a PMI bit of one is useful in determining where longer access times occur.

9.1.3 Ready state

A direct-access device is ready when medium access commands can be executed. A device using removable media is usually not ready until a volume is mounted. Such a device, with a volume not mounted, normally returns CHECK CONDITION status and sets the sense key to NOT READY.

Some direct-access devices may be switched from being ready to being not ready by using the START STOP UNIT command. An initiator may need to issue a START UNIT command to bring a device ready.

9.1.4 Initialization

Many direct-access devices require initialization prior to write or read operations. This initialization is usually performed by a FORMAT UNIT command. Parameters related to the geometry and performance characteristics can be set with the MODE SELECT command prior to the format operation. Some devices are initialized by means not specified in this standard. The time at which this occurs is specific to the implementation of the direct-access device.

Devices using a non-volatile medium typically save the parameters and only need to be initialized once. However, some mode parameters may need to be initialized after each power-on and reset. A catastrophic failure of the direct-access device may require the FORMAT UNIT command to be reissued.

Devices that use a volatile medium may need to be initialized at each power-on prior to the execution of read or write operations. Mode parameters may also need initialization.

9.1.5 Medium defects

Any medium has the potential for defects that can cause user data to be lost. Therefore, each logical block may contain information that allows the detection of changes to the user data caused by defects in the medium or other phenomena, and may also allow the data to be reconstructed following the detection of such a change. Some devices provide the initiator control through use of the mode parameters. Some devices allow the initiator to examine and modify the additional information by using the READ LONG and WRITE LONG commands. Some media having a very low probability of defects may not require these structures.

Defects may also be detected and managed during execution of the FORMAT UNIT command. The FORMAT UNIT command defines four sources of defect information. These defects may be reassigned or avoided during the initialization process so that they do not appear in a logical block.

Defects may also be avoided after initialization. The initiator issues a REASSIGN BLOCKS command to request that the specified logical block address be reassigned to a different part of the medium. This operation can be repeated if a new defect appears at a later time. The total number of defects that may be handled in this manner can be specified in the mode parameters.

Defect management on direct-access devices is usually vendor-specific. Devices not using a removable medium typically optimize the defect management for capacity or performance or both. Devices that use a removable medium typically do not support defect management (e.g. some floppy disk drives) or use defect management that is based on the ability to interchange the medium.

9.1.6 Data cache

Some direct-access devices implement cache memory. A cache memory is usually an area of temporary storage in the direct-access device with a fast access time that is used to enhance performance. It exists separately from the blocks of data stored and is normally not directly accessible by the initiator. Use of cache memory for write or read operations typically reduces the access time to a logical block and can increase the overall data throughput.

During read operations, the direct-access device uses the cache memory to store blocks of data that the initiator may request at some future time. The algorithm used to manage the cache memory is not part of this standard. However, parameters are provided to advise the direct-access device about future requests, or to restrict the use of cache memory for a particular request.

During write operations, the direct-access device uses the cache memory to store data that is written to the medium at a later time. This is called a write-back caching algorithm. Thus the command may complete prior to blocks of data being written to the medium. As a result of using a write-back caching algorithm there is a period of time when the data may be lost if a power or a hardware failure occurs. There is also the possibility of an error occurring during the write operation. If an error occurred during the write, it may be reported as a deferred error on a later command. However, the initiator can request write-through caching to prevent these circumstances from arising.

When the cache memory fills up with blocks of data that are being kept for possible future access, new blocks of data that are to be kept must replace those currently in cache memory. The disable page out (DPO) bit is used to control replacement of logical blocks in the cache. For write operations, setting this bit to one advises the direct-access device to not replace existing blocks in the cache memory with the write data. For read operations, setting this bit to one causes blocks of data that are being read to not replace existing ones in the cache memory.

Sometimes the initiator may wish to have the blocks of data read from the medium instead of from the cache memory. The force unit access (FUA) bit is used to indicate that the direct-access device shall access the physical medium. For a write operation, setting FUA to one causes the direct-access device to complete the data write to the physical medium before completing the command. For a read operation, setting FUA to one causes the logical blocks to be retrieved from the physical medium.

When the DPO and FUA bits are both set to one, write and read operations bypass the cache memory.

When a VERIFY command is executed, a forced unit access is implied, since the blocks of data stored on the medium are being verified. Furthermore, a SYNCHRONIZE CACHE operation (see below) is also implied to write unwritten blocks of data still in the cache memory. These blocks of data must be stored on the medium before the verify operation can begin. The DPO bit is provided since the VERIFY command may cause the replacement of blocks in the cache. The above also applies to the WRITE AND VERIFY command.

Commands may be implemented by the direct-access device that allow the initiator to control other behaviour of the cache memory:

— LOCK UNLOCK CACHE controls whether certain logical blocks will be held in the data cache for future use. Locking a logical block prevents its replacement by a future access. Unlocking a logical block exposes it to possible replacement by a future access (see 9.2.2).

— PRE-FETCH causes a set of logical blocks requested by the initiator to be read into the data cache for possible future access. The blocks fetched are subject to later replacement unless they are locked (see 9.2.3).

— SYNCHRONIZE CACHE forces any pending write data in the requested set of logical blocks to be stored in the physical medium. This command can be used to ensure that the data was written and any errors reported (see 9.2.18).

— The MODE SELECT command defines a page for the control of cache behavior and handles certain basic elements of cache replacement algorithms (see 9.3.3.1).

9.1.7 Reservations

The access enabled or access disabled condition determines when an initiator may store or retrieve user data on all or part of the medium. Access may be restricted for read operations, write operations, or both. This attribute may be controlled by an external mechanism or by the RESERVE and RELEASE commands (see 9.2.12 and 9.2.11).

The RESERVE and RELEASE commands define how different types of restricted access may be achieved, and to whom the access is restricted. This subclause describes the interaction of the initiator that requested the reservation, and the other initiators.

An initiator uses reservations to gain a level of exclusivity in access to all or part of the medium for itself or another initiator. It is expected that the reservation will be retained until released. The direct-access device must ensure that the initiator with the reservation is able to access the reserved media within the operating parameters established by that initiator.

The following paragraphs explain, on a command by command basis, the appropriate target response when a reservation exists. Unless otherwise noted, the appropriate response to an initiator that issues a command to a direct-access device that is reserved to another initiator is RESERVATION CONFLICT status.

The CHANGE DEFINITION command is dealt with as follows. If any initiator has an extent reservation on a direct-access device, no other initiator may affect the operating definition of that initiator by use of this command. If the direct-access device allows different operating definitions for each initiator, then there is no conflict; otherwise, a reservation conflict occurs.

The COMPARE, COPY, and COPY AND VERIFY commands are evaluated for reservation conflict as if they were normal write and read operations even when a direct-access device is requested to copy to or from itself. For example, if a COPY is issued to logical unit 0 that requests the direct-access device to copy from logical unit 0 to logical unit 1, access to logical unit 1 must also be evaluated for conflict.

The FORMAT UNIT, PREVENT ALLOW MEDIUM REMOVAL (with a prevent bit of one), REZERO UNIT, and START STOP UNIT commands return a RESERVATION CONFLICT status if any other initiator has an extent reservation on a direct-access device.

The INQUIRY and REQUEST SENSE commands are not affected by any kind of reservation.

The LOG SELECT, LOG SENSE, MODE SENSE, TEST UNIT READY, READ CAPACITY (PMI set to zero), READ BUFFER, WRITE BUFFER, and READ DEFECT DATA commands are not affected by extent reservations.

The SEEK, LOCK UNLOCK CACHE, PRE-FETCH, and SYNCHRONIZE CACHE commands are evaluated for reservation conflict as if they were normal write or read operations.

The MODE SELECT command is dealt with as follows. If an initiator has an extent reservation on a direct-access device, and another initiator attempts one of these commands, a reservation conflict occurs if the command affects the manner in which access of the extent by the first initiator is performed. If the command does not affect access to the extent, or parameters are saved for each initiator, then a conflict does not occur.

The SEND DIAGNOSTIC, RECEIVE DIAGNOSTIC RESULTS commands conflict with an extent reservation only if they affect access to the extent (as with MODE SELECT).

The REASSIGN BLOCKS command may not reassign a block that is in an extent reserved to another initiator.

The SET LIMITS command generates a reservation conflict if the logical blocks specified are within an extent reserved to another initiator.

ALL other commands are that request read or write operations are evaluated for reservation conflict as described in the RESERVE command.

When a system is integrated with more than one initiator, there must be agreement between the initiators as to how media is reserved and released during operations, otherwise, an initiator may be locked out of access to a target in the middle of an operation. For example, initiator 'A' has a write operation in progress to a direct-access device which has data stored in cache memory. Then, initiator 'B' issues a RESERVE command to the direct-access device. As a result, initiator 'A' is locked out of issuing a SYNCHRONIZE CACHE command to ensure the integrity of the data. To prevent this from happening, initiator 'A' should issue a RESERVE prior to the write command.

9.1.8 Seek and rezero

The SEEK command provides a way for the initiator to position the device in preparation for access to a particular logical block at some later time. Since this positioning action is implicit in other commands, the SEEK command may not be useful with some direct-access devices.

The REZERO UNIT command is provided to bring the direct-access device to a known condition. This standard does not specify the condition. The REZERO UNIT command is used in some devices to position the actuator at cylinder zero. Some devices return GOOD status without attempting any action.

9.1.9 Notched drives

A notched (also called partitioned or zoned) drive has areas of the medium in which the drive geometry changes. In the simplest case, the entire medium consists of a single notch. Multiple notches are often used to increase capacity of the drive. The notch page is used to indicate the notch for assignment of values to the parameters in the format device page. By sequencing the notch page through each notch, the format device parameters of each notch are set. This is usually done prior to initialization by the FORMAT UNIT command.

9.1.10 Rotational position locking

Rotational position locking is an optional feature implemented in some direct-access devices to allow the synchronization of spindles between a number of devices. The rotational position offset feature allows devices to synchronize spindles at offsets from index. This may be useful in improving performance in systems that implement arrays of devices.

9.1.11 Relative addressing

Relative addressing is a technique useful in accessing structured data in a uniform manner. Relative addressing is only allowed when commands are linked. An example of relative addressing and linking for SEARCH DATA commands appropriate to direct-access devices is given in 7.4.3.

The SET LIMITS command is provided to define the limits of a linked chain of relative addressing commands. This gives an additional protection against exceeding a particular set of blocks. The SET LIMITS command has no effect on any other initiator.

9.1.12 Error reporting

If any of the following conditions occur during the execution of a command, the target shall return CHECK CONDITION status. The appropriate sense key and additional sense code should be set. The following list illustrates some error conditions and the applicable sense keys. The list does not provide an exhaustive enumeration of all conditions that may cause the CHECK CONDITION status.

Condition	Sense key
Invalid logical block address	ILLEGAL REQUEST
Unsupported option requested	ILLEGAL REQUEST
Target reset or medium change since last command from this initiator	UNIT ATTENTION

Self diagnostic failed	HARDWARE ERROR
Unrecovered read error	MEDIUM ERROR or HARDWARE ERROR
Recovered read error	RECOVERED ERROR
Overrun or other error that might be resolved by repeating the command	ABORTED COMMAND
Attempt to write on write protected medium	DATA PROTECT

In the case of an invalid logical block address, the sense data information field shall be set to the logical block address of the first invalid address.

In the case of an attempt to read a blank or previously unwritten block, the information field shall be set to the logical block address of the first blank block encountered. The data read up to that block shall be transferred (optical memory and write-once devices only).

In the case of an attempt to write a previously written block when blank checking is enabled, the information field shall be set to the logical block address of the first non-blank block encountered (optical memory and write-once devices only).

9.1.13 Examples

The following examples show some typical variations of the direct-access device. Other variations are possible.

9.1.13.1 Rotating media

The typical application of a direct-access device is a disk drive. The medium is a disk coated with a material in which flux changes may be induced. The disk drive allows direct and random access to the medium. This is done with an actuator that positions the read-write head, and a rotating disk. Data is stored and retrieved through the interaction of the read-write head and the disk.

The disk is typically divided into cylinders. Each cylinder is typically divided into tracks. Each track is typically divided into sectors. A cylinder is a set of tracks that can be accessed without movement of the actuator. A track is a recording path over which the read-write head travels during one rotation of the disk. A sector is a part of a track that contains the stored data blocks.

A logical block is stored in one or more sectors, or a sector may store more than one logical block. A sector is typically made up of a header, data and a trailer. The header contains a preamble used to synchronize read circuits to the data, an address field to identify the sector, flags to use for defect management, and a checksum that validates the header. The data contains the block of data. The trailer contains checksum or error correction information. The checksum or the error correction information allows the correction of data for medium defects.

A disk drive is ready when the disks are rotating at the correct speed and the read-write circuitry is powered and ready to access the disks. Some disks, particularly removable disks, require the user to issue load or start commands to bring the disk drive to the ready state.

A disk drive will typically have to be formatted prior to the initial access. Exceptions to this are drives that are formatted at the factory and some optical drives with pre-formatted media (see 13.1). A disk drive format will typically create the headers for each sector and initialize the data field. The MODE SELECT command is often used at format time to establish the geometry (number of heads and tracks, sectors per track, etc.) and defect management scheme. Disk drives are usually non-volatile.

The defect management scheme of a disk drive is often shielded from the user, though some aspects can be evaluated and controlled by the initiator. The direct-access device will usually reserve some sectors and tracks for recording defect lists and for reassigning defective blocks. The READ LONG and WRITE LONG commands will typically access the user data and checksum portions of the data field so that defects may be induced by the initiator to test the defect detection logic of the direct-access device.

