SASI
Shugart Associates System Interface

Design Specifications

Shugart

Rev. F (18 October 82)

SASI

SHUGART ASSOCIATES SYSTEM INTERFACE

Abstract: The Shugart Associates System Interface is capable of interconnecting small computers with each other and with low to medium performance intelligent peripherals such as rigid disks, flexible disks, magnetic tape devices, and printers.

SASI REVISION LIST

Revision Letter	Description	Date
-	First presentation to ANSI committee X3T9-3 (2 weeks following announcement in Electronic Design)	9-15-81
A	• Revised to reflect changes to meet NCR requirements	1-5-82
	• Further revised to add description of commands	1-12-82
B	Edited and improved for working meeting with NCR, Adaptec and Optimum in Sunnyvale	1-25-82
C	• Rewritten in preparation for presentation to ANSI	1-29-82
	• Addenda and Errata for Rev. C	2-3-82
D	Revised to make all known corrections in preparation for "working session" in Sunnyvale on March 3 and 4	2-19-82
E	Revised to incorporate changes resulting from "working session" in Sunnyvale in preparation for April 26 and 27 meeting in Phoenix.	4-1-82
F	Revised to incorporate changes transacted in ANSI meetingsup to this date.	10-18-82

FOREWORD

The major goal of this interface is to facilitate the integration of small computers and intelligent peripheral devices into computer systems. Since the computers and peripherals envisioned are physically small and the market for such systems is quite cost sensitive, the interface will be suitable for implementation in LSI circuitry and use inexpensive drivers, cables, and connectors, but need only operate over moderate distances (15 m) and data rates (1.5 Mbytes per second). The command set which will be a part of this standard will be a device independent set for hosts and intelligent devices which largely masks the internal structure of the device (cylinders, tracks, sectors, and the like) from the interface.

CONTENTS

Section		Page
1.	Scope	4
2.	Definitions	4
3.	Introduction	5
4.	Physical Path Specification	6
4.1	SASI BUS	6
4.2	SASI Physical Path Philosophy	9
4.3	BUS Signals	11
4.4	BUS Phases	13
4.5	BUS Conditions	19
4.6	Phase Sequencing	20
4.7	Signal Assertions	23
4.8	Timing	25
4.9	Physical Description	26
4.10	Electrical Description	33
5.	Message System Specification	36
5.1	Message Protocol	36
5.2	Messages	37
5.3	Extended Messages	41
6.	Command and Status Definition	42
6.1	SASI Functional Control Philosophy	43
6.2	Command Descriptor Block (CDB)	44
6.3	Commands - Direct Access Devices	46
6.4	Commands - Sequential Access Devices	85
6.5	Commands - Processor Devices	118
6.6	Completion Status	126
6.7	Completion Status Byte	126
6.8	Sense Bytes	127
6.9	Extended Sense Bytes	128

1. Scope

This Shugart Associates System Interface defines a local I/O Bus specification which can be operated at data rates up to an estimated 1.5 megabytes per second, depending upon circuit implementation choices. The primary objective of the interface is to provide host computers with device independence, within a class of devices. Thus, disk drives, tape drives, printers and even communication devices, of different types, can be added to the host computer without requiring modifications to generic system hardware or software. Provision is made for the ready addition of non-generic features and functions.

The interface uses "logical" rather than "physical" addressing for all data structures. All data is addressed as logical blocks up to the maximum number of blocks in a device; and, each device can be interrogated to determine how many blocks it contains.

Provision is made for cable lengths up to 15 meters using differential drivers and receivers. An in-cabinet mounting using cable lengths up to six meters and single ended drivers and receivers is also available.

The interface protocol includes provision for the connection of multiple initiators (**SASI** bus devices capable of initiating an operation) and multiple targets (**SASI** bus devices capable of responding to a request to perform an operation). Arbitration (i.e., bus-contention logic) is built into the architecture of **SASI**. A logical priority system awards interface control to a **SASI** bus device that wins arbitration.

2. Definitions

BUS DEVICE	A host computer or peripheral controller which can be attached to the **SASI** bus.
BYTE	8 bits (Octet).
CDB	Command Descriptor Block
CONNECT	This function occurs when an **INITIATOR** selects a **TARGET**, to start an operation.
CONTROLLER	An **SASI BUS DEVICE** which controls one or more Peripheral Devices.
DISCONNECT	This function occurs when a **TARGET** drops off of the **SASI** bus, after a connect.

ENDING STATUS	One byte of information sent from a **TARGET** to an **INITIATOR** upon completion of one command or a set of linked commands.
INITIATOR	A Bus Device which initiates an operation on the **SASI** bus.
INTERMEDIATE STATUS	One byte of information sent from a **TARGET** to an **INITIATOR** upon completion of each command in a set of linked commands except the last command in the set.
LUN	Logical Unit Number.
ms	Millisecond.
ns	Nanosecond.
PASSIVE SIGNAL STATE	When a signal is not driven by a **SASI BUS DEVICE** but is biased by the cable terminators to the non-asserted state.
PERIPHERAL DEVICE	An **I/O** device such as a disk, printer or magnetic tape unit.
RECONNECT	This function occurs when a **TARGET** selects an **INITIATOR** to continue an operation after a disconnect.
SIGNAL ASSERTION	Driving a signal to the asserted state.
SIGNAL DE-ASSERTION	When a signal is either driven by a **BUS DEVICE** to the non-asserted state or biased by the cable terminators to the non-asserted state.
SIGNAL RELEASE	See **PASSIVE SIGNAL STATE.**
TARGET	A Bus Device on the receiving end of an operation on the **SASI** Bus.
us	Microsecond.

3. Introduction

The main body of this specification is given in Sections 4., 5. and 6.

Section 4. This section contains definitions of physical components of interface. It contains all of the specifications associated with obtaining and maintaining control of the **SASI** bus.

Section 5. This section contains definitions of the message protocol which links the **SASI** bus devices to each other and controls the physical path between them.

Section 6. This section contains definition of the operation of the interface in detail, which in turn defines the operation of the bus devices (controllers or hosts).

"Standard" level **SASI BUS** devices shall implement all Standard (S) commands defined in section 6 for the applicable type of device, and may implement various Extended (E), Optional (O), or Vendor unique (V) commands as well. "Extended" level **SASI** devices shall implement all Standard commands for the applicable device type plus all Extended commands defined in section 6, and may implement various Optional or Vendor unique commands as well. The Extended level devices include commands necessary to write "device independent" software drivers which can "discover" all necessary device attributes without some prior knowledge of the specific device characteristics. The Extended level device also includes commands with a very large address space (2^{32} blocks), adequate for all present and anticipated future devices, while the Standard command set provides for a somewhat restricted address space (2^{21}), which is adequate for present devices but may be insufficient for very large future devices (e.g., optical disks). Standard commands may be used with Extended class **SASI** devices; however, if the Generic device implements an address space greater than 2^{21} blocks, the portion of the address space above block number $2^{21}-1$ cannot be accessed via the Standard commands.

4. Physical Path Specification

This is the physical path definition of the Shugart Associates System Interface (**SASI**), which is designed to provide an efficient method of communication between computers and peripheral input/output devices.

SASI is implemented using an eight-port, daisy chained bus which includes the following key features:

- Single or multiple host computer system.
- Bus contention handled via distributed-arbitration on a prioritized basis.
- Accommodation of multiple peripheral device types.
- Asynchronous communication of up to 1.5 MBytes/sec, at 15 meters cable length.
- Multiple overlap of peripheral device operations.
- Direct copy between peripheral devices.
- Oriented toward intelligent peripheral devices.

4.1 SASI BUS

Communication on the **SASI** Bus is allowed between only two **SASI BUS PORTS** at any given time. There is a maximum of eight (8) **BUS PORTS.** Each port is attached to a **SASI DEVICE** (e.g., peripheral device controller or host computer).

When two **SASI BUS DEVICES** communicate on the bus, one acts as an **INITIATOR** and the other acts as a **TARGET.** The **INITIATOR** (typically a host computer) starts an operation and the **TARGET** (typically a peripheral device controller) performs the operation. A **SASI BUS DEVICE** will usually have a fixed role as an **INITIATOR** or **TARGET,** but some may be able to assume either role.

An **INITIATOR** may address up to eight (8) peripheral **I/O** devices that are connected to a **TARGET.** Three sample system configurations are shown in Figure 1.

SIMPLE SYSTEM

BASIC TWO CONTROL UNIT SYSTEM

COMPLEX SYSTEM

Up to 8 **SASI DEVICES** can be supported by the **SASI** bus. They can be any combination of host CPUs and intelligent controllers.

Figure 1
Sample SASI Configurations

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

Data transfers on the bus are asynchronous and follow a defined **REQUEST/ACKNOWLEDGE handshake** protocol. One eight-bit byte of information may be transferred with each handshake.

4.2 SASI Physical Path Philosophy

Consider the system shown in Figure 2 where an **INITIATOR** and **TARGET** communicate on the **SASI** bus in order to execute some command (e.g., **READ** or **WRITE** data). Since it would be cumbersome to briefly describe these functions at the detailed level of the bus, a higher level of description is used here. Only one of a number of possible partitions of the physical/functional interface is presented for illustration.

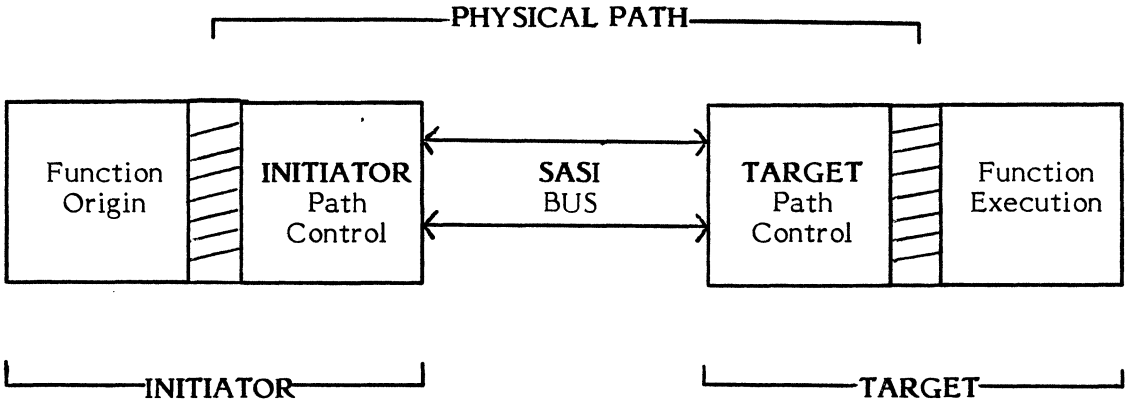


Figure 2
Physical Path

4.2.1 Pointers

In the **SASI** architecture there are three "conceptual" memory address pointers which point to the next byte of command, data or status to be accessed. The pointers are used to represent the state of the interface. The pointers reside in the **INITIATOR** path control.

After the pointers are initially loaded by the **INITIATOR**, their movement is under strict control of the **TARGET**. When the **TARGET** transfers a byte of information to or from the three areas, the corresponding pointer increments.

4.2.2 Typical Functions (External to Path Control)

Listed below are some typical functions that affect the physical path but come from outside the physical path boundary.

- **Establish Path** — This function allows the **INITIATOR** to set up the physical path (the physical and logical connection between the **INITIATOR** and a particular peripheral device for the purpose of executing some peripheral device command). This may involve arbitration to gain control of the **SASI** bus. The "Establish Path" function requires the peripheral device address (i.e., **SASI TARGET** bus address and **LUN** within that address). Also required are the three pointers to the **COMMAND, DATA** and **STATUS** areas. A saved copy of these pointers may also be required (see "Reestablish Path" and "Restore Pointers" functions).

- **Get Command** — This function allows a **TARGET** to fetch a **COMMAND** from the area pointed to by the **COMMAND** pointer.

- **Get Data and Send Data** — These functions allow the **TARGET** to transfer data to or from the area pointed to by the **DATA** pointer.

- **Send Status** — This allows the **TARGET** to send **STATUS** information for a command to the area pointed to by the **STATUS** pointer.

- **End of Command** — This function is sent by the **TARGET** to indicate that the current command terminated and that valid status has been sent. (Note that the current command may be linked to another command.)

- **End Path** — The **TARGET** uses this function to completely clear the physical path with regards to the currently attached peripheral device. A new path can thus be established for this device.

- **Break Path** — This function is done by the **TARGET** to temporarily break the physical path so that others may use the **SASI** bus.

- **Reestablish Path** — The **TARGET** uses this function to establish a physical path connection that was temporarily broken by the "Break Path" function. An indication of the **INITIATOR**'s **SASI** bus address and the peripheral device address are required. **SASI** bus arbitration is also required. The **INITIATOR** must restore the **COMMAND, DATA** and **STATUS** pointers.

- **End of Link** — This function is sent by the **TARGET** to indicate that the current command has terminated but that it was linked to another command. The initiator is then allowed to set up its **COMMAND, DATA** and **STATUS** pointers to the initial state of the next command.

- **Save Data Pointer** — The **TARGET** uses this function to direct the **INITIATOR** to save a copy of the current (active) **DATA** pointer for the currently connected peripheral device.

- **Restore Pointers** — The **TARGET** uses this function to direct the **INITIATOR** to load the current (active) **COMMAND, DATA** and **STATUS** pointers with the last saved values for the currently connected peripheral device.

4.3 BUS Signals

There shall be nine (9) control signals and nine (9) data signals (including parity), as listed below:

- BSY
- SEL
- C/D
- I/O
- MSG
- REQ
- ACK
- ATN
- RST
- DB (7-0, P) (Data Bus)

These signals are described below:

- **BSY (BUSY)** — An "or-tied" signal which generally indicates when the bus is being used.

- **SEL (SELECT)** — An "or-tied" signal used by an **INITIATOR** to select a **TARGET** or by a **TARGET** to reselect an **INITIATOR.**

- **C/D (CONTROL/DATA)** — A signal driven by a **TARGET;** it generally indicates whether **CONTROL** or **DATA** information is on the data bus. Assertion indicates **CONTROL.**

- **I/O (INPUT/OUTPUT)** — A signal driven by a **TARGET** which controls the direction of data movement on the data bus with respect to an **INITIATOR.** Assertion indicates **INPUT** to the **INITIATOR.**

- **MSG (MESSAGE)** — A signal driven by a **TARGET** during the **MESSAGE** phase.

- **REQ (REQUEST)** — A signal driven by a **TARGET** to indicate a request for a **REQ/ACK** data transfer handshake.

- **ACK (ACKNOWLEDGE)** — A signal driven by an **INITIATOR** to indicate an acknowledgement for a **REQ/ACK** data transfer handshake.

- **ATN (ATTENTION)** — A signal driven by an **INITIATOR** to indicate the **ATTENTION** condition.

- **RST (RESET)** — An "or-tied" signal which indicates the **RESET** condition.

- **DB(7-0,P) (DATA BUS)** — Eight data bit signals, plus a parity bit signal which form a **DATA BUS. DB(7)** is the most significant bit and has the highest priority during **ARBITRATION.** Bit number, significance and priority decrease downward to **DB(0).**

  Data parity **DB(P)** is odd. The use of parity is a system option (i.e., a system is configured so that all **SASI DEVICES** on a bus generate parity and have parity detection enabled, or all **SASI DEVICES** have parity detection disabled or not implemented). Parity is not guaranteed valid during the **ARBITRATION** phase.

  Each of the eight data signals **DB(7)** through **DB(0)** is uniquely assigned as a **TARGET** or **INITIATOR**'s own **SASI BUS** address (i.e., **SASI DEVICE ID**). This **SASI DEVICE ID** would normally be assigned and strapped in a **SASI DEVICE** during system configuration.

  During **ARBITRATION,** a **SASI DEVICE** that desires the use of the **SASI BUS** asserts its assigned data bit (**SASI DEVICE ID**) but leaves the other data bits in the passive (non-driven) state. Thus, a **SASI DEVICE** may use one data bit driver for its **SASI DEVICE ID** during **ARBITRATION** and a different driver when driving the complete data bus in other phases of the bus operation.

4.4 BUS Phases

The bus shall have eight (8) distinct operational phases:

- **BUS FREE** Phase
- **ARBITRATION** Phase
- **SELECTION** Phase
- **RESELECTION** Phase
- **COMMAND** Phase
- **DATA** Phase
- **STATUS** Phase
- **MESSAGE** Phase

These phases are collectively termed the **INFORMATION TRANSFER** phases.

The bus can never be in more than one phase at any given time. Unless otherwise noted, the following descriptions assume that bus signals which are not mentioned will not be asserted.

4.4.1 Bus Free Phase

The **BUS FREE** phase is used to indicate that no **SASI DEVICE** is actively using the bus and that the bus is available for subsequent users.

The **BUS FREE** phase is created by the passive release of all bus signals.

SASI DEVICES shall detect the **BUS FREE** phase by the simultaneous (within a **DESKEW DELAY**) condition of both **SEL** and **BSY** not asserted while the **RESET** condition is not active.

During the **BUS FREE** phase, all active **SASI DEVICES** shall immediately release all bus signals (within a **BUS CLEAR DELAY**) after the **BSY** and **SEL** signals are released from the bus.

