Document number 2006013

OHCI P2P Network and Functional Test Specification

June 9, 2008

Sponsored by:

1394 Trade Association

Accepted for publication by

1394 Trade Association Board of Directors

Abstract

Keywords

Expected, May, Reserved, Shall, Should

1394 Trade Association Specification

1394 Trade Association Specifications are developed within Working Groups of the 1394 Trade Association, a non-profit industry association devoted to the promotion of and growth of the market for IEEE 1394-compliant products. Participants in Working Groups serve voluntarily and without compensation from the Trade Association. Most participants represent member organizations of the 1394 Trade Association. The specifications developed within the working groups represent a consensus of the expertise represented by the participants.

Use of a 1394 Trade Association Specification is wholly voluntary. The existence of a 1394 Trade Association Specification is not meant to imply that there are not other ways to produce, test, measure, purchase, market or provide other goods and services related to the scope of the 1394 Trade Association Specification. Furthermore, the viewpoint expressed at the time a specification is accepted and issued is subject to change brought about through developments in the state of the art and comments received from users of the specification. Users are cautioned to check to determine that they have the latest revision of any 1394 Trade Association Specification.

Comments for revision of 1394 Trade Association Specifications are welcome from any interested party, regardless of membership affiliation with the 1394 Trade Association. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments.

Interpretations: Occasionally, questions may arise about the meaning of specifications in relationship to specific applications. When the need for interpretations is brought to the attention of the 1394 Trade Association, the Association will initiate action to prepare appropriate responses.

Comments on specifications and requests for interpretations should be addressed to:

Editor, 1394 Trade Association
315 Lincoln, Suite E
Mukilteo, Wash 98275
USA

1394 Trade Association Specifications are adopted by the 1394 Trade Association without regard to patents which may exist on articles, materials or processes or to other proprietary intellectual property which may exist within a specification. Adoption of a specification by the 1394 Trade Association does not assume any liability to any patent owner or any obligation whatsoever to those parties who rely on the specification documents. Readers of this document are advised to make an independent determination regarding the existence of intellectual property rights, which may be infringed by conformance to this specification.

Published by

1394 Trade Association
315 Lincoln, Suite E
Mukilteo, Wash 98275 USA

Copyright © 2008 by 1394 Trade Association
All rights reserved.

Printed in the United States of America

Contents

1394 TRADE ASSOCIATION
Copyright © 2008, 1394 Trade Association. All rights reserved.

1394 TRADE ASSOCIATION Copyright © 2008, 1394 Trade Association. All rights reserved.

Tables

Figures

Annexes

1394 TRADE ASSOCIATION
Copyright © 2008, 1394 Trade Association. All rights reserved.

Revision history

Revision 0.2 (June 19, 2007)

Please see all changes bars and comments in Revision 0.2.

Revision 0.3 (January 21, 2008)

Formatted and added variables to the Test Parameters Table in Chapter 5.

Editorial change to 8.1.1 to conform HCControl.softReset naming convention.

Added section 8.1.2 Bus Reset Test for all Registers.

Fixed several names to use variable defined in Test Parameters Table in Chapter 5.

Added GUID ROM Register Access Test

Added step RG52 to ATRerties test.

Changed wording for all tests covered by the Base 1394 Test Suite Defintion.

Revision 0.4 (January 24, 2008)

Added P_GUID and P_Outbound_Dual_Phase to Test Parameters Table in Chapter 5.

General Editorial Clean-up

Revision 0.5 (June 9, 2008)

Updated 1394TA Address in document.

Added Test Parameters Table caption in Chapter 5.

Removed BENCH_TEST comment from Specification Reference table column in Register Test chapter.

Removed template information from Annex

1394 TRADE ASSOCIATION

Copyright © 2008, 1394 Trade Association. All rights reserved.

OHCI P2P, Network and Functional Test Specification

1 Scope and purpose

1.1 Scope

This specification defines the Point-to-Point (P2P), Network and Functional tests that must be passed before a 1394 Open Host Controller Interface (OHCI) implementation can earn a 1394 Trade Association FireWire or iLink 1394 Compliant logo.

1.2 Purpose

This specification will be used by the 1394 Trade Association logo program as the mandatory OHCI Point-to-Point, Network and Functional test.

Copyright © 2008, 1394 Trade Association. All rights reserved.

-This page left intentionally blank -

Copyright © 2008, 1394 Trade Association. All rights reserved.

2 Normative references

2.1 Reference scope

The specifications and standards named in this section contain provisions, which, through reference in this text, constitute provisions of this 1394 Trade Association Specification. At the time of publication, the editions indicated were valid. All specifications and standards are subject to revision; parties to agreements based on this 1394 Trade Association Specification are encouraged to investigate the possibility of applying the most recent editions of the specifications and standards indicated below.