Notches find their most typical use in a rotating disk drive. On a disk, the inner tracks are physically shorter than the outer tracks. As a result, if each track is made to store the same number of data bits, the data is packed more densely on the inner tracks than the outer tracks. By using notches, the outer tracks may be made to contain a different number of sectors than the inner tracks, while balancing the data density. This results in increased capacity

9.1.13.2 Sequential media

Some tape devices are implemented as a direct access device so that they can be used in disk oriented operating system environments. These devices are sometimes referred to as random access tape or floppy tape. These devices might be thought of as a disk drive with one or more long tracks. Access time to a logical block is usually longer than for a disk drive, since the tape must be fast forwarded or rewound to the block. As a result, the SEEK command will often be more useful for a tape than for a disk. The only way an initiator may determine if a direct-access device is a tape is by using the medium type code returned by the MODE SENSE command.

9.1.13.3 Memory media

Memory media includes devices that are traditionally used for primary storage within computer systems, such as solid state static or dynamic random access memories (SRAM or DRAM), or magnetic core or bubble memory. These devices are typically non-mechanical, and therefore the entire physical medium may be accessed in virtually the same access time. The data is typically accessed as a bit or byte and this also speeds access time. Memory devices typically store less data than disks or tapes, and are usually volatile when not protected by battery backup.

9.2 Commands for direct-access devices.

The commands for direct-access devices shall be as shown in table 108.

Table 108 - Commands for direct-access devices

Command name	Operation code	Type	Subclause
CHANGE DEFINITION	40h	O	8.2.1
COMPARE	39h	O	8.2.2
COPY	18h	O	8.2.3
COPY AND VERIFY	3Ah	O	8.2.4
FORMAT UNIT	04h	M	9.2.1
INQUIRY	12h	M	8.2.5
LOCK UNLOCK CACHE	36h	O	9.2.2
LOG SELECT	4Ch	O	8.2.6
LOG SENSE	4Dh	O	8.2.7
MODE SELECT(6)	15h	O	8.2.8
MODE SELECT(10)	55h	O	8.2.9
MODE SENSE(6)	1Ah	O	8.2.10
MODE SENSE(10)	5Ah	O	8.2.11
PRE-FETCH	34h	O	9.2.3
PREVENT ALLOW MEDIUM REMOVAL	1Eh	O	9.2.4
READ(6)	08h	M	9.2.5
READ(10)	28h	M	9.2.6
READ BUFFER	3Ch	O	8.2.12
READ CAPACITY	25h	M	9.2.7
READ DEFECT DATA	37h	O	9.2.8
READ LONG	3Eh	O	9.2.9
REASSIGN BLOCKS	07h	O	9.2.10
RECEIVE DIAGNOSTIC RESULTS	1Ch	O	8.2.13
RELEASE	17h	M	9.2.11
REQUEST SENSE	03h	M	8.2.14
RESERVE	16h	M	9.2.12
REZERO UNIT	01h	O	9.2.13
SEARCH DATA EQUAL	31h	O	9.2.14.1
SEARCH DATA HIGH	30h	O	9.2.14.2
SEARCH DATA LOW	32h	O	9.2.14.3
SEEK(6)	0Bh	O	9.2.15
SEEK(10)	2Bh	O	9.2.15
SEND DIAGNOSTIC	1Dh	M	8.2.15
SET LIMITS	33h	O	9.2.16
START STOP UNIT	1Bh	O	9.2.17
SYNCHRONIZE CACHE	35h	O	9.2.18
TEST UNIT READY	00h	M	8.2.16
VERIFY	2Fh	O	9.2.19
WRITE(6)	0Ah	O	9.2.20
WRITE(10)	2Ah	O	9.2.21
WRITE AND VERIFY	2Eh	O	9.2.22
WRITE BUFFER	3Bh	O	8.2.17
WRITE LONG	3Fh	O	9.2.23
WRITE SAME	41h	O	9.2.24

Key: M = Command implementation is mandatory.
O = Command implementation is optional.

The following operation codes are vendor-specific: 02h, 05h, 06h, 09h, 0Ch, 0Dh, 0Eh, 0Fh, 10h, 11h, 13h, 14h, 19h, 20h, 21h, 22h, 23h, 24h, 26h, 27h, 29h, 2Ch, 2Dh and C0h through FFh. All remaining operation codes for direct-access devices are reserved for future standardization.

9.2.1 FORMAT UNIT command

The FORMAT UNIT command (see table 109) formats the medium into initiator addressable logical blocks per the initiator defined options. In addition, the medium may be certified and control structures may be created for the management of the medium and defects. There is no guarantee that the medium has or has not been altered.

Table 109 - FORMAT UNIT command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (04h)							
1	Logical unit number			FmtData	CmpLst	Defect list format		
2	Vendor-specific							
3	(MSB)							
4	Interleave							(LSB)
5	Control							

The simplest mandatory form of the FORMAT UNIT command (with no format data) accomplishes medium formatting with little initiator control over defect management. The target implementation determines the degree of defect management that is to be performed. Two additional mandatory forms of this command increase the initiator's control over defect management. Several optional forms of this command further increase the initiator's control over defect management, by allowing the initiator to specify which defect list(s) are to be used, to specify defect locations (in several formats), to enable target certification, and to specify what to do in the event that defect lists are not accessible.

The FORMAT UNIT command shall be rejected with RESERVATION CONFLICT status if the logical unit is reserved, or any extent reservation, from any initiator, is active in the specified logical unit.

During the format operation, the target shall respond to commands as follows:

a) In response to all commands except REQUEST SENSE and INQUIRY, the target shall return CHECK CONDITION status unless a reservation conflict exists, in which case RESERVATION CONFLICT status shall be returned.

b) In response to the INQUIRY command, the target shall respond as commanded.

c) In response to the REQUEST SENSE command, unless an error has occurred, the target shall return a sense ey of NOT READY and an additional sense code of LOGICAL UNIT NOT READY FORMAT IN PROGRESS, with the sense-key specific bytes set for progress indication (as described in 8.2.14.1). Refer to 8.2.14.2 for a description of deferred error handling that may occur during the format operation.

NOTE 103 It is recommended that MODE SELECT parameters (if any) be set prior to issuing the FORMAT UNIT command.

During the execution of the FORMAT UNIT command, the target may perform a medium defect management algorithm (which can be controlled by the initiator, using optional forms of this command). Four sources of defect location information (hereafter called defects) are defined as follows:

a) **Primary defect list (Plist).** This is the list of defects, usually supplied by the original manufacturer of the device or medium, that are considered permanent defects. The Plist is located outside of the initiator-accessible logical block space. The Plist is accessible by the target (to reference while formatting), but it is not normally accessible by the initiator except through the READ DEFECT DATA command. Once created, the original Plist shall not be subject to change.

b) **Target certification list (CList).** This list includes defects detected by the target during an optional certification process executed during the FORMAT UNIT command. This list shall be added to the Glist.

c) **Data defect list (Dlist).** This list of defect descriptors may be supplied to the target by the initiator in the DATA OUT phase of the FORMAT UNIT command. This list shall be added to the Glist. The defect list length in the defect list header may be zero, in which case there is no Dlist.

d) **Grown defect list (Glist).** The Glist includes all defects sent by the initiator or detected by the target. The Glist does not include the Plist. If the CmpLst bit is zero, the Glist shall include Dlists provided to the target during the previous and the current FORMAT UNIT commands. The Glist shall also include:
   a) defects detected by the format operation during medium certification;
   b) defects previously identified with a REASSIGN BLOCKS command;
   c) defects previously detected by the target and automatically reallocated.

A format data (FmtData) bit of one indicates that the FORMAT UNIT parameter list (see table 110) shall be transferred during the DATA OUT phase. The DATA OUT phase consists of a defect list header (see table 111), followed by an initialization pattern descriptor, followed by zero or more defect descriptors. Each defect descriptor identifies a location on the medium that the target shall map out of the user-accessible area.

Table 110 - FORMAT UNIT parameter list

Bit / Byte	7	6	5	4	3	2	1	0
	Defect list header							
	Initialization pattern descriptor (if any)							
	Defect descriptor(s) (if any)							
0	Defect descriptor 0 (See specific table for length.)							
n								
	. . .							
0	Defect descriptor x (See specific table for length.)							
n								

The defect list header (see table 111) provides several optional format control bits. Targets that implement these bits give the initiator additional control over the use of the four defect sources, and the formatting operation. If the initiator attempts to select any function not implemented by the target, the target shall terminate the command with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN PARAMETER LIST.

Table 111 - Defect list header

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved							
1	FOV	DPRY	DCRT	STPF	IP	DSP	Immed	VS
2	(MSB) Defect list length							
3								(LSB)

A FmtData bit of zero indicates that a DATA OUT phase shall not occur. The source of defect information is nc specified.

A complete list (CmpLst) bit of one indicates that the defect list sent by the initiator is a complete list of defects. An existing defect list except the Plist shall be ignored by the target. As a result, a new Glist is constructed that contains the Dlist (if it is sent by the initiator), and the Clist (if certification is enabled). The target may add any defects i detects during the format operation to this Dlist.

A CmpLst bit of zero indicates that the defect list sent by the initiator is an addition to the existing list of defects. As a result a new Glist is constructed that contains the existing Glist, the Dlist (if it is sent by the initiator), and the Clis (if certification is enabled). The target may add any defects it detects during the format operation to this Dlist.

Table 112 defines the implementation requirements for the FORMAT UNIT command.

Table 112 - FORMAT UNIT defect descriptor format and requirements

FmtData	CmpLst	Defect list format	Defect list length	Type	Comments
0	0	000b	N/A	M	Vendor-specific
BLOCK FORMAT					
1	0	000b	Zero	M	See notes (1) and (3)
1	1	000b	Zero	M	See notes (1) and (4)
1	0	000b	>0	O	See notes (2) and (3)
1	1	000b	>0	O	See notes (2) and (4)
BYTES FROM INDEX FORMAT					
1	0	100b	Zero	O	See notes (1) and (3)
1	1	100b	Zero	O	See notes (1) and (4)
1	0	100b	>0	O	See notes (2) and (3)
1	1	100b	>0	O	See notes (2) and (4)
PHYSICAL SECTOR FORMAT					
1	0	101b	Zero	O	See notes (1) and (3)
1	1	101b	Zero	O	See notes (1) and (4)
1	0	101b	>0	O	See notes (2) and (3)
1	1	101b	>0	O	See notes (2) and (4)
VENDOR-SPECIFIC FORMAT					
1	0	110b			
1	1	110b			
All remaining codes are reserved.					
Key: M = Command implementation is mandatory. O = Command implementation is optional.					
NOTES 1 No Dlist is transferred to the target during the DATA OUT phase. 2 A Dlist is transferred to the target during the DATA OUT phase. Add the Dlist defects to the new Glist. 3 Use the existing Glist as a defect source. Add existing Glist defects to the new Glist. 4 Discard the existing Glist. Do not add existing Glist defects to the new Glist. 5 All the options described in this table cause a new Glist to be created during execution of the FORMAT UNIT command as described in the text.					

The defect list format field specifies which defect descriptor is used if the FmtData bit is one (see table 112).

The interleave field specifies the interleave that is used when performing the format operation. This allows the logical blocks to be related in a way that facilitates matching the transfer rate between the initiator and the peripheral. An interleave of zero specifies that the target use its default interleave. An interleave of one specifies that consecutive logical blocks be placed in contiguous ascending order. All other values are vendor-specific.

A format options valid (FOV) bit of zero indicates that the target shall use its default settings for the DPRY, DCRT, STPF, IP and DSP bits (see below). The initiator shall set these bits to zero. If any of these bits are not zero, the target shall terminate the command with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN PARAMETER LIST.

A FOV bit of one indicates that the target shall examine the setting of the DPRY, DCRT, STPF, IP and DSP bits. When the FOV bit is one, the DPRY, DCRT, STPF, IP and DSP bits are defined as follows.

A disable primary (DPRY) bit of zero indicates that the target shall not use portions of the medium identified as defective in the primary defect Plist for initiator addressable logical blocks. If the target cannot locate the Plist or it cannot determine whether a Plist exists, it shall perform the action specified by the STPF bit. A DPRY bit of one indicates that the target shall not use the Plist to identify defective areas of the medium. The Plist is not deleted.

A disable certification (DCRT) bit of zero indicates that the target shall perform a vendor-specific medium certification operation to generate a Clist. A DCRT bit of one indicates that the target shall not perform any vendor-specific medium certification process or format verification operation while executing the FORMAT UNIT command.

The stop format (STPF) bit controls the behaviour of the target when one of the following events occurs:

a) The target has been requested to use the primary defect list (DPRY is set to zero), or the grown defect list (CmpLst is set to zero) and the target cannot locate the list nor determine whether the list exists.
b) The target has been requested to use the primary defect list (DPRY is set to zero) or the grown defect list (CmpLst is set to zero), and the target encounters an error while accessing the defect list.

A STPF bit of zero indicates that, if one or both of the above conditions occurs, the target shall continue to execute the FORMAT UNIT command. The target shall return CHECK CONDITION status at the completion of the FORMAT UNIT command. The sense key shall be set to RECOVERED ERROR and the additional sense code shall be set to either DEFECT LIST NOT FOUND if the first condition occurred, or DEFECT LIST ERROR if the second condition occurred.

A STPF bit of one indicates that, if one or both of the above conditions occurs, the target shall terminate the FORMAT UNIT command with CHECK CONDITION status. The sense key shall be set to MEDIUM ERROR and the additional sense code shall be set to either DEFECT LIST NOT FOUND if the first condition occurred, or DEFECT LIST ERROR if the second condition occurred.