4.4.2 Arbitration Phase (Optional)

The **ARBITRATION** phase allows one **SASI DEVICE** to gain control of the bus so that this device can assume the role of an **INITIATOR** or **TARGET.**

Note: Implementation of the **ARBITRATION** phase is a system option. Systems with no **ARBITRATION** phase can have only one **INITIATOR.** The **ARBITRATION** phase is required for systems which use the **RESELECTION** phase.

After a **SASI DEVICE** that wants to arbitrate for the bus detects the **BUS FREE** phase it shall wait a minimum of **BUS FREE DELAY** and a maximum of **BUS SET DELAY** in order to assert **BSY** and its own **SASI DEVICE ID** on the bus. (The time required for the device to detect the **BUS FREE** phase shall be included in the measurement of the amount of time that the device waits.)

Note: The **SASI DEVICE ID** shall be asserted on the **DATA BIT** signal that corresponds to the device's unique **BUS ADDRESS.** All other seven **DATA BIT** drivers in this **SASI DEVICE** shall be passive. Data parity is not guaranteed valid during **ARBITRATION.** See 4.7 for restrictions on driving the parity signal.

Any **SASI DEVICE** that is arbitrating shall immediately clear itself from arbitration (within a **BUS CLEAR DELAY** time) by releasing its **BSY** and **ID** signals if **SEL** is asserted by any other **SASI DEVICE** that has won arbitration.

After an **ARBITRATION DELAY** (measured from the assertion of BSY) the **SASI DEVICE** that is arbitrating shall examine the **DATA** bus. If a higher priority **SASI DEVICE ID** is on the bus (**DB(7)** = highest) then the **SASI DEVICE** shall clear itself from **ARBITRATION** by releasing its **BSY** and **ID** signals. If the **SASI DEVICE** determines that its own **ID** is the highest asserted, then it wins **ARBITRATION** and asserts **SEL;** (after the assertion of **SEL** the **SASI DEVICE** shall wait a minimum of two **BUS SETTLE DELAYS** before changing the assertion of any bus signals).

4.4.3 Selection Phase

The **SELECTION** phase allows an **INITIATOR** to select a **TARGET** for the purpose of initiating some **TARGET** function(s), (e.g. read or write data).

Note: All during the **SELECTION** phase the **I/O** signal shall not be asserted so that this phase can be distinguished from the **RESELECTION** phase.

In systems where the **ARBITRATION** phase is not implemented, the **INITIATOR** shall first detect the **BUS FREE** phase and then wait a minimum of **BUS SETTLE DELAY.** Then the **INITIATOR** shall assert the **DATA BUS** with both the desired **TARGET's ID** and its own **INITIATOR ID** (see option note at 4.4.3.1). After two **DESKEW DELAYS** the **INITIATOR** shall assert **SEL.**

In systems with **ARBITRATION** implemented, the **BSY** and **SEL** signals shall be asserted by an **INITIATOR** when going from the **ARBITRATION** phase to the **SELECTION** phase. Also, the **INITIATOR** shall wait a minimum of two **BUS SETTLE DELAYS.** The **INITIATOR** shall then assert the **DATA BUS** with both the desired **TARGET's ID** and its own **INITIATOR ID** (see option note 4.4.3.1). After these assertions, the **INITIATOR** shall wait at least two **DESKEW DELAYS** and then release **BSY.** The **INITIATOR** shall then wait a **BUS SETTLE DELAY** before looking for a response from the **TARGET.**

In all systems, the selected **TARGET** shall detect the simultaneous (within a **DESKEW DELAY**) condition of **SEL** and its own **SASI DEVICE ID** asserted, and both **BSY** and **I/O** not asserted. The selected **TARGET** may sample the **DATA BUS** in order to try to determine the **SASI DEVICE ID** of the **INITIATOR** that is doing the selecting (see option note 4.4.3.1). The selected **TARGET** shall then respond by asserting **BSY** within a **SELECTION RESPONSE TIME** of its actual detection of being selected; this is required for the timeout procedure to work. (Note that in systems with parity implemented, the **TARGET** shall not respond to its **SASI DEVICE ID** if bad parity is detected on the bus. Note also that if more than two **SASI DEVICE IDs** are on the bus, the **TARGET** shall consider this an illegal condition and shall not respond to the selection.)

At least two **DESKEW DELAYS** after the **INITIATOR** detects **BSY** sent from the **TARGET,** it releases **SEL** and may change the **DATA** signals.

4.4.3.1 Option: Initiators that do not implement the **RESELECTION** phase are allowed the option of asserting only one **BUS DEVICE ID** (the **TARGET's ID**) during **SELECTION.**

4.4.3.2 Two optional selection timeout procedures are specified for clearing the **SASI BUS** if the **INITIATOR** waits a minimum of a **SELECTION TIMEOUT PERIOD** and there has been no **BSY** response from the **TARGET:**

(1) The **INITIATOR** may cause the reset condition.

(2) The **INITIATOR** shall leave **SEL** asserted and shall deassert all **BUS ID** signals. If the **INITIATOR** has not detected a **BSY** response from the **TARGET** after trying for at least a **SELECTION RESPONSE TIME** plus two **DESKEW DELAYS,** the **INITIATOR** shall release **SEL** and let the bus go to the **BUS FREE PHASE.**

4.4.4 Reselection Phase (Optional)

RESELECTION is an optional phase which allows a **TARGET** to reconnect to an **INITIATOR** for the purpose of continuing some operation that was previously started by the **INITIATOR** but was interrupted by the **TARGET;** (i.e., the TARGET disconnected by allowing a **BUS FREE** phase to occur before the operation was complete).

4.4.4.1 **RESELECTION** can only be used in systems that have **ARBITRATION** implemented.

After the **TARGET** has gone through the **ARBITRATION** phase, it will be asserting **BSY** and **SEL** and will have waited a minimum of two **BUS SETTLE DELAYS.** The **TARGET** shall then assert the **I/O** signal and also assert the **DATA BUS** with the desired **INITIATOR's ID** and its own **TARGET ID.** After these assertions the **TARGET** shall wait at least two **DESKEW DELAYS** and then release **BSY.** The **TARGET** shall then wait a **BUS SETTLE DELAY** before looking for a response from the **INITIATOR.**

The reselected **INITIATOR** detects the simultaneous (within a **DESKEW DELAY**) condition of **SEL, I/O** and its own **SASI DEVICE ID** asserted, and **BSY** not asserted. The reselected **INITIATOR** may sample the **DATA BUS** to determine the **SASI DEVICE ID** of the **TARGET** that is doing the **RESELECTION.** The reselected **INITIATOR** shall then respond by asserting **BSY** within a **RESELECTION RESPONSE TIME** of its actual detection of be reselected; this is required for the timeout procedure to work. (Note that in systems with parity implemented the **INITIATOR** shall not respond to a **DEVICE ID** that has bad parity. Note also that if more than two **SASI DEVICE IDs** are on the bus, the **INITIATOR** shall consider this an illegal condition and shall not respond to the **RESELECTION.**)

After the **TARGET** detects **BSY** sent from the **INITIATOR,** the **TARGET** shall: (1) also assert **BSY** and continue the assertion until it is done using the bus; and (2) wait at least two **DESKEW DELAYS** and then release **SEL** and possibly change the **I/O** and **DATA** signals.

The reselected **INITIATOR** detects the release of **SEL** and shall release its assertion of **BSY.**

4.4.4.2 Two optional **RESELECTION** timeout procedures are specified for clearing the **SASI BUS** if the **TARGET** waits a minimum of a **SELECTION TIMEOUT PERIOD** and there has been no **BSY** response from the **INITIATOR:**

1. The **TARGET** may cause the **RESET** condition.

2. The **TARGET** shall leave **SELECT** and **I/O** asserted and shall deassert all **BUSY DEVICE ID** signals. If the **TARGET** has not detected a **BSY** response from the **INITIATOR** after trying for at least a **SELECTION RESPONSE TIME** plus two **DESKEW DELAYS,** the **TARGET** shall release **SEL** and **I/O** and let the bus go to the **BUS FREE** phase.

4.4.5 Information Transfer Phases (COMMAND, DATA, STATUS and MESSAGE Phases)

Common Notes: The **COMMAND, DATA, STATUS** and **MESSAGE** phases can all be grouped together as the **INFORMATION TRANSFER** phases because they are all used to transfer data or control information via the **DATA BUS.** The actual contents 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 Table 1.

Table 1
Information Transfer Phases

SIGNAL			PHASE NAME	DIRECTION OF INFORMATION XFER	COMMENT
MSG	C/D	I/O			
0	0	0	DATA OUT PHASE	(INIT to TARG)	DATA PHASES
0	0	1	DATA IN PHASE	(INIT from TARG)	
0	1	0	COMMAND PHASE	(INIT to TARG)	
0	1	1	STATUS PHASE	(INIT from TARG)	
1	0	0	*		
1	0	1	*		
1	1	0	MSG OUT PHASE	(INIT to TARG)	MESSAGE PHASES
1	1	1	MSG IN PHASE	(INIT from TARG)	

Notes: 0 = **SIGNAL DEASSERTION**, 1 = **SIGNAL ASSERTION.**
INIT = **INITIATOR**, **TARG** = **TARGET.**
* = Not used.

The **INFORMATION TRANSFER** phases use one or more **REQ/ACK** handshakes to control the data transfer. Each **REQ/ACK** allows the transfer of one byte of data. The **REQ/ACK** handshake shall start with the **TARGET** asserting the **REQ** signal. The **INITIATOR** shall respond by asserting the **ACK** signal. The **TARGET** shall then deassert the **REQ** signal. The **INITIATOR** shall again respond by deasserting the **ACK** signal.

If the **I/O** signal is asserted, data will be **INPUT** into the **INITIATOR** from the **TARGET.** The **TARGET** shall guarantee that valid data is available on the bus at the **INITIATOR**'s port at least a **DESKEW DELAY** before the assertion of **REQ** is at the **INITIATOR**'s port. The data shall remain valid until the assertion of **ACK** by the **INITIATOR.** It shall be the **TARGET**'s responsibility to compensate for the maximum **CABLE SKEW** and the skew of its own drivers.

If the **I/O** signal is **NOT** asserted, data will be **OUTPUT** from the **INITIATOR** into the **TARGET.** The **INITIATOR** shall guarantee that valid data is available on the bus at the **TARGET**'s port at least a **DESKEW DELAY** before the assertion of **ACK** is at the **TARGET**'s port. Valid data shall remain on the bus until the **TARGET** deasserts **REQ.** It shall be the **INITIATOR**'s responsibility to compensate for the maximum **CABLE SKEW** and the skew of its own drivers.

During each **INFORMATION TRANSFER** phase the **BSY** line shall remain asserted and the **SEL** line shall remain released. Additionally, during each **INFORMATION TRANSFER** phase 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 **REQ** of the first handshake and remain valid until the deassertion of **ACK** at the end of the last handshake.

4.4.6 Command Phase

The **COMMAND** phase allows the **TARGET** to request command information from the **INITIATOR.**

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

4.4.7 Data Phase

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

4.4.7.1 Data In Phase

The **DATA IN** phase allows the **TARGET** to request that data be **INPUT** to the **INITIATOR** from the **TARGET.**

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

4.4.7.2 Data Out Phase

The **DATA OUT** phase allows the **TARGET** to request that data be **OUTPUT** from the **INITIATOR** to the **TARGET.**

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

4.4.8 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 **C/D** and **I/O** and deassert the **MSG** signal during the **REQ/ACK** handshake(s) of this phase.

4.4.9 Message Phase

The **MESSAGE** phase is a term that encompasses the **MESSAGE IN,** and **MESSAGE OUT** phases.

4.4.9.1 Message In Phase

The **MESSAGE IN** phase allows the **TARGET** to request that **MESSAGES** be **INPUT** to the **INITIATOR** from the **TARGET.**

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

4.4.9.2 Message Out Phase

The **MESSAGE OUT** phase allows the **TARGET** to request that a **MESSAGE** be **OUTPUT** from the **INITIATOR** to the **TARGET**. The **TARGET** may invoke this phase at its convenience only in response to the **ATTENTION** condition created by the **INITIATOR**.

In response to the **ATTENTION** condition, the **TARGET** shall assert **C/D** and **MSG** and deassert the **I/O** signal during the **REQ/ACK** handshake(s) of this phase. (See **ATTENTION** condition description 4.5.1.)

4.4.10 Signal Restrictions Between Phases

When the **BUS** is between two phases, the following restrictions shall apply to the bus signals:

- The **BSY, SEL, REQ** and **ACK** signals shall not change.
- The **C/D, I/O, MSG** and **DATA** signals may change.
- The **ATN** and **RST** signals may change as defined under the descriptions for the **ATTENTION** and **RESET** conditions.

4.5 BUS Conditions

The bus has two asynchronous conditions:

- **ATTENTION** Condition.
- **RESET** Condition.

These conditions cause certain **BUS DEVICE** actions and can alter the bus phase sequence.

4.5.1 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 at its convenience 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 **TARGET** may respond with the **MESSAGE OUT** phase.

The **INITIATOR** shall keep **ATN** asserted if more than one byte is to be transferred.

The **INITIATOR** shall deassert the **ATN** signal during: (1) the **RESET** condition, or when the bus goes to a **BUS FREE** phase, or (2) while the **REQ** signal is asserted and the **ACK** signal is not yet asserted during the last **REQ/ACK** handshake of a **MESSAGE OUT** phase.

4.5.2 Reset Condition

The **RESET** condition is used to immediately clear all **SASI DEVICES** from the bus and to reset these devices and their associated equipment (as defined in the **DEVICE** specification).

This condition shall take precedence over all other phases and conditions.

The **RESET** condition is created by the assertion of the **RST** signal.

The **RESET** condition can occur at any time.

The **RESET** condition shall be on for a minimum of a **RESET HOLD TIME.**

Any **SASI DEVICE** (whether active or not) can create the **RESET** condition. The **RESET** condition should be used with caution because of its possible effects.

During the **RESET** condition, no bus signal except **RST** is guaranteed to be in a valid state.

When the **RESET** condition exists, all **SASI DEVICES** shall immediately (within a **BUS CLEAR DELAY**) release all bus signals except **RST** itself.

The specification for the **BUS DEVICE** determines whether or not a specific **I/O** operation is "cleared" by the **RESET** condition. **TARGETs** which are capable of continuing an **I/O** operation interrupted by the **RESET** condition shall clear any **I/O** operation that has not yet been completely identified. Completely identified **I/O** operations are those **I/Os** for which a valid **IDENTIFY** message has been received following the initial selection. Refer to 5.2 for a definition of the **IDENTIFY** message and for a description of additional reset capability.

Regardless of what bus phase may have been interrupted, following the **RESET** condition the bus shall go to a **BUS FREE** phase and then start a normal phase sequence.

4.6 Phase Sequencing

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

In all systems, the **RESET** condition can interrupt any phase and is always followed by the **BUS FREE** phase. Also, any other phase can be followed by the **BUS FREE** phase.

In systems where the **ARBITRATION** phase is not implemented, the allowable sequencing shall be as shown in Figure 3. The normal progression would be from the **BUS FREE** phase to **SELECTION**, and from **SELECTION** to one or more of the **INFORMATION TRANSFER** phases (**COMMAND, DATA, STATUS** or **MESSAGE**).

In systems where the **ARBITRATION** phase is implemented, the allowable sequencing shall be as shown in Figure 4. The normal progression would be 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**).

There are no restrictions on the sequencing between **INFORMATION TRANSFER** phases. A phase may even follow itself (e.g., a **DATA** phase may be followed by another **DATA** phase).

Figure 3
Phase Sequencing
(For systems with no arbitration)

Figure 4
Phase Sequencing
(For systems with arbitration)

4.7 Signal Assertions

Two methods of driving **SASI** Bus signals are specified for use in **SASI BUS DEVICES:** "or-tied" and "non-or-tied." With both methods, signal assertion shall be achieved by having the signal actively driven to the asserted state. For the "non-or-tied" method, signal non-assertion shall be achieved by driving the signal to the non-asserted state. For the "or-tied" method, signal non-assertion shall be achieved by a passive driver condition which allows the cable terminators to bias the signal to the non-asserted state; the signal shall not be driven to the non-asserted state.

Implementation of the **BSY, SEL** and **RST** signals shall always be with the "or-tied" method. Implementation of the other signals may be done with either method.

Table 2 indicates which type of **SASI BUS DEVICE** is allowed to source each signal. No attempt is made to show if this source is driving asserted, driving non-asserted or is passive. All Bus Device drivers that are not active sources shall be in the passive state. Note that the **RST** signal may be sourced by any **SASI BUS DEVICE** at any time.

Table 2

Signal Sources

	Signal				
Bus Phase	**BSY**	**SEL**	**C/D, I/O MSG, REQ**	**ACK/ATN**	**DB(7-0,P)**
Bus Free	None	None	None	None	None
Arbitration	All	One	None	None	ID
Selection	I&T	Init	None	Init	Init
Reselection	I&T	Targ	Targ	Init	Targ
Command	Targ	None	Targ	Init	Init
Data In	Targ	None	Targ	Init	Targ
Data Out	Targ	None	Targ	Init	Init
Status	Targ	None	Targ	Init	Targ
Message In	Targ	None	Targ	Init	Targ
Message Out	Targ	None	Targ	Init	Init

All = The signal shall be driven by all **SASI BUS DEVICES** that are actively arbitrating.