2.2 Approved references

The following approved specifications and standards may be obtained from the organizations that control them.

IEEE Std 1394-1995, Standard for a High Performance Serial Bus

IEEE Std 1394a-2000, Standard for a High Performance Serial Bus—Amendment 1

IEEE Std 1394b-2002, Standard for a High Performance Serial Bus—Amendment 2

1394 Open Host Controller Interface Specification – Release 1.1

Throughout this document, the term “IEEE 1394" shall be understood to refer to IEEE Std 1394-1995 as amended by IEEE Std 1394a-2000 and IEEE Std 1394b-2002.

2.3 References under development

At the time of publication, the following referenced specifications and standards were under development.

2.4 Reference acquisition

The references cited may be obtained from the organizations that control them:

1394 Trade Association, 1560 East Southlake Blvd, Suite 242, Southlake, TX 76092 USA; (817) 416-2200 / (817) 416-2256 (FAX); http://www.1394ta.org/

American National Standards Institute (ANSI), 11 West 42nd Street, New York, NY 10036, USA; (212) 642-4900 / (212) 398-0023 (FAX); http://www.ansi.org/

Institute of Electrical and Electronic Engineers (IEEE), 445 Hoes Lane, PO Box 1331, Piscataway, NJ 08855-1331, USA; (732) 981-0060 / (732) 981-1721 (FAX); http://www.ieee.org/

In addition, many of the documents controlled by the above organizations may also be ordered through a third party:

Global Engineering Documents, 15 Inverness Way, Englewood, CO 80112-5776; (800) 624-3974 / (303) 792-2192; http://www.global.ihs.com/

1394 TRADE ASSOCIATION

Copyright © 2008, 1394 Trade Association. All rights reserved.

3 Definitions and notation

3.1 Definitions

3.1.1 Conformance

Several keywords are used to differentiate levels of requirements and optionality, as follows:

3.1.1.1 expected: A keyword used to describe the behavior of the hardware or software in the design models assumed by this specification. Other hardware and software design models may also be implemented.

3.1.1.2 ignored: A keyword that describes bits, bytes, quadlets, octlets or fields whose values are not checked by the recipient.

3.1.1.3 may: A keyword that indicates flexibility of choice with no implied preference.

3.1.1.4 reserved: A keyword used to describe objects (bits, bytes, quadlets, octlets and fields) or the code values assigned to these objects in cases where either the object or the code value is set aside for future standardization. Usage and interpretation may be specified by future extensions to this or other specifications. A reserved object shall be zeroed or, upon development of a future specification, set to a value specified by such a specification. The recipient of a reserved object shall ignore its value. The recipient of an object defined by this specification as other than reserved shall inspect its value and reject reserved code values.

3.1.1.5 shall: A keyword that indicates a mandatory requirement. Designers are required to implement all such mandatory requirements to assure interoperability with other products conforming to this specification.

3.1.1.6 should: A keyword that denotes flexibility of choice with a strongly preferred alternative. Equivalent to the phrase “is recommended.”

3.1.2 Glossary

The following terms are used in this specification:

3.1.2.1 definition: Press ENTER to create a subsequent numbered definition paragraph …

3.1.3 Abbreviations

The following are abbreviations that are used in this specification:

XXX The “Normal Indent” paragraph style is used for abbreviations

CSR Control and status register [B1]

As exemplified by CSR, abbreviations may cite a bibliographic reference.

3.2 Notation

3.2.1 Numeric values

Decimal and hexadecimal are used within this specification. By editorial convention, decimal numbers are most frequently used to represent quantities or counts. Addresses are uniformly represented by hexadecimal numbers. Hexadecimal numbers are also used when the value represented has an underlying structure that is more apparent in a hexadecimal format than in a decimal format.

1394 TRADE ASSOCIATION

Copyright © 2008, 1394 Trade Association. All rights reserved.

Decimal numbers are represented by Arabic numerals without subscripts or by their English names. Hexadecimal numbers are represented by digits from the character set 0 – 9 and A – F followed by the subscript 16. When the subscript is unnecessary to disambiguate the base of the number it may be omitted. For the sake of legibility hexadecimal numbers are separated into groups of four digits separated by spaces.

As an example, 42 and $2A_{16}$ both represent the same numeric value.