NOTE 104 The use of the FmtData bit, the CmpLst bit, and the defect header allow the initiator to control the source of the defect lists used by the FORMAT UNIT command. Setting the defect list length to zero allows the initiator to control the use of Plist and Clist without having to specify a Dlist.

An initialization pattern (IP) bit of one indicates that an initialization pattern descriptor (see 9.2.1.2) is included in the FORMAT UNIT parameter list immediately following the defect list header. An IP bit of zero indicates that an initialization pattern descriptor is not included and that the target shall use its default initialization pattern.

A disable saving parameters (DSP) bit of one specifies that the target shall not save the MODE SELECT savable parameters to non-volatile memory during the format operation. A DSP bit of zero specifies that the target shall save all the MODE SELECT savable parameters for all initiators to non-volatile memory during the format operation. Pages that are not reported as savable are not affected by the DSP bit (i.e. if pages 03h & 04h are not returned with the PS bit set they may be saved even if DSP is cleared).

An immediate (Immed) bit of zero indicates that status shall be returned after the format operation has completed. An Immed bit value of one indicates that the target shall return status as soon as the command descriptor block has been validated, and the entire defect list has been transferred.

The bit designated VS is vendor-specific.

The defect list length field in the defect list header specifies the total length in bytes of the defect descriptors that follow and does not include the initialization pattern descriptor or initialization pattern, if any. The length of the defect descriptors varies with the format of the defect list. The three formats for the defect descriptor(s) field in the defect lists are shown in 9.2.1.1.

9.2.1.1 Defect list formats

This clause describes the defect list formats used in the FORMAT UNIT, READ DEFECT DATA and translate page of the SEND DIAGNOSTIC and RECEIVE DIAGNOSTIC RESULTS commands.

NOTE 105 The selected reporting format accounts for variables that impact the information in the returned data. For example, the specific location of a defect, while constant in angular and radial location on the device, may change in reported location a format operation with different geometry parameters is performed. It is the responsibility of the initiator to use a defect list format appropriate for the intended operation with the current or future geometry parameters. If the target is able to detect that the selected defect list format would provide inconsistent results, the target may return CHECK CONDITION status.

Each block format defect descriptor (see table 113) specifies a four-byte defective block address that contains the defect.

Table 113 - Defect descriptor - Block format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
	Defective block address							
3								(LSB)

The defect list length is equal to four times the number of defect descriptors.

The defect descriptors should be in ascending order. More than one physical or logical block may be affected by each defect descriptor. A target may return CHECK CONDITION if the defect descriptors are not in ascending order.

Each byte from index defect descriptor (see table 114) specifies the location of a defect that is no more than eight bytes long.

Table 114 - Defect descriptor - Bytes from index format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
	Cylinder number of defect							
2								(LSB)
3	Head number of defect							
4	(MSB)							
	Defect bytes from index							
7								(LSB)

The defect list length is equal to eight times the number of defect descriptors.

Each descriptor is comprised of the cylinder number of defect, the head number of defect, and the defect bytes from index to the defect. The defect descriptors shall be in ascending order. The cylinder number of defect is the most significant part of the address and the defect bytes from index is the least significant part of the address. More than one physical or logical block may be affected by each defect. If the defect bytes from index has a value of FFFFFFFFh, this indicates that the entire track shall be considered defective.

Each physical sector defect descriptor (see table 115) specifies the location of a defect that is the length of a sector.

Table 115 - Defect descriptor - Physical sector format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1	Cylinder number of defect							
2								(LSB)
3	Head number of defect							
4	(MSB)							
5	Defective sector number							
6								
7								(LSB)

The defect list length is equal to eight times the number of defect descriptors.

Each descriptor is comprised of a cylinder number of defect, the head number of defect, and the defective sector number. The defect descriptors shall be in ascending order. The cylinder number of defect is the most significant part of the address and the defective sector number is the least significant part of the address. More than one block may be affected by each defect descriptor. A defective sector number of FFFFFFFFh indicates that the entire track shall be considered defective.

9.2.1.2 Initialization pattern option

The initialization pattern option specifies that the logical blocks contain the specified initialization pattern. The initialization pattern descriptor (see table 116) is sent to the target as part of the FORMAT UNIT parameter list.

Table 116 - Initialization pattern descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	IP modifier		Reserved					
1	Pattern type							
2	(MSB)							
3	Initialization pattern length							(LSB)
4								
n	Initialization pattern							

NOTE 106 The initialization pattern option is not intended for media analysis or certification. This option may only initialize the initiator accessible area of the media to the specified pattern and may not write to any initiator inaccessible areas of the disk.

The IP modifier field specifies the type and location of a header that modifies the initialization pattern (see table 117)

Table 117 - Initialization pattern modifier

IP modifier	Description
00b	No header. The target shall not modify the initialization pattern.
01b	The target shall overwrite the initialization pattern to write the logical block address in the first four bytes of the logical block. The logical block address shall be written with the most significant byte first.
10b	The target shall overwrite the initialization pattern to write the logical block address in the first four bytes of each physical block contained within the logical block. The The lowest numbered logical block or part thereof that occurs within the physical block is used. The logical block address shall be written with the most significant byte first.
11b	Reserved.

The initialization pattern type field (see table 118) indicates the type of pattern the target shall use to initialize each logical block within the initiator accessible portion of the medium. All bytes within a logical block shall be written with the initialization pattern. The initialization pattern is modified by the IP modifier field as described in table 117.

Table 118 - Initialization pattern type

Initialization pattern type	Description
00h	Use default pattern. (note 1)
01h	Repeat the initialization pattern as required to fill the logical block. (note 2)
02h - 7Fh	Reserved
80h - FFh	Vendor-specific

Notes
1) If the initialization pattern length is not zero the target shall terminate the command with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code to INVALID FIELD IN PARAMETER LIST.
2) If the initialization pattern length is zero the target shall terminate the command with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code to INVALID FIELD IN PARAMETER LIST.

The initialization pattern length field indicates the number of bytes contained in the initialization pattern. If the length exceeds the current logical block size the target shall terminate the command with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN PARAMETER LIST. The pattern is modified by the IP modifier field.

9.2.2 LOCK UNLOCK CACHE command

The LOCK UNLOCK CACHE command (see table 119) requests that the target disallow or allow logical blocks within the specified range to be removed from the cache memory by the target's cache replacement algorithm. Locked logical blocks may be written to the medium when modified, but a copy of the modified logical block shall remain in the cache memory.

Table 119 - LOCK UNLOCK CACHE command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (36h)							
1	Logical unit number			Reserved			Lock	RelAdr
2	(MSB)							
3				Logical block address				
4								
5								(LSB)
6	Reserved							
7	(MSB)							
8				Number of blocks				(LSB)
9	Control							

A lock bit of one indicates that any logical block in the specified range that is currently present in the cache memory shall be locked into cache memory. Only logical blocks that are already present in the cache memory are actually locked. A lock bit of zero indicates that all logical blocks in the specified range that are currently locked into the cache memory shall be unlocked, but not necessarily removed.

A relative address (RelAdr) bit of one indicates that the logical block address field is a two's complement displacement. This negative or positive displacement shall be added to the logical block address last accessed on the logical unit to form the logical block address for this command. This feature is only available when linking commands. The feature requires that a previous command in the linked group have accessed a block of data on the logical unit.

A RelAdr bit of zero indicates that the logical block address field specifies the first logical block of the range of logical blocks to be operated on by this command.

The number of blocks specifies the total number of contiguous logical blocks within the range. A number of blocks field of zero indicates that all remaining logical blocks on the logical unit shall be within the range.

Multiple locks may be in effect from more than one initiator. Locks from different initiators may overlap. An unlock of an overlapped area does not release the lock of another initiator.

9.2.3 PRE-FETCH command

The PRE-FETCH command (see table 120) requests that the target transfer the specified logical blocks to the cache memory. No data shall be transferred to the initiator.

Table 120 - PRE-FETCH command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (34h)							
1	Logical unit number			Reserved			Immed	RelAdr
2	(MSB)							
3								
4				Logical block address				
5								(LSB)
6	Reserved							
7	(MSB)							
8				Transfer length				(LSB)
9	Control							

An immediate (Immed) bit of one indicates that status shall be returned as soon as the command descriptor block has been validated. An Immed bit of zero indicates that status shall be returned after the operation is complete.

See 9.2.2 for a definition of the RelAdr bit and the logical block address field.

The transfer length field specifies the number of contiguous logical blocks of data that shall be transferred to the target's cache memory. A transfer length of zero indicates that the contiguous logical blocks up to and including the last logical block of the logical unit shall be transferred to the target's cache memory. Any other value indicates the number of logical blocks that shall be transferred. The target may elect to not transfer logical blocks that already are contained in the cache memory.

If the Immed bit is zero and the specified logical blocks were successfully transferred to the cache memory, the target shall return CONDITION MET status. If the link bit (see 7.2.7) is one, the target shall return INTERMEDIATE-CONDITION MET status.

If Immed is one, and the unlocked cache memory has sufficient capacity to accept all of the specified logical blocks, the target shall return CONDITION MET status. If the link bit (see 7.2.7) is one, the target shall return INTERMEDIATE-CONDITION MET status.

If Immed is one, and the unlocked cache memory does not have sufficient capacity to accept all of the specified logical blocks, the target shall return GOOD status. The target shall transfer to cache memory as many logical blocks as will fit. If the link bit (see 7.2.7) is one, the target shall return INTERMEDIATE status.

9.2.4 PREVENT ALLOW MEDIUM REMOVAL command

The PREVENT ALLOW MEDIUM REMOVAL command (see table 121) requests that the target enable or disable the removal of the medium in the logical unit. This mechanism is independent of device reservations and the target shall not allow medium removal if any initiator currently has medium removal prevented.

Table 121 - PREVENT ALLOW MEDIUM REMOVAL command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (1Eh)							
1	Logical unit number			Reserved				
2	Reserved							
3	Reserved							
4	Reserved							Prevent
5	Control							

The prevention of medium removal shall begin when any initiator issues a PREVENT ALLOW MEDIUM REMOVAL command with a prevent bit of one (medium removal prevented). The prevention of medium removal for the logical unit shall terminate:

a) after all initiators that have medium removal prevented issue PREVENT ALLOW MEDIUM REMOVAL commands with a prevent bit of zero, and the target has successfully performed a synchronize cache operation;
b) upon the receipt of a BUS DEVICE RESET message from any initiator; or
c) upon a hard RESET condition.

While a prevention of medium removal condition is in effect the target shall inhibit mechanisms that normally allow removal of the medium by an operator.

9.2.5 READ(6) command

The READ(6) command (see table 122) requests that the target transfer data to the initiator. The most recent data value written in the addressed logical block shall be returned.

Table 122 - READ(6) command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (08h)							
1	Logical unit number			(MSB)				
2	Logical block address							
3								(LSB)
4	Transfer length							
5	Control							

The cache control bits (see 9.2.6) are not provided for this command. Targets with cache memory may have values for the cache control bits that affect the READ(6) command; however, no default value is defined by this standard. If explicit control is required, the READ(10) command should be used.

The logical block address field specifies the logical block at which the read operation shall begin.

The transfer length field specifies the number of contiguous logical blocks of data to be transferred. A transfer length of zero indicates that 256 logical blocks shall be transferred. Any other value indicates the number of logical blocks that shall be transferred.

9.2.6 READ(10) command

The READ(10) command (see table 123) requests that the target transfer data to the initiator. The most recent data value written in the addressed logical block shall be returned.

Table 123 - READ(10) command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (28h)							
1	Logical unit number			DPO	FUA	Reserved		RelAdr
2	(MSB)							
3	Logical block address							
4								
5								(LSB)
6	Reserved							
7	(MSB)							
8	Transfer length							(LSB)
9	Control							

A disable page out (DPO) bit of one indicates that the target shall assign the logical blocks accessed by this command the lowest priority for being fetched into or retained by the cache. A DPO bit of one overrides any retention priority specified in the cache page (see 9.3.3.1). A DPO bit of zero indicates the priority shall be determined by the retention priority fields in the cache page. All other aspects of the algorithm implementing the cache memory replacement strategy are not defined by this standard.

NOTE 107 The DPO bit is used to control replacement of logical blocks in the cache memory when the host has information on the future usage of the logical blocks. If the DPO bit is set to one, the host knows the logical blocks accessed by the command are not likely to be accessed again in the near future and should not be put in the cache memory nor retained by the cache memory. If the DPO bit is zero, the host expects that logical blocks accessed by this command are likely to be accessed again in the near future.

A force unit access (FUA) bit of one indicates that the target shall access the media in performing the command prior to returning GOOD status. Read commands shall access the specified logical blocks from the media (i.e. the data is not directly retrieved from the cache). In the case where the cache contains a more recent version of a logical block than the media, the logical block shall first be written to the media. Write commands shall not return GOOD status until the logical blocks have actually been written on the media (i.e. the data is not write cached).

An FUA bit of zero indicates that the target may satisfy the command by accessing the cache memory. For read operations, any logical blocks that are contained in the cache memory may be transferred to the initiator directly from the cache memory. For write operations, logical blocks may be transferred directly to the cache memory. GOOD status may be returned to the initiator prior to writing the logical blocks to the medium. Any error that occurs after the GOOD status is returned is a deferred error, and information regarding the error is not reported until a subsequent command.

See 9.2.2 for a definition of the RelAdr bit and the logical block address field.

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. Any other value indicates the number of logical blocks that shall be transferred.

9.2.7 READ CAPACITY command

The READ CAPACITY command (see table 124) provides a means for the initiator to request information regarding the capacity of the logical unit.