ID = A unique data bit (the **SASI BUS DEVICE ID**) shall be driven by each Bus Device that is actively arbitrating; the other seven data bits shall be released (i.e., not driven) by this Bus Device. The parity bit (DB(P)) may be undriven or driven to the asserted state, but shall never be driven to the non-asserted state during this phase.

I&T = The signal shall be driven by the **INITIATOR** and/or **TARGET** as specified in 4.4.2 and 4.4.3.

Init = If this signal is driven, it shall be driven only by the active **INITIATOR.**

None = The signal shall not be driven by any Bus Device (i.e., all drivers shall be passive).

One = The signal shall be driven by the one Bus Device that wins arbitration.

Targ = If this signal is driven, it shall be driven only by the active **TARGET.**

4.8 Timing

Unless otherwise indicated, the delay time measurements for each **SASI DEVICE** shall be calculated from signal conditions existing at that device's own **SASI BUS PORT.** Thus, normally these measurements need not consider delays in the bus cable.

- **ARBITRATION DELAY** (1.7 us)
  The minimum time a **SASI DEVICE** must 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.

- **BUS CLEAR DELAY** (650 ns)
  The maximum time that a **SASI DEVICE** can take to stop driving all bus signals after:
  - The release of **BSY** when going to the **BUS FREE** phase.
  - Another **SASI DEVICE** asserts **SEL** during the **ARBITRATION** phase.

- **BUS FREE DELAY** (100 ns)
  The minimum time that a **SASI DEVICE** must wait from its detection of the **BUS FREE** phase until its assertion of **BSY** when going to the **ARBITRATION** phase.

- **BUS SET DELAY** (1.1 us)
  The maximum time a **SASI DEVICE** can take from its detection of the **BUS FREE** phase to its assertion of **BSY** and its **ID** for the purpose of **ARBITRATION.**

- **BUS SETTLE DELAY** (450 ns)
  The minimum time to wait for the **BUS** to settle after changing certain control signals.

- **CABLE SKEW** (10 ns)
  The maximum difference in propagation time allowed between any two **SASI BUS** signals when measured between any two **SASI BUS PORTS.**

- **DESKEW DELAY** (45 ns)
  Used to calculate the minimum time required for deskew of certain **BUS** signals.

- **RESET HOLD TIME** (25 us)
  The minimum time for which **RST** is asserted. There is no maximum.

- **SELECTION RESPONSE TIME** (200 us)
  The maximum time a **TARGET** (or **INITIATOR**) shall take from its actual detection of being selected (or re-selected) until asserting a **BSY** response.

- **SELECTION TIMEOUT PERIOD** (250 ms) (recommended)
  The minimum time that an **INITIATOR** (or **TARGET**) shall wait for a **BSY** response during the selection (or re-selection) phase before starting the **TIMEOUT PROCEDURE.**

  Note that this is only a recommended time period. Consult device specifications for the actual timing requirements.

4.9 Physical Description

SASI devices are daisy-chained together using a common cable. Both ends of the cable are terminated. All signals are common between all **SASI** devices. Two driver/receiver options are available:

- Single-ended drivers and receivers, which allow a maximum cable length of six meters (primarily for in-cabinet interconnection).

- Differential drivers and receivers, which allow a maximum cable length of fifteen meters (primarily for interconnection outside of a cabinet).

4.9.1 Cable Requirements (Single-Ended Option)

A fifty (50) conductor flat cable (or twisted pair flat cable) shall be used. The maximum cable length shall be 6.0 meters.

Each **SASI BUS PORT** shall have a 0.1 meter maximum stub length of any conductor when measured from the bus cable connector.

Bus termination may be internal to the **SASI BUS DEVICES** that are at the ends of the bus cable.

The cable pin assignment shall be as shown in Figure 7.

Cable Connector Requirements: The connector shall be a fifty (50) conductor flat cable connector which consists of two rows of 25 female contacts on 100 mil centers. See Figure 5. Note that keying is optional.

Bus Device Connector Requirements: The connector shall be a fifty (50) conductor connector which consists of two rows of 25 male pins on 100 mil centers. See Figure 6. Note that a shroud and header body are optional.

4.9.2 Cable Requirements (Differential Option)

A fifty conductor flat cable or twisted pair cable shall be used. The maximum cable length shall be 15 meters.

Each **BUS PORT** shall have a 0.2 meter maximum stub length of any conductor when measured from the bus cable connector. The maximum stub link difference shall be 0.1 meters.

Bus termination may be internal to the **BUS DEVICES** that are at the ends of the bus cable.

The cable pin assignment shall be as shown in Figure 8.

Connector Requirements: The connector shall be a fifty (50) conductor flat cable connector or equivalent for round cable. The connector shall consist of two rows of 25 pins on 100 mil centers.

Figure 5
Cable Connector

Figure 6
Bus Device Connector

Table 3
Dimensions For Cable Connector

	Millimeters	Inches
C1	2.5400	0.100
C2	60.9600	2.400
C3	2.5400	0.100
C4	8.3570	0.329
C5	3.3025	0.130
C6	68.0720	2.680
C7	6.0960	0.240
C8	8.1530	0.321
C9	13.4870	0.531
C10	3.8100	0.150
C11	1.2700	0.050
C12	6.0960	0.240
C13	32.3850	1.275
C14	3.3020	0.130
C15	7.4930	0.295
C16	2.6670	0.105
C17	1.6250	0.064

Note 1: 50 Contacts on 1.27 mm (0.05 in) staggered spacing = 62.23 mm (2.450 in)
Note 2: Tolerances +/- 0.127 mm (.005 in) noncumulative

Table 4
Dimensions For Bus Device Connector

	Millimeters	Inches
D1	2.54	0.100
D2	82.80	3.260
D3	2.54	0.100
D4	4.83	0.190
D5	8.51	0.335
D6	72.64	2.860
D7	78.74	3.100
D8	13.94	0.549
D9	4.19	0.165
D10	6.09	0.240
D11	6.60	0.260

Note 1: 50 Contacts on 2.54 mm (0.100 in) spacing = 60.96 mm (2.40 in)
Note 2: Tolerances +/- 0.127 mm (.005 in) noncumulative

SIGNAL	PIN NUMBER
-DB(0)	2
-DB(1)	4
-DB(2)	6
-DB(3)	8
-DB(4)	10
-DB(5)	12
-DB(6)	14
-DB(7)	16
-DB(P)	18
GROUND	20
GROUND	22
GROUND	24
*TERMPWR	26
GROUND	28
GROUND	30
-ATN	32
GROUND	34
-BSY	36
-ACK	38
-RST	40
-MSG	42
-SEL	44
-C/D	46
-REQ	48
-I/O	50

*Note: This pin is reserved for providing optional terminator power (plus 5 volts).

Note: All odd pins except pin 25 shall be connected to ground. Pin 25 should be left open but may be connected to ground.

Figure 7
Pin Assignments (Single-Ended Option)

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

*Optional shield ground on some cables.

**These pins are reserved for providing optional terminator power (plus 5 volts).

Figure 8
Pin Assignments (Differential Option)

4.10 Electrical Description

Note: For these measurements, bus termination is assumed to be external to the **SASI DEVICE.** A typical **SASI DEVICE** would have the provision for allowing optional internal termination.

4.10.1 Single-Ended Option

All signals shall be asserted low.

All assigned signals are terminated with 220 ohms to +5 volts (nominal) and 330 ohms to ground at each end of the cable. See Figure 9.

All signals use open collector or three-state drivers as noted in Section 4.7.

Each signal driven by a **SASI DEVICE** shall have the following output characteristics when measured at the device's **SASI BUS PORT** connection:

Signal Assertion = 0.0 VDC to 0.4 VDC
Minimum driver output capability = 48 ma (sinking) @ 0.5 VDC
Signal Non-assertion = 2.5 VDC to 5.25 VDC

Each signal received by a **SASI DEVICE** shall have the following input characteristics when measured at the device's **SASI BUS PORT** connection:

Signal Assertion = 0.0 VDC to 0.8 VDC
Maximum total input load = -0.4 ma @ 0.4 VDC
Signal Non-assertion = 2.0 VDC to 5.25 VDC
Minimum input hysteresis shall be 0.2 VDC

4.10.2 Differential Option

All bus signals consist of two lines denoted **SIGNAL (+)** AND **SIGNAL (-).** A signal is **ASSERTED** when **SIGNAL (+)** is more positive than **SIGNAL (-),** while a signal is **NON-ASSERTED** when **SIGNAL (-)** is more positive than **SIGNAL (+).** All assigned signals shall be terminated at each end of the cable as shown in Figure 10.

Each signal driven by a **SASI** Device shall have the following output characteristics when measured at the device's **SASI BUS PORT** connection:

V_{OL} (Low-level output voltage) = 2.0V Max @ I_{OL} = 55 mA.
V_{OH} (High-level output voltage) = 3.0V Min @ I_{OH} = -55 mA.

These two levels shall be as measured between the output terminal and the **SASI DEVICE**'s logic reference.

V_{OD} (Differential voltage = 1.0V Min with common-mode voltage ranges from -7V to +12V).

The output characteristics shall additionally conform to EIA Standards Proposal Number 1488, Draft 9, dated June 3, 1981.

Each signal received by a **SASI DEVICE** shall have the following input characteristics when measured at the device's **SASI BUS PORT** connection. (Note: These characteristics include both receivers and passive drivers):

I_I (Input current on either input) = $\pm$ 2.0 mA max.

This requirement shall be met with the input voltage varying between -7V and +12V and with power on or off.

Hysteresis = 35 mV Min.

The input characteristics shall additionally conform to EIA Standards Proposal Number 1488, Draft 9, dated June 3, 1981.

Figure 9
Termination for Single-Ended Option

Figure 10
Termination for Differential Option

5. Message System Specification

The message system allows communication between an **INITIATOR** and **TARGET** for the purpose of physical path management.

5.1 Message Protocol

All **SASI** bus devices shall implement the **COMMAND COMPLETE** message. A functional **SASI** bus device can be constructed without using any of the other messages if the **LUN** is specified in the CDB.

SASI DEVICES indicate their ability to accommodate more than the **COMMAND COMPLETE** message by asserting or responding to the **ATN** signal. The **INITIATOR** indicates this by creating the **ATTENTION** condition before it releases **BSY** when going through the **SELECTION** phase. The **TARGET** indicates its ability to accommodate more messages by responding to the **ATTENTION** condition with the **MESSAGE OUT** phase after going through the **SELECTION** phase.

Normally, the first message sent by the **INITIATOR** after the **SELECTION** phase is the **IDENTIFY** message. This allows the establishment of the physical path for a particular **LUN** specified by the **INITIATOR**. After **RESELECTION**, the **TARGET**'s first message is also **IDENTIFY**. This allows the physical path to be re-established for the **TARGET**'s specified **LUN**. Under some exceptional conditions, an **INITIATOR** may send the **ABORT** message or the **BUS DEVICE RESET** message instead of the **IDENTIFY** message, as the first message.

Whenever a physical path is established in an **INITIATOR** that can accommodate disconnection and reconnections, the **INITIATOR** must assure that the present pointers of the physical path are equal to the saved pointers for that particular **LUN**.

Note that **BUS DEVICES** that implement any message other than the **COMMAND COMPLETE** message shall also implement the **REJECT** message.

Table 5
Message Codes

Code	Description
00	COMMAND COMPLETE
01	EXTENDED MESSAGE FOLLOWS
02	SAVE DATA POINTER
03	RESTORE POINTERS
04	DISCONNECT
05	INITIATOR DETECTED ERROR
06	ABORT
07	MESSAGE REJECT
08	NO OPERATION
09	MESSAGE PARITY ERROR
0A	LINKED COMMAND COMPLETE
0B	LINKED COMMAND COMPLETE (WITH FLAG)
0C	BUS DEVICE RESET
0D to 7F	RESERVED
80 to FF	IDENTIFY

5.2 Messages

The single byte messages are listed along with their coded values (in HEX) and their definitions.

COMMAND COMPLETE (00) (Standard)

Sent from a **TARGET** to an **INITIATOR** to indicate that the execution of a command (or series of linked commands) has terminated and that valid status has been sent to the **INITIATOR.**

Note: This command may have been executed successfully or unsuccessfully as indicated in the status.

EXTENDED MESSAGE FOLLOWS (01)
(Optional)

Sent from either the **INITIATOR** or **TARGET** to indicate that a multiple byte message will follow. (See 5.3 for extended messages.)

SAVE DATA POINTER (02)
(Optional)

This code is sent from a **TARGET** to direct the **INITIATOR** to save a copy of the present active data pointer for the currently attached **LUN.** See 4.2.1 for a definition of pointers.

RESTORE POINTERS (03)
(Optional)

This code is sent from a **TARGET** to direct the **INITIATOR** to restore the most recently saved pointers (for the currently attached **LUN)** to the active state. Pointers to the **COMMAND, DATA,** and **STATUS** locations for the **LUN** will be restored to the active pointers. **COMMAND** and **STATUS** pointers will be restored at the beginning of the present operation. The **DATA** pointer will be restored at the beginning of the operation or at the point at which the last **SAVE DATA POINTER** message occurred.

DISCONNECT (04)
(Optional)

Sent from a TARGET to inform an **INITIATOR** that the present physical path is going to be broken (the **TARGET** will disconnect by releasing **BSY),** but that a later reconnect will be required in order to complete the current operation. By not sending this message or the **COMMAND COMPLETE** message before going to **BUS FREE** (other than as a result of reset), the **TARGET** indicates that a error condition has occured on the current **I/O.**

INITIATOR DETECTED ERROR (05)
(Optional)

Sent from an **INITIATOR** to inform a **TARGET** that an **INITIATOR** detected retryable error has occured since the last time the state of the **DATA POINTER** was saved.

Note: Commonly, this is for a data parity error.

ABORT (06)
(Optional)

This message is sent from the **INITIATOR** to direct the **TARGET** to:

1. Clear any **I/O** for the specified **LUN** from the selecting **INITIATOR**

2. Cause the bus to go to the **BUS FREE** phase.

No status or ending message shall be sent for the **I/O**. Only the **I/Os** from a single **INITIATOR** are affected. If no **LUN** has been selected by the **IDENTIFY** message, then all pending **I/O** operations for the selected **TARGET** from that **INITIATOR** will be cleared.

MESSAGE REJECT (07)
(Optional)

This message is sent from either the **INITIATOR** or **TARGET** to indicate that the last message it received was inappropriate or has not been implemented.

Note: In order to indicate its intentions of sending this message, the **INITIATOR** must assert the **ATN** signal prior to its release of **ACK** for the **REQ/ACK** handshake of the message that will be rejected. This provides an interlock so that the **TARGET** can determine which message will be rejected. This message shall be implemented if any other optional messages are implemented.

NO OPERATION (08)
(Optional)

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.

MESSAGE PARITY ERROR (09)
(Optional)

Sent from either the **INITIATOR** or **TARGET** to indicate that the last message it received had a parity error.

Note: In order to indicate its intentions of sending this message, the **INITIATOR** must assert the **ATN** signal prior to its release of **ACK** for the **REQ/ACK** handshake of the message that has the parity error. This provides an interlock so that the **TARGET** can determine which message has the parity error.

LINKED COMMAND COMPLETE (0A) (Optional)

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** is then allowed to set up the pointers for the initial state for the next linked command.

LINKED COMMAND COMPLETE (WITH FLAG) (0B) (Optional)

Sent from a **TARGET** to an **INITIATOR** to indicate that the execution of a linked command (with the **FLAG** set) has completed and that status has been sent. The **INITIATOR** is then allowed to set up the pointers for the initial state of the next linked command. Typically the **FLAG** would cause an interrupt in the **INITIATOR.**

BUS DEVICE RESET (0C) (Optional)

This message can be sent from an **INITIATOR** to direct a **TARGET** to reset all current **I/O** operations on that **BUS DEVICE.** This message forces the **BUS DEVICE** to an initial state with no **I/O** operations pending for any **INITIATOR.**

RESERVED (0D to 7F)

Reserved for future definition.

IDENTIFY (80 to FF) (Optional)

This message can be sent by either the **INITIATOR** or **TARGET.** It is used to establish the physical path connection between an **INITIATOR** and **TARGET** for a particular **LUN.**

Bit 7 — This bit is always set to distinguish this message from the others.

Bit 6 — This bit is only set by the **INITIATOR.** When it is set, it indicates that the **INITIATOR** has the ability to accommodate disconnection and reconnection.

Bits 5 - 3 — Reserved.

Bits 2 - 0 — These bits specify an **LUN** address in a **TARGET.**

5.3 Extended Messages

Messages whose first byte is 01 HEX are multiple byte messages that take the following form:

Table 6
Extended Message Format

BYTE	VALUE	FUNCTION
00	01	Extended Message Follows
01	nn	Extended Message Length
02	yy	Extended Message Code

The following field is repeated nn-1 times.

00 to nn-1	xx	Extended Message Arguments

nn = Number of bytes to follow this byte. A value of zero indicates 256 bytes.

Table 7
Extended Message Code

Code	Description
00	MODIFY DATA POINTER (Optional)
01 to 7F	RESERVED
80 to FF	VENDOR UNIQUE

Table 8
Modify Data Pointer

BYTE	VALUE	FUNCTION
00	01	Extended Message Follows
01	05	Extended Message Length
02	00	Modified Data Pointer Code
03	xx	MSB Argument
04	xx	Argument
05	xx	Argument
06	xx	LSB Argument

This message is sent from the **TARGET** to the **INITIATOR** and requests that the signed argument be added (two's complement) to the present value of the data pointer.