3.2.2 Bit, byte and quadlet ordering

This specification uses the facilities of Serial Bus, IEEE 1394, and therefore uses the ordering conventions of Serial Bus in the representation of data structures. In order to promote interoperability with memory buses that may have different ordering conventions, this specification defines the order and significance of bits within bytes, bytes within quadlets and quadlets within octlets in terms of their relative position and not their physically addressed position.

Within a byte, the most significant bit, *msb*, is that which is transmitted first and the least significant bit, *lsb*, is that which is transmitted last on Serial Bus, as illustrated below. The significance of the interior bits uniformly decreases in progression from *msb* to *lsb*.

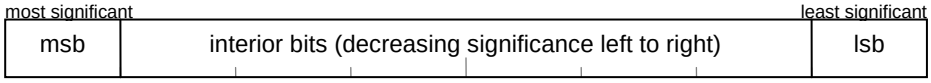

Figure 1 – Bit ordering within a byte

Within a quadlet, the most significant byte is that which is transmitted first and the least significant byte is that which is transmitted last on Serial Bus, as shown below.

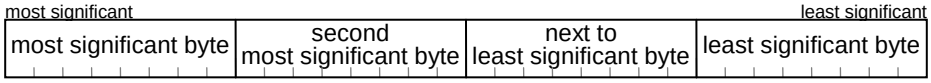

Figure 2 – Byte ordering within a quadlet

Within an octlet, which is frequently used to contain 64-bit Serial Bus addresses, the most significant quadlet is that which is transmitted first and the least significant quadlet is that which is transmitted last on Serial Bus, as the figure below indicates.

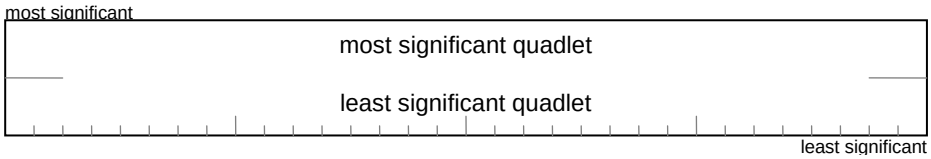

Figure 3 – Quadlet ordering within an octlet

When block transfers take place that are not quadlet aligned or not an integral number of quadlets, no assumptions can be made about the ordering (significance within a quadlet) of bytes at the unaligned beginning or fractional quadlet end of such a block transfer, unless an application has knowledge (outside of the scope of this specification) of the ordering conventions of the other bus.

Copyright © 2008, 1394 Trade Association. All rights reserved.

4 Introduction (informative)

4.1 Overview

The procedures defined in this document can be used to determine if an Open HCI (OHCI) behaves as required when stimulated by compliant OHCI software and 1394 bus activity.

The operation of an OHCI is affected by programmatic control from the host as well as activity which occurs on the 1394 bus. The OHCI specification defines what behavior by the host is legal, whereas the IEEE 1394 specification defines what bus activity is legal. This specification defines tests within those bounds and assumes that the OHCI is connected to a compliant IEEE 1394 PHY. For example, a CRC error on the bus is permitted and the specification defines correct ways of dealing with it. Whereas, the specification requires host software to not prepare DMA programs with pointers to non-existent memory.

Because host software in practice, obeys requirements of the OHCI specification, this test spec does not define tests outside of those bounds. Such testing may be added in the future.

This test specification may require more than one node(s) to facilitate testing. Also a communication mechanism between the implementation under test (IUT) and other Test Node(s) is required. The communication protocol and mechanism is implementation specific, and not constrained by this specification.

This test specification assumes that the 1394 bus on which the Open HCI implementation under test is connected is quiet. That is, no 1394 node outside of the test environment described within this specification shall generate traffic on the 1394 bus.

Unless otherwise specified the following topology is used for all tests:

Figure 4: Basic Topology for OHCI Tests.

These tests do not address host bus behavior or host software behavior. Passing these tests does not guarantee that implementations will not suffer from variable host bus conditions or host software behavior.

1394 TRADE ASSOCIATION

Copyright © 2008, 1394 Trade Association. All rights reserved.

5 Test Parameters

This section contains a form with parameters the tester must input before the test may be executed.

Question	Parameter Name	Value	Comments
Is GUID ROM present?	P_GUID_ROM		
If the GUID ROM is present, what is the GUID ROM value	P_GUID		
What is the OHCI revision supported?	P_Revision		Major version of OHCI supported.
What is the OHCI version supported?	P_Version		Minor version of OHCI supported.
Is outbound dual-phase retry implemented?	P_Outbound_Dual_Phase		
What is the maximum speed supported by the link	P_Link_spd		
What is the maximum write block request that can be received.	P_Max_rec		Shall be 512 or greater
What is the Company ID?	P_vendorCompanyID		
How many Isochronous Transmit channels are supported?	IT_CONTEXT_NUMBER		
How many Isochronous Receive channels are supported?	IR_CONTEXT_NUMBER		

Table 1 – Test Parameters

Copyright © 2008, 1394 Trade Association. All rights reserved.