Table 124 - READ CAPACITY command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (25h)							
1	Logical unit number			Reserved				RelAdr
2	(MSB)							
3				Logical block address				
4								
5								(LSB)
6	Reserved							
7	Reserved							
8	Reserved							PMI
9	Control							

See 9.2.2 for a definition of the RelAdr bit and the logical block address field.

The logical block address shall be zero if the PMI bit is zero. If the PMI bit is zero and the logical block address i not zero, the target shall return a CHECK CONDITION status, the sense key shall be set to ILLEGAL REQUEST an the additional sense code set to ILLEGAL FIELD IN CDB.

A partial medium indicator (PMI) bit of zero indicates that the returned logical block address and the block length i bytes are those of the last logical block on the logical unit.

A PMI bit of one indicates that the returned logical block address and block length in bytes are those of the logic block address after which a substantial delay in data transfer will be encountered. This returned logical block addres shall be greater than or equal to the logical block address specified by the RelAdr and logical block address field in the command descriptor block.

NOTE 108 This function is intended to assist storage management software in determining whether there is sufficient space on the current track, cylinder, etc., to contain a frequently accessed data structure, such as a file directory or file index, without incurring an access delay.

The READ CAPACITY data (see table 125) shall be sent during the DATA IN phase of the command.

Table 125 - READ CAPACITY data

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
- - -				Returned logical block address				
3								(LSB)
4	(MSB)							
- - -				Block length In bytes				
7								(LSB)

9.2.8 READ DEFECT DATA command

The READ DEFECT DATA command (see table 126) requests that the target transfer the medium defect data to the initiator.

Table 126 - READ DEFECT DATA command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (37h)							
1	Logical unit number			Reserved				
2	Reserved			Plist	Glist	Defect list format		
3	Reserved							
4	Reserved							
5	Reserved							
6	Reserved							
7	(MSB) Allocation length							
8								(LSB)
9	Control							

If the target is unable to access any medium defect data, it shall terminate the command with CHECK CONDITION status. The sense key shall be set to either MEDIUM ERROR, if a medium error occurred, or NO SENSE, if the list does not exist; and the additional sense code shall be set to DEFECT LIST NOT FOUND.

NOTE 109 Some targets may not be able to return medium defect data until after a FORMAT UNIT command has been completed successfully.

A primary defect list (Plist) bit of one request that the target return the primary list of defects. A Plist bit of zero requests that the target not return the primary list of defects.

A grown defect list (Glist) bit of one request that the target return the grown defect list. A Glist bit of zero requests that the target not return the grown defect list.

A Plist bit of one and a Glist bit of one requests that the target return the primary and the grown defect lists. The order in which the lists are returned is vendor-specific. Whether the lists are merged or not is vendor-specific.

A Plist bit of zero and a Glist bit of zero requests that the target return only the defect list header.

The defect list format field is used by the initiator to indicate the preferred format for the defect list. This field is intended for those targets capable of returning more than one format, as defined in the FORMAT UNIT command (see 9.2.1.2, defect list format). A target unable to return the requested format shall return the defect list in its default format (see the defect list format field in the defect list header below).

If the requested defect list format and the returned defect list format are not the same, the target shall transfer the defect data and then terminate the command with CHECK CONDITION status. The sense key shall be set to RECOVERED ERROR and the additional sense code shall be set to DEFECT LIST NOT FOUND.

The READ DEFECT DATA defect list (see table 127) contains a four-byte header, followed by zero or more defect descriptors.

Table 127 - READ DEFECT DATA defect list

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved							
1	Reserved			Plist	Glist	Defect list format		
2	(MSB)							
3	Defect list length							(LSB)
	Defect descriptor(s) (if any)							
0	Defect descriptor 0 (See specific table for length.)							
n								
	.							
	.							
0	Defect descriptor x (See specific table for length.)							
n								

A Plist bit of one indicates that the data returned contains the primary defect list. A Plist bit of zero indicates that the data returned does not contain the primary defect list.

A Glist bit of one indicates that the data returned contains the grown defect list. A Glist bit of zero indicates that the data returned does not contain the grown defect list.

The defect list format field indicates the format of the defect descriptors returned by the target. This field is definec in the FORMAT UNIT command (see 9.2.1.2).

NOTE 110 The use of the block format is not recommended. There is no universal model that sensibly defines the meaning of the logical block address of a defect. In the usual case, a defect that has been reassigned no longer has a logical block address.

Defect descriptors returned in the block format are vendor-specific. Defect descriptors returned in the physical sector format may or may not include defects in areas not accessible to the initiator. Defect descriptors returned in bytes-from-index format shall comprise a complete list of defects. A complete list of defects may include defects in areas not within the capacity returned in the READ CAPACITY command.

The defect list length field specifies the length in bytes of the defect descriptors that follow. The defect list length is equal to four or eight times the number of defect descriptors, depending on the format of the returned descriptors (see 9.2.1.1).

If the allocation length is insufficient to transfer all of the defect descriptors, the defect list length shall not be adjusted to reflect the truncation. The target shall not create CHECK CONDITION status. The initiator is responsible for comparing the defect list length and the allocation length to ensure that a partial list was not received.

NOTE 111 The initiator may determine the length of the defect list by sending the READ DEFECT DATA command with an allocation length of four. The target will return the defect list header that contains the length of the defect list.

The defect descriptors may or may not be sent in ascending order. The initiator may determine the exact number of defects by dividing the defect list length by the length of a single defect descriptor for the returned format.

9.2.9 READ LONG command

The READ LONG command (see table 128) requests that the target transfer data to the initiator. The data passed during the READ LONG command is vendor-specific, but shall include the data bytes and the ECC bytes recorded on the medium. The most recent data written in the addressed logical block shall be returned.

Table 128 - READ LONG command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (3Eh)							
1	Logical unit number			Reserved			CORRCT	RelAdr
2	(MSB)							
3				Logical block address				
4								
5								(LSB)
6	Reserved							
7	(MSB)							
8				Byte transfer length				(LSB)
9	Control							

NOTE 112 Any other bytes that can be corrected by ECC should be included (e.g. data synchronization mark within the area covered by ECC). It is not important for the ECC bytes to be at the end of the data bytes; however, they should be in the same order as they are on the media.

A corrected (CORRCT) bit of zero causes a logical block to be read without any correction made by the target. A CORRCT bit of one causes the data to be corrected by ECC before being transferred to the initiator.

See 9.2.2 for a definition of the RelAdr bit and the logical block address field.

The byte transfer length field should specify exactly the number of bytes of data that are available for transfer. If a non-zero byte transfer length does not exactly match the available data length, the target shall terminate the command with CHECK CONDITION status, the sense key shall be set to ILLEGAL REQUEST and an additional sense code set to INVALID FIELD IN CDB. The valid and ILI bits shall be set to one and the information field shall be set to the difference (residue) of the requested length minus the actual length in bytes. Negative values shall be indicated by two's complement notation.

A byte transfer length of zero indicates that no bytes shall be transferred and shall not be considered an error.

9.2.10 REASSIGN BLOCKS command

The REASSIGN BLOCKS command (see table 129) requests the target to reassign the defective logical blocks to another area on the medium set aside for this purpose. The target should also record the location of the defective logical blocks to the grown defect list if such a list is supported. More than one physical or logical block may be relocated by each defect descriptor sent by the initiator. This command does not alter the contents or location of the Plist (see 9.2.1, FORMAT UNIT command).

Table 129 - REASSIGN BLOCKS command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (07h)							
1	Logical unit number			Reserved				
2	Reserved							
3	Reserved							
4	Reserved							
5	Control							

The initiator transfers a defect list that contains the logical block addresses to be reassigned. The target shall reassign the physical medium used for each logical block address in the list. The data contained in the logical blocks specified in the defect list may be altered, but the data in all other logical blocks on the medium shall be preserved.

NOTE 113 The effect of specifying a logical block to be reassigned that previously has been reassigned is to reassign the block again. Over the life of the medium, a logical block can be assigned to multiple physical addresses until no more spare locations remain on the medium.

The REASSIGN BLOCKS defect list (see table 130) contains a four-byte header followed by one or more defect descriptors. The length of each defect descriptor is four bytes.

Table 130 - REASSIGN BLOCKS defect list

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved							
1	Reserved							
2	(MSB) Defect list length							
3								(LSB)
	Defect descriptor(s)							
0	(MSB) Defect logical block address							
3								(LSB)
	.							
0	(MSB) Defect logical block address							
3								(LSB)

The defect list length field specifies the total length in bytes of the defect descriptors that follow. The defect list length is equal to four times the number of defect descriptors and does not include the defect list header length.

The defect descriptor specifies a four-byte defect logical block address that contains the defect. The defect descriptors shall be in ascending order.

If the logical unit has insufficient capacity to reassign all of the logical blocks specified in the defect descriptors, the command shall terminate with CHECK CONDITION status, the sense key shall be set to HARDWARE ERROR and the additional sense code set to NO DEFECT SPARE LOCATION AVAILABLE.

If the logical unit is unable to successfully complete a REASSIGN BLOCKS command, the command shall terminate with CHECK CONDITION status with the appropriate sense information. The logical block address of the first defect descriptor not reassigned shall be returned in the command-specific information field of the sense data. If information about the first defect descriptor not reassigned is not available, or if all the defects have been reassigned, this field shall be set to FFFFFFFFh.

If the REASSIGN BLOCKS command failed due to an unexpected unrecoverable read error that would cause the loss of data in a block not specified in the defect list, the logical block address of the unrecoverable block shall be returned in the information field of the sense data and the valid bit shall be set to one.

NOTE 114 If the REASSIGN BLOCKS command returns CHECK CONDITION status and the sense data command-specific information field contains a valid logical block address, the initiator should remove all defect descriptors from the defect list prior to the one returned in the command-specific information field. If the sense key is MEDIUM ERROR and the valid bit is one (the information field contains the valid block address) the initiator should insert that new defective logical block address into the defect list and reissue the REASSIGN BLOCKS command with the new defect list. Otherwise, the initiator should perform any corrective action indicated by the sense data and then reissue the REASSIGN BLOCKS command with the new defect list.

9.2.11 RELEASE command

The RELEASE command (see table 131) is used to release a previously reserved logical unit, or, if the extent release option is implemented, to release previously reserved extents within a logical unit.

Table 131 - RELEASE command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (17h)							
1	Logical unit number			3rdPty	Third party device ID			Extent
2	Reservation identification							
3	Reserved							
4	Reserved							
5	Control							

The RESERVE and RELEASE commands provide the basic mechanism for contention resolution in multiple-initiator systems. A reservation may only be released by the initiator that made it. It is not an error for an initiator to attempt to release a reservation that is not currently valid, or is held by another initiator. In this case, the target shall return GOOD status without altering any other reservation.

NOTE 115 The reservation queuing option in X3.131-1986 has been removed from SCSI-2.

9.2.11.1 Logical unit release

Implementation of logical unit release is mandatory. If the extent bit is zero, this command shall cause the target to terminate all non-third-party logical unit and extent reservations that are active from the initiator to the specified logical unit. The reservation ID field in the command descriptor block is ignored by the target.

9.2.11.2 Extent release

Implementation of extent release is optional. If the extent bit is one and the extent release option is not implemented, then the RELEASE command shall be terminated with CHECK CONDITION status and the sense key shall be set to ILLEGAL REQUEST. This option shall be implemented if the extent reservation option (see 9.2.12.2) is implemented.

If the extent bit is one and the extent release option is implemented, this command shall cause any reservation from the requesting initiator with a matching reservation identification to be terminated. Other reservations from the requesting initiator shall remain in effect.

9.2.11.3 Third-party release

Implementation of third-party release is mandatory. Third-party release allows an initiator to release a logical unit or extents within a logical unit that were previously reserved using third-party reservation (see 9.2.12.3). Third-party release shall be implemented and is intended for use in multiple-initiator systems that use the COPY command.

If the third-party (3rdPty) bit is zero, then a third-party release is not requested. If the 3rdPty bit is one then the target shall release the specified logical unit or extents, but only if the reservation was made using a third-party reservation by the initiator that is requesting the release for the same SCSI device as specified in the third-party device ID field.

If the 3rdPty bit is one the target shall not modify the mode parameters for commands received from the third-party device even if the target implements the transfer of mode parameters with a third-party RESERVE command.

NOTE 116 If a target implements independent storage of mode parameters for each initiator, a third-party RESERVE command copies the current mode parameters for the initiator that sent the RESERVE command to the current mode parameters for the initiator specified as the third-party device (usually a copy master device). A unit attention condition notifies the third-party of the changed mode parameters due to the reservation. A successful third-party RELEASE command does not return the third-party devices' current mode parameters back to their previous values. The third-party device can issue MODE SENSE and MODE SELECT commands to query and modify the mode parameters.

9.2.12 RESERVE command

The RESERVE command (see table 132) is used to reserve a logical unit or, if the extent reservation option is implemented, extents within a logical unit.

Table 132 - RESERVE command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (16h)							
1	Logical unit number			3rdPty	Third party device ID			Extent
2	Reservation identification							
3	(MSB) Extent list length							
4								(LSB)
5	Control							

The RESERVE and RELEASE commands provide the basic mechanism for contention resolution in multiple-initiator systems. The third-party reservation allows logical units or extents to be reserved for another specified SCSI device.

NOTE 117 The reservation queuing option in X3.131-1986 has been removed from SCSI-2.