6. Command and Status Definition

This is the functional definition of the Shugart Associates System Interface, software command set, as well as specific status information related to the various commands. The document defines the logical structure of the **I/O** subsystem, the functions available from the **I/O** subsystem, and how to request these functions.

By defining a fixed block structure using a simple logical address scheme, the **I/O** interface can support device independence. In addition, by including the address as a component of the command structure, physical requirements such as **SEEK** can be imbedded within the basic **READ** and **WRITE** requests.

Although the interface has been kept quite simple, it has been designed to provide high performance in a multiple host multi-task environment. Functions have been included to enhance random access applications. Multiple block transfers may be accomplished with a single command.

By keeping to a minimum the functions essential to communicate via this protocol, a wide range of devices of varying capability can operate in the same environment. The objective of low cost may be satisfied without precluding that of high performance.

6.1 SASI Functional Control Philosophy

This subsection contains examples of the process of obtaining commands, transferring data, and storing results. Command execution itself (the actual implementation of a command) is device specific and is therefore not described. Because subsets of the full architecture may be implemented, optional functions are noted.

6.1.1 Single Command Example

A typical operation on the **SASI** interface is likely to include a single **READ** to the I/O subsystem. This operation will be described in detail starting with a request from the system to the **INITIATOR** path control logic. Data that is sent back during the **READ** shall be data that most recently was properly written.

The **INITIATOR** path control logic has an active state and a set of stored states representing active disconnected devices (**INITIATOR** path control logic without disconnect capability does not require stored states). Upon receipt of an "Establish Path" request from the system, the **INITIATOR** path control logic sets up the active state for the operation, arbitrates for the bus, and selects the **LUN**. Once this process is completed, the **TARGET** function control logic (a functional component of the **LUN**) assumes control of the operation.

The **TARGET** obtains the command from the **INITIATOR** (in this case a **READ** command) via the "Get Command" request to the **TARGET** path control logic. The **TARGET** then reads the data from the peripheral device and sends it to the **INITIATOR** using the "Send Data" function of the **TARGET** path control logic. At the completion of the **READ** command, the **TARGET** stores the command's completion status in the **INITIATOR** using the "Send Status" path control function. To end the operation, and to acknowledge completion, the **TARGET** path control logic is given the "End Path" function.

6.1.2 Disconnect Example

In the above **READ** example, the length of time necessary to obtain the data may have been considerable (e.g., requiring a time-consuming physical seek). In order to improve system throughput, the **TARGET** may disconnect from the **INITIATOR**, freeing the bus to allow other requests to be sent to other **LUNs**. To do this, the **INITIATOR** path control logic must be reselectable and capable of restoring the device state upon reconnection. The **TARGET** path control logic must be capable of arbitrating for the data bus and reselecting the **INITIATOR**.

After the **TARGET** has received the **READ** command (and determined that there will be a delay), it will request, via path control logic, that the **INITIATOR** save the present state; this state differs from the state when the **TARGET** was initially selected because the command has already been transferred. The **TARGET** then requests that the **TARGET** path control logic break the path (i.e., disconnect).

When data is ready to be transferred, the **TARGET** will request that path control logic reconnect the **INITIATOR**. As a a result of this reconnection, the **INITIATOR** path control logic will restore the "saved" state (last saved by the

reconnecting **LUN**); the **TARGET** will continue (as in the single command example) to finish the operation. At "Path End," the **INITIATOR** path control logic recognizes that the operation is complete.

On those occasions when a **TARGET** could disconnect without first saving the latest state (as might occur if an error was detected while transferring data to the **INITIATOR**), the operation may be repeated by either restoring the previous state or by disconnecting without saving the present state. When reconnection is completed, the previous state will be restored.

6.1.3 Linked Command Example

The "Link" function defines a relationship between commands which, when combined with the **RELATIVE ADDRESS BIT**, allows previous operations to modify subsequent commands. "Link" makes high performance I/O functions possible by providing a relative addressing capability and allowing multiple command execution without invoking the functional component of the **INITIATOR** and without requiring reselection.

If the desired data address (in the previously described **READ** example) is unknown, but a search key defined as some particular bytes of the field is known, then by linking the **READ** to a **SEARCH EQUAL** command this data can be quickly and effectively transferred to the **INITIATOR.**

One additional function must be completed prior to requesting the next command. This function, "End of Link," is sent from the **TARGET** to the **INITIATOR** to acknowledge command completion. The **INITIATOR** then updates the stored state so that subsequent requests from the **TARGET** will reference the next command of the chain. Other than the "End of Link" function and the address modification of linked commands, command processing of linked and single commands is identical.

For example, the successful completion of a **SEARCH EQUAL** will cause the **TARGET** to fetch the linked **READ** command from the **INITIATOR.** If the **RELATIVE ADDRESS BIT** in the **READ** command has been set, and the address field at the **READ** command is set to zero, the **TARGET** will transfer to the **INITIATOR** the block that was just successfully searched.

6.2 Command Descriptor Block (CDB)

An **I/O** request to a device is performed by passing a Command Descriptor Block (CDB) to the **TARGET.** The first byte of the CDB is the command group and operation code. The remaining bytes specify such things as the Logical Unit Numbers (**LUN**), block starting address, control byte, and the number of blocks to transfer.

Commands are in eight categories as follows:

- Group 0 - 6 byte commands including control, data transfer and status.
- Group 1 - 10 Byte Commands.

- Group 2 - Reserved.
- Group 3 - Reserved.
- Group 4 - Reserved.
- Group 5 - 12 Byte Commands.
- Group 6 - Vendor Unique Commands.
- Group 7 - Vendor Unique Commands.

Note: The reserved or **RESRV** used within this document indicates that it is reserved for future use. Reserved bits that are passed over the interface should be set to zero but not checked for zero by the recipient.

Vendor Unique or V.U. used within this document indicates codes which are device specific.

6.2.1 Group Code

The Group Code (Bits 7-6-5, Byte 00) can be 0 to 7.

6.2.2 Operation Code

The Operation Code (Bits 4-0, Byte 00) for each group allows 32 commands (0 to 31).

6.2.3 Logical Unit Number

The Logical Unit Numbers (Bits 7-6-5, Byte 01) designate the primary unit for all groups. Logical Unit Number allows 8 devices per **TARGET** (0 to 7). This method of device addressing is designed for low-end systems that do not implement separate paths for command and address functions. It is not recommended for more sophisticated implementations, that use the **IDENTIFY MESSAGE.**

6.2.4 Logical Block Address

Group 0 commands contain 21 bit starting block addresses. Extended address commands contain 32 bit starting block addresses.

The concept of block implies that the **INITIATOR** and **TARGET** have previously agreed to the number of bytes of data to be transferred.

6.2.5 Number of Blocks

The number of blocks to transfer for the command allows 1 to 256 blocks, for standard commands (1 to 255, and 0 indicates 256 blocks).

Extended address commands allow 1 to 65,536 blocks. (0 indicates 65,536.)

6.2.6 Control Byte (Last Byte in All Commands)

- Bit 7-6 Vendor Unique.
- Bit 5-2 Reserved.
- Bit 1 This bit is only meaningful when Bit 0 is set and means an interrupt is requested for this command in a group of linked commands.
- Bit 0 The use of this bit is optional and means an automatic link to the next command upon successful completion of the current command for this **INITIATOR**. Status shall be returned for each command executed.

6.2.7 Relative Address Bit

The **RELATIVE ADDRESS BIT** (Bit 0 of Byte 1) of the **EXTENDED** commands is set to indicate that the block address portion of the CDB is a two's complement displacement. This displacement is to be added to the Block Address last accessed on the unit to form the Block Address for this **I/O**. 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 device. For an example of the operation of this function, see the section on linked commands.

6.3 Commands - Direct Access Devices

6.3.1 Command Descriptions (Group 00) for Direct Access Devices

STANDARD (S) Commands shall be implemented in order to meet the minimum requirement of this specification.

EXTENDED (E) Commands shall be implemented in addition to Standard commands to meet the extended requirement of this specification.

OPTIONAL (O) Commands, if used, shall be implemented as defined in this specification.

VENDOR UNIQUE (V) Commands available for Vendor Unique implementations. See Appendix C for a list of current Vendor Unique commands. See the vendor specifications where compatibility is desired.

RESERVED (R) Command codes not currently used are reserved for ANSI definition at a future date.

Command operation codes and names are listed in Table 9.

Table 9
Command Codes
Group 00 Commands

HEX OP Code	Type	Description
00	O	TEST UNIT READY
01	O	REZERO UNIT
02	V	
03	S	REQUEST SENSE
04	S/E	FORMAT UNIT
05	V	
06	V	
07	O	RE-ASSIGN BLOCK
08	S	READ
09	V	
0A	S	WRITE
0B	O	SEEK
0C	V	
0D	V	
0E	V	
0F	V	
10	V	
11	V	
12	E	INQUIRY
13	V	
14	V	
15	O	MODE SELECT
16	O	RESERVE UNIT
17	O	RELEASE UNIT
18	O	COPY
19	V	
1A	O	MODE SENSE
1B	O	START/STOP
1C	O	RECEIVE DIAGNOSTIC RESULTS
1D	O	SEND DIAGNOSTIC
1E	R	
1F	R	

Table 10
Test Unit Ready Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	0	0
01	**Logical Unit Number**			Reserved				
02	Reserved							
03	Reserved							
04	Reserved							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

TEST UNIT READY (00) (Optional).

Returns zero status if addressed unit is powered on and ready. This is not a request for unit self test.

Table 11
Rezero Unit Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	0	1
01	**Logical Unit Number**			Reserved				
02	Reserved							
03	Reserved							
04	Reserved							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

REZERO UNIT (01) (Optional).

Sets the unit to a device specific state.

Table 12
Request Sense Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	1	1
01	**Logical Unit Number**			Reserved				
02	Reserved							
03	Reserved							
04	No. of Bytes Allocated							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

REQUEST SENSE (03) (Standard).

Returns unit sense. The sense data will be valid for the check condition (Status) just presented to the **INITIATOR**. This sense data must be preserved in the **TARGET** for the **INITIATOR**. Sense data will be cleared on the reception of any subsequent command to the unit in check from the **INITIATOR** receiving the check condition. (See Section 6.8 for sense data format.) Byte 04 in this command will specify the number of bytes that the **INITIATOR** has allocated for returned sense. All **INITIATOR**s shall be capable of accepting a minimum of four sense bytes (i.e., a value of 0 through 3 defaults to a four-byte allocation).

Check status shall not be returned for this command.

Table 13
Format Unit Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	0	0	1	0	0
01	**Logical Unit Number**			Fmt Data	Cmp Lst	VU Data	Reserved	
02	Reserved							
03	MSB Interleave							
04	LSB Interleave							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

FORMAT UNIT (04) (Standard/Extended).

The format command insures that the media is formatted so that data blocks can be accessed. There is no guarantee that the media has or has not been altered. In addition, the media may be certified and control structures be created for the management of the media and defects.

The interleave field requests that the logical blocks be related in a specific fashion to the physical blocks to facilitate speed matching. An interleave value of zero (0) requests that the **TARGET** use its default interleave. An interleave value of one (1) requests that consecutive logical blocks be placed in consecutive physical order. Values of two or greater are vendor unique.

The following features of this command are part of the extended command set:

Bit 4 of byte 01, if set, means that format data is supplied with the command. The defect list included with this data specifies block addresses which are to be flawed.

Bit 3 of byte 01 in the CDB is the complete list bit. If this bit is on, the defect list is a complete list of all block addresses with known defects using the requested block size (see Mode Select). The controller may add to this list as it formats the media. The effect of the complete list bit is to purge any previously specified defect list and to build a new defect list. If the complete list bit is off, the defect list adds to the previously specified defect list using the current format. Note that the defect list in this case refers to the current block size (and not to the new block size, if it is different). The controller may add to this list as it formats the media.

Bit 2 of byte 01 set means that the format data is vendor unique. If this bit is off, it indicates that the format data sent with the command is as defined in Table 14.

The following **FORMAT DATA** is sent during the data phase of the command:

Table 14

BYTE	
00	Reserved
01	Reserved
02	Length of Defect List (n Bytes)
03	Length of Defect List (n Bytes)

The following defect block addresses may be repeated n/4 within the defined length.

00	Defect List Block Addresses
01	Defect List Block Addresses
02	Defect List Block Addresses
03	Defect List Block Addresses

The additional defect block addresses shall be added to this list in four byte groups for each defect.

Table 15
Re-Assign Block Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	0	0	1	1	1
01	**Logical Unit Number**				MSB Logical Block Address			
02	Logical Block Address							
03	LSB Logical Block Address							
04	Decision							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

RE-ASSIGN BLOCK (07) (Optional)

This command will modify the **STATUS** for the specified block from an error condition to good **STATUS.** Bit 0 of Byte 04 of the CDB, when 0, directs the **TARGET** to re-assign the block to an alternate block. When it is 1, it directs the **TARGET** to re-state the block as valid. Instead of issuing a format unit command to get rid of this bad block, and erasing all information previously stored on the unit, this command only affects the specified block. Information stored on all other blocks is not changed. During a **READ** or **WRITE** operation, after this command is issued, the **TARGET** may either skip the specified block or re-assign it to another location. The re-assigned block may or may not be a better valid block. The information stored on the bad block may or may not be altered by this command.

Table 16
Read Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	0	1	0	0	0
01	**Logical Unit Number**				MSB Logical Block Address			
02	Logical Block Address							
03	LSB Logical Block Address							
04	Number of Blocks							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

READ (08) (Standard).

Transfers to the **INITIATOR** the specified number of blocks starting at the specified logical starting block address. This command shall return the most recent data value written to the specified block(s).

Table 17
Write Command

BYTE	BIT 7	6	5	4	3	2	1	0
00	0	0	0	0	1	0	1	0
01	**Logical Unit Number**				MSB Logical Block Address			
02	Logical Block Address							
03	LSB Logical Block Address							
04	Number of Blocks							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

WRITE (0A) (Standard).

Transfers to the **TARGET** the specified number of blocks starting at the specified logical starting block address.

Table 18
Seek Command

BYTE	BIT 7	6	5	4	3	2	1	0
00	0	0	0	0	1	0	1	1
01	**Logical Unit Number**				MSB Logical Block Address			
02	Logical Block Address							
03	LSB Logical Block Address							
04	Reserved							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

SEEK (0B) (Optional).

Requests that the unit seek to the specified address.

Table 19
Inquiry Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	1	0	0	1	0
01	**Logical Unit Number**			Reserved				
02	Reserved							
03	Reserved							
04	No. of Bytes Allocated							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

INQUIRY (12) (Extended).

This command requests that information regarding **UNIT** and **TARGET** parameters be sent to the **INITIATOR.**

Byte 04 in the CDB of this command will specify the number of bytes that the host has allocated for returned **UNIT** data.

0 indicates 256 bytes.
1 - 255 as indicated.

The following **INQUIRY DATA** is sent during the data phase of the command:

Table 20

BYTE	
00	Device Type Code (00)
01	Device Type Qualifier
02	Length of Additional Bytes (N)

The following unit parameter bytes may be repeated n times.

00 to N-1	Unit Parameter Byte

The device type code (Byte 00) is as follows:

HEX 00 = Direct Access Device
HEX 01 = Sequential Access Device
HEX 02 = Output Only Device
HEX 03 = Processor Device
HEX 04 to 7F are Reserved.
HEX 80 to FF are Vendor Unique.

The device type qualifier (Byte 01) is defined as follows:

Bit 7 set to 1 shall be removeable media.
Bit 7 set to 0 shall be fixed media.
Bit 6 to 0 form a seven bit user specified code. This code may be set with switches or by some other means within the **TARGET.** Devices which do not support this feature shall return all zero bits. The value of this feature allows each user to assign unique coded to each device model that is supported on his system. These codes may then be used by self-configuring software to determine what kind of a device is at each **SASI BUS ADDRESS.** This is especially valuable for systems that support multiple types of removeable media.

If the length of additional bytes (Byte 02) is 0, no unit parameter bytes are returned. All **INITIATOR**s shall be capable of accepting a minimum of 3 bytes of **INQUIRY DATA.**

Table 21
Mode Select Command

BYTE	BIT 7	6	5	4	3	2	1	0
00	0	0	0	1	0	1	0	1
01	Logical Unit Number			Reserved				
02	Reserved							
03	Reserved							
04	Length of Parameter List							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

MODE SELECT (15) (Optional).

This command provides a means to specify media, unit, or device parameters. Byte 04 of the CDB shall specify the length (in bytes) of an associated parameter list which is sent as data. This parameter list contains a list of extent descriptors. Each extent descriptor specifies the number of blocks, block size, and density code for that extent. Additional vendor unique parameters may be appended to the extent descriptor list. The extent descriptor list provides a means to specify media characteristics on media that may contain multiple block sizes or densities such as flexible disks.