6 Point to Point Test

6.1 Purpose

The purpose of this test is to check whether the performance between the Implementation Under Test (IUT) and the reference device is according to expectation when a IUT and a reference device are connected.

- In the case where the devices should be recognized in a certain combination, can the device be recognized?
- Can asynchronous data be read to/from IUT using IP traffic?
- Can asynchronous data be read to/from IUT using SPB2 traffic?
- Can isochronous data be streamed to/from IUT?Are there no issues when 1394 bus resets are initiated during data transfer?

6.2 Determination of reference devices

This section defines how to determine a reference device for IUT. For this test 5 different reference devices is required.

Policy regarding the way to determine reference devices:

1) List the categories of devices that can login into the IUT. (Discovered by operating system).

Example List:

CH1 – Computer Host on host adapter

CH2 – Computer Host on mother board

MS1 – SBP-2 HDD supporting RBC Command Set

ID7 – IIDC v1.31 Digital Camera

AVC2 – DV Device

2) Select 5 reference devices:

- Select devices from each category according to the following order of priority.
- Select the devices with 1394 TA compliance logo from other companies.
- Select the devices from other companies that are (were) available on the market.
- Select the devices with 1394 TA compliance logo from tester's company.
- Select the devices from tester's company that are (were) available on the market.

Copyright © 2008, 1394 Trade Association. All rights reserved.

6.2.1 IEEE-1394B reference devices

If IUT is an IEEE-1394b OHCI device then at least 3/5 of reference devices shall also be IEEE-1394b capable. If IUT is not an IEEE-1394b OHCI device than at least 1/5 of reference devices shall be IEEE-1394b capable.

6.2.2 Multiple platforms/operating systems

If IUT advertises operation under multiple platforms/operating systems one of each platform/operating systems should be tested up to reference device limit of five.

Example:

- Mac PC
- Windows PC

6.3 Common Tests

Common tests must be executed regardless of the type of the IUT.

6.3.1 Common Tests Topology

For the tests listed in section 6.3.3 the following topology shall be used. Each test listed in section 6.3.3 shall be tested five times, once each for each reference device.

IUT ---------- Reference Device ---------- Bus Reset Generator Node

Figure 5. Single port IUT configuration.

or

Reference Device ---------- IUT ---------- Bus Reset Generator Node

Figure 6. Two or more port IUT configuration.

For tests requiring generation of bus reset if both the IUT and Reference device are single port devices then the following topology may be used.

IUT ---------- Bus Reset Generator Node ---------- Reference Device

Figure 7. Single port IUT and reference device configuration.

Copyright © 2008, 1394 Trade Association. All rights reserved.

6.3.2 Common test 1

Test ID	Test Description	Windows Test Result	Macintosh Test Result
PP321	With IUT powered off connect and power the reference device.	-	-
PP322W	Power IUT, was the reference device correctly listed in IUT's device manager/registry?	Yes or No	-
PP322M	Power IUT, was the reference device correctly listed in IUT's device manager/registry?	-	Yes or No
PP323W	Power down IUT, did IUT correctly power down?	Yes or No	-
PP323M	Power down IUT, did IUT correctly power down?	-	Yes or No
PP324W	Repeat steps PP321 through PP323n 4 times. Was each step completed successfully	Yes or No	-
PP324M	Repeat steps PP321 through PP323n 4 times. Was each step completed successfully	-	Yes or No

6.3.3 Common Test 2

Test ID	Test Description	Windows Test Result	Macintosh Test Result
PP331	Connect IUT to reference device. Was the reference device correctly listed in IUT's device manager/registry?	Yes or No	Yes or No
PP332	Could the reference device be unplugged (without unmounting) and correctly removed from IUT's manager/registry?	Yes or No	Yes or No
PP333	Repeat steps PP331 and PP332 four times. Was reference device correctly registered/unregistered each time?	Yes or No	Yes or No
PP334	With IUT active connect the reference device. Put IUT into sleep/suspend state. Did IUT enter sleep/suspend state successfully?	Yes or No	Yes or No
PP335	With IUT asleep/suspended, wake/resume IUT. Did IUT wake/resume successfully?	Yes or No	Yes or No
PP336	Repeat steps PP334 and PP335 4 times. Was each sleep/suspend and wade/resume completed successfully?	Yes or No	Yes or No

1394 TRADE ASSOCIATION

Copyright © 2008, 1394 Trade Association. All rights reserved.

