Working Draft

X3T10 992D

Revision 22A
Sept 19, 1996

Information Technology - SCSI-3 Serial Bus Protocol (SBP)

ASC X3T10 Technical Editor:

Ronald K. Roberts
Sierra-Pac Technology
PO Box 2389
Shingle Springs, CA 95682

TEL: (916) 677-5714
FAX: (916) 677-1218
rkroberts@aol.com

Contributing Editors:

Gerald A. Marazas
IBM PC Company
3039 Cornwallis Road
Research Triangle Park, NC 27709

(919) 543-6892
(919) 543-2219
marazas@bcrvmpc2.vnet.ibm.com

Scott Smyers
Sony Research Labs
3300 Zanker Rd
San Jose, CA 95134
Mail Stop SJ3D3

(408) 955-4573
(408) 955-5180
scotts@lsi.sel.sony.com

Reference number
ISO/IEC ***** : 199x
ANSI X3.268 -1995
Printed Sept 19, 1996

X3T10/992D revision 22

Other Points of Contact:

	X3T10 Chair	X3T10 Vice-Chair
	John B. Lohmeyer	Lawrence J. Lamers
	SYMBIOS Logic, Inc	Adaptec
	4420 ArrowsWest Drive	691 South Milpitas Blvd.
	Colorado Springs, CO 80907-3444	Milpitas, CA 95035
Voice:	(719) 533-7560	(408) 957-7817
Fax:	(719) 533-7036	(408) 957-7193
Email:	john.lohmeyer@ symbios.com	ljlamers@adaptec.com

X3 Secretariat
Lynn Barra
Administrator Standards Processing
X3 Secretariat
1250 Eye Street, NW Suite 200
Washington, DC 20005

Telephone: 202-626-5738
Facsimile: 202-638-4922
E-mail: LBARRA@ITIC.NW.DC.US

SCSI Reflector

Internet address for subscription to the SCSI reflector: majordomo@symbios.com
Internet address for distribution via SCSI reflector: scsi@symbios.com

SCSI Bulletin Board
719-533-7950

IEEE 1394 Reflector

Internet address for subscription: bob.snively@eng.sun.com
Internet address for distribution: IEEE 1394@scsi.eng.sun.com

Document Distribution
Global Engineering
15 Inverness Way East
Englewood, CO 80112-5704

Telephone: 303-792-2181 or
800-854-7179
Facsimile: 303-792-2192

ABSTRACT

This document describes a command and status delivery protocol for controlling the operation of devices attached to an IEEE 1394 Serial Bus physical interface. This protocol is based on a mapping of Parallel SCSI commands and messages to the IEEE 1394 environment.

PATENT STATEMENT

CAUTION: The developers of this standard have requested that holder's of patents that may be required for the implementation of the standard, disclose such patents to the publisher. However, neither the developers nor the publisher have undertaken a patent search in order to identify which, if any, patents may apply to this standard.

As of the date of publication of this standard and following calls for the identification of patents that may be required for the implementation of the standard, no such claims have been made. No further patent search is conducted by the developer or the publisher in respect to any standard it processes. No representation is made or implied that licenses are not required to avoid infringement in the use of this standard.

Revision History

Rev 17 vs Rev 18

Rev 18 has revised or deleted the following clauses:

Annex B.(Rev 17) Device drive considerations (informative) has been removed.

Annex D. (Rev 17) Examples of CDS delivery (informative) has been removed.

Remaining annex's have been re-numbered and re-ordered

Clause 5 has been modified for clarification's and comment response

Clauses 6 and 7 have been rewritten based on comments received from reviews of revision 16 and 17.

Clause 10 has been modified to incorporate isochronous transfer controls, also identifies rules for handling CDB's based on comments received.

Annex C has had additional information added.

Rev 18 vs Rev 19

Revision 19 is 1st public review revision

Revision 19B contains all changes known as of July 7, 1995. Includes responses to public review comments and ANSI editor requested changes.

Revision 20 contains changes to implement Isochronous transfers. Clause 10 and the Annexes were modified by the following contributing editors:

Peter Johansson	Dick Scheel
Congruent Software	Sony Research Labs

Revision 21 resulted from the Sept. 18, 1995 editors meeting. No major technical or editorial changes were implemented in the normative clauses. However, clarification of the Isochronous clauses were included in 5.3.2. and sub-clause 10.5. Other changes included updating the Forward and Introduction clauses to meet X3 standards.

Revision 22 is 2nd public review revision

Revision 22A contains new Clause 2 Normative Refeences and address updates on pageii.

Contents

Revision History iii
Contents iv
Forward ix
Introduction x
1. Scope 1
2. Normative references 2
2.1. Approved References 2
2.2. References under development 2
2.3. Other references 2
3. Definitions, symbols and abbreviations 3
3.1 Definitions 3
3.2 Symbols and Abbreviations 4
4. General 4
4.1. Conventions 5
5. Model of serial bus protocol 7
5.1 IEEE 1394 MEMORY MAP and ADDRESS SPACE 7
5.1.1 Configuration ROM 7
5.1.2 Command FIFO 7
5.2 Command Data Structures (CDS) 7
5.2.1 TAP slots 8
5.2.2 CDS chains 8
5.2.3 CDS movement 8
5.2.4 Task sets 9
5.2.5 Time stamp 9
5.3 Data transfers 10
5.3.1 Asynchronous transfers 10
5.3.2 Isochronous transfers 10
5.4 Auto sense 11
5.5 Asynchronous event notification 11
5.6 Exception handling 11
6. SCSI-3 Architecture model services 11
6.1 Command execution services 11
6.1.1 Tap protocol 12
6.1.2 Fetch protocol 12
6.2 Data transfer services 12
6.2.1 Data transfer from target to destination 12
6.2.2 Data transfer from source to target 12
6.3 Task management services 12
6.3.1 Abort task 13
6.3.2 Abort task set 13
6.3.3 Clear task set 13
6.3.4 Clear auto contingent allegiance 14
6.3.5 Target reset 14
6.3.6 Terminate task 14
6.4 Exception and error processing 15
6.4.1 Tap transfer exceptions 15
6.4.2 Fetch transfer exceptions 15
6.4.3 Asynchronous data transfer exceptions 15
6.4.4 Isochronous data transfer exceptions 15
7.0 SBP Transaction management 15
7.1 Transaction services 15
7.1.1 Transaction data request (TR_DATA.request) 16
7.1.2 Transaction data confirmation (TR_DATA.confirmation) 16

7.1.3 Transaction data indication (TR_DATA.indication) 16
7.1.4 Transaction data response (TR_DATA.response) 16
7.2 Tap transaction 16
7.3 Fetch transaction 17
7.4 Asynchronous data transfer 18
7.4.1 Application data transfer target to destination / from source 18
7.4.2 Login data transfer from target to initiator 19
7.4.3 Status data transfer from target to initiator 19
7.5 Isochronous data transfers 20
7.5.1 Link layer services 20
7.5.2 Isochronous status reporting 20
7.5.3 Ignore and continue 21
7.5.4 Report and continue 21
7.5.5 Report and stop 22
8.0 Target memory 23
8.1 Configuration ROM 23
8.2 Asynchronous FIFOs 25
8.2.1 Asynchronous login FIFO 25
8.2.2 Asynchronous normal FIFO 25
8.2.3 Asynchronous urgent FIFO 25
8.2.4 Asynchronous ACA FIFO 25
8.3 Isochronous FIFO's 26
8.3.1 Isochronous login FIFO 26
8.3.2 Isochronous normal FIFO 26
8.3.3 Isochronous control FIFO 26
8.3.4 Isochronous ACA FIFO 26
9. Initiator memory 27
9.1 Status FIFO 27
9.2 CDS sense buffer 28
9.3 Asynchronous event buffer 30
10. Command data structures 31
10.1 General CDS format 31
10.1.1 CDS codes field 32
10.1.2 Data transfer control field 33
10.2 SCSI CDS FORMATS 33
10.2.1 SCSI CDS 33
10.2.2 Isochronous SCSI CDS 37
10.3 LOGIN CDS 38
10.4 MANAGEMENT CDS 40
10.5 Isochronous control CDS 41
11.0 Control procedures 44
11.1 Login procedure 44
11.1.1 Asynchronous login procedure 44
11.1.2 Isochronous login procedure 45
11.1.3 Tap slot allocation procedure 45
11.2 Logout procedure 46
11.2.1 Asynchronous logout 46
11.2.2 Isochronous logout 46
11.3 Sign-in procedure 47
Annex A (Normative) Isochronous transfers 48
A.1 Isochronous transfers 48
A.1.1 Packet boundary location 48
A.2 Isochronous Storage Formats 48
A.2.1 Packet format 48
A.2.2 NULL packet format 50
A.2.3 CYCLE_MARK packet format 50

Annex B (Informative) Isochronous Transfer Pseudo Code 52
Annex C (Informative) Configuration ROM 53
C.1 Scope 53
C.1.1 First quadlet 55
C.1.2 Bus information block 55
C.1.3 Root directory 55
C.1.4 Node unique ID leaf 56
C.1.5 Unit directory 56
C.1.6 Unit dependent directory 56

Figures

FIGURE 1 SCOPE OF SCSI-3 STANDARDS x
FIGURE 2 BLOCK DIAGRAM OF SERIAL SCSI DEVICE 23
FIGURE A.1 - DATA PACKET STORAGE FORMAT 49
FIGURE A.2 NULL PACKET FORMAT 50
FIGURE A.3 - CYCLE_START PACKET FORMAT 51
FIGURE B.1 - PSEUDO CODE FOR RECORDING 52
FIGURE B.2 - PSEUDO CODE FOR PLAYBACK 52
FIGURE C.1 - CONFIGURATION ROM LAYOUT 53

Tables

Table 1 - Bit ordering in a byte 6
Table 2 - Byte ordering within a quadlet 6
Table 3 Error log entry 22
Table 4 Isochronous status codes 22
Table 5 Unit Directory 24
Table 6 Unit Dependent Directory 24
Table 7 Status block 27
Table 8 CDS status 27
Table 9 SBP status 28
Table 10 CDS sense data format 29
Table 11 Status quadlet 29
Table 12 Valid flags field 29
Table 13 Command data structure format 31
Table 14 CDS codes field 32
Table 15 CDS identifier 32
Table 16 Tap type 32
Table 17 CDS type field 33
Table 18 Data transfer control field 33
Table 19 Data transfer rate 33
Table 20 SCSI CDS 34
Table 21 Task codes field 34
Table 22 Task attribute 34
Table 23 Protocol flags field 35
Table 24 Scatter/gather list 36
Table 25 ISOCHRONOUS SCSI CDS 37
Table 26 Protocol flags field 37
Table 27 LOGIN CDS 38
Table 28 Login flags field 38
Table 29 LOGIN data format 39
Table 30 Management CDS 40
Table 31 Management flags field 40
Table 32 SBP flags field 40
Table 33 ISOCHRONOUS CONTROL CDS 41
Table 34 Control Function Field 41
Table 35 Error Reporting Field 42
Table 36 Stream Event Field 43
Table 37 Channel Mask Field 44
Table 38 Channel Map Field 44
Table A.1 type field values 48
Table A.2 Tag field encoding 50
Table C.1 - Example of SBP configuration ROM entries 54

Forward

The SCSI-3 Serial Bus Protocol standard document is organized in the following clauses.

Clause 1 defines the scope of the Serial Bus Protocol
Clause 2 specifies the normative references
Clause 3 defines the definitions, symbols and abbreviations
Clause 4 contains general information
Clause 5 contains the model of serial bus protocol
Clause 6 defines the SCSI-3 services
Clause 7 defines the SBP transaction management
Clause 8 defines the target memory and command FIFO's
Clause 9 defines the initiator memory and status blocks
Clause 10 defines the command data structures
Clause 11 defines the control procedures
Annexes A, B, are normative
Annex C is informative

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

This standard was processed and approved for submittal to ANSI by Accredited Standards Committee on Information Process Systems, X3. Committee approval of the standard does not necessarily imply that all committee members voted for approval.

At the time of approval of this standard the X3 committee had the following members.

Introduction

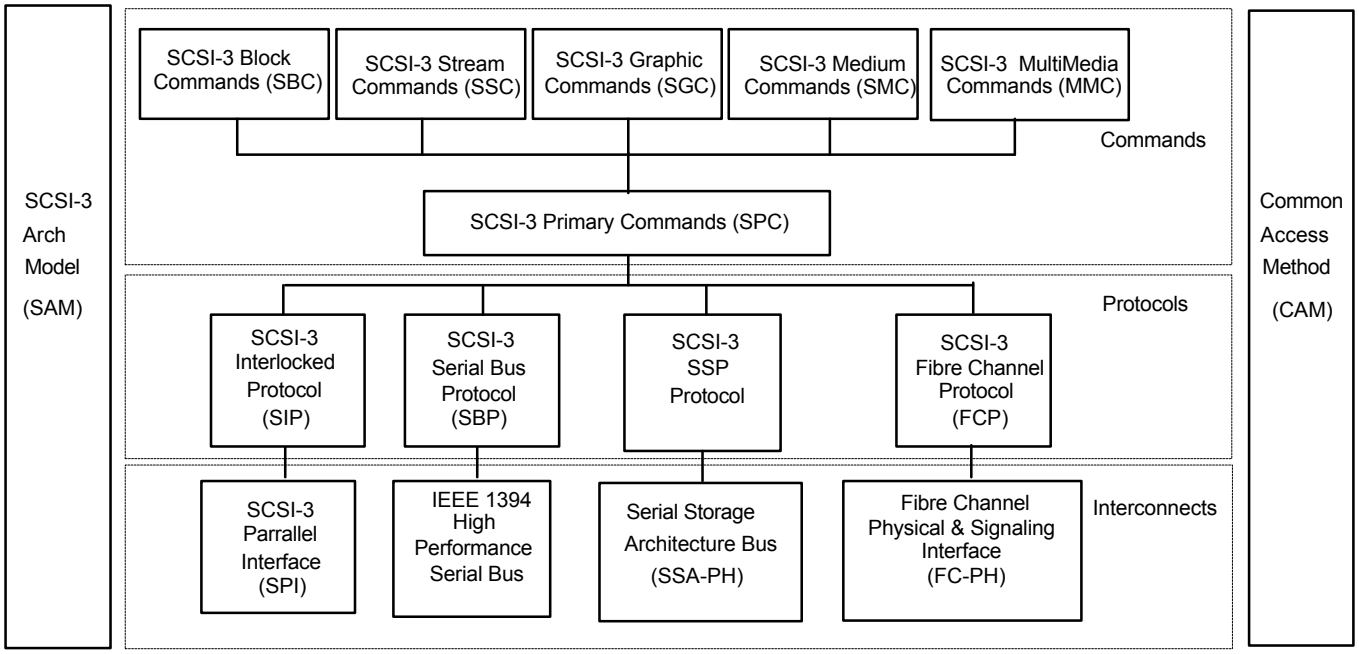
A major goal of this protocol is to define a model acceptable to device vendors, looking for an evolution from parallel SCSI, and systems designers looking for opportunities to more fully exploit the capabilities inherent to the IEEE 1394 High Performance Serial Bus.

In striking a balance between support for existing parallel SCSI functions, and providing a growth path to exploit the new features of the serial SCSI environment, a new fixed length CDS has been defined. This CDS provides support for advanced features of today's SCSI such as automatic sense data reporting and scatter-gather host data buffer handling. Support is provided for the interesting new features of the IEEE 1394 High Performance Serial Bus such as isochronous data transfer. SBP does not require atomic or "locked" memory operations for list pointer update or management.

This protocol has been developed in conjunction with the development of an IEEE 1394 High Performance Serial Bus, and although this protocol is intended to reside on a physical interface not covered in the family of standards outlined below, it shall follow the architecture defined in the SCSI-3 Architecture Model (SAM).

Figure 1 is intended to show the relationship of this document to other SCSI standards. The figure is not intended to imply a relationship such as a hierarchy, protocol stack, or system architecture. It indicates the applicability of a standard to the implementation of a given transport.

Figure 1 Scope of SCSI-3 Standards



The term SCSI is used wherever it is not necessary to distinguish between the versions of SCSI. The original Small Computer System Interface Standard, X3.131-1986, is referred to herein as SCSI-1. SCSI-1 was revised resulting in the Small Computer System Interface - 2, X3.131-1994, referred to herein as SCSI-2 and approved by ANSI on January 31, 1994.

The term SCSI-3 refers collectively to the following documents that fall under the jurisdiction of X3T10 (formerly X3T9.2)

SCSI-3	Parallel Interface (SPI)	X3.253-1995
SCSI-3	Interlocked Protocol (SIP)	X3T10/856D
SCSI-3	Fiber Channel Protocol (FCP)	X3.269-1995
SCSI-3	Serial Bus Protocol (SBP)	X3.268-1995
SCSI-3	Generic Packetized Protocol (GPP)	X3T10/991D
SCSI-3	Architecture Model (SAM)	X3.270-1995
SCSI-3	Primary Commands (SPC)	X3T10/995D
SCSI-3	Block Commands (SBC)	X3T10/996D
SCSI-3	Stream Commands (SSC)	X3T10/997D
SCSI-3	Graphic Commands (SGC)	X3T10/998D
SCSI-3	Medium Changer Commands (SMC)	X3T10/999D
SCSI-3	SCSI-3 Multimedia Command Set(MMC)	X3T10/1048D

With any technical document there may arise questions of interpretation as new products are implemented. The X3 Committee has established procedures to issue technical opinions concerning the standards developed by the X3 organization. These procedures may result in SCSI Technical Information Bulletins being published by X3.