The following **MODE SELECT** parameter list is sent during the data phase of the command:

Table 22

BYTE	
00	Reserved
01	Reserved
02	Reserved
03	Length in Bytes of Descriptor List (nn)

The Extent Descriptor list follows in Bytes 04 through 4+nn. This list shall contain nn/8 Extent Descriptors defined as follows:

00	Density Code
01	Reserved
02	Number of Blocks
03	Number of Blocks
04	Block Size
05	Block Size
06	Block Size
07	Block Size

Additional Vendor Unique parameters may be sent in Bytes 4+nn through N.

4+nn to N	Vendor Unique Parameters

Bytes 02 and 03 (number of blocks in extent descriptor) - A zero value indicates all remaining blocks of the unit.

Byte 00 (Density Code in extent descriptor) - Defined as follows:

- 00 - Default (unit's default density)
- 01 - First commonly used density for the media type. (Flexible disk single density.)
- 02 - Second commonly used density for the media type. (Flexible disk double density.)
- 03 to FF - Increasing densities for the media type.

Table 23
Reserve Unit Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	1	0	1	1	0
01	**Logical Unit Number**				Reserved			
02	Reserved							
03	Reserved							
04	Reserved							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

RESERVE UNIT (16) (Optional).

Reserves this unit for use by the requesting **INITIATOR** until a release unit command is received. No other **INITIATOR** can perform any fucntion on this unit.

Table 24
Release Unit Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	1	0	1	1	1
01	**Logical Unit Number**				Reserved			
02	Reserved							
03	Reserved							
04	Reserved							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

RELEASE UNIT (17) (Optional).

Releases this unit from the requesting **INITIATOR.** This **INITIATOR** must have issued the previous reserve unit command.

Table 25
Copy Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	1	1	0	0	0
01	**Logical Unit Number**				Reserved			
02	MSB Length of Parameter List							
03	Length of Parameter List							
04	LSB Length of Parameter List							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

COPY (18) (Optional).

This copy command shall allow disk to disk data movement without host involvement in the data transfer.

CDB Bytes 2 - 4 will specify the byte length of the command parameter list to be retrieved from the **INITIATOR.**

The following **COPY** parameter list data is sent during the data phase of the command:

Table 26

BYTE	BIT 7	6	5	4	3	2	1	0
00	Copy Function						Priority	
01	Reserved							
02	Reserved							
03	Reserved							

The following segment descriptors may be repeated n times within the length descriptor above.

00	Source Controller I.D.				Reserved	Source LUN		
01	Dest. Controller I.D.				Reserved	Destination LUN		
02	Reserved							
03	Reserved							
04	MSB Number of Blocks							
05	Number of Blocks							
06	Number of Blocks							
07	LSB Number of Blocks							
08	MSB Logical Block Address (Source)							
09	Logical Block Address (Source)							
10	Logical Block Address (Source)							
11	LSB Logical Block Address (Source)							
12	MSB Logical Block Address (Dest.)							
13	Logical Block Address (Dest.)							
14	Logical Block Address (Dest.)							
15	LSB Logical Block Address (Dest.)							

Bits 7 to 3 of Byte 00 specify the copy function as follows:

Copy function = 00, Back Up (disk to tape).
Copy function = 01, Restore (tape to disk).
Copy function = 02, Copy (disk to disk).
Copy function = 03 to 0F (reserved).
Copy function = 10 to 1F (vendor unique).

Bits 2 to 0 of Byte 00 specify the priority of this command for use of the tape or disk resource as follows:

Priority = 0, copy is not interruptable.
Priority = 1, copy is interruptable.

Note: It is presumed that the disk **TARGET** will preserve the disk data during the copy command by use of the reserve command. If the copy is interruptable, the **TARGET** shall execute commands specifying **LUN**s other than the **LUN** involved in the copy during the execution of the copy command.

The segment descriptors start after Byte 03 of the parameter list for 16 bytes. Additional segment descriptors may be sent to specify additional data segments to be transferred.

Bits 7 to 5 of the first byte of the segment descriptor specify the address of the source disk controller.
Bits 4 and 3 are reserved and should be set to zero.
Bits 2 and 0 specify the **LUN** of the source disk unit.
Bits 7 to 5 of the second byte specify the address of the destination disk controller.
Bits 4 and 3 are reserved and should be set to zero.
Bits 2 to 0 specify the **LUN** of the destination disk unit.

The third through sixth bytes specify the number of disk blocks to be transferred. The seventh through tenth bytes specify the logical starting block address of the source disk.

The eleventh through the fourteenth bytes specify the logical starting block address of the destination disk.

Table 27
Mode Sense Command

BYTE	BIT 7	6	5	4	3	2	1	0
00	0	0	0	1	1	0	1	0
01	**Logical Unit Number**			Reserved				
02	Reserved							
03	Reserved							
04	Length of Data List							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

MODE SENSE (1A) (Optional).

This command provides a means for a **TARGET** to report its media, unit, or device parameters. It is a complementary command to the mode select command for support of media that may contain multiple block sizes or densities such as flexible disks. Byte 04 of the CDB shall specify the maximum number of bytes that may be returned by the command as data.

The following **MODE SENSE** data is sent during the data phase of the command:

Table 28

BYTE	
00	Length (in Bytes) of the Following Data (N)
01	Media Type
02	Reserved
03	Length in Bytes of Extent Descriptor List (nn)

The Extent Descriptor list follows in Bytes 04 through 4+nn. This list shall contain nn/8 Extent Descriptors defined as follows:

00	Density Code
01	Reserved
02	Number of Blocks
03	Number of Blocks
04	Block Size
05	Block Size
06	Block Size
07	Block Size

Additional Vendor Unique parameters may be sent in Bytes 4+nn through N.

4+nn to N	Vendor Unique Parameters

Bytes 02 and 03 (number of blocks in extent descriptor) - A zero value indicates all remaining blocks on the unit.

Byte 00 (Density Code in extent descriptor) - Defined as follows:

- 00 - Default (unit's default density)
- 01 - First commonly used density for the media type. (Flexible disk single density.)
- 02 - Second commonly used density for the media type. (Flexible disk double density.)
- 03 to FF - Increasing densities for the media type.

Byte 01 (Media Type) - Defined as follows:

- 00 - Default (unit only supports one media type).
- 01 - First commonly used media type for the unit (Flex disk single sided diskette).
- 02 - Second commonly used media type for the unit (Flex disk double sided diskette).
- 03 to FF - Additional media types.

Table 29
Start/Stop Unit Command

BYTE	BIT 7	6	5	4	3	2	1	0
00	0	0	0	1	1	0	1	1
01	**Logical Unit Number**			Reserved				
02	Reserved							
03	Reserved							
04	Decision							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

START/STOP UNIT (1B) (Optional)

If Byte 4 of the CDB is 1, this is a start command. Completion will be returned when unit is ready or after a determined delay depending upon the unit. If Byte 4 is 0, then this is a stop unit command. Completion will be returned after stopping or after a determined delay depending upon the unit.

Table 30
Receive Diagnostic Results Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	1	1	1	0	0
01	**Logical Unit Number**				Reserved			
02	Reserved							
03	Data Variable Length Indicator							
04	Data Variable Length Indicator							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

Note: This command allows the operating system to be independent of vendor unique diagnostic functions. The diagnostic software becomes more portable to various operating systems.

RECEIVE DIAGNOSTIC RESULTS (1C) (Optional).

Sends analysis data to **INITIATOR** after completion of a write diagnostic command. The analysis data is vendor unique with Bytes 3 and 4 of the CDB specifying the length of the data.

Table 31
Send Diagnostic Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	1	1	1	0	1
01	**Logical Unit Number**			Reserved		Self Test	Dev Ofl	Unt Ofl
02	Reserved							
03	Data Variable Length Indicator							
04	Data Variable Length Indicator							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

Note: This command allows the operating system to be independent of vendor unique diagnostic functions. The diagnostic software becomes more portable to various operating systems.

SEND DIAGNOSTIC (1D) (Optional).

Sends data to the **TARGET** to specify diagnostic tests for **TARGET** and peripheral units. The analysis data is vendor unique with Bytes 3 and 4 of the CDB specifying the length of the data.

Bit 00 of Byte 01 (Unit Off Line) enables write operations on user media or operations that affect user visable media positioning.

Bit 1 of Byte 01 (Device Off Line) enables diagnostic operations which may adversely affect **I/O** operations to other units on the same controller.

Bit 2 of Byte 01 (Self Test) directs the **TARGET** to complete its default self test.

6.3.2 Command Descriptions (Group 01) for Direct Access Devices

STANDARD (S)	Commands shall be implemented in order to meet the minimum requirement of this specification.
EXTENDED (E)	Commands shall be implemented in addition to Standard commands to meet the extended requirement of this specification.
OPTIONAL (O)	Commands, if used, shall be implemented as defined in this specification.
VENDOR UNIQUE (V)	Commands available for Vendor Unique implementations. See Appendix C for a list of current Vendor Unique commands. See the vendor specifications where compatability is desired.
RESERVED (R)	Command codes not currently used are reserved for ANSI definition at a future date.

Command operation codes and names are listed in Table 32.

Table 32
Command Codes
Group 01 Commands

HEX OP Code	Type	Description
00	V	
01	V	
02	V	
03	V	
04	V	
05	E	READ CAPACITY
06	V	
07	V	
08	E	READ
09	V	
0A	E	WRITE
0B	V	
0C	V	
0D	V	
0E	O	WRITE AND VERIFY
0F	O	VERIFY
10	O	SEARCH DATA HIGH
11	O	SEARCH DATA EQUAL
12	O	SEARCH DATA LOW
13	R	
14	R	
15	R	
16	R	
17	R	
18	R	
19	R	
1A	R	
1B	R	
1C	R	
1D	R	
1E	R	
1F	R	

Table 33
Read Capacity Command

BYTE	BIT 7	6	5	4	3	2	1	0
00	0	0	1	0	0	1	0	1
01	**Logical Unit Number**				Reserved			Rel Add
02	MSB Logical Block Address							
03	Logical Block Address							
04	Logical Block Address							
05	LSB Logical Block Address							
06	Reserved							
07	Reserved							
08	Full or Partial Media Indicator							
09	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

READ CAPACITY (05) (Extended).

If Byte 08 of the CDB is zero, this command will return the address and size of the last block of the unit. If Byte 08 of the CDB is one, this command will return the address and size of the last block after the specified address before a substantial delay in data transfer will be encountered (e.g., a cylinder boundary).

The following **READ CAPACITY** data is sent during the data phase of the command:

Table 34

BYTE	
00	Block Address
01	Block Address
02	Block Address
03	Block Address
04	Block Size
05	Block Size
06	Block Size
07	Block Size

Table 35
Read Command

BYTE	BIT 7	6	5	4	3	2	1	0
00	0	0	1	0	1	0	0	0
01	**Logical Unit Number**				Reserved			Rel Add
02	MSB Logical Block Address							
03	Logical Block Address							
04	Logical Block Address							
05	LSB Logical Block Address							
06	Reserved							
07	Number of Blocks							
08	Number of Blocks							
09	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

READ (08) (Extended).

Transfers to the **INITIATOR** the specified number of blocks starting at the specified logical starting block address. This command shall return the most recent data value written to the specified block(s).

Table 36
Write Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	1	0	1	0	1	0
01	**Logical Unit Number**				Reserved			Rel Add
02	MSB Logical Block Address							
03	Logical Block Address							
04	Logical Block Address							
05	LSB Logical Block Address							
06	Reserved							
07	Number of Blocks							
08	Number of Blocks							
09	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

WRITE (0A) (Extended).

Transfers to the **TARGET** the specified number of blocks starting at the specified logical starting block address.

Table 37
Write and Verify Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	1	0	1	1	1	0
01	Logical Unit Number				Reserved			Rel Add
02	MSB Logical Block Address							
03	Logical Block Address							
04	Logical Block Address							
05	LSB Logical Block Address							
06	Reserved							
07	Number of Blocks							
08	Number of Blocks							
09	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

WRITE AND VERIFY (0E) (Optional).

Writes and verifies the data for the specified number of blocks.

Table 38
Verify Command

BYTE	BIT 7	6	5	4	3	2	1	0
00	0	0	1	0	1	1	1	1
01	**Logical Unit Number**			Reserved				Rel Add
02	MSB Logical Block Address							
03	Logical Block Address							
04	Logical Block Address							
05	LSB Logical Block Address							
06	Reserved							
07	Number of Blocks							
08	Number of Blocks							
09	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

VERIFY (0F) (Optional).

Verifies the data for the specified number of blocks. No data is transferred with this command.

Table 39
Search Data High Command

BYTE	BIT 7	6	5	4	3	2	1	0
00	0	0	1	1	0	0	0	0
01	**Logical Unit Number**				Reserved		Spn Dat	Rel Add
02	MSB Logical Block Address							
03	Logical Block Address							
04	Logical Block Address							
05	LSB Logical Block Address							
06	Reserved							
07	Number of Blocks							
08	Number of Blocks							
09	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

SEARCH DATA HIGH (10) (Optional).

This command performs the same function as the search data equal command, but is satisfied by a compare of high or equal.

See **SEARCH DATA EQUAL** for Search Data Format.

Table 40
Search Data Equal Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	1	1	0	0	0	1
01	**Logical Unit Number**			Reserved			Spn Dat	Rel Add
02	MSB Logical Block Address							
03	Logical Block Address							
04	Logical Block Address							
05	LSB Logical Block Address							
06	Reserved							
07	Number of Blocks							
08	Number of Blocks							
09	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

SEARCH DATA EQUAL (11) (Optional).

This command is used to search one or more blocks for equality to a data pattern. The search commands contain the concept of records within a data block to allow multiple records within a block to be searched. The Spanned Flag, the starting block address, and the number of blocks to search are specified in the Command Descriptor Block. The record size, starting record offset, and search data pattern are specified in the search parameter list which is passed to the target as data. The Spanned Flag (Bit 1, Byte 01), when set, indicates that records span block boundaries. Thus, a record may start in one block and end in the next or a subsequent block. When the Spanned Flag is not set, each record shall be wholly contained within a single block. Any unused space at the end of a block which is smaller than the record size shall be ignored by the Search commands.

If a command is linked to the Search command, and the search is successful, then the next command is fetched and executed. In this case, if the Relative Address bit (Bit 0, Byte 01) is on, the address portion of the command is used as a displacement from the block address at which the search was satisfied. If a linked search was not satisfied, the link is broken and ending status is presented.

When a Search command is not linked to another command, if the search is satisfied, it will terminate with a condition met status. A Request Sense command can then be issued to determine the block address and record offset of the matching record. A Request Sense following a successful Search command shall:

1. Report a Sense Key of equal if this condition occured; otherwise, report a Sense Key of No Sense.

2. Set the Valid bit to 1.

3. Report the address of the block containing the first matching record in the Information Bytes.

4. Report the record offset within block in the first four bytes of Additional Sense Bytes.

A Request Sense following an unsuccessful Search command shall:

1. Report a Sense key of No Sense, provided no errors occurred.

2. Set the Valid Bit to zero.

The following **SEARCH** parameter list will be sent during the data phase of the command:

Table 41

BYTE	
00	Length of Records (Bytes)
01	Length of Records (Bytes)
02	Length of Records (Bytes)
03	Length of Records (Bytes)
04	Number of Bytes to Skip in First Block
05	Number of Bytes to Skip in First Block
06	Number of Bytes to Skip in First Block
07	Number of Bytes to Skip in First Block
08	Length of Search Argument (Bytes)
09	Length of Search Argument (Bytes)

The following search argument may be repeated n times within the length defined above.

00	Displacement
01	Displacement
02	Displacement
03	Displacement
04	Length of Field to Compare (MM Bytes)
05	Length of Field to Compare (MM Bytes)
06-MM	Data to Compare

Bytes 00 through MM define the search argument and may be repeated.

Bytes 04 and 05 define the number of bytes in the search argument (Bytes 06-MM).

Table 42
Search Data Low Command

BYTE	BIT 7	6	5	4	3	2	1	0
00	0	0	1	1	0	0	1	0
01	**Logical Unit Number**			Reserved			Spn Dat	Rel Add
02	MSB Logical Block Address							
03	Logical Block Address							
04	Logical Block Address							
05	LSB Logical Block Address							
06	Reserved							
07	Number of Blocks							
08	Number of Blocks							
09	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

SEARCH DATA LOW (12) (Optional).

This command performs the same function as the search data equal command, but is satisfied by a compare of low or equal.

See **SEARCH DATA EQUAL** for Search Data Format.

6.3.4 Command Descriptions (Group 05) for Direct Access Devices

STANDARD (S)	Commands shall be implemented in order to meet the minimum requirement of this specification.
EXTENDED (E)	Commands shall be implemented in addition to Standard commands to meet the extended requirement of this specification.
OPTIONAL (O)	Commands, if used, shall be implemented as defined in this specification.
VENDOR UNIQUE (V)	Commands available for Vendor Unique implementations. See Appendix C for a list of current Vendor Unique commands. See the vendor specifications where compatability is desired.
RESERVED (R)	Command codes not currently used are reserved for ANSI definition at a future date.

Command operation codes and names are listed in Table 43.