6.3.4 Common Test 3

Test ID	Test Description	Windows Test Result	Macintosh Test Result
PP341	Connect IUT to IP 1394 capable reference device. Was the reference device correctly listed in IUT's device manager/registry?	Yes or No	Yes or No
PP342	Have IUT write one giga-byte or larger file to the reference device. Was file transfer completed successfully?	Yes or No or N/A	Yes or No or N/A
PP343	Have IUT read one giga-byte or larger file from the reference device. Was file transfer completed successfully?	Yes or No or NA	Yes or No or NA
PP344	Repeat steps PP342 and PP343 4 times. Was each file transfer completed successfully?	Yes or No or NA	Yes or No or NA
PP345	Have the reference device write one giga-byte or larger file to IUT. Was file transfer completed successfully?	Yes or No or N/A	Yes or No or N/A
PP346	Have the IUT read one giga-byte or larger file from the reference device. Was file transfer completed successfully?	Yes or No or NA	Yes or No or NA
PP347	Repeat steps PP345 and PP346 4 times. Was each file transfer completed successfully?	Yes or No or NA	Yes or No or NA
PP348	Have the reference device and IUT write and read one giga-byte or larger file to each other at the same time. Were the file transfers completed successfully?	Yes or No or N/A	Yes or No or N/A
PP349	Repeat step PP348 4 times. Was each file transfer completed successfully?	Yes or No or NA	Yes or No or NA

6.3.5 Common Test 4

Test ID	Test Description	Windows Test Result	Macintosh Test Result
PP351	Connect IUT to SBP2 HDD reference device. Was the reference device correctly listed in IUT's device manager/registry?	Yes or No	Yes or No
PP352	Have IUT write one giga-byte or larger file to the reference device. Was file transfer completed successfully?	Yes or No or N/A	Yes or No or N/A
PP353	Have IUT read one giga-byte or larger file from the reference device. Was file transfer completed	Yes or No	Yes or No

Copyright © 2008, 1394 Trade Association. All rights reserved.

	successfully?		
PP354	Repeat steps PP352 and PP353 4 times. Was each file transfer completed successfully?	Yes or No	Yes or No

6.3.6 Common Test 5

Test ID	Test Description	Windows Test Result	Macintosh Test Result
PP361	Connect IUT to DV camera reference device. Was the reference device correctly listed in IUT's device manager/registry?	Yes or No	Yes or No
PP362	Transfer isochronously 1 minute of video/audio from the reference device to the IUT. Was video/audio transfer completed successfully?	Yes or No	Yes or No
PP363	Transfer isochronously 1 minute of video/audio from IUT to the reference device. Was video/audio transfer completed successfully?	Yes or No	Yes or No
PP364	Repeat steps PP362 and PP363 4 times. Was each transfer completed successfully	Yes or No	Yes or No

6.3.7 Common Test 6

Test ID	Test Description	Windows Test Result	Macintosh Test Result
PP371	Load movie (example .avi file) with video and sound on to the **IP1394** reference device. Play movie using Window Media Player or Quicktime on IUT. Initiate 10 long bus resets within 20 seconds while movie is playing. Did the movie continue with no interruption?	Yes or No	Yes or No
PP372	Load movie (example .avi file) with video and sound on to the **IP1394** reference device. Play movie using Media Player or Quicktime on IUT. Initiate 10 short bus resets within 20 seconds while movie is playing. Did the movie continue with no interruption?	Yes or No	Yes or No
PP373	Load movie (example .avi file) with video and sound on to the **SBP2 HDD** reference device. Play movie using Window Media Player or Quicktime on IUT. Initiate 10 long bus resets within 20 seconds while movie is playing. Did the movie continue with no interruption?	Yes or No	Yes or No
PP374	Load movie (example .avi file) with video and sound on to the **SBP2 HDD** reference device. Play movie using Media Player or Quicktime on	Yes or No	Yes or No

1394 TRADE ASSOCIATION
Copyright © 2008, 1394 Trade Association. All rights reserved.

	IUT. Initiate 10 short bus resets within 20 seconds while movie is playing. Did the movie continue with no interruption?		
PP375	Transfer isochronously 1 minute of video/audio from the **DV camera** reference device to the IUT. Initiate 10 long bus resets within 20 seconds while movie is being transferred. Did the movie continue with no interruptions?	Yes or No	Yes or No
PP376	Transfer isochronously 1 minute of video/audio from the **DV camera** reference device to the IUT. Initiate 10 short bus resets within 20 seconds while movie is being transferred. Did the movie continue with no interruptions?	Yes or No	Yes or No

Copyright © 2008, 1394 Trade Association. All rights reserved.