Any such bulletins, while reflecting the opinion of the Technical Committee that developed the standard, are intended solely as supplementary information to other users of the standard. This standard, as approved through the publication and voting procedures of the American National Standards Institute, is not altered by these bulletins. Any subsequent revision to this standard may or may not reflect the contents of any such Technical Information Bulletins

Current X3 practice is to make Technical Information Bulletins available through:

Global Engineering
15 Inverness Way East
Englewood, CO 80112-5704

Telephone: 303-792-2181 or
800-854-7179
Facsimile: 303-792-2192

Information Processing Systems - SCSI-3 Serial Bus Protocol

1. Scope

This document defines a protocol to deliver SCSI command, data, and status over an IEEE 1394 High-Performance Serial Bus physical interface. This document conforms to the SCSI-3 Architectural Model (SAM) X3.270 and the ISO/IEC 13213:1994 Control and Status Registers Control and Status Registers standards. All devices implementing SBP shall meet the mandatory requirements of IEEE 1394 High Performance Serial Bus, ISO/IEC 13213:1994 Control and Status Registers, and SAM.

An element of SBP is that SCSI CDB's are contained in a larger data structure called a CDS. In addition to the CDB, the CDS contains additional information, such as that used to link a sequence of CDS's into a chain and pointers into initiator memory for various other operations.

The protocol described in this document has the following functional capabilities:

1. Allows the initiator to form a chain of commands without limitation of the queue depth in a target;
2. Target does not need to interrupt the processing of one prior command in order to queue a subsequent command;
3. Ability to design target devices that have varying levels of support for command queuing or command processing without affecting the design of the initiator;
4. Distribute the DMA context handling from the initiator adapter to the targets to minimize device and adapter complexity for higher subsystem performance.

This protocol has the following elements to ease the transition to a Serial SCSI:

1. Integrates SCSI command descriptor blocks (CDB) within the SBP command data structure (CDS);
2. Follows the SCSI-3 Architectural Model.

2. Normative references

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

Copies of the following documents can be obtained from ANSI: Approved ANSI standards, approved and draft international and regional standards (ISO, IEC, CEN/CENELEC, ITUT), and approved and draft foreign standards (including BSI, JIS, and DIN). For further information, contact ANSI Customer Service Department at 212-642-4900 (phone), 212-302-1286 (fax) or via the World Wide Web at http://www.ansi.org.

Additional availability contact information is provided below as needed.

2.1. Approved References

The following approved ANSI standards, approved international and regional standards (ISO, IEC, CEN/CENELEC, ITUT), may be obtained from the international and regional organizations who control them.

American National Standard X3.131-1994-Small Computer System Interface-2

ISO/IEC 13213:1994 Control and Status Registers

IEEE1394/Draft 8.0v2, IEEE Standard for a High Performance Serial Bus

2.2. References under development

At the time of publication, the following referenced standards were still under development. For information on the current status of the document, or regarding availability, contact the relevant standards body or other organization as indicated.

SCSI-3 Fiber Channel Protocol (FCP) [X3.269]
SCSI-3 Architecture Model (SAM) [X3.270]
SCSI-3 Primary Commands (SPC) [X3T10/0995-D]
SCSI-3 Block Commands (SBC) [X3T10/0996-D]
SCSI-3 Stream Commands (SSC) [X3T10/0997-D]
SCSI-3 Medium Changer Commands (SMC) [X3T10/0999-D]

For more information on the current status of the above documents, contact (Secretariat). To obtain copies of these documents, contact Global Engineering or (Secretariat).

2.3. Other references

None

3. Definitions, symbols and abbreviations

3.1 Definitions

3.1.1 acquire: The act of transferring a CDS from an initiator to a target.

3.1.2 asynchronous ACA FIFO: An address to which a CDS is sent during the auto-contingent allegiance condition to deliver ACA tagged commands.

3.1.3 asynchronous data transfer: A data transfer mode of the High Performance Serial Bus that is characterized by unscheduled, acknowledged delivery of data packets over the IEEE 1394 bus.

3.1.4 asynchronous login FIFO: An address to which a login CDS is sent during the asynchronous login procedure.

3.1.5 asynchronous normal FIFO: An address to which a CDS is sent that the initiator expects to have a normal execution.

3.1.6 asynchronous urgent FIFO: An address to which a CDS is sent that the initiator expects to have priority execution.

3.1.7 baseline CDS: The first 64 bytes of a CDS.

3.1.8 chain: A list of CDS's connected together via a forward pointer from one CDS to the next following CDS.

3.1.9 command data structure (CDS): A structure that an initiator creates to describe an operation to be performed by a target.

3.1.10 command FIFO: A write only data structure in the target at a fixed address within the target's address space. Initiators write CDS's to the address of a command FIFO as part of the command delivery protocol.

3.1.11 cooperative multi-initiator environment: A collection of initiators such that each and every one of them restrict their usage of Tap Slots at a given target to no more than the number of Tap Slots allocated to them by that target at login time.

3.1.12 displacement: The offset of a bit within a field. For example, a displacement of zero indicates the leftmost or most significant bit of a field.

3.1.13 enter policy: The rules governing the entering of commands to a task set.

3.1.14 extended CDS: The portion of the CDS following the first 64 bytes.

3.1.15 fetch: A read operation by a target to acquire a CDS.

3.1.16 initiator identifier: An 8-bit value assigned by the login procedure. See the SCSI-3 Architectural Model for definition.

3.1.17 isochronous ACA FIFO: An address to which a CDS is sent during the auto-contingent allegiance condition to deliver ACA tagged commands.

3.1.18 isochronous data transfer: A data transfer mode of the High Performance Serial Bus that is characterized by scheduled, non-acknowledged delivery of data packets over the bus.

3.1.19 isochronous login FIFO: An address to which a login CDS is sent during the isochronous login procedure.

3.1.20 isochronous normal FIFO: An address to which an isochronous CDS is sent that the initiator expects to have normal execution.

3.1.21 isochronous control FIFO: The destination address for any isochronous control CDS.

3.1.22 logical unit number: The contents of a field in the CDS that identifies logical units in the target. See the SCSI-3 Architectural Model X3T10/994D for definition.

3.1.23 login: The procedure by which an initiator obtains the addresses of the appropriate FIFO's and an identifier.

3.1.24 node identifier: The IEEE 1394 node_ID.

3.1.25 sign-in: A procedure by which an initiator requests notification of asynchronous events.

3.1.26 status block: A data structure returned by the target that contains status information for a command.

3.1.27 status FIFO: An address in the initiator to which the target returns the status block.

3.1.28 stream identifier: An 8-bit value assigned by the isochronous login procedure.

3.1.29 tag value: The 64-bit address of a CDS. See the SCSI-3 Architectural Model for definition.

3.1.30 tap: A write operation by an initiator to transfer the baseline portion of the first CDS of a chain.

3.1.31 tap slot: A resource within a target used to hold the baseline portion of a CDS.

3.1.32 target identifier: The 64-bit address of the unit dependent directory. See the SCSI-3 Architectural Model for definition.

3.1.33 task identifier: The entity used to identify a task for a task management function. See the SCSI-3 Architectural Model for definition.

3.2 Symbols and Abbreviations

lsq	-	least significant quadlet
msq	-	most significant quadlet
isoch	-	Isochronous
async	-	asynchronous
CDS	-	command data structure
ACA	-	Auto Contingent Allegiance

4. General

SBP cannot operate until the IEEE 1394 bus is initialized. Following the bus initialization, the transaction layer services of IEEE 1394 are used to transfer commands, data, and status.

SBP requires the implementation of the general ROM format in each device as specified in ISO/IEC 13213:1994 Control and status registers for the configuration ROM.

Upon successful completion of initialization of the IEEE 1394 bus, each node will have a unique node identifier. IEEE 1394 hardware provides means to detect the physical removal or addition of a node (i.e., not power on or off). In either case, appropriate bus initialization occurs, and this may involve a given node obtaining new, unique node identifier, possibly different from the node identifier obtained during a prior initialization.

After IEEE 1394 bus initialization, the initiator reads the configuration ROM of each node looking for nodes implementing SBP. The configuration ROM contains a data structure, which when parsed, describes the existence of one or more units, each of which may implement SBP or some other protocol.

The configuration ROM provides information to construct a world-wide unique 64-bit address for the node. This provides the means to maintain a system wide knowledge of each device as a unique entity on the bus in the face of different node identifier values obtained from one bus initialization to the next.

The configuration ROM also provides the addresses of the login FIFO's for each of the supported types of operation (asynchronous and/or isochronous).

After the initiator has parsed the configuration ROM and determined that the target supports SBP, the Serial Bus Protocol (SBP) can now be initiated. The first step is to perform a login procedure.

A login procedure (see clause 11) can either be for isochronous or asynchronous data transfers. A login procedure is performed prior to the transfer of a command data structure (CDS) to the target other than the login CDS. SBP defines the format of all CDS's.

The CDS (see clause 10) is the construct that is used for the delivery of command and control information from the initiator to the target. A CDS may exist in a list called a chain. The initiator writes the first CDS in a chain to a command FIFO using the tap protocol (see clause 6.1.1). The target fetches the second and subsequent CDS's in a chain using the fetch protocol (see clause 6.1.2).

The CDS contains information that the target uses to return status for that CDS. The CDS also contains information that the target uses to return sense data to the initiator if an exception occurs.

All exception handling (isochronous and asynchronous) and the asynchronous command task set management rules are defined in the SCSI-3 Architecture Model (SAM) X3.270-1995.

Target implementation determines the number of taps that can be accepted before a reject occurs. SBP provides management facilities by which cooperating initiators and targets can eliminate the possibility of a tap being rejected by a target. Sub-clause 11.1.3 defines the tap slot allocation procedure.

Initiators that want to receive notification of asynchronous events can sign in with a target and thereby provide information by which the target can inform the initiator of the asynchronous event. Sub-clause 11.3 defines the sign-in procedure.

4.1. Conventions

Certain words and terms used in this standard have a specific meaning beyond the normal English meaning. These words and terms are defined either in the glossary or in the text where they first appear. Lowercase is used for words having the normal English meaning.

Fields containing only one bit are usually referred to as the "named" flag instead of the "named" field.

Data units follow the naming conventions used in IEEE 1394, byte (byte), four bytes (quadlet), eight bytes (octlet). These data units also constitute a continuous stream of bits to which a bit numbering and binary weighting scheme are applied. Within in a collection of continuous bits, the bit having the highest binary weighting is referenced as bit zero. Remaining bits in the stream of bits are referenced in terms of displacement from bit zero.

Numbers that are not immediately followed by lowercase "b" or "h" are decimal values.

Numbers immediately followed by lowercase "b" (xxb) are binary values.

Numbers immediately followed by lowercase "h" (xxh) are hexadecimal values.

Table 1 illustrates the bit ordering used in SCSI and in IEEE 1394 and how they relate to each other.

Table 1 - Bit ordering in a byte

SCSI	7	6	5	4	3	2	1	0
1394	0	1	2	3	4	5	6	7
	msb							lsb

Table 2 illustrates the byte ordering within a quadlet used in SCSI and in IEEE 1394 and the bit significance within the bytes.

Table 2 - Byte ordering within a quadlet

Note 1	most significant byte						least significant byte	
Note 2	byte 0		byte 1		byte 2		byte 3	
Note 3	msb	lsb	msb	lsb	msb	lsb	msb	lsb
Note 4	0	7	8	15	16	23	24	31
Note 5	7	0	7	0	7	0	7	0

NOTES:

1: This row depicts the location within the quadlet of the byte having the highest binary weighting (most significant) and the location of the byte having the lowest binary weighting (least significant).

2: This row depicts the naming convention for bytes within a quadlet.

3: This row depicts the location of the most significant bit and least significant bit within each byte comprising a quadlet.

4: This row depicts the naming convention used by ANSI/IEEE 1212/IEEE 1394 for the individual bits within a quadlet.

5: This row depicts the naming convention used by SCSI for the individual bits within a byte. As there are four bytes in a quadlet this illustrates the application of this convention to each of the bytes.

5. Model of serial bus protocol

This clause describes a model of how the Serial Bus Protocol (SBP) operates on the IEEE 1394 High Performance Serial Bus.

The following description of SBP applies to both asynchronous and isochronous operation, as described in IEEE 1394 High Performance Serial Bus standard. Elements that are common to these two modes of operation are the protocols for command and status delivery, with the key difference between the two modes of operation being in the transport of media data. The differences are cited in the appropriate clauses with in this document.

5.1 IEEE 1394 MEMORY MAP and ADDRESS SPACE

The Serial Bus is in accordance with ISO/IEC 13213:1994 CONTROL AND STATUS REGISTERS for 64-bit fixed addressing, where the upper 16 bits of each address represents the node_ID. This allows address space for up to 64K nodes. The Serial Bus divides the node_ID into two smaller fields: the higher order 10 bits specify a bus_ID and the lower-order bits specify a physical_ID.

5.1.1 Configuration ROM

All IEEE 1394 nodes that implement SBP shall have a configuration ROM. The configuration ROM is read during bus initialization and may be read at anytime thereafter. (See Annex C) IEEE 1394 defines the base address of this ROM and the ISO/IEC 13213:1994 Control and Status Registers defines the format of the information contained in this ROM. Annex C contains an example of a configuration ROM that is compliant with IEEE 1394, ISO/IEC 13213:1994 Control and Status Registers and SBP. Other implementations are possible.

5.1.2 Command FIFO

A key element of the logical model of SBP is the Command FIFO. A Command FIFO is a write only data structure located at a fixed address within the target's address space and is used in support of a logical set of SBP functions (command, control, and sense information). In order for the Initiator to obtain the addresses of the Command FIFO's it shall first perform a Login operation. (See sub-clause 11.1) Processing the Login CDS returns the addresses of requested (asynchronous or isochronous) Command FIFO's.

Although the protocol requires multiple Command FIFO's, it is possible to build a conforming device that maps all FIFO's to a single address. At the other end of the spectrum each login may result in a different address assigned to the normal, urgent, and ACA FIFO's.

5.2 Command Data Structures (CDS)

Another element of SBP is a data structure called a command data structure (CDS) that may carry the SCSI CDB. SBP also defines a number of CDS’s for various control and management functions, which do not carry CDB's. The CDS contains information, such as that used to link a sequence of CDS’s into a chain, initiator identifier, logical unit identifier, and pointers into initiator memory for various other operations.

Initiators write CDS’s to the address of a Command FIFO as part of the command delivery protocol. This operation is called a TAP. Targets may read CDS’s from Initiator memory also as part of the command delivery protocol. This operation is called a FETCH.

The format of the SCSI CDS, contains space for a 16-byte SCSI CDB, and is defined in sub-clause 10.2.1. The SCSI CDS shall only be written to any one of the collection of asynchronous command FIFO's. For SCSI CDS’s written to an asynchronous command FIFO, the carried CDB shall be entered into a task set. The rules governing CDB handling, for CDS’s in a task set, are defined in SAM.

The format of the isochronous SCSI CDS, contains space for a 16 byte SCSI CDB, and is defined in sub-clause 10.2.2. The key distinction of the isochronous SCSI CDS is that it shall only be written to any one of the collection of isochronous command FIFO's. For Isochronous SCSI CDS's written to an isochronous command FIFO, the carried CDB shall not be entered into a task set. The rules governing CDB handling, for CDB's carried in an isochronous SCSI CDS, are defined within this document (SBP).

5.2.1 TAP slots

A TAP slot is a resource (e.g. buffer space in memory) within a target. One or more tap slots may be associated with each Command FIFO. The TAP slot is allocated to an initiator, if requested, during the login procedure. However the initiator is not required to request an allocation of tap slots prior to writing a CDS. During the login procedure, the target returns the number of slots allocated to the initiator in the new tap slot count field of the login data. When the allocation is made the target removes that number of tap slots from the pool of tap slots. In normal usage the target does not enforce the tap slot allocation (i.e., restrict an initiator to use only those tap slots requested). Allocated tap slots will be reused by multiple taps coming from a given initiator. In a cooperative multi-initiator environment, the use of tap slot allocations can eliminate the rejection of a tap. There may also exist a pool of tap slots that have not been allocated. These are available to be used by any initiator on a first-come, first-served basis. The use of a tap slot from the pool is for the duration of a CDS chain. Once the CDS chain has completed, then the tap slot returns to the pool of tap slots not allocated. In a cooperative multi-initiator environment, a target shall never reject a tap.

5.2.2 CDS chains

In certain applications, a high level logical work item results in many physical level I/O commands that originate over such a small interval of time that the multiple commands constitute a block of work to be done by the target device. It becomes a natural as well as an efficient solution to process these commands by forming them into a CDS chain. Individual commands within a CDS chain do not need to have any particular logical relationship to one another. However, in many instances, there is some relation among the commands as viewed by the initiator.