Table 43
Command Codes
Group 05 Commands

HEX OP Code	Type	Description
00	V	
01	V	
02	V	
03	V	
04	V	
05	V	
06	V	
07	V	
08	V	
09	O	SET LIMITS
0A	V	
0B	V	
0C	V	
0D	V	
0E	V	
0F	V	
10	R	
11	R	
12	R	
13	R	
14	R	
15	R	
16	R	
17	R	
18	R	
19	R	
1A	R	
1B	R	
1C	R	
1D	R	
1E	R	
1F	R	

Table 44
Set Limits Command

BYTE	BIT 7	6	5	4	3	2	1	0
00	1	0	1	0	1	0	0	1
01	**Logical Unit Number**				Reserved			
02	MSB Logical Block Address							
03	Logical Block Address							
04	Logical Block Address							
05	LSB Logical Block Address							
06	2nd MSB Logical Block Address							
07	2nd Logical Block Address							
08	2nd Logical Block Address							
09	2nd LSB Logical Block Address							
10	Reserved							
11	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

SET LIMITS (09) (Optional).

This command defines the addresses outside of which any following linked commands may not operate. A second set block limits command may not be linked to a chain of commands in which a set block limits command has already been issued. The two low order bits of the Byte 10 define the legal operations within the limits of the specified addresses. Bit 0 indicates write inhibit, and Bit 1 indicates read inhibit.

6.3.5 Command Descriptions (Groups 06 and 07) for Direct Access Devices

Commands in these classes are vendor unique commands.

6.4 Commands - Sequential Access Devices

6.4.1 Command Descriptions (Group 00) for Sequencial Access Devices

STANDARD (S)	Commands shall be implemented in order to meet the minimum requirement of this specification.
EXTENDED (E)	Commands shall be implemented in addition to Standard commands to meet the extended requirement of this specification.
OPTIONAL (O)	Commands, if used, shall be implemented as defined in this specification.
VENDOR UNIQUE (V)	Commands available for Vendor Unique implementations. See Appendix C for a list of current Vendor Unique commands. See the vendor specifications where compatability is desired.
RESERVED (R)	Command codes not currently used are reserved for ANSI definition at a future date.

Command operation codes and names are listed in Table 45.

Table 45
Command Codes
Group 00 Commands

HEX OP Code	Type	Description
00	O	TEST UNIT READY
01	S	REWIND
02	V	
03	S	REQUEST SENSE
04	R	
05	E	READ BLOCK LIMITS
06	V	
07	V	
08	S	READ
09	V	
0A	S	WRITE
0B	O	TRACK SELECT
0C	V	
0D	V	
0E	V	
0F	O	READ REVERSE
10	S	WRITE FILE MARK
11	O	SPACE
12	E	INQUIRY
13	O	VERIFY
14	O	RECOVER BUFFERED DATA
15	O	MODE SELECT
16	O	RESERVE UNIT
17	O	RELEASE UNIT
18	O	COPY
19	O	ERASE
1A	O	MODE SENSE
1B	O	LOAD/UNLOAD
1C	O	RECEIVE DIAGNOSTIC RESULTS
1D	O	SEND DIAGNOSTIC
1E	R	
1F	R	

Table 46
Test Unit Ready Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	0	0
01	**Logical Unit Number**				Reserved			
02	Reserved							
03	Reserved							
04	Reserved							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

TEST UNIT READY (00) (Optional).

Returns zero status if addressed unit is powered on and ready. This is not a request for unit self test. A fast response is expected.

Table 47
Rewind Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	0	1
01	**Logical Unit Number**				Reserved			Immed
02	Reserved							
03	Reserved							
04	Reserved							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

REWIND (01) (Standard).

The specified unit is to be rewound to beginning of tape (BOT) or load point. Status shall be returned after the rewind is completed unless the Immediate Status bit (Bit 0, Byte 1) is set. If the Immediate Status bit is set, status may be returned as soon as the **REWIND** is initiated.

Table 48
Request Sense Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	1	1
01	**Logical Unit Number**			Reserved				
02	Reserved							
03	Reserved							
04	Number of Bytes Allocated							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

REQUEST SENSE (03) (Standard).

Returns unit sense. The sense data will be valid for the check condition (status) last presented to the **INITIATOR**. This sense data shall be preserved in the **TARGET** for the **INITIATOR**. Sense data shall be cleared on the reception of any subsequent command to the unit in check from the **INITIATOR** receiving the check condition. (See Section 6.8 and 6.9 for sense data format.) Byte 04 in this command will specify the number of bytes that the **INITIATOR** has allocated for returned sense. All **INITIATOR**s shall be capable of accepting a minimum of four sense bytes (i.e., a value of 0 through 3 defaults to a four byte allocation).

Check status shall not be returned for this command.

Table 49
Read Block Limits Command

BIT / BYTE	7	6	5	4	3	2	1	0
00	0	0	0	0	0	1	0	1
01	**Logical Unit Number**			Reserved				
02	Reserved							
03	Reserved							
04	Reserved							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag Req	Link

READ BLOCK LIMITS (05) (Extended).

Requests that the block length limits be returned for the addressed unit as data. The limit data shall be returned as follows:

The following **READ BLOCK LIMITS** data is sent during the data phase of the command:

Table 50

BYTE	
00	MSB Maximum Block Length
01	LSB Maximum Block Length
02	MSB Minimum Block Length
03	LSB Minimum Block Length

If the Maximum Block Length = the Minimum Block Length, fixed length blocks are specified. Otherwise, variable length blocks are specified. For variable length blocks, if the Maximum Block Length = 0, no upper limit is specified.

Table 51
Read Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	0	1	0	0	0
01	**Logical Unit Number**			Reserved				Fixed
02	Number of Bytes/Blocks to Transfer							
03	Number of Bytes/Blocks to Transfer							
04	Number of Bytes/Blocks to Transfer							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

READ (08) (Standard).

When the Fixed bit (Bit 0 of Byted 01) is zero, this command shall cause the next block on the addressed unit to be transferred to the **INITIATOR**. Bytes 2-4 of the CDB specify the number of bytes the **INITIATOR** has allocated for the returned data. If the actual block size is different from the specified number of bytes, check status shall be sent to the **INITIATOR** and the Incorrect Length Indicator (ILI) shall be set to one in the Extended Sense. The Information Bytes in the Extended Sense shall be set to the difference (residue) between the requested length and the actual block size. Controllers which do not support negative residues shall set the ILI to one and the residue to zero when the actual block size is larger than the requested length.

When the Fixed bit is one, this command shall cause a number of blocks on the addressed unit to be transferred to the **INITIATOR**. The number of blocks to be transferred is specified in Bytes 2-4 of the CDB. Fixed mode **READ** commands are valid only if the unit is currently operating in Fixed Block Mode. Units are said to be in Fixed Block Mode when either of the following conditions are true:

1. The unit reports the same value for Minimum Block Size and Maximum Block Size in response to the **READ BLOCK LIMITS** command.
2. A variable block size unit has been instructed to use fixed block sizes with the **MODE SELECT** command.

If neither of the above conditions are true, units are said to be in Variable Block Mode.

If a block is read which is larger or smaller than the unit's current block size, check status shall be sent to the **INITIATOR** and the Incorrect Length Indicator (ILI) shall be set to one in the Extended Sense. A unit may, at its option, report the actual block size

encountered by setting the Valid bit to one and the Information Bytes to the difference (residue) between the unit's current block size and the actual block size. Units which do not report the actual block size shall set the Valid bit to zero.

If unit reads a File Mark prior to completing a fixed mode Read command, the unit shall send check status to the **INITIATOR** and set the File Mark bit in Extended Status to one. The Information Bytes in the Extended Sense shall be set to the difference (residue) between the requested number of blocks and the actual number of blocks read (not including the File Mark).

Table 52
Write Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	0	1	0	1	0
01	**Logical Unit Number**			Reserved				Fixed
02	Number of Bytes/Blocks to Transfer							
03	Number of Bytes/Blocks to Transfer							
04	Number of Bytes/Blocks to Transfer							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

WRITE (0A) (Standard).

When the Fixed bit (Bit 0 of Byte 01) is zero, this command shall cause a single block to be transferred from the **INITIATOR** and written to the addressed unit beginning at the current media position. Bytes 2-4 of the CDB specify the size of the block to be written. The requested block size must match the unit's current block size if it is in Fixed Block Mode or must be within the unit's Minimum and Maximum Block Size range if it is in Variable Block Mode. If the above conditions are not true, the controller shall send Check status to the **INITIATOR** and shall set the Sense Key in the Extended Sense to Illegal Request.

When the Fixed bit is one, this command shall cause a number of blocks to be transferred from the **INITIATOR** to the **TARGET** to be written on the addressed unit beginning at the current media position. Bytes 2-4 of the CDB specify the number of blocks to be transferred. Fixed Mode **WRITE** commands are valid only if the unit is currently operating in Fixed Block Mode (see the **READ** command description).

If the Early Warning End-of-Tape condition is encountered while writing, the controller shall attempt to finish writing any buffered data and shall terminate with Check status. The EOM bit in Extended Sense shall be set to one. If the physical End-of-Tape is encountered prior to writing all buffered data, the controller shall additionally set the Volume Overflow Sense Key in the Extended Sense. When the End-of-Tape condition is encountered on a Fixed Mode **WRITE** command, the Valid bit in Extended Sense shall be set to one and the Information Bytes shall be set to the difference (residue) between the requested number of blocks and the actual number of blocks written on the media.

Table 53
Track Select Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	0	1	0	1	1
01	**Logical Unit Number**			Reserved				
02	Reserved							
03	Reserved							
04	Track Value							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

TRACK SELECT (0B) (Optional).

This command requests that the specified track (value in CDB Byte 04) in the addressed Logical Unit be selected.

Table 54
Read Reverse Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	0	1	1	1	1
01	**Logical Unit Number**			Reserved				Fixed
02	Number of Bytes/Blocks to Transfer							
03	Number of Bytes/Blocks to Transfer							
04	Number of Bytes/Blocks to Transfer							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

READ REVERSE (0F) (Optional).

This command functions identically to the Read Command except that tape motion is in the reverse direction. Thus, the block(s) and bytes within the block(s) are transferred in the reverse order. This command shall terminate with Check status and the EOM bit in Extended Sense shall be set to one if BOT or Load point is encountered. The Valid bit shall be set to one and the Information Bytes shall contain the difference (residue) of the requested number of bytes/blocks and the actual number of bytes/blocks transferred before BOT was encountered. File Mark handling is the same as in the Read Command.

Table 55
Write File Marks Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	1	0	0	0	0
01	**Logical Unit Number**				Reserved			
02	Number of File Marks							
03	Number of File Marks							
04	Number of File Marks							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

WRITE FILE MARKS (10) (Standard).

This command causes one or more File Marks to be written on the specified logical unit. The number of File Marks to be written is specified by Bytes 2-4 of the CDB. A zero in this field indicates that no File Marks are to be written.

Table 56
Space Command

BYTE	BIT 7	6	5	4	3	2	1	0
00	0	0	0	1	0	0	0	1
01	**Logical Unit Number**			Reserved			Code	
02	Count							
03	Count							
04	Count							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

SPACE (11) (Optional).

This command provides a variety of positioning functions which are determined by the code and count fields in the CDB. Both forward (toward End-of-Tape) and reverse (toward Beginning-of-Tape) positioning are provided, although some controllers will only support a subset of this command. Such controllers shall return check status and set the Sense key in Extended Sense to Illegal Request in response to any attempt to invoke function which are not supported.

The code field is defined as follows:

Code	Meaning
00	Blocks
01	File Marks
10	Sequential File Marks
11	Physical End-of-Data

When spacing over Blocks or File Marks Bytes 2-4 of the CDB specify the number of Blocks or File Marks to be spaced over. A positive value N shall cause forward tape movement over N Blocks or File Marks ending on the EOT side of the last Block or File Mark. A zero value shall cause no tape movement. A negative value -N (2's complement notation) shall cause reverse tape movement over N Blocks or File Marks ending on the BOT side of the last Block or File Mark.

If a File Mark is encountered while spacing over Blocks, tape movement shall be stopped. The media shall be positioned on the ETO side of the File Mark if movement was in the forward direction and on the BOT side of the File Mark if movement was in the reverse direction. Check status shall be sent to the **INITIATOR** and the File Mark and Valid bits in Extended Sense shall be set to one. The Information Bytes shall be set to the difference (residue) in the request number of blocks and the actual number of blocks spaced over (not including the File Mark).

When spacing over Sequential File Marks, the count field (Bytes 2-4 of the CDB) is interpreted as follows:

A positive value N shall cause forward tape movement to the first occurrence of N consecutive File Marks stopping after the Nth File Mark.

A zero value shall cause no tape movement.

A negative value -N (2's complement notation) shall cause reverse tape movement to the first occurrence of N consecutive File Marks stopping on the BOT side of the Nth File Mark.

When spacing to Physical End-of-Data, the count field of the CDB is ignored. Forward tape movement shall occur until the unit encounters Physical End-of-Data as defined by the unit. Some units define Physical End-of-Data as a DC-erased area on the media; however, other definitions are not precluded. Units which implement this function shall leave the media positioned such that a subsequent Write command would append data to the last file on the volume.

Table 57
Inquiry Command

BYTE	BIT 7	6	5	4	3	2	1	0
00	0	0	0	1	0	0	1	0
01	**Logical Unit Number**			Reserved				
02	Reserved							
03	Reserved							
04	Number of Bytes Allocated							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

INQUIRY (12) (Extended).

This command sends to the **INITIATOR** information regarding unit and **TARGET** parameters.

Byte 04 in the CDB of this command will specify the number of bytes that the host has allocated for returned unit data.

A value of zero indicates 256 bytes. A value of from 1 to 255 is as indicated.

The following **INQUIRY** data is sent during the data phase of the command:

Table 58

BYTE	
00	Device Type Code
01	Device Type Qualifier
02	Length of Additional Bytes (N)

The following unit parameter bytes may be repeated N times.

00 to N-1	Unit Parameter Byte

The device type code (Byte 00) is as follows:

HEX 00 = Direct Access Device
HEX 01 = Sequential Access Device
HEX 02 = Output Only Device
HEX 03 = Processor Device
HEX 04 to 7F = Are Reserved.
HEX 80 to FF = Are Vendor Unique.

The device type qualifier (Byte 01) is defined as follows:

Bit 7 set to 1 = Removable Media.
Bit 7 set to 0 = Fixed Media.
Bits 6 to 0 form a seven bit user specified code. This code may be set with switches or by some other means within the **TARGET.** Devices which do not support this feature shall return all zero bits. The value of this feature allows each user to assign unique codes to each device model that is supported on his system. These codes may then be used by self-configuring software to determine what kind of a device is at each **SASI BUS** address. This is specially valuable for systems that support multiple types of removable media.

If the length of additional bytes (Byte 02) is 0, no unit parameter bytes are returned. All **INITIATOR**s shall be capable of accepting a minimum of 3 bytes of **INQUIRY DATA.**

Table 59
Verify Command

BYTE	BIT 7	6	5	4	3	2	1	0
00	0	0	0	1	0	0	1	1
01	**Logical Unit Number**			Reserved				
02	Number of Blocks MSB							
03	Number of Blocks							
04	Number of Blocks LSB							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

VERIFY (13) (Optional).

This command verifies the data on the addressed logical unit for the number of blocks specified in the CDB Bytes 2-4 starting with the next block on the tape. Bit 7 of Byte 02 shall = 0.

Table 60
Recover Buffered Data Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	1	0	1	0	0
01	**Logical Unit Number**			Reserved				Fixed
02	Number of Bytes/Blocks							
03	Number of Bytes/Blocks							
04	Number of Bytes/Blocks							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

RECOVER BUFFERED DATA (14) (Optional).

This command is used to read data that has been written to the controller data buffer but not yet written on the media. It is normally only used to recover from error or exception conditions that make it impossible to write the buffered data on the media. This command functions similarly to the Read command except that the data is transferred from the controller data buffer instead of the media. The order in which block(s) are transferred is the same as they would have been transferred to the media.

One or more Recover Buffered Data commands may be used to read the unwritten buffered data blocks. Controllers shall return Check status and set the EOM bit in Extended Sense to one whenever an attempt is made to recover more blocks than are contained in the buffer. Additionally, if the Fixed bit in the CDB is one, the controller shall set the Valid bit in Extended Sense to one and shall set the Information Bytes to the difference (residue) between the requested number of blocks and the actual number of blocks transferred.

Table 61
Mode Select Command

BYTE	BIT 7	6	5	4	3	2	1	0
00	0	0	0	1	0	1	0	1
01	**Logical Unit Number**				Reserved			
02	Reserved							
03	Reserved							
04	Length of Parameter List							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

MODE SELECT (15) (Optional).

This command provides a means to specify media, unit, or device parameters. Byte 04 of the CDB shall specify the length (in bytes) of an associated parameter list which is sent as data. This parameter list contains a list of extent descriptors. Each extent descriptor specifies the number of blocks, block size, and density code for that extent. Additional vendor unique parameters may be appended to the extent descriptor list. The extent descriptor list provides a means to specify media characteristics on media that may contain different densities such as half inch tapes.

The following **MODE SELECT** parameter list is sent during the data phase of the command:

Table 62

BYTE		
00	Reserved	
01	Reserved	
02	Buffered Mode	Speed
03	Length in Bytes of Descriptor List (nn)	

The Extent Descriptor list follows in Bytes 04 through 4+nn. This list shall contain nn/8 Extent Descriptors defined as follows:

00	Density Code
01	Reserved
02	Number of Blocks
03	Number of Blocks
04	Reserved
05	Reserved
06	Block Size
07	Block Size

Additional Vendor Unique parameters may be sent in Bytes 4+nn through N.