7 Network Test

7.1 Purpose

The purpose of this test is to check that other devices on the bus are not adversely affected when the IUT is connected to the bus or begins operation.

7.2 Basic configuration and topology

Two basic configurations are used for this test, Windows OS and Mac OS Computer. If IUT is specified to operate in one or the other but not both, testing maybe restricted to the one appropriate configuration. Otherwise, IUT shall be tested in both.

Connect the following reference devices to the bus (one from each)

Windows PC

Minimum Requirements:

- Maybe 1394a or b depending on reference device requirements.
- Latest version of Windows with all relevant updates.
- At least Pentium 4 or Celeron Processor - 1.5GHz, 512Mbytes Memory, HDD 40GB, Motherboard or add-in IEEE-1394 card.
- Amcap, Windows Media Player or Quicktime

Mac PC

Minimum Requirements:

- Maybe 1394a or b depending on reference device requirements.
- Latest version of Mac OS with the with all relevant updates.
- At least G4 – 1 GHz, 512Mbytes Memory, HDD 40 GB, Motherboard or add-in 1394 card.
- iMovie, Quicktime

DV device (camcorder)

- This device shall be IEEE-1394a only (not 1394b)

SBP-2 HDD

- Maybe 1394a or b depending on reference device requirements.

Bus analyzer or equivalent

1394 TRADE ASSOCIATION

Copyright © 2008, 1394 Trade Association. All rights reserved.

Hub(s)

If only three port hubs are available (example 1394b) then multiple hubs maybe used or branching from other devices in the topology is acceptable.

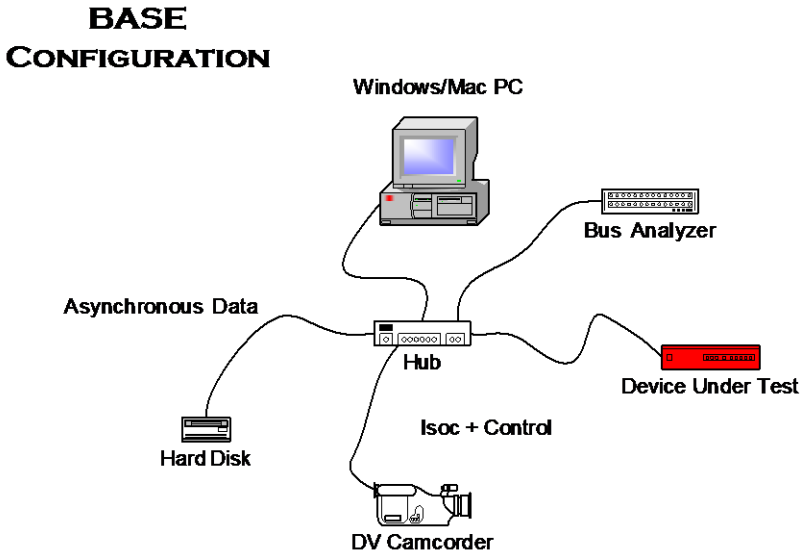

Figure 8. Base network topology diagram.

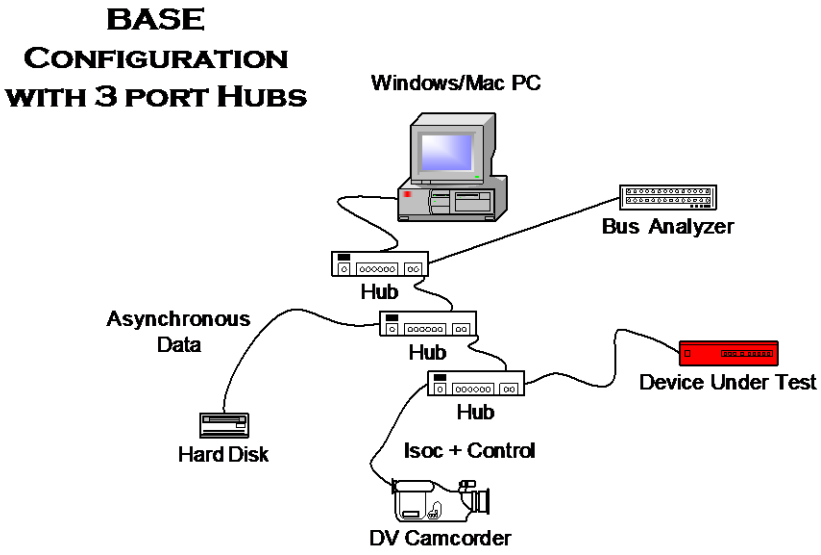

Figure 9. Base network topology with 3 port hub.