The CDS chain is a data structure that is created by the initiator and maintained within initiator memory space. A wide range of choices is given to the initiator as to the relationship between CDS's in a chain. For example, the same chain may contain CDS's intended for different logical units supported by the given target, or the initiator may organize a chain so that it contains CDS's intended for one of the several logical units implemented at the target. The same initiator may direct different chains to the same logical unit identifier. Every CDS within the same chain is directed to the same target. Asynchronous and isochronous command chains are both options within SBP. Initiators shall not mix asynchronous, isochronous, and task management CDS's within the same chain. An asynchronous chain can support all of the normal SCSI-2, and SCSI-3 commands as well as certain SBP-defined unit-management functions. Some CDS formats have restrictions about how they can exist in a chain. Consult the CDS format definitions in this standard for details.

5.2.3 CDS movement

There are two methods of moving a CDS from an initiator to a target. One method is as a result of unsolicited delivery (of a CDS) which uses the tap protocol. (See 5. 2.1, TAP Slots). A TAP is a 64-byte IEEE 1394 write request transaction, originated by an initiator. The other method of moving a CDS is a FETCH, which is a read request transaction, originated by a target, for the purpose of obtaining all or part of a CDS. In all cases the initiator shall not modify the CDS (with the exception of the abort bit) until the target returns status for that CDS.

There are three states defined in the movement of a CDS from an initiator to a target. In the first state, the CDS exists only in initiator memory and the target has no knowledge of it. In the second state, the CDS has been acquired by the target, via a tap or fetch, and a copy of the CDS exists in both the initiator and the target. The third state has multiple definitions depending on the content of the CDS.

A SCSI CDS, which carry CDB's, enters the third state when the command (SCSI CDB) contained in the CDS, is entered into a task set. Task management or other SBP control CDS's (CDS's that do not carry a CDB) enter the third state when the target schedules the time at which it processes the information carried by the CDS. Isochronous SCSI CDS's, which carries a CDB but is directed to an isochronous command FIFO, enters the third state when the target processes the information provided by the CDS.

The delivery of a tap, or the fetch of a CDS, marks the transition from the first state to the second. Once this transition has occurred the initiator is not allowed to modify any CDS within the chain with the exception of the abort task flag.

A target may fetch all or part of a CDS as many times as needed until the SBP status block is returned for that CDS. A target may also discard all or part of a CDS after it is fetched except for the address of the CDS and the task attributes. The task attribute is in effect at the time of the first acquisition of any part of the CDS.

5.2.4 Task sets

The Task Set is a SCSI construct and its operation is governed by the SCSI Architecture Model. The Task Set applies to the mode of SBP operation in which application (media) data is to be transported by the IEEE 1394 asynchronous transfer mechanism. For the mode of SBP operation in which application (media) data is to be transported by the IEEE 1394 isochronous transfer mechanism, there is no isochronous task set defined, even though some of the exception handling methods of the task set are utilized. In particular, while the Isochronous SCSI CDS employs a SCSI CDB to define media extents, and uses SCSI defined error conditions for exception handling, the CDB is not entered into the SCSI Task Set.

For the SCSI CDS, the transition from the second state to the third state occurs when a target enters the command (SCSI CDB) contained in the CDS into a task set. Entry into a task set is done prior to execution of the SCSI CDB. Acquisition of a CDS by the target is a different event than the entry of the command it contains into a task set. In the event that multiple task sets exist a target shall enter the command into only one task set.

> Note: The login and task management CDS's are not entered into a task set since they do not contain CDB's.

The target shall conform to the SCSI Architectural Model (SAM) X3.270 as to the order in which commands, in the task set, are processed.

5.2.5 Time stamp

Each CDS chain has a time stamp based on the order of arrival of the tap at the target. Unless use is made of the SCE flag, each and every CDS within a chain has the same time stamp. Additionally, chains are to have CDS's fetched from them in the order specified by the implied stamp. A chain with earlier time stamp has CDS's fetched from it before CDS's are fetched from chains with a later time stamp.

If the SCE flag is set, then the present CDS marks the end of one sub-chain and the next CDS (within this chain) marks the start of a new sub-chain. Setting the SCE flag, causes the time stamp to be advanced to the present time for all CDS's remaining within the chain. The remaining CDS's are viewed as existing within a new chain, a sub-chain, and this sub-chain now has a time stamp later than the time stamp of all other chains known to the target. If no other CDS has the SCE flag set, CDS's are fetched (in time sequence order) from all other chains before any CDS is fetched from the new sub-chain.

It is useful to consider an example of a limiting case for the setting of the SCE-flag in a set of CDS's. In this case, the SCE-flag is set in every CDS. The effect is that one CDS is fetched from each chain in a round robin procedure based on the time stamp associated with each of the chains. This case only affects the fetch policy; the order of completion is still governed by the SCSI-3 Architecture Model.

5.3 Data transfers

The target transfers target data to a destination or from a source as commanded by a CDS from the initiator. Target data is defined as any data that an initiator requests the target to transfer. This includes data such as login response data, device media data, scanner data, etc. The transfer of application data can use either asynchronous transfer mode or isochronous transfer mode. All commands, status blocks, and sense data shall use asynchronous transfer mode. Within each command CDS, the initiator specifies the address for each data buffer to which the target will execute the block read or block write transaction. The data transfer control field in a CDS provides control information regarding transfer rate, alignment, and packet size. The data transfer rate field specifies the maximum speed to be used when transferring data. The target shall use IEEE 1394 block read and block write transaction services when transferring data via the asynchronous transfer mode.

For certain application (media) data, the transfer means may be via the IEEE 1394 isochronous transfer mode. The isochronous data transfers make use of a simplified Link layer protocol supported by the LK_ISO.request and the LK_ISO.indication link layer services defined within the IEEE 1394 standard.

5.3.1 Asynchronous transfers

Asynchronous data transfers provide for the movement of data without a guarantee of bandwidth but with an acknowledgment protocol establishing correct receipt of all data transferred. Asynchronous data packets are defined in IEEE 1394.

The rate of data transfer shall be as specified in the data transfer control field of the CDS associated with the CDB currently in execution. Normal completion of the CDB occurs when the full amount of data specified by the CDB and associated CDS transfer length field, is transferred. Such completion may require multiple asynchronous data transfer packets with the division into IEEE 1394 packets being the responsibility of the SBP application layer in the target.

5.3.2 Isochronous transfers

Isochronous data transfers provide for the broadcast of data with a guaranteed bounded worst case latency. Isochronous data packets are defined in IEEE 1394. Isochronous data packets are not acknowledged and may be received by zero, one, or more than one node.

Related isochronous transfers between the serial bus and the storage medium constitute a logical entity called a stream. A 1394 storage device may be able to handle zero, one, or more than one stream simultaneously, depending upon its capabilities. The stream is created by a successful login process, and is dissolved by a logout process. The login process associates target resources with the stream, such as buffer space, DMA channels, etc. During a login procedure, the initiator may request a unique “initiator id. The target may assign the requested value or another value and return this value in the login data. This value, when an isochronous login is requested shall also be the stream identifier (stream id)

The storage device will reject a login if it is not capable of supporting the requested stream.

After the stream has been created, the initiator issues control CDS’s to specify which channels are to be read or written and how channel numbers are to be mapped. SCSI read or write CDS’s are issued to specify the location and amount of data to be transferred. Control CDSís are issued to start, pause, or stop the data transfer associated with a stream. Control CDSís control the flow of data to or from the 1394 bus, whereas SCSI CDS’s control the flow of data to or from the storage medium.

At any one time a stream’s data transfer may proceed in only one direction, but the initiator may (for example) first record one set of channels in one location on the storage medium and then stop that transfer and begin a playback transfer from a different location with different channels. Annex A specifies the format of stored packets, and Annex B gives pseudo code examples of the recording and playback processes.

The manner of scheduling isochronous transfers is not given by SAM. The application based requirement is to transfer isochronous data in the exact order specified by the initiator in all cases. SCSI CDB's are conveyed to the target in order to specify the area on target media that will receive data if the target is to be an isochronous listener, or the area on target media that will supply the data if the target is to be an isochronous talker. The target shall process the SCSI CDBs in the exact order as specified by the initiator. It is particularly convenient for both the initiator and the target if the initiator forms the SCSI CDS's into one or more CDS chains. In this manner, the target can control the point in time at which a given CDS is fetched and subsequently processed.

The isochronous SCSI CDS's describe a time-ordered set of data that the target transmits or receives using the IEEE 1394 isochronous data transfer mode. Timing information is added to the isochronous data stream as it is transferred to the storage medium. This timing information is stripped when the data is read from the storage medium, and is used to recreate the isochronous data stream at the proper rate. Start, Stop, and Pause information for an isochronous stream is specified in a Isochronous Control CDS and effectively controls the scheduling of transmission of isochronous data packets.

Completion of a given isochronous SCSI CDS occurs when all the data specified by the command have been transferred. Multiple isochronous packets may be required to transfer the data associated with one SCSI CDB. Status upon completion of an isochronous SCSI CDS is reported in the usual manner defined by SBP. Special error logging conventions are defined in 7.5 as applied to isochronous data transfer.

5.4 Auto sense

SBP provides a means to enable sense data to be returned without the initiator sending a REQUEST SENSE command. Sense data is sent to the address specified by the command CDS sense buffer address field. The initiator may indicate that it does not want the target to send the sense data automatically by setting the auto sense length field to zero. When the target returns auto sense data, the sense data sent (SDS) flag shall be set to one.

The sense data format in SBP is compatible with the sense data format used in the Fibre Channel Protocol.

5.5 Asynchronous event notification

The initiator can enable the notification of asynchronous events using the login procedure or the sign-in procedure. In either case, the procedure establishes an AE buffer within initiator memory to which the AE sense data is sent. Following notification of an asynchronous event, the initiator is required to implement a sign-in procedure to be notified of a subsequent AE. See SCSI-3 Architectural Model document for further information.

5.6 Exception handling

SBP conforms to all of the exception handling defined in the SCSI-3 Architectural Model (SAM) X3.270 document. SBP implements all of the required SAM services for exception handling. Refer to the SAM document for the model that shall be used in SBP and to clause 6 of this document for SBP specific implementation.

6. SCSI-3 Architecture model services

This clause defines the services that are implemented in SBP. These services are defined in detail in the SCSI-3 Architectural Model (SAM) X3.270 document. Services consist of the command execution service, data delivery services, and task management services. While isochronous operations are beyond the scope of SAM, SBP asynchronous operations conform to SAM.

6.1 Command execution services

All CDS's are transmitted from the initiator to the target, either by a tap or a fetch. The CDS is the means by which the initiator transfers SCSI command (CDB's) to the target.

Exception conditions and error processing associated with command execution are defined in clause 6.4.

6.1.1 Tap protocol

The tap protocol is the means by which an initiator sends the first 64 bytes of the first CDS in a chain to the target. The destination address shall be the address of one of the target's command FIFO's. The target shall acquire subsequent CDS's in the chain, if any, using the fetch protocol defined below. A target shall always have the ability to receive a CDS and examine its contents even if all tap slots are allocated and in use. In this manner, a target which has committed its entire set of normal priority Tap slots can still accommodate SAM task management services such as Clear Task Set or Abort Task.

If the target does not accept a tap, the target shall return a status block with a CDS status of TAP SLOT NOT AVAILABLE, (Table 9 code 12h). The target shall not execute the CDS and shall discard it once CDS status has been sent. In a cooperative multi-initiator environment, where no initiator exceeds its tap slot allocation, a target shall never reject a tap.

6.1.2 Fetch protocol

The fetch protocol is the means by which the target acquires subsequent CDS's, within a chain, from the initiator. A fetch is an IEEE 1394 transaction data request of type READ with a data length of 64. The address, within the initiator, of the CDS to be fetched is identified in the previously acquired CDS. The target is allowed to fetch any CDS within a chain until it returns a status block for the CDS. After the target returns a status block for a CDS, the target shall not fetch any portion of that CDS, nor shall it access any data structure referenced by that CDS including the sense data buffer.

6.2 Data transfer services

The data transfer protocol is the means by which data is transferred to or from the target in response to a CDS. Data is defined as any data which a CDS specifies. This includes data such as login response data, disk drive media data, scanner data, etc. Data may be moved using one of two transfer modes, asynchronous or isochronous.

6.2.1 Data transfer from target to destination

The target shall transfer data to the destination node's address space as specified within the CDS. The address specified within the CDS may or may not be associated with the initiator that issued the CDS. The bus speed, transfer length, and alignment of each data request shall comply with the settings in the data transfer control field of the CDS.

Each request shall be to an address within the range of addresses specified by the data buffer address field and the transfer length field in the CDS. No transaction shall extend beyond or start before the buffer indicated by the data buffer address and transfer length CDS fields.

6.2.2 Data transfer from source to target

The target shall read data from the source node's address space as specified within the CDS. The bus speed, transfer length, and alignment of each data request shall comply with the settings in the data transfer control field of the CDS.

Each request shall be to an address within the range of addresses specified by the data buffer address field and the transfer length field in the CDS. No transaction shall extend beyond or start before the buffer indicated by the data buffer address and transfer length CDS fields.

6.3 Task management services

Task management services are issued by the initiator to control the flow and execution of one or more tasks within the target. These services shall conform to the task management functions defined in the SCSI-3 Architectural Model (SAM) X3.270 document.

6.3.1 Abort task

This subclause describes the protocol by which an SBP initiator performs a SAM ABORT TASK, task management function.

To issue an Abort Task, task management function, the initiator performs the following actions in the order listed:

1. In the CDS residing in initiator's address space, that the initiator wishes to abort, set the ABORT TASK bit
2. Send a tap to the target, containing a task management CDS with the abort task flag set and the tag value field set to the start address of the CDS to be aborted.

The target responds to the task management CDS in the following way:

1. Returns a status block for the task management CDS with a CDS status of SCSI status invalid and an SBP status of no status to report.
2. If the CDS identified in the tag value field of the task management CDS is in a task set in the target, the target shall perform the following actions in the order listed:
   a) Stop issuing transaction data requests for the affected CDS
   b) Wait for any pending transaction data responses to complete
   c) Return a status block for the affected CDS with a CDS status of SCSI status invalid and an SBP status of no status to report.
3. If the CDS identified in the task management CDS is not known to the target, the target shall continue normal operation.

Whenever the target encounters a CDS with the abort flag set, the target shall not make that CDS active. The target shall return a status block, for that CDS, with a CDS status of SCSI status invalid and an SBP status of no status to report.

The initiator shall not reuse the address space occupied by the aborted CDS until the target returns a status block for that CDS.

6.3.2 Abort task set

This subclause describes the protocol by which SBP accomplishes the SAM ABORT TASK SET, task management function.

To perform an abort task set function an initiator sends a task management CDS with the abort task set flag set (byte 22 bit 6= 1).

For the specified task set the target response shall be :

a) stop issuing transaction data requests for the affected task set;
b) wait for any pending transaction data responses to complete;
c) return SBP status block with SCSI status invalid set in the CDS status byte, and SBP status byte set to no status to report.

The target shall not transfer SBP status, data, or sense information for any pending CDS acquired prior to the time the abort task set function was received.

6.3.3 Clear task set

This subclause describes the protocol by which SBP accomplishes the SAM CLEAR TASK SET task management function.

To perform a clear task set function, an initiator sends a task management CDS with the clear task set flag set (byte 22 bit 5= 1).

For the specified task set the target response shall be :

a) stop issuing transaction data requests for the affected task set;

b) wait for any pending transaction data responses to complete;

c) return SBP status block with SCSI status invalid set in the CDS status byte, and SBP status byte set to no status to report.;

d) establish a unit attention condition for all task identifiers;

The target shall not transfer SBP status, data, or sense information for any pending CDS acquired prior to the time the clear task set function was received.

6.3.4 Clear auto contingent allegiance

This subclause describes the protocol by which SBP accomplishes the SAM CLEAR ACA, task management function.

To perform a clear ACA function an initiator sends a task management CDS with the clear ACA flag set (byte 22 bit 1= 0). As a restriction, while the ACA condition continues to be declared by a target: (a) CDS fetch operations, by the target, shall only take place in relationship to an ACA tap message received at the ACA FIFO, (b) CDS execution shall only take place in relation to CDS's having the ACA queue type.

Once the ACA condition has been cleared, an SBP status block is returned with a SCSI status of command terminated, and an SBP status of no status to report. For further details of the ACA conditions, refer to the SCSI-3 Architectural Model (SAM) document.

6.3.5 Target reset

This subclause describes the protocol by which SBP accomplishes the SAM TARGET RESET, task management function.

To perform a target reset function an initiator send a task management CDS with the target reset flag set (byte 22 bit 2= 1). When the target has completed the reset, an SBP status block is returned with a SCSI status of good, and an SBP status of no status to report. An SBP target shall conform to the specified hard reset defined in the SCSI-3 Architectural Model (SAM) document.

6.3.6 Terminate task

This subclause describes the protocol by which SBP accomplishes the SAM TERMINATE TASK, task management function.

Support for this function is a logical unit option. To perform a terminate task function, an initiator sends a task management CDS with the terminate task flag set (byte 22 bit 4 = 1). In conformity to the SAM requirements established for this task management function, the logical unit supporting this option shall, with the following SAM specified exceptions, complete the specified task and return SCSI status of COMMAND TERMINATED and an SBP status of no status to report. The sense key, additional sense code, and qualifier shall be set as specified by SAM.

If the work performed by the terminated task involves the transfer of data, the supporting logical unit shall proceed as specified by the SAM document with regard to setting the valid bit in the sense data and the information field.