4+nn to N	Vendor Unique Parameters

Buffered Mode (Bits 4-7 of Byte 02) is defined as follows:

0 - The unit shall not report good status on Write commands until data blocks are actually written on the media.

1 - The unit may report good status on Write commands as soon as the data block has been transferred to the controller's buffer. One or more blocks may be buffered prior to writing the block(s) to the media.

2 to F - Reserved.

Speed (Bits 0-3 of Byte 02) is defined as follows:

0 - Default (Use the unit's default speed).
1 - Use the unit's lowest speed.
2 to F - Use increasing unit speeds.

Density Code (Byte 00 of the Extent Descriptor) is defined as follows:

00 - Default (Use the unit's default density).
01 - First commonly used density for the media type. (Half inch - 800 BPI - NRZI).
02 - Second commonly used density for the media type. (Half inch -1600 BPI -PE).
03 - Third commonly used densith for the media type. (Half inch 6250 BPI = GCR).
04 to FF - Increasing densities for the media type.

A zero in the Number of Blocks field (Bytes 02-03) of an Extent Descriptor shall indicate that all remaining blocks on the unit are to be formatted as specified in that Extent Descriptor. A zero in the Block Size field (Bytes 06-07) of an Extent Descriptor shall indicate that blocks in that extent are variable in size.

Table 63
Reserve Unit Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	1	0	1	1	0
01	**Logical Unit Number**				Reserved			
02	Reserved							
03	Reserved							
04	Reserved							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

RESERVE UNIT (16) (Optional).

This command reserves this unit for use by the requesting **INITIATOR** until a **RELEASE UNIT COMMAND** is received. No other **INITIATOR** can perform any function on this unit.

Table 64
Release Unit Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	1	0	1	1	1
01	**Logical Unit Number**				Reserved			
02	Reserved							
03	Reserved							
04	Reserved							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

RELEASE UNIT (17) (Optional).

This command releases this unit from the requesting **INITIATOR.** This **INITIATOR** must have issued the previous **RESERVE UNIT COMMAND.**

Table 65
Copy Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	1	1	0	0	0
01	**Logical Unit Number**				Reserved			
02	MSB Length of Parameter List							
03	Length of Parameter List							
04	LSB Length of Parameter List							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

COPY (18) (Optional).

This copy command shall allow tape to disk and disk to tape data movement without host involvement in the data transfer. Host involvement shall be required to insert label names, file marks, etc., between copy commands.

CDB Bytes 2 - 4 will specify the byte length of the command parameter list to be retrieved from the **INITIATOR.**

The following **COPY** parameter list data is sent during the data phase of the command:

Table 66

BYTE	BIT 7	6	5	4	3	2	1	0
00	Copy Function						Priority	
01	Reserved							
02	Tape Block Size							
03	Tape Block Size							

One or more Segment Descriptors may follow beginning in Byte 04. The Segment Descriptors are defined as follows:

00	Source Controller I.D.	Reserved	Logical Unit Number
01	Reserved		
02	Reserved		
03	Reserved		
04	MSB Number of Blocks		
05	Number of Blocks		
06	Number of Blocks		
07	LSB Number of Blocks		
08	MSB Logical Block Address		
09	Logical Block Address		
10	Logical Block Address		
11	LSB Logical Block Address		

Copy Function (Bits 3-7 of Byte 00) is defined as follows:

00	- Backup (disk to tape)
01	- Restore (tape to disk)
02	- Copy (disk to disk; see 6.3.1)
03 to 0F	- Reserved
0F to 1F	- Vendor Unique

Priority (Bits 0-2 of Byte 00) establishes the relative priority of this Copy Command to other commands being executed by the same controller. All non-Copy commands are assumed to have a priority of 1. Priority 0 is the highest priority with increasing priority numbers indicating lower priorities.

Tape Block Size (Bytes 02-02) specifies the Block Size to be used on the Tape unit during the Copy for Variable Block Mode units. This field should match the current Block Size for Fixed Block Mode units. If this field does not match, the controller shall send Check status to the **INITIATOR** and set the Sense Key in Extended Sense to Illegal Request.

Controller Device ID (Bits 5-7 of Byte 00 of the Segment Descriptor) specifies the **SASI BUS DEVICE** number of the disk unit to be transferred to/from in the current segment. Some controllers may not support Controller Device IDs other than their own. Such controllers shall send Check status to the **INITIATOR** and set the Sense Key in the Extended Sense toIllegal Request when a Segment Descriptor contains an unsupported Controller Device ID.

Logical Unit Number (Bits 0-2 of Byte 00 of the Segment Descriptor) specifies the Logical Unit Number of the disk unit to be transferred to/from in the current segment.

Number of Blocks (Bytes 04-07 of the Segment Descriptor) specifies the number of blocks in the current segment. A zero value indicates that no blocks are to be transferred in this segment.

Logical Block Address (Bytes 08-11 of the Segment Descriptor) specifies the starting block address of the segment.

If the Early Warning End-of-Tape condition is encountered during a Backup Copy command, the controller shall:

1. Terminate with Check status.

2. Set the Valid and EOM bits in Extended Sense to one.

3. Set Byte 01 of Extended Sense to the current Segment Descriptor number. The Segment Descriptors are identified by ascending numbers beginning with 0.

4. Set the Information Bytes of the Extended Sense to the difference (residue) between the Number of Blocks field for the current segment and the actual number of blocks written on the media.

If the File Mark is encountered during a Restor Copy command, the controller shall:

1. Terminate with Check status.

2. Set the Valid and File Mark bits in Extended Sense to one.

3. Set Byte 01 of Extended Sense to the current Segment Descriptor number. The Segment Descriptors are identified by ascending numbers beginning with 0.

4. Set the Information Bytes of the Extended Sense to the difference (residue) between the Number of Blocks field for the current segment and the actual number of blocks transferred to the disk.

Table 67
Erase Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	1	1	0	0	1
01	Logical Unit Number			Reserved				Long
02	Reserved							
03	Reserved							
04	Reserved							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

ERASE (19) (Optional).

This command causes part or all of the remaining media on the volume to be "erased" beginning from the current media position. As used here, "erased" means either the media shall be DC-erased or a pattern shall be written on the media that appears as gap to the controller. The distance to be erased is controlled by the Long Bit (Bit 0 of Byte 01) in the CDB. If the Long bit is one, all remaining media on the volume shall be erased. If the Long bit is zero, a controller defined portion of the media shall be erased. Normally, short erases are used to create an extended gap for software controlled error recovery or for support of update in place functions.

Note: Some controllers will reject long Erase commands if the unit is not positioned at BOT.

Table 68
Mode Sense Command

BYTE	BIT 7	6	5	4	3	2	1	0
00	0	0	0	1	1	0	1	0
01	**Logical Unit Number**				Reserved			
02					Reserved			
03					Reserved			
04					Length of Data List			
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

MODE SENSE (1A) (Optional).

This command provides a means for a **TARGET** to report its media, unit, or device parameters. It is a complementary command to the mode select command for support of media that may contain different densities such as half inch tapes. Byte 04 of the CDB shall specify the maximum number of bytes that may be returned by the command as data.

The following **MODE SENSE** data is sent during the data phase of the command:

Table 69

BYTE		
00	Length (in Bytes) of the Following Data (N)	
01	Media Type	
02	Buffered Mode	Speed
03	Length in Bytes of Extent Descriptor List (nn)	

The Extent Descriptor list follows in Bytes 04 through 4+nn. This list shall contain nn/8 Extent Descriptors defined as follows:

00	Density Code
01	Reserved
02	Number of Blocks
03	Number of Blocks
04	Reserved
05	Reserved
06	Block Size
07	Block Size

Additional Vendor Unique parameters may be sent in Bytes 4+nn through N.

4+nn to N	Vendor Unique Parameters

Media Type (Byte 01) is defined as follows:

00	- Default (Unit only supports one media type).
01	- First commonly used media type for the unit.
02	- Second commonly used media type for the unit.
03 to FF	- Additional media types for the unit.

Buffered Mode (Bits 4-7 of Byte 02) is defined as follows:

0	- The unit will not report good status on Write commands until data blocks are actually written on the media.
1	- The unit may report good status on Write commands as soon as the data block has been transferred to the controller's buffer. One or more blocks may be buffered prior to writing the block(s) to the media.
2 to F	- Reserved.

Speed (Bits 0-3 of Byte 02) is defined as follows:

0	- Default (unit only supports one speed).
1	- Slowest unit speed.
2 to F	- Increasing unit speeds.

Density Code (Byte 00 of the Extent Descriptors) is defined as follows:

00	- Default (unit only supports one density).
01	- First commonly used density for the media type. (Half inch 800 BPI - NRZI)
02	- Second commonly used density for the media type. (Half inch - 1600 BPI - PE)
03	- Third commonly used density for the media type. (Half inch - 6250 BPI - GCR)
04 to FF	- Increasing densities for the media type.

A zero in the Number of Blocks field (Bytes 02-03) of an Extent Descriptor means that all remaining blocks on the unit are formatted as specified in that Extent Descriptor. A zero in the Block Size field (Bytes 06-07) of an Extent Descriptor means that the blocks in that extent are variable in size.

Table 70
Load/Unload Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	1	1	0	1	1
01	**Logical Unit Number**				Reserved			
02	Reserved							
03	Reserved							
04	Reserved							Load
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

LOAD/UNLOAD (1B) (Optional).

If the Load bit (Bit 0 of Byte 04) is on, this command requests that the media on the addressed logical unit be loaded and positioned to BOT or Load Point as determined by the unit.

If the Load bit is off, this command requests that the media on the addressed logical unit be positioned for removal from the drive.

Table 71
Receive Diagnostic Results Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	1	1	1	0	0
01	**Logical Unit Number**				Reserved			
02	Reserved							
03	Data Variable Length Indicator							
04	Data Variable Length Indicator							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

Note: This command allows the operating system to be independent of vendor unique

diagnostic functions. The diagnostic software becomes more portable to various operating systems.

RECEIVE DIAGNOSTIC RESULTS (1C) (Optional).

This command sends analysis data to **INITIATOR** after completion of a write diagnostic command. The analysis data is vendor unique with bytes 3 and 4 of the CDB specifying the length of the data.

Table 72
Send Diagnostic Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	1	1	1	0	1
01	**Logical Unit Number**			Reserved			Dev Ofl	Unt Ofl
02	Reserved							
03	Data Variable Length Indicator							
04	Data Variable Length Indicator							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

Note: This command allows the operating system to be independent of vendor unique diagnostic functions. The diagnostic software becomes more portable to various operating systems.

SEND DIAGNOSTIC (1D) (Optional).

This command sends data to the **TARGET** to specify diagnostic tests for **TARGET** and peripheral units. The analysis data is vendor unique with Bytes 3 and 4 of the CDB specifying the length of the data.

Unit off line (Bit 0 of Byte 01) enables diagnostic operations which affect the current tape position or write on the tape.

Device off line (Bit 1 of Byte 01) enables diagnostic operations which may adversely affect **I/O** operations to other units on the same controller.

6.5 Commands - Processor Devices

6.5.1 Command Descriptions (Group 00) for Processor Devices

STANDARD (S)	Commands shall be implemented in order to meet the minimum requirement of this specification.
EXTENDED (E)	Commands shall be implemented in addition to Standard commands to meet the extended requirement of this specification.
OPTIONAL (O)	Commands, if used, shall be implemented as defined in this specification.
VENDOR UNIQUE (V)	Commands available for Vendor Unique implementations. See Appendix C for a list of current Vendor Unique commands. See the vendor specifications where compatability is desired.
RESERVED (R)	Command codes not currently used are reserved for ANSI definition at a future date.

Command operation codes and names are listed in Table 73.

Table 73
Command Codes
Group 00 Commands

HEX OP Code	Type	Description
00	V	
01	V	
02	V	
03	S	REQUEST SENSE
04	V	
05	V	
06	V	
07	V	
08	O	RECEIVE
09	V	
0A	O	SEND
0B	V	
0C	V	
0D	V	
0E	V	
0F	V	
10	V	
11	V	
12	S	INQUIRY
13	V	
14	V	
15	V	
16	V	
17	V	
18	R	
19	R	
1A	R	
1B	R	
1C	O	RECEIVE DIAGNOSTIC RESULTS
1D	O	SEND DIAGNOSTIC
1E	R	
1F	R	

Table 74
Request Sense Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	1	1
01	**Logical Unit Number**				Reserved			
02	Reserved							
03	Reserved							
04	Number of Bytes Allocated							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

REQUEST SENSE (03) (Standard).

Returns unit sense. The sense data will be valid for the check condition (status) just presented to the **INITIATOR**. This sense data must be preserved in the **TARGET** for the **INITIATOR**. Sense data will be cleared on the reception of any subsequent command to the unit in check from the **INITIATOR** receiving the check condition (see section 6.8 for the sense data format). Byte 04 in this command will specify the number of bytes that the **INITIATOR** has allocated for returned sense. All **INITIATOR**s shall be capable of accepting a minimum of four sense bytes (i.e., a value of 0 through 3 defaults to a four byte allocation).

Check status shall not be returned for this command.

Table 75
Receive Command

BYTE	BIT 7	6	5	4	3	2	1	0
00	0	0	0	0	1	0	0	0
01	**Logical Unit Number**				Reserved			
02	Number of Bytes Allocated							
03	Number of Bytes Allocated							
04	Number of Bytes Allocated							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

RECEIVE (08) (Optional).

Transfers to the **INITIATOR** the specified number of bytes. Bytes 02 to 04 of the CDB specify the number of bytes allocated.

Table 76
Send Command

BYTE	BIT 7	6	5	4	3	2	1	0
00	0	0	0	0	1	0	1	0
01	**Logical Unit Number**				Reserved			
02	Number of Bytes Allocated							
03	Number of Bytes Allocated							
04	Number of Bytes Allocated							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

SEND (0A) (Optional).

Transfers to the **TARGET** the specified number bytes. Bytes 02 to 04 of the CDB specify the number of bytes the **INITIATOR** has to send to the **TARGET.**

Table 77
Inquiry Command

BYTE	BIT 7	6	5	4	3	2	1	0
00	0	0	0	1	0	0	1	0
01	**Logical Unit Number**			Reserved				
02	Reserved							
03	Reserved							
04	Number of Bytes Allocated							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

INQUIRY (12) (Standard).
This command sends to the **INITIATOR** information regarding unit and **TARGET** parameters.

Byte 04 in the CDB of this command will specify the number of bytes that the host has allocated for returned unit data. (0 indicates 256 bytes, and 1 to 255 as indicated).

The following **INQUIRY** Data is sent during the data phase of the command:

Table 78

BYTE	
00	Device Type Code
01	Device Type Qualifier
02	Length of Additional Bytes (N)
The following unit parameter bytes may be repeated N times.	
00 to N-1	Unit Parameter Byte

The device type code (Byte 00) is as follows:

HEX 00 = Direct Access Device
HEX 01 = Sequential Access Device
HEX 02 = Output Only Device
HEX 03 = Processor Device
HEX 04 to 7F are reserved.
HEX 00 to FF are Vendor Unique.

The device type qualifier (Byte 01) is defined as follows:

Bit 7 set to 1 shall be removeable media.
Bit 7 set to 0 shall be fixed media.
Bits 6 to 0 form a seven bit user specified code. This code may be set with switches or by some other means within the **TARGET.** Devices which do not support this feature shall return all zero bits. The value of this feature allows each user to assign unique codes to each device model that is supported on his system. These codes may then be used by self-configuring software to determine what kind of a device is at each **SASI BUS** address. This is specially valuable for systems that support multiple types of removeable media.

If the length of additional bytes (Byte 02) is 0, no unit parameter bytes are returned. All **INITIATORs** shall be capable of accepting a minimum of 3 bytes of **INQUIRY DATA.**

Table 79
Receive Diagnostic Results Command

BYTE	BIT 7	6	5	4	3	2	1	0
00	0	0	0	1	1	1	0	0
01	**Logical Unit Number**			Reserved				
02	Reserved							
03	Data Variable Length Indicator							
04	Data Variable Length Indicator							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

RECEIVE DIAGNOSTIC RESULTS (1C) (Optional).

Sends analysis data to **INITIATOR** after completion of a send diagnostic command. The analysis data is vendor unique with bytes 3 and 4 of the CDB specifying the length of the data.

Table 80
Send Diagnostic Command

BYTE \ BIT	7	6	5	4	3	2	1	0
00	0	0	0	1	1	1	0	1
01	**Logical Unit Number**			Reserved			Dev Ofl	Unt Ofl
02	Reserved							
03	Data Variable Length Indicator							
04	Data Variable Length Indicator							
05	VU	VU	Reserv	Reserv	Reserv	Reserv	Flag	Link

Note: This command allows the operating system to be independent of vendor unique diagnostic functions. The diagnostic software becomes more portable to various operating systems.

SEND DIAGNOSTIC (1D) (Optional).

Sends data to the **TARGET** to specify diagnostic tests for **TARGET** and peripheral units. The analysis data is vendor unique with bytes 3 and 4 of the CDB specifying the length of the data.

Bit 0 of bytes 01 enables diagnostic operations which affect the current tape position or write on the tape.

Bit 1 of byte 01 enables diagnostic operations which may adversely affect **I/O** operations to other units on the same controller.