Copyright © 2008, 1394 Trade Association. All rights reserved.

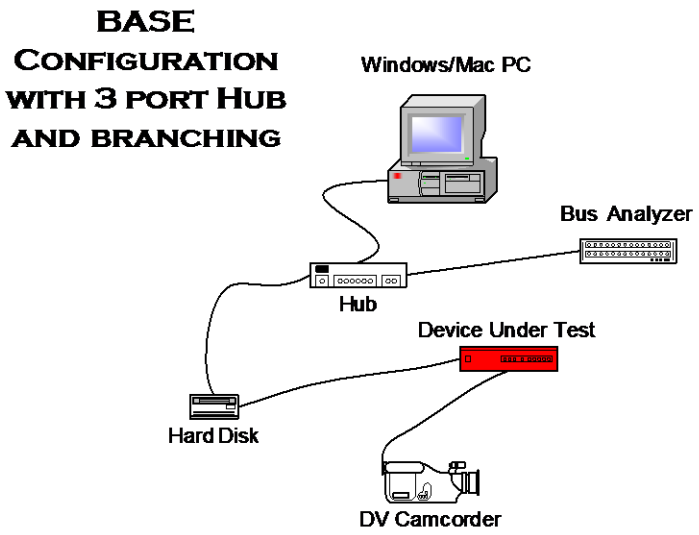

Figure 10. Base network topology with 3 port hub and branching.

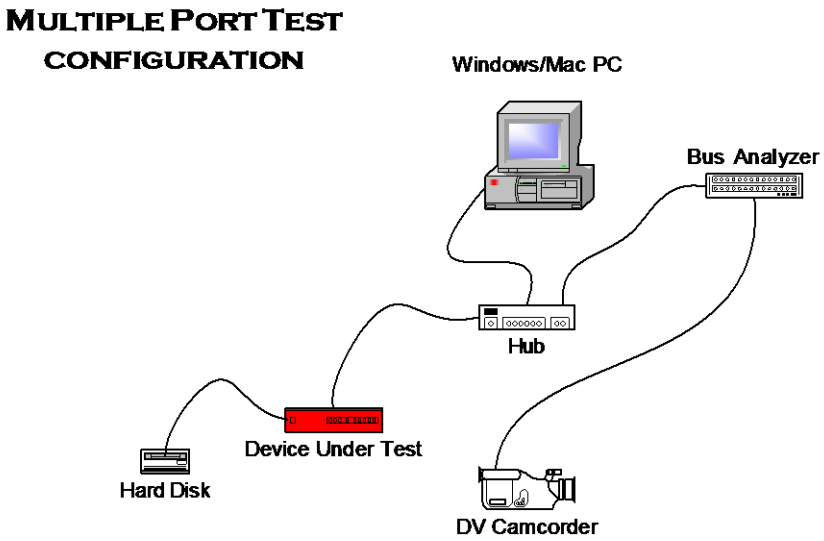

Figure 11. Multiple port test topology

Copyright © 2008, 1394 Trade Association. All rights reserved.

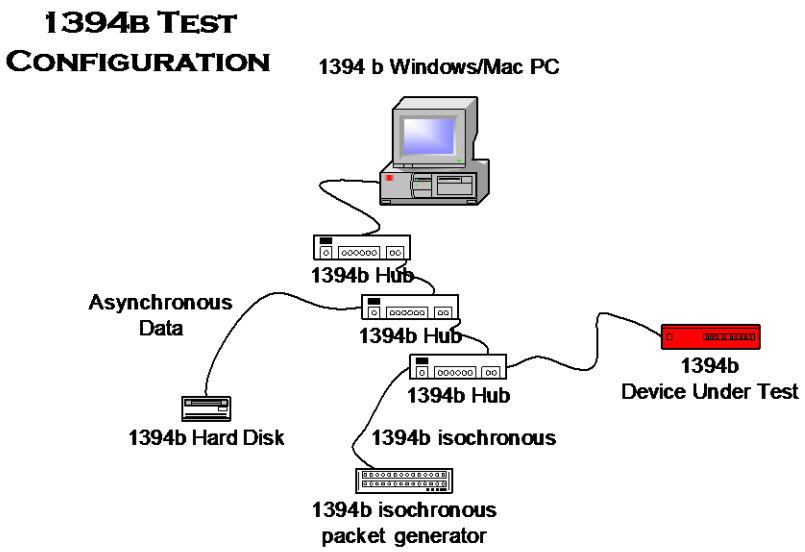

Figure 12. 1394b Test Configuration

7.3 Determination of reference device

This section is about how to select reference devices:

a) Select the device from each category according to the following order of priority.

b) Select the device with 1394 TA compliance logo from other company.

c) Select the device from other company that is (was) available on the market.

d) Select the device with 1394 TA compliance logo from tester's company.

e) Select the device from tester's company that is (was) available on the market.