If an error is detected for the specified task, the supporting logical unit shall ignore the Terminate Task request and shall return a service response of Function complete.

If the target does not support the Terminate Task function or is unable to stop the task, the target shall return a service response of Function Complete.

6.4 Exception and error processing

Exception conditions and errors associated with the transfer of commands and/or data shall be processed as specified in the following clauses. Distinctions are made and possibly different actions are taken based upon whether errors are detected during the asynchronous or the isochronous transfer modes. The following set of general rules govern exception and error processing with SBP.

For transaction that do have idempotence property there shall be no retries;

SBP specifies target retry behavior for transactions that do not have the idempotence property;

Isochronous transfers involve only application data and It is left to application and/or implementation decision as to the exception handling processing.

6.4.1 Tap transfer exceptions

For specification of exception processing, refer to 7.2.

6.4.2 Fetch transfer exceptions

For specification of exception processing, refer to 7.3.

6.4.3 Asynchronous data transfer exceptions

For specification of exception processing, refer to 7.4.

6.4.4 Isochronous data transfer exceptions

For specification of exception processing, refer to 7.5.

7.0 SBP Transaction management

In the IEEE 1394 physical interface environment, communications with the SBP is accomplished via the services and facilities provided by the IEEE 1394 Transaction layer component. The transaction layer provides a complete request-response protocol to communicate with the application layer. Within this structure SBP is defined as the application layer.

7.1 Transaction services

The SBP (application) layer uses the following Transaction layer services:

Transaction Data Request (TR_DATA.request)

Transaction Data Confirmation (TR_DATA.confirmation)

Transaction Data Indication (TR_DATA.indication)

Transaction Data Response (TR_DATA.response)

A transaction is started by an application layer (requester) issuing a TR_DATA.request to its local transaction layer. The transaction layer sends a request packet via the IEEE 1394 bus. When the receiving Transaction Layer receives the request packet, a TR_DATA.indication is issued to the application layer (responder). The responder application layer reacts in an appropriate manner to the TR_DATA.indication, and issues a TR_DATA.response to the responder transaction layer. The transaction layer sends a response packet via the IEEE 1394 bus. When the receiving Transaction Layer receives the response packet, a TR_DATA.confirmation is issued to the application layer (requester), thus completing the transaction.

The transaction layer does not provide any services to the application layer for Isochronous transfers. These services are provided by the IEEE 1394 Link Layer component.

7.1.1 Transaction data request (TR_DATA.request)

The SBP application layer issues a TR_DATA.request to its Transaction layer to start a transaction. The following request types are supported by the IEEE 1394 bus.

WRITE	Used by SBP
READ	Used by SBP
BROADCAST WRITE	Not used by any SBP protocol
LOCK	Not used by any SBP protocol

7.1.2 Transaction data confirmation (TR_DATA.confirmation)

The SBP application layer (requester) receives a TR_DATA.conformation from its Transaction Layer in order to confirm the completion of a transaction. The following status values are supported:

COMPLETE

TIME-OUT

ACKNOWLEDGE MISSING

RETRY LIMIT

DATA ERROR

Exception condition processing by the SBP application layer is discussed in 7.2 through 7.5. This processing is dependent upon the specific SBP protocol in which the confirmation is received.

7.1.3 Transaction data indication (TR_DATA.indication)

The SBP application layer receives a TR_DATA.indication from its Transaction Layer to indicate the arrival of a request packet. The following indication types are supported by the IEEE 1394 bus.

WRITE	Used by SBP
READ	Used by SBP
BROADCAST WRITE	Not used by any SBP protocol
LOCK	Not used by any SBP protocol

7.1.4 Transaction data response (TR_DATA.response)

The SBP application layer issues a TR_DATA.response to its Transaction Layer to cause a response packet to be transferred to the requester. The following response types are supported by the IEEE 1394 bus.

WRITE	Used by SBP
READ	Used by SBP
LOCK	Not used by any SBP protocol

The disposition of the request is contained within the response code carried by the response packet.

7.2 Tap transaction

A tap transaction is TR_DATA.request of type WRITE, with a data length of 64, issued by the initiator application layer. The requester is the initiator, and the responder is the target. The destination address shall

be the address of one of the target's FIFO's. The SBP application layer at the requester takes the following exception condition processing upon receipt of the confirmation associated with the original transaction request.

COMPLETE -

TARGET CASE: Not Applicable

INITIATOR CASE: Successful transaction, no exception condition processing needed

TIME-OUT -

TARGET CASE: Not Applicable

INITIATOR CASE: Shall not retry. The initiator may abort task, reset target, or issue an IEEE 1394 command reset to the target. The choice of action is initiator implementation dependent.

ACKNOWLEDGE MISSING -

TARGET CASE: Not Applicable

INITIATOR CASE: Shall not retry. The initiator may abort task, reset target, or issue an IEEE 1394 command reset to the target. The choice of action is initiator implementation dependent.

RETRY LIMIT -

TARGET CASE: Not Applicable

INITIATOR CASE: May retry again, flag CDS aborted, or issue an IEEE 1394 command reset to the target. The choice of action is initiator implementation dependent.

DATA ERROR. (no distinction is made between under-runs, overruns, or CRC type errors)

Note: A data error indicates a transient condition affecting target/initiator communications. The target shall not use this response to reject a tap.

TARGET CASE: Not Applicable

INITIATOR CASE: May retry again. The choice of action is initiator implementation dependent.

7.3 Fetch transaction

A fetch transaction is a TR_DATA.request of type READ, with a data length of 64, issued by the target application layer. The source address and data length shall be within the limits of the address range of the CDS within initiator memory space. The requester shall be a target, and the responder shall be an initiator.

The SBP application layer at the requester takes the following exception condition action upon receipt of the given confirmation associated with the original transaction request. The exception condition processing by the initiator applies only to the response sub-action needed to satisfy the read request issued by the target.

COMPLETE -

TARGET CASE: Successful transaction, no exception condition processing needed

INITIATOR CASE: Successful completion of read response sub-action, no exception condition processing needed.

TIME-OUT -

TARGET CASE: Retry request again, continue to retry until successful or until directed otherwise by the initiator.

INITIATOR CASE: Not Applicable A time out can not occur on a response sub-action.

ACKNOWLEDGE MISSING -

TARGET CASE: Assume pending ack. This may result in a time-out condition. If so, target shall take time-out condition action.

INITIATOR CASE: Shall not retry.

RETRY LIMIT

TARGET CASE: Retry request again, continue to retry until successful or until directed otherwise by the initiator.

INITIATOR CASE: No further retry, retry limit is initiator implementation decision.

DATA ERROR. (no distinction is made between under-runs, overruns, or CRC type errors)

TARGET CASE: Retry request again, continue to retry until successful or until directed by the initiator to do otherwise.

INITIATOR CASE: Shall not retry.

7.4 Asynchronous data transfer

Asynchronous data transfer transactions are started by the SBP application layer at the target issuing TR_DATA.request. Because of the need to perform differing exception condition processing depending on the category of data to be transferred, the following different categories are discussed separately below:

application data, sense data, AE report data

login response data,

status data

7.4.1 Application data transfer target to destination / from source

The target, as part of command processing, shall transfer data to/from the destination/source node's address space by generating one or more transaction data requests. If data is to move from the target to the destination node, then the target SBP layer issues a TR_DATA.request of type WRITE. If data is to move from the source node to the target, the target issues a TR_DATA.request of type READ. The length shall be consistent both with the transfer length specified within the data transfer control field of the associated SCSI CDS and the maximum data packet length for the IEEE 1394 bus speed used. The SBP application layer at the target takes the following exception condition processing upon receipt of the given confirmation associated with the original transaction request. The exception condition processing by the destination/source node applies only to the response sub-action needed to satisfy the read or write request issued by the target.

COMPLETE -

TARGET CASE: Successful transaction, no exception condition processing needed.

DESTINATION/SOURCE CASE: Successful transaction. Successful completion of read response sub-action, no exception condition processing needed.

TIME-OUT -

TARGET CASE: Retry request again, continue to retry until successful or until directed otherwise by the initiator.

DESTINATION/SOURCE CASE: Not Applicable A time out can not occur on a response sub-action.

ACKNOWLEDGE MISSING -

TARGET CASE: Assume pending ack. This may result in a time-out condition. If so, target shall take time-out condition action.

DESTINATION/SOURCE CASE: Shall not retry.

RETRY LIMIT

TARGET CASE: Retry request again, continue to retry until successful or until directed otherwise by the initiator.

DESTINATION/SOURCE CASE: No further retry, retry limit is initiator implementation decision.

DATA ERROR. (no distinction is made between under-runs, overruns, or CRC type errors)

TARGET CASE: Retry request again, continue to retry until successful or until directed by the initiator to do otherwise.

DESTINATION/SOURCE CASE: Shall not retry.

7.4.2 Login data transfer from target to initiator

The target shall transfer login data using an TR_DATA.request of type WRITE. The SBP application layer at the target shall take the following exception condition processing upon reception of the given confirmation. There shall be at most one successful transfer of any given login data block.

COMPLETE -

TARGET CASE: Successful transaction, no exception condition processing needed.

INITIATOR CASE: NA

TIME-OUT -

TARGET CASE: Shall not retry. Target shall abort the login CDS and shall return status block with SBP status code login error.

INITIATOR CASE: NA

ACKNOWLEDGE MISSING -

TARGET CASE: Shall not retry. Target shall abort the login CDS and shall return status block with SBP status code login error.

INITIATOR CASE: NA

RETRY LIMIT

TARGET CASE: Shall not retry. Target shall abort the login CDS and shall return status block with SBP status code login error.

INITIATOR CASE: NA

DATA ERROR. (no distinction is made between under-runs, overruns, or CRC type errors)

TARGET CASE: Shall not retry. Target shall abort the login CDS and shall return status block with SBP status code login error.

INITIATOR CASE: NA

7.4.3 Status data transfer from target to initiator

The target shall transfer a status block using an TR_DATA.request of type WRITE with a data length of 16. The SBP application layer at the target shall take the following exception condition processing upon reception of the given confirmation. There shall be at most one successful transfer of any given status block.

COMPLETE -

TARGET CASE: Successful transaction, no exception condition processing needed.

INITIATOR CASE: NA

TIME-OUT -

TARGET CASE: Shall not retry. No further processing of CDS by the target.

INITIATOR CASE: NA

ACKNOWLEDGE MISSING -

TARGET CASE: Shall not retry. No further processing of CDS by the target.

INITIATOR CASE: NA

RETRY LIMIT

TARGET CASE: Retry request again, continue to retry until successful or until directed otherwise by the initiator..

INITIATOR CASE: NA

DATA ERROR. (no distinction is made between under-runs, overruns, or CRC type errors)

TARGET CASE: Retry request again, continue to retry until successful or until directed otherwise by the initiator..

INITIATOR CASE: NA

7.5 Isochronous data transfers

Isochronous data transfers are accomplished via the services and facilities provided by the IEEE 1394 Link layer component. The Link layer provides a simplified protocol to support isochronous data transfer. Within this structure, SBP is defined as the application layer.

7.5.1 Link layer services

Link Layer services, to/from the application layer, consist of the following:

Link Isochronous Control request (LK_ISO_CONTROL.request)

Link Cycle Start Indication (LK_CYCLE.indication) (hardware detected)

Link Isochronous indication (LK_ISO.indication)

Link Isochronous request (LK_ISO.request)

The SBP application layer queues the data to be transferred in the isochronous packet. When an Isochronous Control CDS is decoded, and data is available, an Isochronous transfer will begin as indicated in the stream event field of the CDS. When the Link Layer receives the Isochronous data block a LK_ISO.indication is sent to the application indicating receipt of the packet. The target or application layer does not respond to the LK_CYCLE.indication.

Multiple channels are possible within the application layer. The number and identification of the available channels is communicated to the Link Layer via the LK_ISO_CONTROL.request. When the Link Layer receives an Isochronous data block (from the physical interface) with one of the channel numbers identified by the application layer, a LK_ISO.indication is generated to begin the transfer of the data block to the application layer.

7.5.2 Isochronous status reporting

This subclause describes target status reporting mechanisms for isochronous SCSI read and isochronous SCSI write commands. The target reports status for isochronous SCSI read and isochronous SCSI write commands in the following situations:

Command Completion

Missed isochronous cycle of data - intermediate status reporting

The target always reports status on completion of an isochronous SCSI read or an isochronous SCSI write command. The protocol for doing this are the same as those for status reporting on completion of non-isochronous SCSI read and SCSI write commands.

While executing an isochronous SCSI read command (target sources isochronous data), if the target misses a cycle of data due to an internal error or transient condition, the target invokes the isochronous status reporting mechanism defined below. This mechanism is called intermediate isochronous status reporting.

While executing an isochronous SCSI write command (target sinks isochronous data), if the target misses an isochronous cycle of data due to an internal error or transient condition, the target invokes the isochronous status reporting mechanism defined in following sub-clauses. This mechanism is called intermediate isochronous status reporting.

There are three modes for intermediate isochronous status reporting. The Isochronous Login command declares the status reporting mode to be used for a given stream. The mode remains in effect for the entire life of the stream. The three modes differ in the way that the target handles error conditions which occur during isochronous data transfer. The three modes are:

Ignore and Continue - Target ignores all errors and continues isochronous data transfer

Report and Continue - Target logs errors as they occur and continues isochronous data transfer

Report and Stop - Target reports the first error that occurs and stops isochronous data transfer

7.5.3 Ignore and continue

The target generates no status during isochronous SCSI read or isochronous SCSI write command execution. Upon encountering any error while reading or writing isochronous data, the target ignores the error and continues.

There is no intermediate status block associated with this mode of operation.

7.5.4 Report and continue

If an error occurs during isochronous SCSI read or isochronous SCSI write command execution, the target generates an entry in an error log. The address of the error log is contained in the CDS sense buffer address field of the isochronous SCSI read and isochronous SCSI write CDS's.

If the target generates any entries in the error log during execution of an isochronous SCSI read or isochronous SCSI write command, the target shall set the Sense Data Sent bit in the status block for that command.

If the target completely fills the allocated sense buffer, the target stops writing error log entries, but continues executing the command to completion. Upon completing the command, the target reports a status code in the status block indicating that the sense buffer is filled with error log entries and some entries were discarded due to overflow.

Each entry in the error log reports an event from the following list:

Start of gap in the isochronous stream;

End of gap in the isochronous stream.

A gap is defined as one or more consecutive isochronous cycles during which no isochronous data packet was transmitted or received on a given channel number. The start of a gap is signified by one isochronous cycle during which an isochronous data packet for a given isochronous channel is not generated (due to a talker problem) or is not received (due to a listener problem, an isochronous header or data field CRC error,). The end of a gap is signified by the first isochronous cycle following the start of a gap in which a valid isochronous data packet is transmitted or received for a given channel number. Note that a stream gap may appear at one device and not another.

As an example, if an isochronous stream is interrupted for multiple consecutive isochronous cycles and then resumed, the target will log 2 entries to the error log regardless of the duration of the stream gap: One error log entry to report the start of the stream gap, and another error log entry to report the end of the stream gap.

Each entry in the error log is 2 quadlets long. The format of each entry in the error log is defined in table 3.

Table 3 Error log entry

Byte	Function		
0 to 3	Seconds high count		reserved
4 to 7	Seconds count/cycle count	reserved	status

The Seconds High Count is a 25-bit number that contains the value in the seconds_hi_count field of the Bus_Time CSR register on the cycle on which the error occurred.

The Seconds count/cycle count is a 20-bit field that contains the value in the seconds count and cycle count fields of the isochronous cycle start packet on which the error occurred.

The status identifies the event that occurred on the reported isochronous cycle time according to table 4.

Table 4 Isochronous status codes

Status Code	isochronous cycle time field contains the cycle time of:
0	cycle time of first valid cycle of data following the stream gap
1	Start of gap due to an unidentified error condition
2	start of gap due to target unable to source/sink isochronous data
3	start of gap due to missing isochronous data block (target is listener)
4	start of gap due to data field CRC error (target is listener)

The error log contains one entry for each start of gap detected and one entry for each end of gap detected (i.e., two error log entries per stream gap). In the case where the target is the talker, the target reports errors to the error log if it is unable to generate isochronous data packets for whatever reason. If the target is a listener, the target reports errors to the error log if it is unable to receive a packet, due to an internal error, a missing packet, or a data or header CRC error.

7.5.5 Report and stop

In this operational mode, upon encountering a stream gap , the target generates an entry in the error log and immediately stops execution of the current command. The target returns a status block with the sense data sent flag set for the command indicating the error condition.

8.0 Target memory

The target memory space contains configuration ROM and CSR's required by IEEE 1394. Also in target memory, as specified by the Serial Bus Protocol, are two sets of FIFO's, one set for asynchronous operations and one set for each channel used for isochronous operations. Figure 3 is a illustration of a device with a single task set. It serves as a conceptual framework and is not intended to imply an implementation.