9.2.12.1 Logical unit reservation

Implementation of logical unit reservation is mandatory. If the extent bit is zero, this command shall request that the entire logical unit be reserved for the exclusive use of the initiator until the reservation is superseded by another valid RESERVE command from the initiator that made the reservation or until released by a RELEASE command from the same initiator that made the reservation, by a BUS DEVICE RESET message from any initiator, by a hard RESET condition, or by a power on cycle. A logical unit reservation shall not be granted if the logical unit or any extent is reserved by another initiator. It shall be permissible for an initiator to reserve a logical unit that is currently reserved by that initiator. If the extent bit is zero, the reservation identification and the extent list length shall be ignored.

If the logical unit, or any extent within the logical unit is reserved for another initiator, the target shall return RESERVATION CONFLICT status.

If, after honouring the reservation, any other initiator attempts to perform any command on the reserved logical unit other than an INQUIRY, REQUEST SENSE, PREVENT ALLOW MEDIUM REMOVAL (with a prevent bit of zero), or a RELEASE command the command shall be rejected with RESERVATION CONFLICT status.

9.2.12.2 Extent reservation

Implementation of extent reservation is optional. The reservation identification field provides a means for an initiator to identify each extent reservation. This allows an initiator in a multiple tasking environment, to have multiple reservations outstanding. The reservation identification is used in the RELEASE command to specify which reservation is to be released. It is also used in superseding RESERVE commands to specify which reservation is to be superseded.

If the extent reservation option is implemented, then the extent release option (see 9.2.11.2) shall also be implemented. These options permit multiple extents within the logical unit to be reserved, each with a separate reservation type.

If the extent bit is one, and the extent reservation option is implemented, then the target shall process the reservation request as follows:

a) The extent list shall be checked for the number of extents in the reservation request. If the extent list length is zero, no current reservations shall be changed, no new reservations shall be created, and this condition shall not be treated as an error. If the extent list contains more extents than are supported on the logical unit, the command shall be terminated with CHECK CONDITION status and the sense key shall be set to ILLEGAL REQUEST. If the extent list contains more extents than are currently available on the logical unit, then the target shall return a RESERVATION CONFLICT status.
b) The extent list shall be checked for valid extent logical block addresses. If any logical block address is invalid for this logical unit, the command shall be terminated with CHECK CONDITION status and the sense key shall be set to ILLEGAL REQUEST. The extent list shall be checked for invalid extent overlaps (as defined by reservation type) with other extent descriptors in the extent list and if invalid overlaps are found, the command shall be terminated with CHECK CONDITION status and the sense key shall be set to ILLEGAL REQUEST.
c) If the requested reservation does not conflict with an existing reservation, the extents specified shall be reserved until superseded by another valid RESERVE command from the initiator that made the reservation or until released by a RELEASE command from the same initiator, by a BUS DEVICE RESET message from any initiator, or by a hard RESET condition. If either of the last two conditions occur, the next command from each initiator shall be terminated with CHECK CONDITION status and the sense key shall be set to UNIT ATTENTION.
d) If the reservation request conflicts with an existing reservation, then the target shall return a RESERVATION CONFLICT status.

If the extent bit is one, and the extent reservation option is not implemented, then the RESERVE command shall be rejected with CHECK CONDITION status and the sense key shall be set to ILLEGAL REQUEST.

The size of the extent list shall be defined by the extent list length field. The extent list shall consist of zero or more descriptors as shown in table 133. Each extent descriptor defines an extent beginning at the specified logical block address for the specified number of blocks. If the number of blocks is zero, the extent shall begin at the specified logical block address and continue through the last logical block address on the logical unit.

Table 133 - Data format of extent descriptors

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved					RelAdr	Reservation type	
1	(MSB)							
2	Number of blocks							
3								(LSB)
4	(MSB)							
5	Logical block address							
6								
7								(LSB)

The reservation type field shall define the type of reservation in effect for the extent. The types of reservation are defined in table 134.

Table 134 - Reservation types

Reservation type	Description
00b	Read shared
01b	Write exclusive
10b	Read exclusive
11b	Exclusive access

a) **Read exclusive.** While this reservation is active, no other initiator shall be permitted read operations to the indicated extent. This reservation shall not inhibit write operations from any initiator or conflict with a write

exclusive reservation; however, read exclusive, exclusive access, and read shared reservations that overlap this extent shall conflict with this reservation.

b) **Write exclusive.** While this reservation is active, no other initiator shall be permitted write operations to the indicated extent. This reservation shall not inhibit read operations from any initiator or conflict with a read exclusive reservation from any initiator. This reservation shall conflict with write exclusive, exclusive access, and read shared reservations that overlap this extent.

c) **Exclusive access.** While this reservation is active, no other initiator shall be permitted any access to the indicated extent. All reservation types that overlap this extent shall conflict with this reservation.

d) **Read shared.** While this reservation is active, no write operations shall be permitted by any initiator to the indicated extent. This reservation shall not inhibit read operations from any initiator or conflict with a read shared reservation. Read exclusive, write exclusive, and exclusive access reservations that overlap with this extent shall conflict with this reservation.

If the relative address bit is one, the logical block address in the extent descriptor shall be treated as a two's complement displacement. This displacement shall be added to the logical block address last accessed on the logical unit to form the logical block address for this extent. This feature is only available when linking commands and requires that a previous command in the linked group has accessed a logical block on the logical unit; if not, the RESERVE command shall be terminated with CHECK CONDITION status and the sense key shall be set to ILLEGAL REQUEST.

If an initiator attempts a command to a logical block that has been reserved and that access is prohibited by the reservation, the command shall not be performed and the command shall be terminated with a RESERVATION CONFLICT status. If a reservation conflict precludes any part of the command, none of the command shall be performed. COPY commands shall be terminated with CHECK CONDITION status and the sense key shall be set to DATA PROTECT if any part of the copy operation is prohibited by an active reservation. If any extent in a logical unit is reserved in any way, by any initiator, a FORMAT UNIT command shall be rejected with a RESERVATION CONFLICT status.

9.2.12.3 Third-party reservation

Implementation of third-party reservation is mandatory. The third-party reservation for the RESERVE command allows an initiator to reserve a logical unit or extents within a logical unit for another SCSI device. This is intended for use in multiple-initiator systems that use the COPY command. Third-party reservation is required.

If the third-party (3rdPty) bit is zero, then a third-party reservation is not requested. If the 3rdPty bit is one then the target shall reserve the specified logical unit or extents for the SCSI device specified in the third-party device ID field. The target shall preserve the reservation until it is superseded by another valid RESERVE command from the initiator that made the reservation or until it is released by the same initiator, by a BUS DEVICE reset message from any initiator, or a hard reset condition. The target shall ignore any attempt to release the reservation made by any other initiator.

If independent sets of parameters are implemented, a third party reservation shall cause the target to transfer the set of parameters in effect for the initiator of the RESERVE command to the parameters used for commands from the third party device. Any subsequent command issued by the third-party device is executed according to the mode parameters in effect for the initiator that sent the RESERVE command.

NOTE 118 This transfer of the mode parameters is applicable to target devices which store mode information independently for different initiators. This mechanism allows an initiator to set the mode parameters of a target for the use of a copy master (i.e. the third-party device). The third-party copy master may subsequently issue a MODE SELECT command to modify the mode parameters.

9.2.12.4 Superseding reservations

Implementation of superseding reservations is mandatory. An initiator that holds a current reservation (unit or extent) may modify that reservation by issuing another RESERVE command (unit or extent) to the same logical unit. The superseding RESERVE command shall release the previous reservation state (unit or extent) when the new reservation request is granted. If the superseding reservation is for an extent reservation and the current reservation is also an extent reservation, the current extent reservation identification value is used for the superseding reservation. The current reservation shall not be modified if the superseding reservation request cannot be granted. If the superseding reservation cannot be granted because of conflicts with a previous reservation (other than the reservation being superseded), then the target shall return RESERVATION CONFLICT status.

NOTE 119 Superseding reservations allow the SCSI device ID to be changed on a reservation using the third-party reservation option. This capability is necessary for certain situations when using COMPARE, COPY, and COPY AND VERIFY commands.

9.2.13 REZERO UNIT command

The REZERO UNIT command (see table 135) requests that the target set the logical unit to a specific state. See vendor specifications for details.

Table 135 - REZERO UNIT command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (01h)							
1	Logical unit number			Reserved				
2	Reserved							
3	Reserved							
4	Reserved							
5	Control							

9.2.14 SEARCH DATA commands

The SEARCH DATA commands (see table 136) search one or more logical blocks for equality or inequality to a data pattern. The concept of records within a logical block is used to allow multiple records within a logical block to be searched.

Table 136 - SEARCH DATA commands

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (31h 30h 32h)							
1	Logical unit number			Invert	Reserved		SpnDat	RelAdr
2	(MSB)							
3				Logical block address				
4								
5								(LSB)
6	Reserved							
7	(MSB)							
8				Number of blocks to search				(LSB)
9	Control							

The invert bit determines whether the search condition is to be inverted. See 9.2.14.1 through 9.2.14.3 for a description of the search conditions for the individual SEARCH DATA commands.

A spanned data (SpnDat) bit of zero indicates that each record shall be wholly contained within a single block. Any space at the end of a block that is smaller than the record length is ignored by the SEARCH DATA commands. A SpnDat bit of one indicates that records span block boundaries (i.e. record may start in one block and end in the next or a subsequent block.

The number of blocks to search field specifies the maximum number of contiguous logical blocks to be searched. A value of zero indicates that no logical blocks shall be searched. This condition shall not be considered an error. Any other value indicates the maximum number of logical blocks that shall be searched.

A link bit (see 7.2.7) of zero indicates a non-linked command and if the search is satisfied, the command shall be terminated with a CONDITION MET status. A REQUEST SENSE command can then be issued to determine the logical block address and record offset of the matching record. If the search is not satisfied and no error occurs, the command shall be terminated with GOOD status.

A link bit (see 7.2.7) of one indicates a command is linked to the SEARCH DATA command and if the search is satisfied, INTERMEDIATE-CONDITION MET status is returned and the next command is executed. If the RelAdr bit in the next command is one, the logical block address of the next command is used as a displacement from the logical block address at which the search was satisfied. If a linked search is not satisfied, the command is terminated with CHECK CONDITION status. A REQUEST SENSE command may then be issued.

A REQUEST SENSE command following a satisfied SEARCH DATA command shall:

a) return a sense key of EQUAL if the search was satisfied by an exact match. If the search was satisfied by an inequality then a sense key of NO SENSE shall be returned.
b) return the valid bit set to one.
c) return the logical block address of the logical block containing the first matching record in the information field.
d) return the record offset of the matching record in the command-specific information field.

A REQUEST SENSE command following a SEARCH DATA command that is not satisfied shall:

a) return a sense key of NO SENSE, if no errors occurred during the command execution.
b) return the valid bit set to zero.

The SEARCH DATA parameter list (see table 137) contains a fourteen-byte header, followed by one or more search argument descriptors.

Table 137 - SEARCH DATA parameter list

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3	Logical record length							(LSB)
4	(MSB)							
7	First record offset							(LSB)
8	(MSB)							
11	Number of records							(LSB)
12	(MSB)							
13	Search argument length							(LSB)
	Search argument descriptor(s)							
0	(MSB)							
3	Displacement							(LSB)
4	(MSB)							
5	Pattern length							(LSB)
6								
n	Pattern							

The logical record length field specifies the record length in bytes.

The first record offset field specifies the number of bytes that shall be ignored in the first logical block before the search begins. If the value of the first record offset field shall is larger than the logical block length the target shall terminate the command with a CHECK CONDITION status, set the sense key to ILLEGAL REQUEST and set the additional sense code to INVALID FIELD IN PARAMETERS LIST. Subsequent logical blocks shall be searched beginning with the first byte in the logical block. This permits one or more records to be skipped initially.

The number of records field specifies the maximum number of records that shall be searched by this command. An search shall terminate when the search pattern is found or when the number of records is exhausted or when the number of blocks to search is exhausted.

The search argument length field specifies the length in bytes of all the search argument descriptors that follow.

NOTE 120 Since the pattern length can vary, there is no fixed multiple of the search argument descriptor to determine the search argument length.

The search argument descriptors specify one or more search conditions to execute within a single record in order to satisfy the search. Each search argument descriptor is made up of a displacement field, a pattern length field, and a pattern field.

The displacement field specifies the displacement in bytes of the first byte of the data to be compared from the start of the logical record.

The pattern length field specifies the length in bytes of the pattern that follows.

The pattern field specifies the data to compare to the logical record.

9.2.14.1 SEARCH DATA EQUAL command

The SEARCH DATA EQUAL command (see table 136, operation code 31h) shall be satisfied by the first logical record searched that contains data that satisfies all of the search argument descriptor(s). If the invert bit in the command descriptor block is zero, the search argument descriptor(s) shall be satisfied by data in the logical record being equal to the data in the pattern. If the invert bit is one, the search argument descriptor(s) shall be satisfied by data in the logical record being not equal to the data in the pattern. (See 9.2.14.)

9.2.14.2 SEARCH DATA HIGH command

The SEARCH DATA HIGH command (see table 136, operation code 30h) shall be satisfied by the first logical record searched that contains data that satisfies all of the search argument descriptor(s). If the invert bit in the command descriptor block is zero, the search argument descriptor(s) shall be satisfied by data in the logical record being greater than the data in the pattern. If the invert bit is one, the search argument descriptor(s) shall be satisfied by data in the logical record being less than or equal to the data in the pattern. (See 9.2.14.)

9.2.14.3 SEARCH DATA LOW command

The SEARCH DATA LOW command (see table 136, operation code 32h) shall be satisfied by the first logical record searched that contains data that satisfies all of the search argument descriptor(s). If the invert bit in the command descriptor block is zero, the search argument descriptor(s) shall be satisfied by data in the logical record being less than the data in the pattern. If the invert bit is one, the search argument descriptor(s) shall be satisfied by data in the logical record being greater than or equal to the data in the pattern. (See 9.2.14.)