6.6 Completion Status

Status shall always be stored at the end of a command or set of linked commands. Intermediate status shall be stored at the completion of a linked command. Any abnormal conditions encountered during command execution shall cause command termination and ending status.

6.7 Completion Status Byte

Byte 00

Bit 0 — Vendor unique.

Bit 1 — Check condition. Sense is available. See Request Sense Command.

Bit 2 — Condition met. Will be set to indicate a test condition was met, in particular a **SEARCH DATA** command satisfied its search condition.

Bit 3 — Busy. Device is busy or reserved. Busy status will be stored whenever a target is unable to accept a command from an **INITIATOR.** The most common instance of this condition arises when an **INITIATOR** that does not allow reconnection requests an operation from a reserved or busy device.

Bit 4 — Intermediate status sent. This bit is set for any intermediate status sent during a series of linked commands. This bit will not be set for any ending status.

Bits 5 - 6 — Vendor unique.

Bit 7 — Extended status. This bit set means that the second status byte is valid.

Byte 01

Bit 0 — Host adapter detected error. Reserved for the host adapter H.A. to flag the host for errors not detected by the **TARGET.**

Bits 1 - 6 — Reserved.

Bit 7 — Extended status. This bit, when set, means that the third status byte is valid.

BYTE	BIT 7	6	5	4	3	2	1	0
00	Ext Stat	VU	VU	Int Sta	Busy	Cdt. Met	Check	VU
01	Ext Stat							H.A. Err

6.8 Sense Bytes

The **SENSE BYTES** are retrieved using the **REQUEST SENSE** command. The format of these bytes is determined by the first **SENSE BYTE.** Bits 6 - 4 of this byte specify the Error Class and bits 3 - 0 of this byte specify the Error Code. Error classes 0 - 6 specify the following format:

For Error Class 7, the format is determined by the Error Code bits. The code 0 format **(EXTENDED SENSE BYTES)** is defined below. Codes 1 through 14 are undefined and are reserved for future definition. Code 15 is available for vendor unique **SENSE BYTE** formats.

	BIT 7	6	5	4	3	2	1	0
00	Ad Valid	Error Class (0-6)			Error Code			
01	Reserved				MSB Logical Block Address			
02					Logical Block Address			
03					LSB Logical Block Address			

6.9 Extended Sense Bytes

The **EXTENDED SENSE BYTES** are defined as follows:

BIT	7	6	5	4	3	2	1	0
00	Valid	Class 7			Code 0			
01	Segment Number							
02	File Mk	EOM	ILI	Reserved	Sense Key			
03	Information Byte 1							
04	Information Byte 2							
05	Information Byte 3							
06	Information Byte 4							
07	Additional Sense Length							
08 - NN	Additional Sense Bytes							

The Valid bit (bit 7 of byte 0) indicates that the Information Bytes contain valid device type specific information. For type 0 devices, the Information Bytes specify the Logical Block Address associated with the Sense Key. For type 1 devices, the Information Bytes specify the difference (residue) of the requested length and the actual length in either bytes or blocks, as determined by the command. This field is not defined for other device types or if the Valid bit is not set.

The Segment Number (Byte 1) contains the current Segment Number if the Extended Sense is in response to a Copy command. Up to 256 segments are suported beginning with segment zero.

The Filemark bit (bit 7 of byte 2) indicates that the current **I/O** read a Filemark Block. This bit is only used for type 1 devices.

The End of Media bit (bit 6 of byte 2) indicates that an End of Media (End of Tape, Beginning of Tape, Out of Paper, etc.) has occurred on a sequential device. For type 1 devices, this bit indicates End of Tape if the direction was forward and Beginning of Tape if the direction was reverse. Direct Access devices will NOT use this bit; instead, these devices will report attempts to access beyond the end of media as **ILLEGAL REQUEST** Sense Key (see below).

The Incorrect Length Indicator (Bit 5 of Byte 2) indicates that the requested data transfer did not match the length of data on the media.

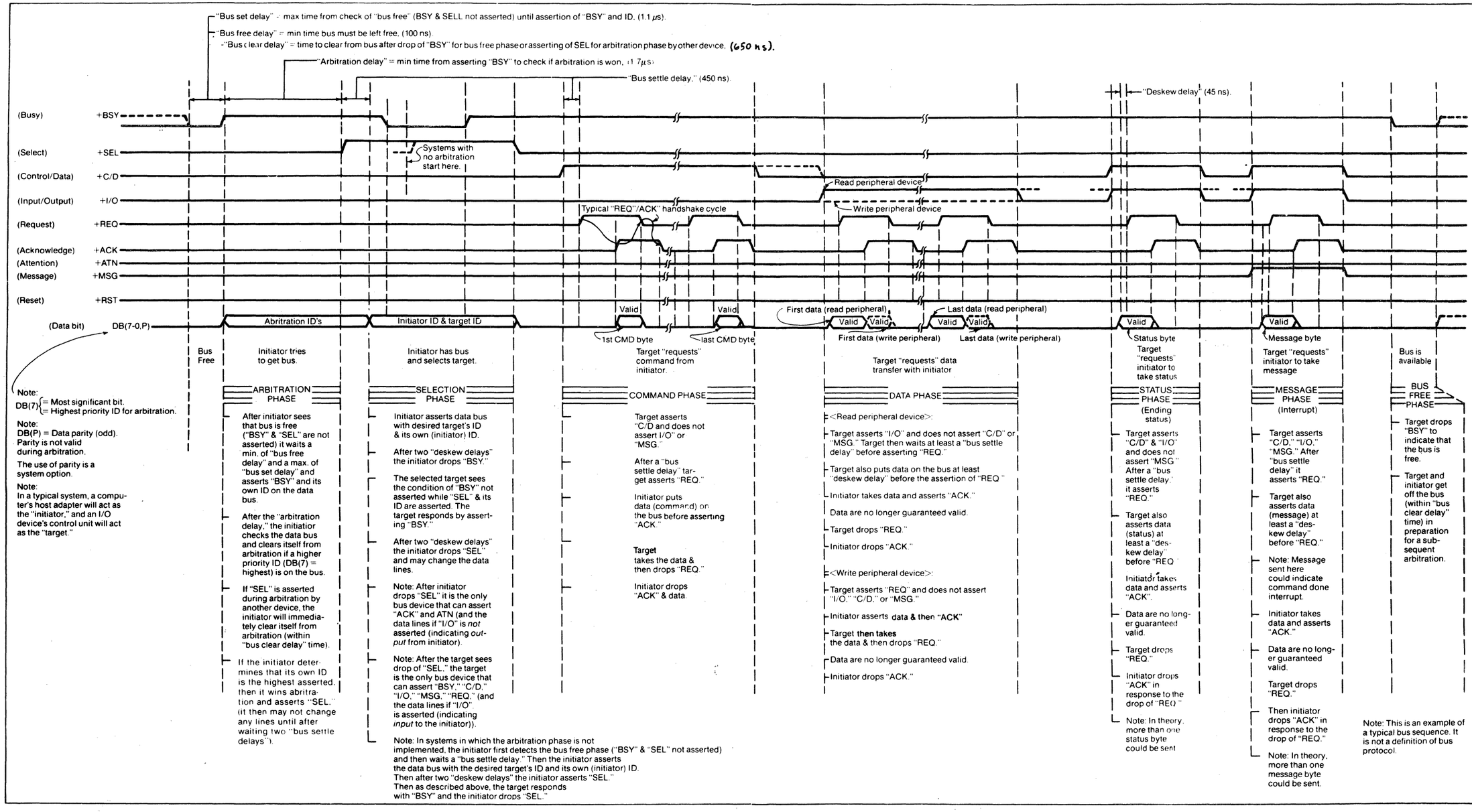
The Sense Key is defined as follows:

Sense Key	Meaning
0	**NO SENSE.** There is no sense information currently locked for the designated unit. This means that the last **I/O** from the current host to the addressed **LUN** completed with a Status that had the Check Condition bit off. This would be the case for a Good **I/O.**
1	**RECOVERABLE ERROR.** The last **I/O** completed successfully with some recovery action required. Details can be determined by examining the Additional Sense bytes and the Logical Block Address field.
2	**NOT READY.** The addressed unit cannot be accessed. Operator intervention may be required to correct this condition.
3	**MEDIA ERROR.** The **I/O** completed with a Non-recoverable error condition which was probably caused by a flaw on the media.
4	**HARDWARE ERROR.** This condition indicates that the **TARGET** detected a Non-recoverable hardware failure while attempting to perform the command.
5	**ILLEGAL REQUEST.** This condition 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.).
6	**MEDIA CHANGE.** This status indicates that the removeable media may have been changed since the last command was issued to the addressed unit. This status will be reported the FIRST time that a command which reads or writes on the media is issued after the condition is detected and the requested command will not be performed. This condition is cleared on the next **I/O** from the same host. Note: The **MEDIA CHANGE** condition MUST be reported to ALL hosts in a multi-host system.
7	**WRITE PROTECT.** This condition indicates that a command which writes on the media was attempted on a "write protected" unit. The write was not performed.

Sense Key	Meaning
8	**DIAGNOSTIC UNIQUE.** This condition is provided to report diagnostic unique conditions that should be handled by diagnostic programs instead of the Operating System. This condition should only be used with diagnostic commands. If this status occurs on a non-diagnostic command, the Operating System should treat it the same as **HARDWARE ERROR.**
9	**VENDOR UNIQUE.** This condition is provided to report vendor unique conditions.
A	**POWER UP FAILED.** This condition indicates that the Bus Device has failed its Power-Up self-test. This condition should be reported to ALL hosts.
B	**ABORTED COMMAND.** The **TARGET** aborted the command. The Operating System may be able to recover by trying the command again.
C	**EQUAL.** Will be sent when search data equal is satisfied.
D	**VOLUME OVERFLOW.** This condition indicates that a buffered divice has reached the End of Media and data remains in the buffer which has not been written to the media. Recover Buffered Data command(s) may be issued to read the unwritten data from the buffer.
E-F	These sense keys are reserved

The Additional Sense Length (byte 7) specifies the number of Additional Sense Bytes. The Additional Sense Bytes contain device specific data that further define the nature of the Check Condition.

APPENDIX A: SHUGART ASSOCIATES SYSTEM INTERFACE
"This Appendix is included for informational purposes only and is not a part of the **SCSI** Standard."



SASI TIMING CHART

Appendix - A. Timing Sequence Example
(Command without Disconnect/Reselect)

APPENDIX B:

"This Appendix is included for informational purposes only and is not a part of the SASI Standard."

(To be completed.)

APPENDIX C: Typical Bus Phase Sequence

"This Appendix is included for informational purposes only and is not a part of the **SASI** Standard."

Table C
Read Data Transfer With No Disconnect

	Signals											
Bus Phase	**BSY**	**SEL**	**ATN**	**MSG**	**C/D**	**I/O**	**REQ**	**ACK**	**RST**	**DB (7-0)**	**DB (P)**	**Comment**
Bus Free	o	o	o	o	o	o	o	o	o	o	o	**BUS** is available.
Arbitration	1	o	o	o	o	o	o	o	o	ID	X	**INITIATOR** tries to to get **BUS.**
"		1										
Selection	1	1	1	o	o	o	o	0	o	$ID_{I,T}$	V	**INITIATOR** has **BUS** and selects **TARGET.** **ATN** is on.
"	o	1								$ID_{I,T}$	V	
"	1	1								$ID_{I,T}$	V	
"	1	o								X	X	
Message Out	1	o	1	1	1	0	0	0	o	X	X	**TARGET** has control of **BUS** and gets **IDENTIFY** message from **INITIATOR.**
"			1				1	0		X	X	
"			0				1	0		X	X	
"			0				1	1		Message	V	
"			0				0	1		X	X	
"			0				0	0		X	X	
Command	1	o	0	0	1	0	0	0	o	X	X	**TARGET** gets command from **INITIATOR.** (This phase is repeated for each byte.)
"							1	0		X	X	
"							1	1		Command	V	
"							0	1		X	X	
"							0	0		X	X	

Table C (continued)
Read Data Transfer With No Disconnect

	Signals											
Bus Phase	**BSY**	**SEL**	**ATN**	**MSG**	**C/D**	**I/O**	**REQ**	**ACK**	**RST**	**DB (7-0)**	**DB (P)**	**Comment**
Data In	1	o	0	0	0	1	0	0	o	X	X	**TARGET**
"							1	0		Read Data	V	sends data
"							1	1		X	X	to **INITIATOR.**
"							0	1		X	X	(This phase
"							0	0		X	X	is repeated for each byte.)
Status	1	o	0	0	1	1	0	0	o	X	X	**TARGET**
"							1	0		Status	V	sends
"							1	1		X	X	status to
"							0	1		X	X	**INITIATOR.**
"							0	0		X	X	
Message In	1	o	0	1	1	1	0	0	o	X	X	**TARGET**
"							1	0		Message	V	sends
"							1	1		X	X	**COMMAND**
"							0	1		X	X	**COMPLETE**
"							0	0		X	X	message to **INITIATOR.**
Bus Free	o	o	o	o	o	o	o	o	o	o	o	**BUS** is available.

o = Signal driver is passive.

0 = Signal is not asserted.

1 = Signal is asserted.

"Blank" = Signal condition is the same as the previous line.

ID = BUS Device ID for arbitration.

$ID_{I,T}$ = **TARGET**'s ID and **INITIATOR**'s ID.

V = Parity is valid.

X = The signals are not guaranteed to be a known state.

APPENDIX D:

"This Appendix is included for informational purposes only and is not a part of the SASI Standard."

(To be completed.)

APPENDIX E:

"This Appendix is included for informational purposes only and is not a part of the SASI Standard."

(To be completed.)

APPENDIX F: Error Codes in Sense Bytes

"This Appendix is included for informational purposes only and is not a part of the **SASI** Standard."

Table F1
Error Codes in Sense Bytes

Class 00 Errors
Drive Errors

ERROR CODE	
00	NO SENSE
01	NO INDEX SIGNAL
02	NO SEEK COMPLETE
03	WRITE FAULT
04	DRIVE NOT READY
05	DRIVE NOT SELECTED
06	NO TRACK 0
07	MULTIPLE DRIVES SELECTED
08	NO ADDRESS ACKNOWLEDGE
09	MEDIA NOT LOADED
0A	INSUFFICIENT CAPACITY
0B	VENDOR UNIQUE
0C	VENDOR UNIQUE
0D	VENDOR UNIQUE
0E	
0F	

Table F2
Error Codes in Sense Bytes

Class 01 Errors
Target Errors

ERROR CODE	
00	I.D. CRC ERROR
01	UNCORRECTABLE DATA ERROR
02	I.D. ADDRESS MARK NOT FOUND
03	DATA ADDRESS MARK NOT FOUND
04	RECORD NOT FOUND
05	SEEK ERROR
06	DMA TIMEOUT ERROR
07	WRITE PROTECTED
08	CORRECTABLE DATA CHECK
09	BAD BLOCK FOUND
0A	INTERLEAVE ERROR
0B	DATA TRANSFER INCOMPLETE
0C	VENDOR UNIQUE
0D	VENDOR UNIQUE
0E	VENDOR UNIQUE
0F	

Table F3
Error Codes in Sense Bytes

Class 02 Errors
System Related Errors

ERROR CODE	
00	INVALID COMMAND
01	ILLEGAL BLOCK ADDRESS
02	VENDOR UNIQUE
03	VENDOR UNIQUE
04	VENDOR UNIQUE
05	
06	
07	
08	
09	
0A	
0B	
0C	
0D	
0E	
0F	

Table F4
Error Codes in Sense Bytes

Class 07 Errors
Extended Sense

ERROR CODE	
00	EXTENDED SENSE
01	
02	
03	
04	
05	
06	
07	
08	
09	
0A	
0B	
0C	
0D	
0E	
0F	VENDOR UNIQUE SENSE

APPENDIX G: Commands Currently in Use

"This Appendix is included for informational purposes only and is not a part of the **SASI** Standard."

The Shugart Associates Systems Interface (SASI) is an ANSI X3T9.2 project with the objective to inter-connect multiple small computers and peripherals. The **SASI** is oriented to a future environment of distributed logic which is device-independent.

The original working document for the **SASI** was based on the Shugart Associates System Interface (SASI). The **SASI** is currently in use by a number of manufacturers and there was a strong industry desire to enhance and extend the **SASI.** Different manufacturers were beginning to evolve from the **SASI** in different ways, and this was the root of the desire for an ANSI interface, so that all could grow to a common goal.

At this time (circa 1982) there are a number of different **SASI** implementations which have many device-dependent characteristics embodied therein. Even though the functional differences affect interchangeability there is a high degree of hardware compatibility.

It is the intention of X3T9.2 to ensure that the current implementations of the **SASI** will form a viable, compatible subset of the **SASI** functions as device-dependencies move towards device-independence.

The addendums in this appendix represents a composite of all commands known to be currently in use by different manufacturers (presented in the same form as Table 2 in Section 6.3) and descriptions provided by **SASI** manufacturers of their command repertoires.

Addendum:
- I Summary of Commands in Use (To be completed)
- II Cynthia Peripherals (See: ANSI X3T9.2/82-11)
- III DTC (See: ANSI X3T9.2/82-12)
- IV OMTI (See ANSI X3T9.2/82-13)