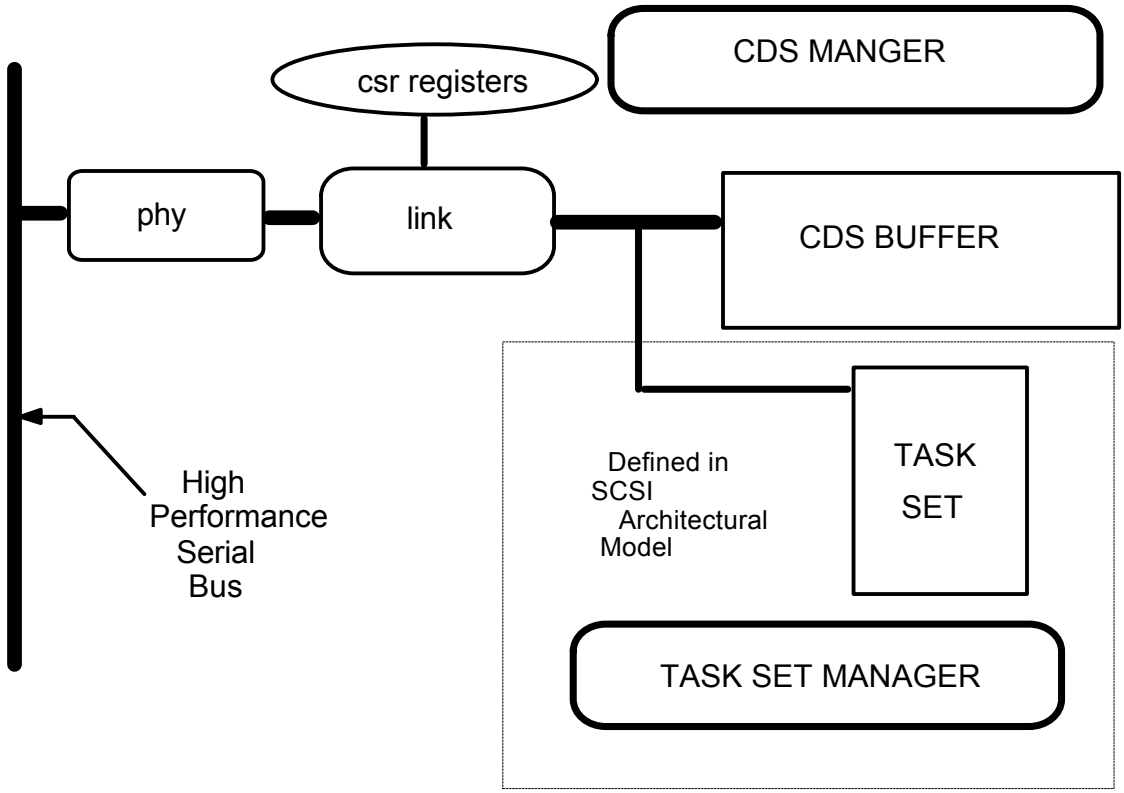


Figure 2 Block diagram of Serial SCSI Device

8.1 Configuration ROM

All IEEE 1394 nodes that implement SBP shall have a configuration ROM. The IEEE 1394 standard defines the base address of this ROM and the ISO/IEC 13213:1994 CONTROL AND STATUS REGISTERS standard defines the format of the information contained in this ROM. Annex A contains an example of a configuration ROM which is compliant with IEEE 1394, ISO/IEC 13213:1994 CONTROL AND STATUS REGISTERS and SBP. Other implementations are possible.

The configuration ROM is read during bus initialization and may be read at anytime thereafter.

Configuration ROM in the target contains the address in target memory space of the login FIFO.

An SBP device is required to provide a data structure for each unit directory and unit dependent directory supported. The data structures are shown in tables 5 and 6.

Table 5 Unit Directory

Quadlet	Description			
1	Directory length			Directory CRC
2	00b	Key (12h)	Unit specifier id	
3	00b	Key (13h)	Unit software version	
4 - n	11b	Key (14h)	Unit dependent directory offset	

The unit directory contains a unit software version, a unit specifier id, and pointers to one or more unit dependent directories.

The directory length field contains the total number of quadlets in this directory entry minus one. For example, a value of one in the directory length field indicates that the unit directory is comprised of two quadlets.

The directory CRC field contains the CRC of this directory. The CRC is calculated beginning with the next quadlet and continuing for directory length quadlets.

A six-bit key field value of 12h indicates that the low-order 24-bits of this quadlet contain the unit specifier ID.

The unit specifier ID field contains the identification number of the entity defining the architectural interface of this unit. A value of XXXXXXh indicates that the X3 Secretariat is responsible for the software interface definition.

NOTE: At the time of writing, the IEEE Registration Authority Committee (RAC) has agreed to grant a Organizationally Unique ID (OUID) to the X3 Secretariat for use in X3 standards that are IEEE Std 1212:1994 compliant. The actual value of this 24-bit ID is expected to be available before final publication of SBP as an ANSI approved standard.

A six-bit key field value of 13h indicates that the low order 24-bits of this quadlet contain the unit software version for this unit.

The unit software version field contains the value 0103E0h to indicate an SBP device compliant with this standard.

A six-bit key field value of 14h indicates that the low order 24-bits of this quadlet contain the unit dependent directory offset

The unit dependent directory offset field contains the offset from this entry, in quadlets, of the unit dependent directory.

Table 6 Unit Dependent Directory

Quadlet	Description			
1	Directory length			Directory CRC
2	01b	Key (21h)	Offset of asynchronous login FIFO	
3	01b	Key (22h)	Offset of isochronous login FIFO	

The unit dependent directory contains the offsets of the Asynchronous login FIFO and isochronous login FIFO. At a minimum, the unit dependent directory contains at least one login FIFO offset and may contain multiple offsets for asynchronous or isochronous.

The unit dependent directory length field contains the number of quadlets in this directory entry, minus one.

The directory CRC field contains the CRC of the data in this directory calculated beginning with the next quadlet and continuing for unit dependent directory length quadlet's.

A six-bit key field value of 21h indicates that the low order 24-bits of this quadlet contain the offset of the asynchronous login FIFO.

The offset of asynchronous login FIFO field contains the offset, in quadlets, from the base of CSR space to the asynchronous login FIFO.

A six-bit key field value of 22h indicates that the low order 24-bits of this quadlet contain the offset of the isochronous login FIFO.

The offset of isochronous login FIFO field contains the offset, in quadlets, from the base of CSR space to the isochronous login command FIFO.

NOTE - The asynchronous login FIFO and the isochronous login FIFO may be at the same address or at different addresses from each other.

8.2 Asynchronous FIFOs

This section describes the target FIFOs used for initiating and controlling an asynchronous task set.

The target supporting IEEE 1394 asynchronous data transfer mode shall implement all FIFOs described in this clause.

8.2.1 Asynchronous login FIFO

The asynchronous login FIFO shall have its offset specified in the unit dependent directory if asynchronous data transfers are supported. This FIFO is used for the asynchronous login procedure (see 11.1.1). The FIFO is a write only address that accepts a 64-byte tap.

8.2.2 Asynchronous normal FIFO

The asynchronous normal FIFO is used for receiving taps that contain an SCSI CDS defining a command to be executed. The address is provided as part of the login data during the login procedure.

The target may reject a tap sent to the asynchronous normal FIFO if it is unable to accept the CDS. The initiator should retry the tap until the target has resources available to accept the CDS. In a cooperative multi-initiator environment, where no initiator exceeds its tap slot allocation, a target shall never reject a tap.

The number of asynchronous normal FIFOs available is implementation dependent, but shall be at least one per asynchronous task set.

8.2.3 Asynchronous urgent FIFO

The asynchronous urgent FIFO is used for receiving taps that contain an SCSI CDS that defines a command to be fetched and entered into the task set on a priority basis. The address is provided as part of the login data during the login procedure.

NOTE - The order of completion of commands is determined by the rules governing task sets in the SCSI-3 Architecture Model.

Any of the taps that may be sent to the normal FIFO may also be sent to the urgent FIFO. The difference is in the priority of fetching the associated CDS's.

The number of asynchronous urgent FIFOs shall be one per asynchronous task set.

8.2.4 Asynchronous ACA FIFO

The asynchronous ACA FIFO is used for receiving taps that contain a tap type of ACA and an SCSI CDS with an ACA task attribute for the purposes of recovering from an exception condition associated with asynchronous operation. The address is provided as part of the login data during the login procedure.

If a CDS is received that does not have a tap type of ACA and an ACA task attribute then the target shall not execute the CDS and return a status of ILLEGAL REQUEST. Once an ACA condition is established, and for

the duration of the time the ACA condition exists, the target may only fetch CDS's directed to the asynchronous ACA FIFO.

No restriction shall exist with regard to fetching CDS's directed to an isochronous FIFO based upon existence of an asynchronous ACA.

The number of asynchronous ACA FIFO's shall be one per asynchronous task set.

8.3 Isochronous FIFO's

This section describes the target FIFO's used for initiating and controlling an isochronous stream.

A target supporting the IEEE 1394 isochronous data transfer shall implement all FIFO's described in this clause.

8.3.1 Isochronous login FIFO

The isochronous login FIFO shall have its offset specified in the unit dependent directory if isochronous data transfers are supported. This FIFO is used for the isochronous login procedure (see 11.1.2). The FIFO is a write only address that accepts a 64-byte tap.

8.3.2 Isochronous normal FIFO

The isochronous normal FIFO is used for receiving taps that contain an isochronous SCSI CDS defining a command to be executed. The address is provided as part of the login data during the login procedure.

The target may reject a tap sent to the isochronous normal FIFO if it is unable to accept the CDS. The initiator is expected to retry the tap until the target has resources available to accept the CDS.

8.3.3 Isochronous control FIFO

The isochronous control FIFO is used for receiving taps that contain an isochronous control CDS. The address is provided as part of the login data during the login procedure.

8.3.4 Isochronous ACA FIFO

The isochronous ACA FIFO is used for receiving taps that contain a tap type of ACA and an SCSI CDS with an ACA task attribute for the purposes of recovering from an exception condition associated with the isochronous operation. The address is provided as part of the login data during the login procedure.

If a CDS is received that does not have a tap type of ACA and an ACA task attribute then the target shall not execute the CDS and return a status of ILLEGAL REQUEST. Once an ACA condition is established, and for the duration of the time the ACA condition exists, the target may only fetch CDS's directed to the isochronous ACA FIFO.

No restriction shall exist with regard to fetching CDS's directed to an asynchronous FIFO based upon existence based on the existence of an isochronous ACA.

9. Initiator memory

The initiator memory space contains the CDS's, status FIFO, sense data buffers, the asynchronous event buffers and the buffers associated with transfer of data. Initiator memory also contains at least the general configuration ROM as well as various ISO/IEC 13213:1994 CONTROL AND STATUS REGISTERS and IEEE 1394 specified CSR's.

The target knows the location of the first CDS in a chain as a result of acquiring the CDS through a tap. The location of subsequent CDS's in the chain are found from the address of the next CDS field.

The address of the status FIFO and sense data buffers are stated within the CDS to which they apply. The initiator shall manage the address space so that the status FIFO and sense data buffer can be associated with a specific CDS.

The asynchronous event buffers are used if an initiator requests notification of asynchronous events during the login procedure. The address of the AE buffers is specified in the LOGIN CDS.

9.1 Status FIFO

The initiator shall provide the address of a status FIFO within each CDS. The data structure sent to the status FIFO is defined in table 7.

Table 7 Status block

Bytes	Function			
0 to 3	CDS Address (msq)			
4 to 7	CDS Address (lsq)			
8 to 11	SCSI status	Sense key	ASC	ASCQ
12 to 15	CDS status	Reserved		SBP status

IMPLEMENTOR NOTE: The location of the status FIFO is implementation dependent. If an initiator desires an interrupt when the status block is received then the FIFO may be located within memory space that is capable of generating an interrupt.

The CDS address is the value contained in the address of this CDS field for the associated CDS.

The SCSI status, sense key, ASC and ASCQ are defined in the SCSI-3 Architectural Model standard and the SCSI-3 Primary Command Set standard.

The CDS status is defined in table 8.

Table 8 CDS status

Displacement	Description
0 to 4	Reserved
5	SCSI Status Invalid (SSI)
6	Sense data sent flag (SDS)
7	Tap slot available flag (TSA)

The SCSI status invalid (SSI) flag, when set, indicates that the SCSI status byte in the SBP status block is invalid. This flag is used when an abort function has been requested by the initiator.

The sense data sent (SDS) flag, when set, indicates that sense data has been transferred for the associated CDS.

The tap slot available (TSA) flag, when set, indicates that a tap slot previously used by the initiator is available. See tap slot allocation procedure (11.1.3).

The SBP status is defined in table 9.

Table 9 SBP status

Value	Description
00h	No status to report
01h-0Fh	Reserved
10h	Unsupported data rate requested
11h	CDS has mismatch in tap type and task attribute
12h	Tap rejected
13h	Login Error
14h-EFh	Reserved
F0h-FEh	Vendor-specific
FFh	Exception occurred but no status code defined

SBP status code 00h is returned to the initiator when the target has completed the requested operation without detecting an exception condition.

SBP status code 10h is returned when an initiator has requested a data transfer rate that is not supported by the target.

SBP status code 11h is returned when the target has determined there is a mismatch in tap types and task attributes. The target may allow the various tap types to be directed to any FIFO, or may return this code if the addressed FIFO is not allowed.

SBP status code 12h is returned when the target does not have sufficient resources to honor a requested Tap.

SBP status code 13h is returned when a login is rejected. For example the resources requested by the initiator are not available or could not be supplied.

SBP status code F0h - FEh These codes are vendor specific status codes. See the appropriate vendor specification for the meaning and or definition.

SBP status code FFh is returned when an exception has occurred and the target could not determine the proper status code or no status code was identified.

9.2 CDS sense buffer

If the CDS sense length field contains a non-zero value the initiator shall provide a CDS sense buffer. The data sent to the CDS sense buffer is defined in table 10. The total length (i.e., the value of m) shall not exceed 65,535 bytes.

> NOTE - The format for bytes 0 - 23 is taken from the Fibre Channel Protocol (FCP) document describing status return and is intended to provide for increased compatibility between FCP and SBP regarding format of status and sense data.

Table 10 CDS sense data format

Bytes	Description	
0 to 3	Reserved	
4 to 7	Reserved	
8 to 11	Status	
12 to 15	Residual Count	
16 to 19	REQUEST SENSE data length	Reserved
20 to 23	Response length	Reserved
24 to n	Response information	
n+1 to m	REQUEST SENSE data	

The status quadlet field (see table 11) is normally zero upon successful completion of a command. A zero value indicates that no error occurred and no other information is present in the response information.

Table 11 Status quadlet

Byte	Description
0	Reserved
1	Reserved
2	Valid flags
3	SCSI status

The valid flags field is defined in table 12.

Table 12 Valid flags field

Displacement	Description
0	Reserved
1	Reserved
2	Reserved
3	Reserved
4	Residue field valid - underrun
5	Residue field valid - overrun
6	REQUEST SENSE information valid
7	Response information valid

The Residue field valid-underrun flag indicates that requested amount of data transferred was less than the CDS transfer length.

The Residue field valid-overrun flag indicates that the amount of data available to transfer was more than the CDS transfer length.

The REQUEST SENSE information valid flag indicates that REQUEST SENSE data is valid.

The Response information valid flag indicates that the response information is valid.

The SCSI status is the status byte as defined by the SCSI-3 Architecture Model. The status byte reported in the status block and that reported in the sense data shall be the same value for the CDS.

The residue field contains a count of the number of residual data bytes that were not transferred for this CDS. This is the difference between the byte transfer length in the CDS and the number of bytes that were or could have been transferred. A residue of zero is normal upon successful completion. Devices having indeterminate data lengths may have a non zero residue even with successful completion of the command.

The REQUEST SENSE data length field specifies the length of the of the REQUEST SENSE data. The length shall be an integral multiple of four and not exceed 100h. A length of zero indicates that REQUEST SENSE data is not provided.

NOTE - A REQUEST SENSE data length of zero is normal but not required upon successful completion of a command.

The response data length field specifies the length of the response information. The length shall be an integral multiple of four. A length of zero specifies that response information is not provided.

NOTE - A response data length of zero is normal but not required upon successful completion of a command.

The sense information field contains the information specified by the SCSI-2 Standard for presentation by the REQUEST SENSE command.

The response information field is reserved.

9.3 Asynchronous event buffer

If asynchronous event reporting is enabled the initiator shall provide a status FIFO and a sense data buffer for asynchronous event notification. AE reporting is enabled through the login procedure. See clause 11.1.

After reporting an asynchronous event, AE reporting is disabled until the initiator performs another login procedure. The target shall not reuse the sense data buffer.

10. Command data structures

All command data structures (CDS) formats are 72 bytes long. A tap contains the first 64 bytes of a CDS. CDS's are transported from the initiator to a target either as the contents of a tap, or via the fetch protocol. When the target fetches a CDS, it may read all or any part (from 1 to 72 bytes).

See the definition of the target fetch protocol in clause 6.1.2 for more information. A target may re-fetch a CDS until it returns a status block for that CDS.

The last eight bytes of all CDS formats contain the CDS sense buffer address field. The target shall return CDS sense data to this sense buffer when using the auto sense procedure.

Some CDS formats contain an SCSI command descriptor block. This is a 16-byte field designated as a CDB(0) through CDB(15). This field shall contain a command descriptor block as defined in SCSI-2 or one of the SCSI-3 command set documents.

There are multiple formats of the CDS. These formats are specified by the value of the function code in the CDS codes field.