9.2.15 SEEK(6) and SEEK(10) commands

The SEEK(6) (see table 138) and SEEK(10) (see table 139) commands request that the logical unit seek to the specified logical block address.

Table 138 - SEEK(6) command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (0Bh)							
1	Logical unit number			(MSB)				
2	Logical block address							
3								(LSB)
4	Reserved							
5	Control							

Table 139 - SEEK(10) command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (2Bh)							
1	Logical unit number			Reserved				
2	(MSB)							
3				Logical block address				
4								
5								(LSB)
6	Reserved							
7	Reserved							
8	Reserved							
9	Control							

9.2.16 SET LIMITS command

The SET LIMITS command (see table 140) defines the range within which subsequent linked commands may operate. A second SET LIMITS command may not be linked to a chain of commands in which a SET LIMITS command has already been issued.

Table 140 - SET LIMITS command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (33h)							
1	Logical unit number			Reserved			RdInh	WrInh
2	(MSB)							
3				Logical block address				
4								
5								(LSB)
6	Reserved							
7	(MSB)							
8				Number of blocks				(LSB)
9	Control							

The read inhibit (RdInh) bit of one indicates that read operations within the range shall be inhibited.

A write inhibit (WrInh) bit of one indicates that write operations within the range shall be inhibited.

The logical block address field specifies the starting address for the range.

The number of blocks field specifies the number of logical blocks within the range. A number of blocks of zero indicates that the range shall extend to the last logical block on the logical unit.

Any attempt to access outside of the restricted range or any attempt to perform an inhibited operation within the restricted range shall not be performed. The command shall be terminated with CHECK CONDITION status and the sense key shall be set to DATA PROTECT. A second SET LIMITS command within a linked list of commands shall be rejected with CHECK CONDITION status and the sense key shall be set to DATA PROTECT.

9.2.17 START STOP UNIT command

The START STOP UNIT command (see table 141) requests that the target enable or disable the logical unit for media access operations.

Table 141 - START STOP UNIT command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (1Bh)							
1	Logical unit number			Reserved				Immed
2	Reserved							
3	Reserved							
4	Reserved						LoEj	Start
5	Control							

An immediate (Immed) bit of one indicates that status shall be returned as soon as the command descriptor block has been validated. An Immed bit of zero indicates that status shall be returned after the operation is completed.

A load eject (LoEj) bit of zero requests that no action be taken regarding loading or ejecting the medium. A LoEj bit of one requests that the medium shall be unloaded if the start bit is zero. A LoEj bit of one requests that the medium is to be loaded if the start bit is one.

A start bit of one requests the logical unit be made ready for use. A start bit of zero requests that the logical unit be stopped (media cannot be accessed by the initiator).

Targets that contain cache memory shall implicitly perform a SYNCHRONIZE CACHE command for the entire medium prior to executing the STOP UNIT command.

9.2.18 SYNCHRONIZE CACHE command

The SYNCHRONIZE CACHE command (see table 142) ensures that logical blocks in the cache memory, within the specified range, have their most recent data value recorded on the physical medium. If a more recent data value for a logical block within the specified range exists in the cache memory than on the physical medium, then the logical block from the cache memory shall be written to the physical medium. Logical blocks are not necessarily removed from the cache memory as a result of the synchronize cache operation.

Table 142 - SYNCHRONIZE CACHE command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (35h)							
1	Logical unit number			Reserved			Immed	RelAdr
2	(MSB)							
3				Logical block address				
4								
5								(LSB)
6	Reserved							
7	(MSB)							
8				Number of blocks				(LSB)
9	Control							

An immediate (Immed) bit of one indicates that the target shall return status as soon as the command descriptor block has been validated. An Immed bit of zero indicates that the status shall not be returned until the operation has been completed. If the Immed bit is one and the target does not support it then the command shall terminate with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN CDB.

See 9.2.2 for a definition of the RelAdr bit and the logical block address field.

The number of blocks field specifies the total number of contiguous logical blocks within the range. A number of blocks of zero indicates that all remaining logical blocks on the logical unit shall be within the range.

A logical block within the specified range that is not in cache memory is not considered an error.

9.2.19 VERIFY command

The VERIFY command (see table 143) requests that the target verify the data written on the medium.

Table 143 - VERIFY command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (2Fh)							
1	Logical unit number			DPO	Reserved	Reserved	BytChk	RelAdr
2	(MSB)							
3				Logical block address				
4								
5								(LSB)
6	Reserved							
7	(MSB)							
8				Verification length				(LSB)
9	Control							

If the MODE SELECT command is implemented, and the verify error recovery parameters page is also implemented, then the current settings in that page specifies the verification error criteria. If the verify error recovery parameters page is not implemented, then the verification criteria is vendor-specific.

A byte check (BytChk) bit of zero causes a medium verification to be performed with no data comparison. A BytChk bit of one causes a byte-by-byte compare of data written on the medium and the data transferred from the initiator. If the compare is unsuccessful for any reason, the target shall return CHECK CONDITION status with the sense key set to MISCOMPARE.

See 9.2.2 for a definition of the RelAdr bit and the logical block address field.

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

See 9.2.6 for a description of the cache control bit (DPO).

9.2.20 WRITE(6) command

The WRITE(6) command (see table 144) requests that the target write the data transferred by the initiator to the medium.

Table 144 - WRITE(6) command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (0Ah)							
1	Logical unit number			(MSB)				
2	Logical block address							
3								(LSB)
4	Transfer length							
5	Control							

The cache control bits (see 9.2.6) are not provided for this command. Targets with cache memory may have values for the cache control bits which may affect the WRITE(6) command, however no default value is defined by this standard. If explicit control is required, the WRITE(10) command should be used.

The logical block address field specifies the logical block at which the write operation shall begin.

The transfer length field specifies the number of contiguous logical blocks of data to transferred. A transfer length of zero indicates that 256 logical blocks shall be transferred. Any other value indicates the number of logical blocks that shall be transferred.

9.2.21 WRITE(10) command

The WRITE(10) command (see table 145) requests that the target write the data transferred by the initiator to the medium.

Table 145 - WRITE(10) command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (2Ah)							
1	Logical unit number			DPO	FUA	Reserved	Reserved	RelAdr
2	(MSB)							
3	Logical block address							
4								
5								(LSB)
6	Reserved							
7	(MSB)							
8	Transfer length							(LSB)
9	Control							

See READ(10) command (9.2.6) for a definition of the cache control bits (DPO and FUA).

See 9.2.2 for a definition of the RelAdr bit and the logical block address field.

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.

9.2.22 WRITE AND VERIFY command

The WRITE AND VERIFY command (see table 146) requests that the target write the data transferred from the initiator to the medium and then verify that the data is correctly written. The data is only transferred once from the initiator to the target.

Table 146 - WRITE AND VERIFY command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (2Eh)							
1	Logical unit number			DPO	Reserved	Reserved	BytChk	RelAdr
2	(MSB)							
3				Logical block address				
4								
5								(LSB)
6	Reserved							
7	(MSB)							
8				Transfer length				(LSB)
9	Control							

If the MODE SELECT command is implemented, and the verify error recovery page is also implemented (see 9.3.3.8), then the current settings in that page along with the AWRE bit from the read-write error recovery page specify the verification error criteria. If these pages are not implemented, then the verification criteria is vendor-specific.

A byte check (BytChk) bit of zero requests a medium verification to be performed with no data comparison. A BytChk bit of one requests a byte-by-byte compare of data written on the medium and the data transferred from the initiator. If the compare is unsuccessful for any reason, the target shall return CHECK CONDITION status with the sense key set to MISCOMPARE.

See 9.2.2 for a definition of the RelAdr bit and the logical block address field.

See WRITE(10) command (see 9.2.21) for a definition of the transfer length field.

See 9.2.6 for a description of the cache control bit (DPO).

NOTE 121 The WRITE AND VERIFY command specifically states that the data are not to be transferred twice (i.e once for the write pass, and once for the verify pass) when performing a byte compare. If there is a need for two transfers to occur (e.g. to ensure the integrity of the path to the media), then the initiator should issue a WRITE command with a LINK bit of one followed by a VERIFY command with a BytCmp bit of one, transferring the same data on each command.

9.2.23 WRITE LONG command

The WRITE LONG command (see table 147) requests that the target write the data transferred by the initiator to the medium. The data passed during the WRITE LONG command is implementation specific, but shall include the data bytes and the ECC bytes.

Table 147 - WRITE LONG command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (3Fh)							
1	Logical unit number			Reserved				RelAdr
2	(MSB)							
3				Logical block address				
4								
5								(LSB)
6	Reserved							
7	(MSB)							
8				Byte transfer length				(LSB)
9	Control							

NOTE 122 Any other bytes that can be corrected by ECC should be included (e.g. a data synchronization mark within the area covered by ECC). The READ LONG command is usually issued before issuing a WRITE LONG command. The WRITE LONG data should be the same length and in the same order as the data returned by the READ LONG command.

See 9.2.2 for a definition of the RelAdr bit and the logical block address field.

The byte transfer length field should specify the number of bytes of data that the target would return for the READ LONG command. If a non-zero byte transfer length does not exactly match the data length the target would return for the READ LONG command, then the target shall terminate the command with CHECK CONDITION status and a sense key of ILLEGAL REQUEST and an additional sense code of INVALID FIELD IN CDB. The ILI and valid bits shall be set to one and the information field shall be set to the difference (residue) of the requested length minus the actual length in bytes. Negative values shall be indicated by two's complement notation. A transfer length of zero indicates that no bytes shall be transferred and shall not be considered an error.

9.2.24 WRITE SAME command

The WRITE SAME command (see table 148) requests that the target write the single block of data transferred by the initiator to the medium multiple times.

Table 148 - WRITE SAME command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (41h)							
1	Logical unit number			Reserved			PBdata	LBdata	RelAdr
2	(MSB)							
3				Logical block address				
4								
5								(LSB)
6	Reserved							
7	(MSB)							
8				Number of blocks				(LSB)
9	Control							

NOTE 123 This command is useful if large areas of the medium need to be written, prepared for certification, or otherwise initialized without the initiator having to transfer all the data.

A logical block data (LBdata) bit of one requests that the target replace the first four bytes of the data to be written to the current logical block with the logical block address of the block currently being written.

A physical block data (PBdata) bit of one requests that the target replace the first eight bytes of the data to be written to the current physical sector with the physical address of the sector currently being written using the physical sector format (see 9.2.1.1).

If PBdata and LBdata are one the command shall be terminated with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST.

See 9.2.2 for a definition of the RelAdr bit and the logical block address field.

The number of blocks field specifies the number of contiguous logical blocks to be written. A number of blocks field of zero requests that all the remaining logical blocks on the medium be written.

9.3 Parameters for direct-access devices

9.3.1 Diagnostic parameters

This subclause defines the descriptors and pages for diagnostic parameters used with direct-access devices.

The diagnostic page codes for direct-access devices are defined in table 149.

Table 149 - Diagnostic page codes

Page code	Description	Subclause
00h	Supported diagnostic pages	8.3.1.1
40h	Translate address page	9.3.1.1
01h - 3Fh	Reserved (for all device type pages)	
41h - 7Fh	Reserved	
80h - FFh	Vendor-specific pages	

9.3.1.1 Translate address page - SEND DIAGNOSTIC

The translate address page allows the initiator to translate a logical block address, physical sector address or physica bytes from index address into any one of the other formats. The address to be translated is passed to the target with the SEND DIAGNOSTIC command and the results are returned to the initiator by the RECEIVE DIAGNOSTIC RESULTS command. The format of the translate address page - SEND DIAGNOSTIC is shown in table 150. The translated address is returned in the translate address page - RECEIVE DIAGNOSTIC RESULTS (see 9.3.1.2).

Table 150 - Translate address page - SEND DIAGNOSTIC

Bit / Byte	7	6	5	4	3	2	1	0
0	Page code (40h)							
1	Reserved							
2	(MSB)							
3	Page length (000Ah)							(LSB)
4	Reserved					Supplied format		
5	Reserved					Translate format		
6	Address to translate							
...								
13								

The supplied format field specifies the format of address to translate field. Valid values for this field are defined in the FORMAT UNIT command (see 9.2.1). If the target does not support the requested format it shall terminate the SEND DIAGNOSTIC command with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and an additional sense code shall be set to INVALID FIELD IN PARAMETER LIST.

The translate format field specifies which format the initiator would like the address to be translated to. Valid values for this field are defined in the FORMAT UNIT command (see 9.2.1). If the target does not support the requested format it shall terminate the command with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and an additional sense code shall be set to INVALID FIELD IN PARAMETER LIST.

The address to translate field contains a single address the initiator is requesting the target to translate. The format of this field depends on the value in the supplied format field. The formats are described in 9.2.1.1. If the logical block format is specified the block address shall be in the first four bytes of the field with the remaining bytes set to zero.

9.3.1.2 Translate address page - RECEIVE DIAGNOSTIC

The translate address page allows the initiator to translate a logical block address, physical sector address, or physical bytes from index address into any one of the other formats. The address to be translated is passed to the target with the SEND DIAGNOSTIC command and the results are returned to the initiator by the RECEIVE DIAGNOSTIC RESULTS command. The translated address is returned in the translate address page - RECEIVE DIAGNOSTIC (see table 151).