10.1 General CDS format

The general format of the Command Data Structure (CDS) is defined in table 13. The CDS format specific fields are defined for a specific use of the CDS within sub-clauses 10.2 through 10.5. Only those fields common to the CDS format are defined in this clause.

Table 13 Command data structure format

Bytes	Description			
0 to 3	Address of next CDS (msq)			
4 to 7	Address of next CDS (lsq)			
8 to 11	Address of this CDS (msq)			
12 to 15	Address of this CDS (lsq)			
16 to 19	Reserved	Identifier	Logical unit number	
20 to 23	CDS codes	CDS specific	CDS specific	CDS specific
24 to 27	CDS format specific			
28 to 31	CDS format specific			
32 to 35	CDS format specific			
36 to 39	CDS format specific			
40 to 43	CDS transfer length			
44 to 47	Data transfer control		CDS sense length	
48 to 51	CDS format specific			
52 to 55	CDS format specific			
56 to 59	CDS status FIFO address (msq)			
60 to 63	CDS status FIFO address (lsq)			
64 to 67	CDS sense buffer address (msq)			
68 to 71	CDS sense buffer address (lsq)			

The address of next CDS field specifies the address of the next CDS in the chain. Some CDS types do not allow chaining and as a result reserved this field. See the specific CDS type for details.

The address of this CDS field specifies the address of this CDS.

The identifier field contains the initiator id returned by the target in response to the login request.

The logical unit number field specifies the logical unit number for this CDS.

NOTE - The logical unit number field is defined as a 16-bit unsigned integer value within SBP. Other protocols using SCSI commands define the size of this field differently. The mechanism for dealing with the different size fields for logical units is not specified in this standard.

The CDS transfer length specifies the maximum size in bytes of the data transfer associated with this CDS.

The CDS sense length field specifies the size in bytes of the CDS sense data buffer. It is an unsigned integer with a maximum value of FFFFh (i.e., 65,535 bytes of sense data). If the CDS sense length value is zero, CDS sense data shall not be returned for this CDS via the auto sense procedure.

The CDS status FIFO address field contains the address to which the CDS status block is sent for this CDS.

The CDS sense buffer address field contains the address of the buffer associated with the CDS to which CDS sense data is returned. The address in the CDS sense buffer address field is not valid if the CDS sense length field has a value of zero. The target is not required to fetch the CDS sense buffer address field unless it needs to return CDS sense data for this CDS. The format of the CDS sense data returned is defined in table 10.

10.1.1 CDS codes field

The CDS codes field specifies format, CDS type, and tap type, of the CDS (see table 14)

Table 14 CDS codes field

Displacement	Description
0 to 1	CDS identifier
2 to 4	Tap type
5 to 7	CDS type

The CDS identifier field value of 1h identifies this CDS as following the general format specified in Table 13. In the future other CDS formats may be defined and will be identified by other values in this field. The current value is defined in table 15 below.

Table 15 CDS identifier

Value	Description
0h	Reserved
1h	Format specified by Table 13
2h	Reserved
3h	Reserved

The tap type field specifies the tap associated with this CDS (see table 16). This allows the target to identify the tap type of the CDS when multiple FIFO address fields point to the same address (shared FIFOs).

Table 16 Tap type

Value	Description
0h	Asynchronous normal tap
1h	Asynchronous login tap
2h	Asynchronous ACA tap
3h	Asynchronous urgent tap
4h	Isochronous normal tap
5h	Isochronous login tap
6h	Isochronous ACA tap
7h	Isochronous control tap

The CDS type field specifies the format of the CDS as defined in table 17.

Table 17 CDS type field

Value	Description
0h	SCSI CDS
1h	Login CDS
2h	Management CDS
3h	Isochronous SCSI CDS
4h	Isochronous Control CDS
5h - 7h	Reserved

10.1.2 Data transfer control field

The data transfer control field provides control information regarding transfer rate, alignment, and packet size as defined in table 18. A value of 0h means maximum packet size for bus speed.

Table 18 Data transfer control field

Displacement	Description
0 to 3	Reserved
4 to 7	Data transfer rate
8 to 11	Alignment
12 to 15	Maximum packet size

The data transfer rate field specifies the maximum speed to be used when transferring data for this CDS. The values of this field are defined in table 19.

NOTE - The transfer rates of S100, S200, and S400 identify speeds that are defined for IEEE 1394 PHY data rates.

Table 19 Data transfer rate

Value	Description
0h	Transport medium base rate
1h	S100
2h	S200
3h	S400
4h - Fh	Reserved

The alignment field indicates the address boundary that is not to be crossed by a single data transfer packet. A value of zero specifies that there are no restrictions on address boundary. All other values, 0001b through and including 1111b represent the value of an exponent N such that, the number two (2) raised to the power (Decimal equivalent of N plus 1) represents the address boundary expressed in bytes. The limit case, is N=15 Decimal, with the result that two raised to power 16 indicates an address boundary of 64 KBytes.

The maximum packet size field indicates the maximum size of packet to be used by the device in accomplishing a requested data transfer. All other values, 0001b through and including 1111b represent the value of an exponent N such that, the number two (2) raised to the power (Decimal equivalent of N plus 1) represents the maximum size packet expressed in units of bytes. The limit case, is N=15 Decimal, with the result that two raised to power 16 indicates an maximum packet size of 64 KBytes.

10.2 SCSI CDS FORMATS

Two formats are defined below for SCSI CDS's. The two types of CDS formats are identified as a SCSI CDS and an ISOCHRONOUS SCSI CDS. Both CDS types carry a SCSI CDB within the format. The differences are identified below.

10.2.1 SCSI CDS

The rules for handling CDB's, included in a SCSI CDS, are as follows:

The CDB carried by the CDS shall be entered into a task set (identified in SAM);

The rules governing the scheduling of execution of the CDB are specified in SAM;

No additional information governing scheduling of execution is required.

Table 20 defines the format for the SCSI CDS.

Table 20 SCSI CDS

Bytes	Description			
0 to 3	Address of next CDS (msq)			
4 to 7	Address of next CDS (lsq)			
8 to 11	Address of this CDS (msq)			
12 to 15	Address of this CDS (lsq)			
16 to 19	Reserved	Identifier	Logical unit number	
20 to 23	CDS codes	Task codes	Reserved	Protocol flags
24 to 27	CDB 0	CDB 1	CDB 2	CDB 3
28 to 31	CDB 4	CDB 5	CDB 6	CDB 7
32 to 35	CDB 8	CDB 9	CDB 10	CDB 11
36 to 39	CDB 12	CDB 13	CDB 14	CDB 15
40 to 43	CDS transfer length			
44 to 47	Data transfer control		CDS sense length	
48 to 51	Data buffer address (msq)			
52 to 55	Data buffer address (lsq)			
56 to 59	CDS status FIFO address (msq)			
60 to 63	CDS status FIFO address (lsq)			
64 to 67	CDS sense data buffer address (msq)			
68 to 71	CDS sense data buffer address (lsq)			

The address of the next CDS is valid if the M-flag is set. (see table 23, Protocol flags field)

The identifier field contains the initiator id returned by the target in response to the login request.

The task codes field for use with the SCSI CDS is defined in table 21.

Table 21 Task codes field

Displacement	Description
0 to 4	Reserved
5 to 7	Task attribute

The task attribute field specifies the type of queue tag. The values of this field are defined in table 22. The task attributes are defined in the SCSI-3 Architecture Model standard.

Table 22 Task attribute

Value	Description
0h	Simple
1h	Head of queue
2h	Ordered
3h	Reserved
4h	ACA
5h	Untagged
6h	Reserved
7h	Reserved

The protocol flags field is defined in table 23.

Table 23 Protocol flags field

Displacement	Description
0	Abort task flag (A)
1	Linked flag (L)
2	Data transfer order flag (O)
3	Scatter-gather flag (S)
4	next CDS address valid flag (M)
5	end of sub-chain flag (SCE)
6	Read data flag
7	Write data flag

The Abort task flag when set indicates the that this CDS shall be aborted. The Abort task flag is the only field that an initiator may modify once a tap for the chain contain the CDS has been sent. The initiator shall not clear the Abort task flag once it has been set until the target has released the affected CDS (via status or other action).

The Linked flag when set indicates that this is a linked CDS. This flag shall have a value of zero for the last CDS in a series of linked CDS's. When the Linked flag is cleared, this is not a linked CDS.

Within a chain, once the Linked flag is set, all CDS's to the end of the chain shall have the Linked flag set. The Linked flag does not affect the function of the next CDS address valid flag. When the Linked flag is set, the end of sub-chain flag shall be cleared.

> IMPLEMENTOR'S NOTE - The Linked flag, in conjunction with the linked bit in the command descriptor block, can be used to implement a SCSI style set of linked commands. However, the Linked flag affects the fetch policy regardless of the setting of the linked bit in the command descriptor block.

The Data transfer order flag when set to one indicates that the target shall perform read or write requests to the data buffer in sequential order, beginning at the base address of the data buffer. When clear, the target may perform read or write requests in any order while transferring data.

> Note: The target shall transfer media data to the data buffer such that the data with the lowest LBA goes to the starting address of the data buffer, increasing to the end of the data buffer.

The Scatter-gather flag when set to one indicates that the initiator memory Scatter-gather function is being invoked for the present command.

When the Scatter-gather flag is cleared:

> The transfer length field contains the requested number of bytes of data to be transferred;
>
> The data buffer address field contains a 64-bit address that the data is transferred to or from. This data address is not required to be associated with the initiator that sent the CDS.

When the Scatter-gather flag is set:

> The transfer length field indicates the size, in bytes, of the scatter-gather list found at the address specified by the value in the data buffer address.
>
> The data buffer address field contains the address of the scatter-gather list. This address has no specific alignment restriction.

The scatter-gather list shall consist of one or more consecutive 16-byte constructs in the format shown in table 24.

Table 24 Scatter/gather list

Byte	Function
0 to 3	Data buffer address (msq)
4 to 7	Data buffer address (lsq)
8 to 11	Reserved
12 to 15	Transfer length

The data buffer address field contains an address that data is written to or read from. This data address may or may not be associated with the initiator that sent the command.

The Transfer length field contains the length in bytes of the data buffer segment. The data buffer segment is located at the Data buffer address.

The M-flag, when set, indicates that the address of next CDS field contains the address of the next CDS in the chain. The M-flag, when cleared, indicates that this CDS is the last CDS in the chain. If the M-flag is cleared, the SCE-flag is ignored. If the M-flag is cleared and the link bit in the SCSI command descriptor block is set, the command is terminated with CHECK CONDITION status.

The SCE-flag when set, indicates the end of a sub-chain and advances the time stamp for the remainder of the chain to the current time. A CDS with the SCE flag set marks the end of one sub-chain. If the M-flag is set in the current CDS, the address of next CDS field contains the address of the first CDS in the next sub-chain.

10.2.2 Isochronous SCSI CDS

The rules for handling CDB's, carried within an ISOCHRONOUS SCSI CDS, are as follows:

The CDB carried by the CDS shall not be entered into any task set;

SAM does not provide rules governing the scheduling of execution of the CDB;

Scheduling of execution of the CDB is based on information contained in an Isochronous Control CDS.

Table 25 defines the format for the ISOCHRONOUS SCSI CDS.

Table 25 ISOCHRONOUS SCSI CDS

Bytes	Description			
0 to 3	Address of next CDS (msq)			
4 to 7	Address of next CDS (lsq)			
8 to 11	Address of this CDS (msq)			
12 to 15	Address of this CDS (lsq)			
16 to 19	Reserved	Identifier	Logical unit number	
20 to 23	CDS codes	Reserved	Reserved	Protocol flags
24 to 27	CDB 0	CDB 1	CDB 2	CDB 3
28 to 31	CDB 4	CDB 5	CDB 6	CDB 7
32 to 35	CDB 8	CDB 9	CDB 10	CDB 11
36 to 39	CDB 12	CDB 13	CDB 14	CDB 15
40 to 43	CDS transfer length			
44 to 47	Data transfer control		CDS sense length	
48 to 51	Reserved		Starting Offset	
52 to 55	Reserved			
56 to 59	CDS status FIFO address (msq)			
60 to 63	CDS status FIFO address (lsq)			
64 to 67	CDS sense data buffer address (msq)			
68 to 71	CDS sense data buffer address (lsq)			

The address of the next CDS is valid if the M-flag is set. (See table 23, Protocol flags field)

The identifier field contains the initiator id returned by the target in response to an isochronous login request. All CDS's (Isochronous SCSI CDS and Isochronous Control CDS) having the same value in the identifier field are associated with the same isochronous stream.

The protocol flags field is defined in table 26.

Table 26 Protocol flags field

Displacement	Description
0	Abort task flag (A)
1	Reserved
2	Reserved
3	Reserved
4	next CDS address valid flag (M)
5	Reserved
6	Read data flag
7	Write data flag

The Abort task flag when set indicates the that this CDS shall be aborted. The Abort task flag is the only field that an initiator may modify once a tap for the chain contain the CDS has been sent. The initiator shall not clear the Abort task flag, once it has been set, until the target has released the affected CDS (via status or other action).

The M-flag, when set, indicates that the address of next CDS field contains the address of the next CDS in the chain. The M-flag, when cleared, indicates that this CDS is the last CDS in the chain. If the M-flag is cleared, the SCE-flag is ignored.

The Starting Offset field specifies the offset from the start of the first logical specified in the CDB carried by this CDS, that the data for this CDS is to be read/written. The Starting Offset shall be a multiple of 4 bytes.

10.3 LOGIN CDS

The LOGIN CDS (table 27) performs the login function that allocates resources in a target and provides an identifier for the initiator to use.

Table 27 LOGIN CDS

Bytes	Description			
0 to 3	Reserved			
4 to 7	Reserved			
8 to 11	Address of this CDS (msq)			
12 to 15	Address of this CDS (lsq)			
16 to 19	Reserved	Identifier	Logical unit number	
20 to 23	CDS codes	Reserved	Reserved	Login flags
24 to 27	Reserved			Max Isoch channels
28 to 31	Max isochronous payload size		Number of slots	AE sense length
32 to 35	AE buffer address (msq)			
36 to 39	AE buffer address (lsq)			
40 to 43	CDS transfer length			
44 to 47	Data transfer control		CDS sense length	
48 to 51	Data buffer address (msq)			
52 to 55	Data buffer address (lsq)			
56 to 59	CDS status FIFO address (msq)			
60 to 63	CDS status FIFO address (lsq)			
64 to 67	CDS sense buffer address (msq)			
68 to 71	CDS sense buffer address (lsq)			

The login flags are defined in table 28 below.

Table 28 Login flags field

Displacement	Description
0	I-flag
1	Reserved
2	Reserved
3	Reserved
4	Reserved
5	Reserved
6	Reserved
7	Reserved

If the I-flag is set, the identifier field contains the proposed identifier. If the I-flag is cleared, the target shall ignore this field and assign an identifier.

The Max isochronous channels field specifies the maximum number of channels that the stream shall be able to transfer at any one time.

The Max isochronous payload size field is the maximum value that will be used in the data length field of the isochronous packets of the stream.

The number of slots field identifies the number of tap slots requested by this identified initiator.

The AE sense length specifies the length in bytes of the sense data buffer.

The AE buffer address specifies the location within initiator memory to which the AE sense data shall be sent.

The CDS transfer length field shall specify the size of the buffer allocated for the return of login data to the initiator. The length shall be specified in bytes.

Table 29 LOGIN data format

Bytes	Function	
0 to 3	Normal FIFO address (msq)	
4 to 7	Normal FIFO address (lsq)	
8 to 11	Control FIFO address (msq)	
12 to 15	Control FIFO address (lsq)	
16 to 19	ACA FIFO address (msq)	
20 to 23	ACA FIFO address (lsq)	
24 to 27	Minimum transfer length	Identifier
28 to 31	Prior tap slot count	New tap slot count

The Normal FIFO address field contains the address of the normal command FIFO.

The Control FIFO address field contains the address of the control command FIFO.

The ACA FIFO address field contains the address of the ACA command FIFO.

The Minimum transfer length field is valid only for isochronous login data. The value in the minimum transfer length field specifies the minimum CDS transfer length, in the Isochronous SCSI CDS required by the target in order to sustain the isochronous data rate requested in the Isochronous Login CDS. In order to assure that the target can sustain the requested data transfer rate, all isochronous SCSI CDSís for this stream shall have a CDS transfer length field equal to or greater than the value in the minimum transfer length field. The initiator is not required to follow this rule, however, if it does not , the target may lose some isochronous data.

The identifier field contains the identifier assigned to the initiator.

The prior tap slot count field contains the number of tap slots assigned to the initiator at the time the LOGIN CDS was received.

The new tap slot count field contains the number of tap slots assigned to the initiator as a result of processing the LOGIN CDS.

The data buffer address field, in the Login CDS, is the location to which the returned data is sent.

10.4 MANAGEMENT CDS

Table 30 Management CDS

Bytes	Description			
0 to 3	Reserved			
4 to 7	Reserved			
8 to 11	Address of this CDS (msq)			
12 to 15	Address of this CDS (lsq)			
16 to 19	Reserved	Identifier	Logical unit number	
20 to 23	CDS codes	Reserved	Management flags	SBP flags
24 to 27	Reserved			
28 to 31	Reserved			
32 to 35	Tag value (msq)			
36 to 39	Tag value (lsq)			
40 to 43	CDS transfer length			
44 to 47	Data transfer control		CDS sense length	
48 to 51	Reserved			
52 to 55	Reserved			
56 to 59	CDS status FIFO address (msq)			
60 to 63	CDS status FIFO address (lsq)			
64 to 67	CDS sense buffer address (msq)			
68 to 71	CDS sense buffer address (lsq)			

The management flags field used with the management CDS is defined in table 31.

Table 31 Management flags field

Displacement	Description
0	RESERVED
1	CLEAR ACA
2	TARGET RESET
3	RESERVED
4	TERMINATE TASK
5	CLEAR TASK SET
6	ABORT TASK SET
7	ABORT TASK

The task management functions are defined in the SCSI-3 Architecture Model and in clause 6 of this document.

Table 32 SBP flags field

Displacement	Description
0	Reserved
1	Reserved
2	Reserved
3	Reserved
4	Reserved
5	Reserved
6	Logout flag
7	Reserved

The logout flag, if set, causes the target to release all the resources for the specified identifier.

The pause flag, if set, causes the target to suspend data transfer for the specified identifier. The pause flag, if cleared, causes the target to resume a suspended data transfer for the specified identifier. If the pause flag is cleared and a suspended data transfer does not exist for the specified identifier, then no action is required.

The tag value field contains the address of the CDS on which the management function is to be performed.

10.5 Isochronous control CDS

The Isochronous Control CDS has a CDS type of "Isochronous Control CDS" in the CDS type field. The ISOCHRONOUS CONTROL CDS (table 31) shall be delivered to the isochronous control FIFO for the stream for which it applies.

Table 33 ISOCHRONOUS CONTROL CDS

Bytes	Description			
0 to 3	Address of next CDS (msq)			
4 to 7	Address of next CDS (lsq)			
8 to 11	Address of this CDS (msq)			
12 to 15	Address of this CDS (lsq)			
16 to 19	Reserved	Identifier	Logical unit number	
20 to 23	CDS codes	Reserved	Reserved	Control Function
24 to 27	Error Handling	Stream Event	sy	Reserved
28 to 31	Reserved			
32 to 35	Seconds High Count			
36 to 39	Seconds count/cycle count			
40 to 43	Reserved			
44 to 47	Data transfer control		CDS error log length	
48 to 51	Control Function specific			
52 to 55	Control Function specific			
56 to 59	CDS status FIFO address (msq)			
60 to 63	CDS status FIFO address (lsq)			
64 to 67	CDS error log address (msq)			
68 to 71	CDS error log address (lsq)			

The ISOCHRONOUS CONTROL CDS requests the target to change the state of an isochronous stream. The state of an isochronous stream includes the set of enabled channels, whether or not the stream is running, paused or stopped, a mapping from source channels to destination channels and the method of error handling (including the provision of an optional error log).

The identifier field contains the initiator id returned by the target. All CDS's (Isochronous SCSI CDS and Isochronous Control CDS) having the same value in the identifier field are associated with the same isochronous stream. This identifier value is returned during the isochronous login operation.

The control function field contains one of the following values:

Table 34 Control Function Field

Value	Description
0h	Start
1h	Stop
2h	Pause
3h	Update Channel Mask
4h	Remap Channels
5h	Set Error Mode
6h	Query Stream Status
7h - FFh	Reserved

For some of the control functions, the time at which the requested action occurs may be modified by the Stream Event field.

The Start control function instructs the target to commence or resume talking or listening. The time at which the action is to take place may be modified by the Stream Event field.

The Stop control function instructs the target to terminate the isochronous stream (either as a talker or a listener) and to flush its buffers. If the target had been a listener, any isochronous data already received shall be made available to the SCSI commands previously sent to the isochronous normal FIFO. If the target has been a talker, any isochronous data already obtained from SCSI commands shall be discarded. The time at which the action is to take place may be modified by the Stream Event field.

The Pause control function instructs the target to suspend the transfer of isochronous data with the expectation that the data transfer will resume. If the target is a talker, the target shall pause on the requested stream event and not send any data for that stream while in the paused state. Subject to target implementation limitations, data produced by SCSI commands previously sent to the isochronous normal FIFO may continue to accumulate while the isochronous stream is paused. If the target is a listener, the target shall stop listening to the stream of data on the requested stream event. The target may flush isochronous data in order to make the data available to SCSI commands previously sent to the isochronous normal FIFO. The time at which the action is to take place may be modified by the Stream Event field.

The Update Channel Mask function requests the target to change the set of enabled channels, specified by the Channel Mask field, in accordance with the time indicated by the Stream Event field.

The Remap Channels function requests the target to update the 64-entry channel map array maintained for the isochronous stream. The channel map array specifies a transformation from each source channel to a destination channel. If the isochronous stream is configured as a listener, the source channels observed on Serial Bus may be transformed to different channel numbers recorded on the medium. When isochronous data is played back from the storage medium, the source channels embedded with the recorded data may be transformed to different channels broadcast on Serial Bus.

> Note: It is possible for a mapping of two or more source channels into a single destination channel to be meaningful. For example, isochronous data recorded at different times and from different channels may be concatenated on the medium and then played back on a single channel.

The Set Error Mode function requests the target to configure its mode of error handling as specified by the Error Handling field. The address and size of an error log may be specified as well.

The Query Stream Status function may be used to ascertain whether or not a stopped or paused isochronous stream is ready to accept a Start function. The ready state of a stream is determined by the state of the isochronous SCSI commands associated with the stream.

> Note: If a target is instructed to listen to isochronous data and to transfer the data to the medium at a nonzero offset within a logical block, some implementations may require time to retrieve previously recorded data from the medium before it is possible to commence recording the new data. The Query Stream Status function may be used to determine when the target is ready to listen.

The Error Handling field is valid only if the control function field contains a value of Set Error Mode. The Error Handling field contains one of the following values:

Table 35 Error Reporting Field

Value	Description
0	Report error and stop
1	Report error and continue
2	Ignore all errors
3-15	reserved

The definition for these values is described in the section 7.5.2. Isochronous status reporting

The stream event field is valid only if the control function field has a value of Start, Stop, Pause or Update Channel Mask. The stream event field contains one of the following values:

Table 36 Stream Event Field

Value	Description
0h	Immediately
1h	Cycle time
2h	sy field match (listener only)
3h	First observed data (listener only)
4h - FFh	Reserved

The Immediately value instructs the target to perform the action specified by control function as soon as possible, within the capabilities of the target implementation.

The Cycle time value instructs the target to perform the action specified by control function on the cycle time specified by the seconds high count and seconds count/cycle count fields.

The sy field match value only has meaning when the target is acting as a listener. The sy field match value instructs the target to perform the action specified by a control function on the isochronous cycle on which the isochronous packet header for any enabled channel contains the sy value specified in the sy field of this CDS.

The “First observed data” value has an effect similar to that of “Immediately” in that it is possible for data packets to be recorded immediately. The difference between the two stream events occurs during recording if no data packets are being received when the control CDS is processed. If “Immediately” is specified then CYCLE MARK packets are recorded as cycle start packets as received from the IEEE 1394 bus. If “First observed data” is specified then no CYCLE MARK packets are recorded until the first DATA packet is observed. When the first DATA packet is received, a CYCLE_MARK packet with values from the most recent CYCLE_START bus packet is recorded followed by the initial DATA packet.

The sy field has meaning only if the target is acting as a listener. It is used in conjunction with the stream event field to modify the operations of the start and update channel mask control functions.

The seconds high count is a 25-bit field which has meaning only if the control function field contains a value of Start, Stop, Pause or Update Channel Mask and the stream event field contains a value of cycle time. This field is used in conjunction with the seconds count/cycle count field described below.

The seconds count/cycle count field is a 20-bit field which has meaning only if the control function field contains a value of Start, Stop, Pause or Update Channel Mask and the stream event field contains a value of cycle time. This field is used in conjunction with the seconds high count field described above.

When both fields are valid, the seconds high count field contains a value to be compared against the seconds high count field of the BUS_TIME register. A cycle time match occurs on the isochronous cycle on which these two fields are equal, and the values in the seconds Count/cycle Count fields are equal to the seconds count and cycle count fields of the cycle start packet.

The CDS error log length field specifies the size in bytes of the buffer into which the target may report isochronous stream errors. This field is valid only if the control function field has a value of Set Error Mode. If this field is zero, the target shall not report any isochronous stream errors.

The eight bytes of control function specific data, from bytes 48 to 55 of the isochronous control CDS, assume different meanings according to the value in the control function field. If the control function field has a value of Update Channel Mask, these fields are the Channel Mask. If the control function field has a value of Remap Channels, these fields are the Channel Map fields.

The Channel Mask field is valid only if the control function field contains a value of Update Channel Mask. The channel mask is a bit map that represents enabled isochronous channels, as shown below:

Table 37 Channel Mask Field

Bytes	Description
48 to 51	Channel Mask (msq)
52 to 55	Channel Mask (lsq)

A one in the bit position that corresponds to the channel number indicates that the channel is to be enabled. If an isochronous channel is enabled for listening, isochronous packets observed for that channel are transferred to the storage medium under control of isochronous SCSI CDS's with the same stream identifier. Conversely, if an isochronous channel is enabled for talking, a data stream is obtained from the storage medium and isochronous packets with matching channel numbers are broadcast on Serial Bus.

The Channel Map field is valid only if the control function field contains a value of Remap Channels. The Channel Map field is a four entry array, as shown below:

Table 38 Channel Map Field

Bytes	Description			
48 to 51	Destination channel	Source channel	Destination channel	Source channel
52 to 55	Destination channel	Source channel	Destination channel	Source channel

Each entry in the array specifies a transformation from a source channel to a destination channel. At the time a stream identifier is obtained, by means of an Isochronous Login CDS, the 64-entry channel map for the stream is reset to the identity map.

The CDS error log address field specifies the location of a buffer available to the target to report isochronous stream errors. The address of this buffer shall be specified only if the control function is Set Error Mode.

11.0 Control procedures

Additional protocols are required within the Serial SCSI environment to support the ability of the initiator to interact with targets regarding management of:

tap slot resources

sign-in for notification of asynchronous events

Targets are not allowed to respond to most SBP control protocol requests from an initiator unless that initiator has previously obtained from the target either an identifier for use with the asynchronous transfer protocol or an identifier for use with the isochronous transfer protocol. The target is allowed to respond normally to an initiator without a valid identifier only when the request is for login.

11.1 Login procedure

The login procedure is the method by which the initiator obtains an identifier and the FIFO addresses needed to deliver CDS's to the target. The initiator shall not attempt to deliver any CDS's other than the login CDS prior to logging in.

The initiator may have multiple identifiers outstanding for a given target for both asynchronous login and isochronous login.

11.1.1 Asynchronous login procedure

Using the tap protocol, the initiator sends an ASYNCHRONOUS LOGIN CDS to the target's asynchronous login FIFO, thereby requesting an identifier and the addresses for the FIFO needed to support the asynchronous command, control, status and data transfer protocols.

If the target is able to honor the initiator's login request, the target returns the following information to the initiator using the data transfer protocol:

address of the normal command FIFO

address of the urgent command FIFO

address of the ACA command FIFO

unique identifier

The target then returns status for the asynchronous login CDS using the status transfer protocol.

If the target is unable to accept a login request, the target shall return sense data using the sense data transfer protocol, and then return status for the asynchronous login CDS using the status transfer protocol.

After successful completion of the asynchronous login request, the initiator may issue asynchronous CDS's to the target.

11.1.2 Isochronous login procedure

Using the tap protocol, the initiator sends an ISOCHRONOUS LOGIN CDS to the targets isochronous login FIFO, thereby requesting an identifier and the addresses for the FIFO and registers needed to support the isochronous command, control, status and data transfer protocols.

If the target is able to honor the initiator's isochronous login request, the target returns the following information to the initiator using the data transfer protocol:

address of the isochronous command FIFO

address of the isochronous control FIFO

address of the isochronous ACA command FIFO

unique stream identifier

minimum transfer length

The target then returns status for the isochronous login CDS using the status transfer protocol.

If the target is unable to accept an isochronous login request, the target shall return sense data using the sense data transfer protocol, then return status for the isochronous login CDS using the status transfer protocol.

After successful completion of the isochronous login request, the initiator may issue isochronous CDS's to the target.

The initiator may generate isochronous SCSI CDSís with a transfer length field equal to or greater than the value of the minimum transfer length field for this stream. The initiator is not required to do this, however, if the initiator does not follow this rule there will be a greater likelihood of losing isochronous data.

11.1.3 Tap slot allocation procedure

The tap slot allocation procedure provides an initiator the means to request a specific number of tap slots be reserved for its use. A tap slot allocation request can be made as a part of the login procedure or at any time thereafter up until a logout procedure is performed for the identifier. This allows the initiator to modify the number of tap slots it has been allocated to meet its workload requirements. The LOGIN CDS is used for the tap slot allocation procedure.

There is no requirement upon an initiator to request allocation of tap slots prior to sending CDS's to the target. A target supporting asynchronous transfer operation shall support the tap slot allocation procedure.

Initiators are not required to limit their use of tap slots to the number they have been allocated. In a cooperating, multi-initiator environment where all initiators stay within their allocation of tap slots, the target shall not reject a tap.

A target shall not limit an initiator to its allocated number of tap slots if additional tap slots are available in the unallocated pool of tap slots. The target shall notify the initiator of the release of a tap slot by setting the tap slot available (TSA) flag in one of the SBP status blocks returned for a chain. The following rules govern the setting of this flag:

The tap slot available flag is set in only one of the status blocks returned for a chain.

When the tap slot available flag is set then the target shall have the tap slot for the associated chain available to accept a new tap.

The tap slot available flag is set only for chains that are using allocated tap slots. If the chain is not using an allocated tap slot then no tap slot available indication is returned.

If a chain is using an allocated tap slot the target shall set tap slot available flag in a status block on or before completion of the chain.

Initiators are not limited in their use of tap slots to the number they have been allocated. Initiators are not guaranteed availability of a tap slot even if they are within their allocation.

11.2 Logout procedure

The logout procedure is the method by which the initiator relinquishes an identifier and all resources associated with that identifier in a target.

The initiator performs the asynchronous logout procedure in order to release its assigned identifier, any sign-in reservations and any tap slot resources. The initiator performs the isochronous logout procedure in order to release its assigned identifier and any resources previously reserved for isochronous data transfer for the associated stream. The following sub-clauses describe these two procedures.

11.2.1 Asynchronous logout

Using the tap protocol defined in this standard, the initiator sends a tap containing the management CDS with the logout flag set to the target's asynchronous login FIFO. The target's configuration ROM contains the address of the asynchronous login FIFO.

The target returns a status block and then releases resources reserved for the identifier. The resources include:

tap slot allocations

sign-in reservations

normal, urgent and ACA FIFO's

initiator identifier

After performing the asynchronous logout procedure, the initiator shall not use the identifier nor shall it deliver any asynchronous CDS’s to the target using the normal, urgent or ACA command FIFO's that were released at the time of the asynchronous logout without performing another login.

11.2.2 Isochronous logout

Using the tap protocol defined in this standard, the initiator sends a tap containing the management CDS with the logout flag set to the target's isochronous login FIFO. The target's configuration ROM contains the address of the isochronous login FIFO.

The target releases resources reserved for the identifier and returns a status block using the status transfer protocol defined in this standard. The resources include:

internal resources necessary to support the isochronous stream

sign-in reservations

the normal, urgent, and ACA FIFO's

stream identifier

The target then returns status using the status transfer protocol.

After performing the isochronous logout procedure, the initiator shall not use the identifier nor shall it send any isochronous CDS's to the isochronous normal, urgent, or ACA FIFO's that were released at the time of the isochronous logout without performing another login.

11.3 Sign-in procedure

A sign-in provides the initiator with the means to deal with asynchronous events. An AE sense length greater than zero indicates that the initiator requests asynchronous event notification. An AE sense length of zero indicates that the initiator requests notification of asynchronous events be terminated. The LOGIN CDS is used for the sign-in procedure. The initiator may sign-in as part of the initial login procedure or at later time.

The sign-in procedure applies to the occurrence of one asynchronous event. When a target performs an asynchronous event notification, all notified initiators perform a sign-in procedure again to receive notification of the next asynchronous event.

An initiator may send to the target a second request for notification of asynchronous events even though a prior request for notification is still valid and in effect. The new request supersedes the previous request with the new AE buffer address and length replacing the previous address and length.

The format of the sense data returned on an asynchronous event report is the same as it is in the SCSI-3 REQUEST SENSE command.

Annex A (Normative) Isochronous transfers

A.1 Isochronous transfers

Multiple Isochronous channels can be recorded on a storage medium as a single stream. A channel mask controls which channels are recorded at any instant, and the channel mask may be altered any number of times during the recording of a stream. Similarly, during playback a channel mask determines which channels in a recorded stream are transmitted on the bus. In actual implementations the number of channels per stream may be limited.

During transfer in either direction the channel numbers may be mapped. This permits playback when the bus does not have the same channels available as when the stream was recorded. It also allows implementations to have conventions for channel numbers on the recording medium without regard for the actual bus channel number during recording or playback.

A 1394 storage device may be capable of handling one or more streams at a time, or may only be able to handle asynchronous data. The device will respond to the isochronous login request indicating whether it has the resources required to handle the requested stream.