Table 151 - Translate address page - RECEIVE DIAGNOSTIC

Bit / Byte	7	6	5	4	3	2	1	0
0	Page code (40h)							
1	Reserved							
2	(MSB) Page length							
3	Page length (LSB)							
4	Reserved					Supplied format		
5	RAREA	ALTSEC	ALTTRK	Reserved		Translated format		
	Translated addresses							
6 — 13	Translated address 1							
	. . .							
n — n+7	Translated address x (if required)							

The translate address page contains a four byte page header which specifies the page code and length followed by two bytes which describe the translated address followed by zero or more translated address(s).

The page length field contains the number of parameter bytes which follow.

The supplied format field contains the value from the SEND DIAGNOSTIC command supplied format field (see 9.3.1.1).

A reserved area (RAREA) bit of one indicates that all or part of the translated address falls within a reserved area of the medium (e.g. speed tolerance gap, alternate sector, vendor reserved area, etc.). If the entire translated address falls within a reserved area the target may not return a translated address. An RAREA bit of zero indicates that no part of the translated address falls within a reserved area of the medium.

An alternate sector (ALTSEC) bit of one indicates that the translated address is physically located in an alternate sector of the medium. If the target cannot determine if all or part of the translated address is located in an alternate sector it shall set this bit to zero. An ALTSEC bit of zero indicates that no part of the translated address is located in an alternate sector of the medium or that the target is unable to determine this information.

An alternate track (ALTTRK) bit of one indicates that part or all of the translated address is located on an alternate track of the medium or the target cannot determine if all or part of the translated address is located on an alternate track. An ALTTRK bit of zero indicates that no part of the translated address is located on an alternate track of the medium.

The translated format field contains the value from the SEND DIAGNOSTIC command translate format field (see 9.3.1.1).

The translated address field contains the address(s) the target translated from the address supplied by the initiator in the SEND DIAGNOSTIC command. This field shall be in the format specified in the translate format field. The different formats are described in 9.2.1.1. If the logical block format is specified the block address shall be in the first four bytes of the field and the remaining bytes shall be set to zero.

If the returned data is in the logical block or physical sector format and the address to be translated covers more than one address after it has been translated (e.g. accounting for speed tolerance or multiple physical sectors within a single logical block or multiple logical blocks within a single physical sector) the target shall return all possible addresses which are contained in the area specified by the address to be translated.

If the returned data is in bytes from index format the target shall return a pair of translated values for each of the possible addresses which are contained in the area specified by the address to translate field. Of the pair of translated values returned, the first indicates the starting location and the second the ending location of the area.

9.3.2 Log parameters

This subclause defines the descriptors and pages for log parameters used with direct-access devices.

The log page codes for direct-access devices are defined in table 152.

Table 152 - Log page codes

Page code	Description	Subclause
01h	Buffer over-run/under-run page	8.3.2.1
03h	Error counter page (read) page	8.3.2.2
05h	Error counter page (verify) page	8.3.2.2
02h	Error counter page (write) page	8.3.2.2
07h	Last n error events page	8.3.2.3
06h	Non-medium error page	8.3.2.4
00h	Supported log pages	8.3.2.5
04h	Reserved	
08h - 2Fh	Reserved	
3Fh	Reserved	
30h - 3Eh	Vendor-specific pages	

9.3.3 Mode parameters

This subclause defines the descriptors and pages for mode parameters used with direct-access devices.

The mode parameter list, including the mode parameter header and mode block descriptor are described in 8.3.3.

The medium-type code field is contained in the mode parameter header (see 8.3.3). Table 153 defines this field for direct-access devices.

Table 153 - Direct-access medium-type codes

Code value	Medium type				
00h	Default medium type (currently mounted medium type)				
01h	Flexible disk, single-sided; unspecified medium				
02h	Flexible disk, double-sided; unspecified medium				
	Flexible disks				
	Diameter mm (in)	Bit density Bits/radian	Track density /mm (/in)	Number of sides	ANSI reference standard
05h	200 (8,0)	6 631	1,9 (48)	1	ANSI X3.73-1980
06h	200 (8,0)	6 631	1,9 (48)	2	(Note 1)
09h	200 (8,0)	13 262	1,9 (48)	1	None
0Ah	200 (8,0)	13 262	1,9 (48)	2	ANSI X3.121-1984
0Dh	130 (5,25)	3 979	1,9 (48)	1	ANSI X3.82-1980
12h	130 (5,25)	7 958	1,9 (48)	2	ANSI X3.125-1985
16h	130 (5,25)	7 958	3,8 (96)	2	ANSI X3.126-1986
1Ah	130 (5,25)	13 262	3,8 (96)	2	ISO DIS8630-1985
1Eh	90 (3,5)	7 958	5,3 (135)	2	ANSI X3.137-1988
	Direct-access magnetic tapes				
	Width mm (in)	Tracks	Density ftpmm (ftpi)	ANSI reference standard	
40h	6,3 (0,25)	12	394 (10 000)	Note 1	
44h	6,3 (0,25)	24	394 (10 000)	Note 1	
Code values 80h - FFh are vendor-specific. All remaining code values are reserved.					
NOTE 1 See annex c for additional standards information.					

The device specific parameter field (see table 154) is contained in the mode parameter header (see 8.3.3).

Table 154 - Device specific parameter

Bit	7	6	5	4	3	2	1	0
	WP	Reserved		DPOFUA	Reserved			

When used with the MODE SELECT command the write protect (WP) bit is not defined.

When used with the MODE SENSE command a WP bit of zero indicates that the medium is write enabled. A WP bit of one indicates that the medium is write protected.

When used with the MODE SELECT command, the DPOFUA bit is not used and the field is reserved.

When used with the MODE SENSE command, a DPOFUA bit of one indicates that the target supports the DPO and FUA bits (see 9.2.6). When used with the MODE SENSE command, a DPOFUA bit of zero indicates that the target does not support the DPO and FUA bits.

The density code field is contained in the mode parameter block descriptor (see 7.3.3). This field is reserved for direct-access devices.

The mode page codes for direct-access devices are shown in table 155.

Table 155 - Mode page codes

Page code	Description	Subclause
08h	Caching page	9.3.3.1
0Ah	Control mode page	8.3.3.1
02h	Disconnect-reconnect page	8.3.3.2
05h	Flexible disk page	9.3.3.2
03h	Format device page	9.3.3.3
0Bh	Medium types supported page	9.3.3.4
0Ch	Notch and partition page	9.3.3.5
09h	Peripheral device page	8.3.3.3
01h	Read-write error recovery page	9.3.3.6
04h	Rigid disk geometry page	9.3.3.7
07h	Verify error recovery page	9.3.3.8
06h	Reserved	
0Dh - 1Fh	Reserved	
00h	Vendor-specific (does not require page format)	
20h - 3Eh	Vendor-specific (page format required)	
3Fh	Return all pages (valid only for the MODE SENSE command)	

For direct-access devices, if the notch page is not supported, or if the active notch field in the notch page is zero then each page descriptor specifies mode parameters for the target to use for subsequent operations on the specified logical unit. If the notch page is supported, and the active notch is not zero, then each page descriptor specifies parameters for the target to use for subsequent operations on the disk notch specified by the current value of the active notch field on the specified logical unit.

9.3.3.1 Caching page

The caching parameters page (see table 156) defines the parameters that affect the use of the cache.

Table 156 - Caching page

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	Reserved	Page code (08h)					
1	Page length (0Ah)							
2	Reserved					WCE	MF	RCD
3	Demand read retention priority				Write retention priority			
4	(MSB)							
5	Disable pre-fetch transfer length							(LSB)
6	(MSB)							
7	Minimum pre-fetch							(LSB)
8	(MSB)							
9	Maximum pre-fetch							(LSB)
10	(MSB)							
11	Maximum pre-fetch ceiling							(LSB)

The parameters savable (PS) bit is only used with the MODE SENSE command. This bit is reserved with the MODE SELECT command. A PS bit of one indicates that the target is capable of saving the page in a non-volatile vendor-

specific location. If the PS is set to one in MODE SENSE data then the page shall be savable by issuing a MODE SELECT command with SP set to one.

A write cache enable (WCE) bit of zero specifies that the target shall return GOOD status for a WRITE command after successfully writing all of the data to the medium. A WCE bit of one specifies that the target may return GOOD status for a WRITE command after successfully receiving the data and prior to having successfully written it to the medium.

A multiplication factor (MF) bit of zero specifies that the target shall interpret the minimum and maximum pre-fetch fields in terms of the number of logical blocks for each of the respective types of pre-fetch. An MF bit of one specifies that the target shall interpret the minimum and maximum pre-fetch fields to be specified in terms of a scaler number which, when multiplied by the number of logical blocks to be transferred for the current command, yields the number of logical blocks for each of the respective types of pre-fetch.

A read cache disable (RCD) bit of zero specifies that the target may return data requested by a READ command by accessing either the cache or media. A RCD bit of one specifies that the target shall transfer all of the data requested by a READ command from the medium (i.e. data cannot be transferred from the cache).

The demand read retention priority field (see table 157) advises the target on the retention priority to assign data read into the cache that has also been transferred from the target to the initiator.

Table 157 - Demand read and write retention priority

Value	Description
0h	Indicates the target should not distinguish between retaining the indicated data and data placed into the cache memory by other means (e.g. pre-fetch).
1h	Demand read retention priority: Data put into the cache via a READ command should be replaced sooner (has lower priority) than data placed into the cache by other means (e.g. pre-fetch). Write retention priority: Data put into the cache during a WRITE or WRITE AND VERIFY command should be replaced sooner (has lower priority) than data placed into the cache by other means (e.g. pre-fetch).
Fh	Demand read retention priority: Data put into the cache via a READ command should not be replaced if there is other data in the cache that was placed into the cache by other means (e.g. pre-fetch) and it may be replaced (i.e. it is not locked). Write retention priority: Data put into the cache during a WRITE or WRITE AND VERIFY command should not be replaced if there is other data in the cache that was placed into the cache by other means (e.g. pre-fetch) and it may be replaced (i.e. it is not locked).
2h - Eh	Reserved

The write retention priority field (see table 157) advises the target on the retention priority to assign data written into the cache that has also been transferred from the cache memory to the medium.

An anticipatory pre-fetch occurs when data is placed in the cache that has not been requested. This usually happens in conjunction with the reading of data that has been requested. All the following parameters give an indication to the target how it should manage the cache based on the last READ command. An anticipatory pre-fetch may occur based on other information. All the remaining caching parameters are only recommendations to the target and should not cause a CHECK CONDITION to occur if the target cannot satisfy the request.

The disable pre-fetch transfer length field specifies the selective disabling of anticipatory pre-fetch on long transfer lengths. The value in this field is compared to the number of blocks requested by the current READ command. If the number of blocks is greater than the disable pre-fetch transfer length, then an anticipatory pre-fetch is not done for the command. Otherwise the target should attempt an anticipatory pre-fetch. If the pre-fetch disable transfer length is set to zero, then all anticipatory pre-fetching is disabled for any request for data, including those for zero logical blocks.

The minimum pre-fetch field indicates either a number of blocks or a scaler multiplier of the transfer length, depending upon the setting of the MF bit. In either case, the resulting number of blocks is the number to pre-fetch regardless of the delays it might cause in executing subsequent commands.

The pre-fetching operation begins at the logical block immediately after the last logical block of the previous READ command. Pre-fetching shall always halt before the end of the media. Errors that occur during the pre-fetching operation shall not be reported to the initiator unless that target cannot, as a result of the error, execute subsequent commands correctly. In this case the error may be reported either immediately as an error for the current READ command, or as a deferred error, at the discretion of the target and according to the rules for reporting deferred errors.

If pre-fetch has read more than the amount of data indicated by the minimum pre-fetch then pre-fetching should be terminated whenever another command is ready to execute. This consideration is ignored when the minimum pre-fetch is equal to the maximum pre-fetch.

The maximum pre-fetch field indicates either a number of blocks or a scaler multiplier of the transfer length depending upon the setting of the MF bit. In either case, the resulting number of blocks is the number to pre-fetch if there are no delays in executing subsequent commands.

The maximum pre-fetch field contains the maximum amount of data to pre-fetch into the cache as a result of one READ command. It is used in conjunction with the disable pre-fetch transfer length and maximum pre-fetch ceiling parameters to trade off pre-fetching new data with displacing old data already stored in the cache.

The maximum pre-fetch ceiling field specifies an upper limit on the number of logical blocks computed as the maximum pre-fetch. If this number of blocks is greater than the maximum pre-fetch, then the number of logical blocks to pre-fetch shall be truncated to the value stored in the maximum pre-fetch ceiling field.

NOTE 124 If the MF bit is one the maximum pre-fetch ceiling field is useful in limiting the amount of data to be pre-fetched.

9.3.3.2 Flexible disk page

The flexible disk page (see table 158) contains parameters for control and reporting of flexible disk drive parameters.

Table 158 - Flexible disk page

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	Reserved	Page code (05h)					
1	Page length in bytes (1Eh)							
2	(MSB) Transfer rate							
3								(LSB)
4	Number of heads							
5	Sectors per track							
6	(MSB) Data bytes per sector							
7								(LSB)
8	(MSB) Number of cylinders							
9								(LSB)
10	(MSB) Starting cylinder-write precompensation							
11								(LSB)
12	(MSB) Starting cylinder-reduced write current							
13								(LSB)
14	(MSB) Drive step rate							
15								(LSB)
16	Drive step pulse width							
17	(MSB) Head settle delay							
18								(LSB)
19	Motor on delay							
20	Motor off delay							
21	TRDY	SSN	MO	Reserved				
22	Reserved						SPC	
23	Write compensation							
24	Head load delay							
25	Head unload delay							
26	Pin 34						Pin 2	
27	Pin 4						Pin 1	
28	Medium rotation rate							
29								(LSB)
30	Reserved							
31	Reserved							

The parameters savable (PS) bit is only used with the MODE SENSE command. This bit is reserved with the MODE SELECT command. A PS bit of one indicates that the target is capable of saving the page in a non-volatile vendor-specific location.

NOTE 125 This page is mainly intended for defining parameters of flexible disk drives, but may be used for other devices, if applicable.

The transfer rate indicates the data rate of the peripheral device. See table 159 for examples of common transfer rates.

Table 159 - Examples of transfer rates

Value	Transfer rate
00FAh	250 kbit/s transfer rate
012Ch	300 kbit/s transfer rate
01F4h	500 kbit/s tranfer rate
03E8h	1 mbit/s transfer rate
07D0h	2 mbit/s transfer rate
1388h	5 mbit/s transfer rate

The number of heads field specifies the number of heads used for reading and writing data on the medium. Heads used exclusively for servo information are excluded.

The sectors per track field specifies the number of sectors per revolution per head.

The data bytes per sector field specifies the number of bytes of data per sector that an initiator can read or write.

The number of cylinders field specifies the number of cylinders used for data storage.

The starting cylinder for write precompensation field specifies the cylinder at which write precompensation is to begin. Cylinders are numbered starting with zero. If the starting cylinder for write precompensation is equal to the value in the number of cylinders field, write precompensation shall be disabled by the target.

The starting cylinder for reduced write current field specifies cylinder at which write current is reduced. Cylinders are numbered starting with zero. If the starting cylinder for reduced write current is equal to the value in the number of cylinders field, reduced write current shall be disabled by the target.

The drive step rate field specifies the step rate in units of 100 μs. This value may be rounded as defined in 7.5.4. A value of zero requests the target to set its default value.

The drive step pulse width field specifies the width of the step pulse in microseconds. This value may be rounded as defined in 7.5.4. A value of zero requests the target to set its default value.

The head settle delay field specifies the head settle time in units of 100 μs. This value may be rounded as defined in 7.5.4. A value of zero requests the target to set its default value.

If a true ready signal is not available, the motor on delay field specifies in tenths of a second the time that the target shall wait before attempting to access the medium after the motor on signal is asserted. If a true ready signal is available, the motor on delay field specifies in tenths of a second the time that the target shall wait for drive ready status before aborting an attempt to access the medium. This value may be rounded as defined in 7.5.4.

The motor off delay field specifies in tenths of a second the time that the target shall wait before releasing the motor on signal after an idle condition exists. A value of FFh indicates that the motor on signal shall not be released. The START STOP UNIT command is not affected by this parameter. This value may be rounded as defined in 7.5.4.

A true ready (TRDY) bit of one specifies that a signal is provided that indicates the medium is ready to be accessed.

A start sector number (SSN) bit of one specifies that sectors are numbered starting with one. An SSN bit of zero specifies that sectors are numbered starting with zero.

A motor on (MO) bit of one specifies that pin 16 (motor on) shall remain released. This bit shall be set to one when using high capacity (192 tracks per inch) drives and their pre-formatted diskettes. An MO bit of zero indicates that pin 16 (motor on) shall be asserted.

The step pulse per cylinder (SPC) field is used to specify the number of additional step pulses required per cylinder. Non-zero values allow a drive to read a diskette formatted on a drive with a lower number of tracks per inch. For example, a value of one allows a 96 track-per-inch drive to access tracks on a diskette that was formatted for 48 tracks per inch.

The write compensation field is used to specify the amount of write compensation to be used starting at the cylinder specified in the starting cylinder for write precompensation field. The correlation of any values used in this field to actual write precompensation time values is vendor-specific. If a zero is specified in this field the target shall use its default write precompensation value. This value may be rounded as defined in 7.5.4.

The head load delay field specifies the head loading time in milliseconds. This value may be rounded as defined in 7.5.4. A value of zero requests the target to set its default value.

The head unload delay field specifies the head unloading time in milliseconds. This value may be rounded as defined in 7.5.4. A value of zero requests the target to set its default value.

The Pin 34 field defines the usage of pin 34 of the flexible disk drive interface. This use of this pin varies among vendors and drives. The settings allow the initiator to select how pin 34 shall be used by the interface. See table 160.

Table 160 - Pin 34 field

Bit 7 6 5 4	Description of pin 34 use
P 0 0 0	Open
P 0 0 1	Ready
P 0 1 0	Disk changed

NOTES
1 P is a polarity bit, where 0 is active low and 1 is active high.
2 All undefined values are reserved.

The pin 4 field defines the usage of pin 4 of the flexible disk drive interface. This use of this pin varies among drive vendors and drives. The settings allow the initiator to specify how pin 4 shall be used by the interface. See table 161.

Table 161 - Pin 4 field

Bit 7 6 5 4	Description of pin 4 use
P 0 0 0	Open
P 0 0 1	In use
P 0 1 0	Eject
P 0 0 0	Head load

NOTES
1 P is a polarity bit, where 0 is active low and 1 is active high.
2 All undefined values are reserved.

The pin 1 field defines the usage of pin 1 of the flexible disk drive interface. This use of this pin varies among vendors and drives. The settings allow the initiator to specify how pin 1 shall be used by the interface. See table 162.

Table 162 - Pin 34 field

Bit 7 6 5 4	Description of pin 34 use
P 0 0 0	Open
P 0 0 1	Disk change reset

NOTES
1 P is a polarity bit, where 0 is active low and 1 is active high.
2 All undefined values are reserved.

The medium rotation rate field specifies the speed at which the medium rotates. The unit of measure is rotations pe minute (e.g. 2 400 rpm). This field cannot be changed by a MODE SELECT command.

9.3.3.3 Format device page

The format device page (see table 163) contains parameters which specify the medium format.

Table 163 - Format device page

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	Reserved	Page code (03h)					
1	Page length (16h)							
2	(MSB)							
3	Tracks per zone							(LSB)
4	(MSB)							
5	Alternate sectors per zone							(LSB)
6	(MSB)							
7	Alternate tracks per zone							(LSB)
8	(MSB)							
9	Alternate tracks per logical unit							(LSB)
10	(MSB)							
11	Sectors per track							(LSB)
12	(MSB)							
13	Data bytes per physical sector							(LSB)
14	(MSB)							
15	Interleave							(LSB)
16	(MSB)							
17	Track skew factor							(LSB)
18	(MSB)							
19	Cylinder skew factor							(LSB)
20	SSEC	HSEC	RMB	SURF	Reserved			
21 - 23	Reserved							

The parameters savable (PS) bit is only used with the MODE SENSE command. This bit is reserved with the MODE SELECT command. A PS bit of one indicates that the target is capable of saving the page in a non-volatile vendor-specific location.

NOTE 126 If the initiator changes any of the current physical parameters defined below, the target may not be able to access the media until a FORMAT UNIT command has been successfully completed.

If the defect handling format parameters (tracks per zone, alternate sectors per zone, alternate tracks per zone and alternate tracks per logical unit) requested by the initiator are not supported by the target the target may round these fields to acceptable values as described in 7.5.4.

The tracks per zone field specifies the number of tracks per zone to use in dividing the capacity of the device for the purpose of allocating alternate sectors. A value of zero means that one zone is defined for the entire device. The last zone on the device might not contain the same number of tracks as the previous zone(s).

The alternate sectors per zone field specifies the number of sectors per zone the target shall reserve for defec handling. The target shall de-allocate these sectors from the initiator addressable blocks during the FORMAT UNI command. If the notch page is implemented and the ND bit of the notch page is one and the active notch field c the notch page is zero, then a value of zero indicates that no alternate sectors shall be reserved. Otherwise, a valu of zero indicates that the number of alternate sectors is target specific.

The alternate tracks per zone field specifies the number of tracks per zone the target shall reserve for defect handlinç The target shall de-allocate these tracks from the initiator addressable blocks during the FORMAT UNIT commanc If the notch page is implemented and the ND bit of the notch page is one and the active notch field of the notch pag is zero, then a value of zero indicates that no alternate tracks shall be reserved. Otherwise, a value of zero indicate that the number of alternate tracks is target specific.

The alternate tracks per logical unit field specifies the number of tracks per logical unit the target shall reserve fc defect handling. The target shall de-allocate these tracks from the initiator addressable blocks during the FORMA UNIT command. If the notch page is implemented and the ND bit of the notch page is one and the active notch fiel of the notch page is zero, then a value of zero indicates that no alternate tracks shall be reserved. Otherwise, a valu of zero indicates that the number of alternate tracks is target specific.

The sectors per track field specifies the number of physical sectors included within each track. This numbe includes any alternate sectors the target may allocate. A value of zero during MODE SELECT indicates that the targe shall define the number of sectors per track. For devices with a variable number of sectors per track, the value ir MODE SELECT shall be zero and the value reported in MODE SENSE for the number of sectors per track is vendor specific.

The data bytes per physical sector field specifies the number of data bytes per physical sector that the target sha use. This value may be different than the logical block size reported in the MODE SELECT data. The target sha return CHECK CONDITION status if it determines that the combination of this field and the sectors per track fielc exceed the capability of the medium. A value of zero indicates that the data bytes per physical sector is defined by the target.

For MODE SENSE the interleave field returns the same parameter passed in the FORMAT UNIT command, The targe shall report this field as target defined in the corresponding MODE SENSE command. For MODE SELECT this fielc is ignored.

NOTE 127 It is recommended that this field be marked non-changeable and that initiators send the value returned in MODE SENSE. This allows migration to specifying interleave as a mode parameter instead of in the FORMAT UNIT command.

The track skew factor field specifies the number of physical sectors between the last logical block of one track anc the first logical block on the next sequential track of the same cylinder.

The cylinder skew factor field specifies the number of physical sectors between the last logical block of one cylinde and the first logical block on the next sequential cylinder.

The SSEC bit set to one indicates that the target shall use soft sector formatting.

The HSEC bit set to one indicates that the target shall use hard sector formatting. The HSEC bit and the SSEC bi are mutually exclusive in MODE SELECT commands.

The combinations sector formatting supported that are reported in response to a request for default values are defined in table 164.

Table 164 - Reporting of default sector formatting support

SSEC	HSEC	Description
0	0	Target shall not return this combination
1	0	Target supports soft sector formatting only
0	1	Target supports hard sector formatting only
1	1	Target supports both soft and hard sector formatting

The combinations sector formatting supported that are reported in response to a request for changeable values are defined in table 165.

Table 165 - Reporting of changeable sector formatting support

SSEC	HSEC	Description
0	0	Sector formatting not changeable
1	0	Target shall not return this combination
0	1	Target shall not return this combination
1	1	Target supports both soft and hard sector formatting

The removable medium (RMB) bit set to one indicates that the logical unit supports removable media. A RMB bit set to zero indicates that the logical unit does not support removable media. The status of this bit shall be reflected in the INQUIRY command RMB bit.

The surface (SURF) bit set to zero indicates that the target shall allocate progressive addresses to all logical blocks within a cylinder prior to allocating addresses on the next cylinder. A SURF bit set to one indicates that the target shall allocate progressive addresses to all logical blocks on a surface prior to allocating sector addresses on the next surface.

NOTE 128 If the target supports savable parameters, all savable parameters for this initiator, including those in page codes 3, 4, and 5, are saved to non-volatile memory when the save parameters bit (SP) in the command descriptor block is set to one. The savable parameters may also be saved to non-volatile memory during a FORMAT UNIT command (see 9.2.1).

9.3.3.4 Medium types supported page

The medium types supported page (see table 166) contains a list of the medium types implemented by the target for logical units.

Table 166 - Medium types supported page

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	Reserved	Page code (0Bh)					
1	Page length (06h)							
2	Reserved							
3	Reserved							
4	Medium type one supported							
5	Medium type two supported							
6	Medium type three supported							
7	Medium type four supported							

The parameters savable (PS) bit is only used with the MODE SENSE command. This bit is reserved with the MODE SELECT command. A PS bit of one indicates that the target is capable of saving the page in a non-volatile vendor-specific location.

The code values for each medium type supported by the target (up to four maximum), are reported in ascending order. If only the default medium type is supported zero is reported. If less than four medium types are supported the unused entries shall be returned as zero.

9.3.3.5 Notch and partition page

The notch page (see table 167) contains parameters for direct-access devices which implement a variable number of blocks per cylinder and support this page. Each clause of the logical unit with a different number of blocks per cylinder is referred to as a notch.

Table 167 - Notch page

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	Reserved	Page code (0Ch)					
1	Page length (16h)							
2	ND	LPN	Reserved					
3	Reserved							
4	(MSB)	Maximum number of notches						
5								(LSB)
6	(MSB)	Active notch						
7								(LSB)
8	(MSB)	Starting boundary						
11								(LSB)
12	(MSB)	Ending boundary						
15								(LSB)
16	(MSB)	Pages notched						
23								(LSB)

The parameters savable (PS) bit is only used with the MODE SENSE command. This bit is reserved with the MODE SELECT command. A PS bit of one indicates that the target is capable of saving the page in a non-volatile vendor-specific location.

A notched drive (ND) bit of zero indicates that the device is not notched and that all other parameters in this page shall be returned as zero by the target. A ND bit of one indicates that the device is notched. For each supported active notch value this page defines the starting and ending boundaries of the notch.

A logical or physical notch (LPN) bit of zero indicates that the boundaries are based on the physical parameters of the logical unit. The cylinder is considered most significant, the head least significant. A LPN bit of one indicates that the notch boundaries are based on logical blocks of the logical unit.

The maximum number of notches field indicates the maximum number of notches supported by the logical unit. This field shall be reported as unchangeable.