A.1.1 Packet boundary location

When seeking forward or back through a recorded stream of isochronous data, it may be necessary to locate the boundary between packets starting at an arbitrary location. Data may be examined at quadlet boundaries for valid header patterns. Validity checks such as data length pointing to another valid header quadlet, allowed field values, etc. permit identifying packet boundaries with high confidence after scanning through a small number of packets.

A.2 Isochronous Storage Formats

As an isochronous data stream is transferred to a storage medium, DATA packets of enabled channels are stored along with CYCLE MARK packets and optionally NULL packets. If there is any error receiving the packet from the IEEE 1394 bus (including header CRC, data CRC, or data length errors), then the packet shall be treated as if it had not been received and shall not be recorded. The packet types are distinguished on the storage medium by a type field in the first quadlet. The values stored in the type field are given in Table A.1 below.

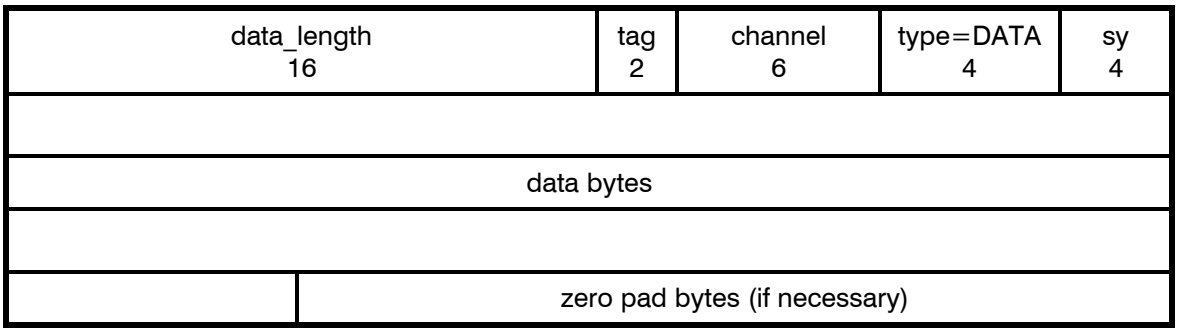
Table A.1 type field values

Value	Name	Description
0	NULL	Null packet
1	CYCLE_MARK	Cycle start packet marker
2	DATA	Isochronous data packet
3 - 15	—	Reserved

A.2.1 Packet format

The format of an isochronous packet as recorded on the storage medium is shown in figure A.1.

Figure A.1 - DATA packet storage format



The packet shall be stored on a quadlet boundary in the storage medium.

The stored packet is almost entirely a copy of the packet as it was received from the 1394 bus. Only the channel and tcode fields are altered during the storage/playback process.

The channel number field may be replaced if the channel mapping function is used.

The 4 bit type field contains the DATA value; see Table A.1 for its corresponding numerical value. This replaces the tcode field in the received isochronous packet.

All other fields of the stored packet have the same meanings as the standard 1394 bus packet. These field descriptions (below) are copied from the IEEE 1394 standard document for reference.

The data length field shall specify the length of the data field.

The data field shall contain the data to be transferred.

If the data length field specifies a length that is not a multiple of four, the sender shall pad the data field with 00_{16} bytes such that the data field is composed of full quadlets.

The tag field provides a high level label for the format of data carried by the isochronous packet.

Table A.2 Tag field encoding

value (binary)	meaning
00	Data field unformatted.
01	Reserved.
10	Reserved.
11	Reserved.

The sy (synchronization) code is an application-specific control field. The details of its use are beyond the scope of this standard.

NOTE: A synchronization point may be the defined as boundary between video or audio frames, or any other point in the data stream the application may consider appropriate.

A.2.2 NULL packet format

The NULL packet provides a mechanism for inserting one or more quadlets within the recorded byte stream, as illustrated in figure A.2. These are ignored during playback. The use of NULL packets is optional. They serve no purpose for SBP, but may be useful in some applications. Note that excessive null packets may cause some implementations to not be able to play back data at full speed.

Figure A.2 NULL packet format

data_length 16	reserved 8	type=NULL 4	reserved 4
(data_length/4 quadlets) 32			

The packet shall be stored on a quadlet boundary in the storage medium. The length of a NULL packet is a multiple of four bytes and the minimum size is 4 bytes (only the header is provided). The 16 bit data_length field, which shall be a multiple of 4 (zero is a valid value), specifies the number of bytes within the packet following the header.

The 4 bit type field contains the NULL value; see table A.1 for its corresponding numerical value.

The content of the quadlets following the header is not specified by this standard.

A.2.3 CYCLE_MARK packet format

The CYCLE_MARK packet shall be recorded when a cycle start packet is received from the serial bus. The CYCLE_START packet layout is illustrated in figure A.3.

During recording of a stream, each time a cycle start packet is seen on the bus a CYCLE_MARK packet shall be recorded. Note that this only occurs when recording is enabled (not stopped or paused). Even if the channel mask is set to all zero bits (no channels enabled for transfer), if the stream is enabled then CYCLE_MARK packets shall be recorded for each cycle start packet on the bus.

During playback of a stream, a CYCLE_MARK packet indicates to the device that a new cycle was encountered at that point during recording. To make the timing of the data during playback be the same as

when it was recorded, the device waits for the next isochronous cycle start before transmitting the next packet. See Annex B for pseudo code illustrating this function.

Figure A.3 - CYCLE_START packet format

second_count 7	cycle_count 13	reserved 4	type=CYCLE_START 4	reserved 4

The packet shall be stored on a quadlet boundary in the storage medium. The length of an CYCLE_MARK packet is four bytes.

The 4 bit type field contains the CYCLE_MARK value; see table A.1 for its corresponding numerical value.

The 7 bit second_count field and the 13 bit cycle_count fields are set to the corresponding fields of the most recent cycle start packet.

Note: The second_count and cycle_count fields are not required for isochronous playback, but may be useful for editing and for packet boundary location when seeking through a recorded stream.

Annex B (Informative) Isochronous Transfer Pseudo Code

Figures B.1 and B.2 below contain pseudo code fragments for recording and playback. These fragments only show packet handling, not login, logout, or control & command handling.

Figure B.1 - Pseudo code for recording

```
repeat
      when cycle_start
            packet.type = CYCLE_MARK
            packet.second_count = CYCLE_TIME.second_count
            packet.cycle_count = CYCLE_TIME.cycle_count
            move_to_storage(packet)
      when packet_received
            packet = move_from_1394_bus()
            if channel_mask[packet.channel]
                packet.type = DATA
                packet.channel = channel_map[packet.channel]
                move_to_storage(packet)
            endif
until forever
```

Figure B.2 - Pseudo code for playback

```
repeat                              // synchronize to cycle mark in file
    packet = move_from_storage()
    until (packet.type == CYCLE_MARK)
        repeat
            packet = move_from_storage()
            when cycle_start
                repeat
                    if (packet.type == DATA)
                        if channel_mask[packet.channel]
                            packet.channel = channel_map[packet.channel]
                            packet.type = ISOCHRONOUS_DATA_BLOCK
                            move_to_1394_bus(packet)
                        endif
                    endif
                packet = move_from_storage()
        until (packet.type == CYCLE_MARK)
until end_of_file
```

Annex C (Informative) Configuration ROM

C.1 Scope

The IEEE 1394 Serial Bus interface carries with it certain implementation requirements for CSR registers and configuration ROM information. These register and ROM implementation requirements follow ANSI/IEEE 1212, which describes certain baseline registers and ROM entries and also defines a framework for additional registers and ROM entries for bus specific and device specific applications.

This annex provides a description of the configuration ROM for advisory purposes only. An example of a simple implementation of SBP configuration ROM is provided. IEEE 1394 and ISO/IEC 13213:1994 CONTROL AND STATUS REGISTERS define the registers and ROM entries required to support SBP.

The base configuration ROM structure exists at a 48-bit node identifier address (FFFF F000 0400_{16}) found in every IEEE 1394 node.

Table C-1 shows an example of SBP configuration ROM entries. This is only an example and not a definitive statement of a compliant configuration ROM. Note that the configuration ROM is parsed for pointers to the locations of the node unique ID, the unit directories, and the unit dependent directories. ISO/IEC 13213:1994 CONTROL AND STATUS REGISTERS formally defines these registers, and IEEE 1394 describes the use of some of these registers in the IEEE 1394 environment.

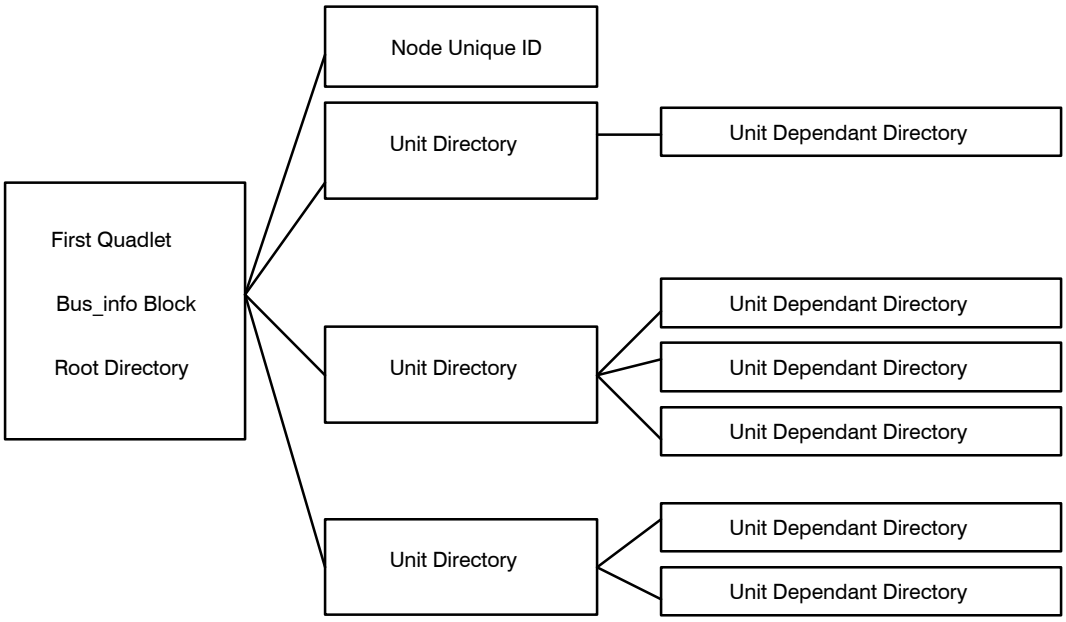


Figure C.1 - Configuration ROM Layout

Table C.1 - Example of SBP configuration ROM entries

Block	Bytes	Description		
First Quadlet	0 to 3	bus_info_length 04h	crc_length 13h	ROM_CRC_value (calculated)
Bus_Info Block	4 to 7	'1394' in ASCII 31 33 39 34h		
	8 to 11	Node options 00 FF 00 00h		
	12 to 15	node_vendor_id (00 00 00h) (serial number hi)		chip_id-hi (serial number) (00)
	16 to 19	chip_id_lo (serial number)		
Root Directory	20 to 23	Directory length 00 04h		Directory CRC (calculated)
	24 to 27	vendor ID key 03h	Module Vendor ID (ID of module vendor)	
	28 to 31	Node Capability key 0Ch	Node Capabilities 00 83 80h	
	32 to 35	Node Unique ID key 8Dh	Node Unique ID offset 00 00 02h	
	36 to 39	Unit Directory key D1h	Unit Directory offset 00 00 04h	
Node Unique ID leaf	40 to 43	length_of_ leaf 00 02h		Leaf CRC (calculated)
	44 to 47	node_vendor_id (00 00 00h) (serial number hi)		chip_id-hi (serial number) (00)
	48 to 51	chip_id_lo (serial number lo)		
Unit Directory	52 to 55	Directory length 00 03h		Directory CRC (calculated)
	56 to 59	unit_sw_version_key 13h	unit_sw_version (SW version number)	
	60 to 63	unit_spec_ key 12h	unit_spec_ID (Specifier ID)	
	64 to 67	unit_dep_dir_ key D4h	unit_dep_dir_ offset 00 00 01h	
Unit Dependent Directory	68 to 71	Unit Dependent Directory length 00 02h		Directory CRC (calculated)
	72 to 75	asyn_login_FIFO_key 41h	offset of asynchronous login FIFO 00 40 00h	
	76 to 79	isoch_login_FIFO_key 42h	offset of isochronous login FIFO 00 40 00h	

The text below describes the parameters in the table above. Much of this text is implementation specific. Please refer to ISO/IEC 13213:1994 Control and Status Registers and IEEE 1394 High Performance Serial Bus for a more formal and general description of these registers.

C.1.1 First quadlet

bus_info_length is the length, in quadlets, of the Bus_Info_Block portion of the data structure.

crc_length is the number of quadlets over which the following CRC is calculated. In this case, the following CRC field protects the entire data structure shown in the above table.

ROM_CRC_value is calculated on the number of quadlets contained in the crc_length field, beginning with the quadlet immediately following the ROM_CRC_value field.

C.1.2 Bus information block

'1394' in ASCII defines this to be a 1394 device.

Node options contains a field of bits that determine the 1394 options that this device implements. The example value shown in the table above indicates that this device implements no options. Note that most of these options have to do with isochronous data transfer and bus management capabilities.

node_vendor_id the upper 24 bits of the quadlet contains the node vendor ID number. The vendor ID field contains a serial number unique for all nodes manufactured by the vendor whose ID appears in the vendor ID field.

chip_id_hi, the low order 8 bits of the quadlet, contains the most significant byte of a unique chip id number.

chip_id_lo contains the least significant quadlet of the unique chip id number.

C.1.3 Root directory

directory length is the length, in quadlets, of the root directory. This length does not include the first quadlet in the root directory.

Directory CRC is calculated on the entire root directory (i.e., directory length quadlets, beginning with the quadlet immediately following the Directory CRC entry).

Vendor ID key indicates that the low order 24 bits of this quadlet contain the vendor ID.

Module Vendor ID is the 24-bit identification number of the vendor of this module.

Node Capab. key indicates that the low order 24 bits of this quadlet contain a field of bits defined in ISO/IEC 13213:1994 CONTROL AND STATUS REGISTERS which indicate the capabilities of this node.

Node Capabilities describes the capabilities of this node. The bit settings appearing in the table above indicate that this device has the following capabilities:

spt bit set indicating that this device implements a split time-out register in the core register block.

64 bit set indicating that this node uses 64 bit addressing.

fix bit set indicating that this node implements the fixed addressing scheme.

lst bit set indicating that this node implements the lost bit in the state bits register (accessed using the State_Set and State_Clear registers in the core register block).

NOTE - These capabilities represent a minimal baseline feature set.

Node Unique ID key indicates that the low order 24 bits of this quadlet contain the offset to the node unique ID leaf.

Node Uniq ID offset contains the offset, in quadlets, from this entry to the node unique ID leaf.

Unit Directory Key indicates that the low order 24 bits of this quadlet contain the offset from this entry to the unit directory.

unit directory off contains the offset, in quadlets, from this entry to the unit directory.

C.1.4 Node unique ID leaf

length of leaf contains the number of quadlets in this leaf.

leaf CRC contains the CRC of this leaf, calculated beginning with the next quadlet and continuing for length of leaf quadlets.

node_vendor_id, the upper 24 bits of the quadlet, contains the node vendor ID number. The vendor ID field contains a serial number unique for all nodes manufactured by the vendor whose ID appears in the vendor ID field.

chip_id_hi, the low order 8 bits of the quadlet, contains the most significant byte of a unique chip id number.

chip_id_lo contains the least significant quadlet of the unique chip id number.

C.1.5 Unit directory

directory length contains the number of quadlets in this directory entry.

directory CRC contains the CRC of this directory, calculated beginning with the next quadlet and continuing for directory length quadlets.

unit_sw_version key indicates that the low-order 24 bits of this quadlet contain the software version number for this node.

unit_sw_version contains the version of the software of this device.

unit_spec_id key indicates that the low-order 24 bits of this quadlet contain the vendor ID of the entity defining the architectural interface of this node.

unit spec ID contains the identification number of the entity defining the architectural interface of this unit.

unit dep dir key indicates that the low-order 24 bits of this quadlet contain the offset from this entry, in quadlets, of the unit dependent directory.

unit dep dir offset contains the offset from this entry, in quadlets, of the unit dependent directory.

C.1.6 Unit dependent directory

Unit dep dir length contains the number of quadlets in this directory entry.

directory CRC contains the CRC of the data in this directory calculated beginning with the next quadlet and continuing for unit dep dir length quadlets.

asynchronous login FIFO key indicates that the low-order 24 bits of this quadlet contain the offset, in quadlets, from the base of CSR space to the asynchronous login FIFO. The presence of the asynchronous login FIFO entry in the unit-dependent directory indicates that the target implements and supports all mandatory elements of the asynchronous data transfer protocol.

offset of asynchronous login FIFO contains the offset, in quadlets, from the base of CSR space to the asynchronous login command FIFO.

isochronous login FIFO key indicates that the low-order 24 bits of this quadlet contain the offset, in quadlets, from the base of CSR space to the isochronous login FIFO. The presence of the isochronous login FIFO entry in the unit dependent directory indicates that the target implements and supports all mandatory elements of the isochronous data transfer protocol.

offset of isochronous login FIFO contains the offset, in quadlets, from the base of CSR space to the isochronous login command FIFO.