7.3.1 IEEE-1394b reference devices

If IUT is an IEEE-1394b capable host then the Hub, HDD, and Windows/Mac PC shall be IEEE-1394b capable and the camcorder shall be IEEE-1394a. If IUT is not an IEEE-1394b capable host then the Hub and Windows/Mac PC are recommended to be IEEE-1394b capable. If a five port Hub is not available then multiple Hubs maybe used to enable the appropriate number of connections. If IUT is IEEE-1394b capable then Common test 6 is required. Common test 6 requires all devices be IEEE-1394b. Note: Due to a lack of availability the DV camcorder maybe replaced by a 1394b isochronous packet generator.

7.4 Common Test

Common test must be executed regardless of the types of the IUT.

This set of tests checks that both Isochronous and Asynchronous transmissions of the other devices are not affected while they are operating on the bus, when the IUT is connected and disconnected. Operations such as "file copy" may be accomplished by any procedure.

1394 TRADE ASSOCIATION
Copyright © 2008, 1394 Trade Association. All rights reserved.

7.4.1 Common test 1

Test ID	Test Description	Windows Test Result	Macintosh Test Result
NT411	Establish base topology, without IUT, as shown in Figure 8 or Figure 9 or Figure 10.	-	-
NT412	Wait for all bus traffic to stop, except cycle start and isochronous packets.	-	-
NT413	Verify Camcorder or Windows/Mac PC is root.	-	-
NT414	Connect IUT to Hub and verify:	-	-
NT415W	Is number of bus resets less than three (3)?	Yes or No	-
NT415M	Is number of bus resets less than three (3)?	-	Yes or No
NT416W	Is Camcorder or Windows PC root?	Yes or No	-
NT416M	Is Camcorder or Mac PC root?	-	Yes or No
NT417W	Is number of self-id's equal to number of nodes in topology[1]?	Yes or No	-
NT417M	Is number of self-id's equal to number of nodes in topology[1]?	-	Yes or No
NT418	Repeats steps NT414 through NT417x two more times. Did each test complete successfully?	Yes or No	Yes or No

7.4.2 Common test 2

Test ID	Test Description	Windows Test Result	Macintosh Test Result
NT421	Establish base topology, without IUT, as shown in Figure 8 or Figure 9 or Figure 10.	-	-
NT422	Wait for all bus traffic to stop, except cycle start and isochronous packets.	-	-
NT423	Verify Camcorder or Windows/Mac PC is root.	-	-
NT424	Set IUT's RHB using PHY Configuration packet, initiate bus reset and verify:	-	-
NT425W	Is number of bus resets less than three (3)?	Yes or No	-
NT425M	Is number of bus resets less than three (3)?	-	Yes or No

[1] If any nodes present in the topology has more than 3 ports the number of self-ids will increase accordingly.

Copyright © 2008, 1394 Trade Association. All rights reserved.

NT426W	Is Camcorder or Windows PC root?	Yes or No	-
NT426M	Is Camcorder or Mac PC root?	-	Yes or No
NT427W	Is number of self-id's equal to number of nodes in topology?	Yes or No	-
NT427M	Is number of self-id's equal to number of nodes in topology?	-	Yes or No
NT428	Repeats steps NT424 through NT429x two more times. Did each test complete successfully?	Yes or No	Yes or No

7.4.3 Common test 3

Test ID	Test Description	Windows Test Result	Macintosh Test Result
NT431	Establish base topology, without IUT, as shown in Figure 8 or Figure 9 or Figure 10.	-	-
NT432	Wait for all bus traffic to stop, except cycle start and isochronous packets.	-	-
NT433	Verify Camcorder or Windows/Mac PC is root.	-	-
NT434	Start capture of isochronous stream from camcorder to Windows/Mac PC	-	-
NT435	Connect IUT to Hub and verify:		
NT436W	Did connection of IUT cause little or no noticeable interruption[2] of audio/video?	Yes or No	-
NT436M	Did connection of IUT cause little or no noticeable interruption of audio/video?	-	Yes or No
NT437	Repeats steps NT426 through NT437x four more times. Did each test complete successfully?	Yes or No	Yes or No

[2] Within IEEE-1394 networks expectation of audio/video reliability are very high. However, given OS inefficiencies connection of new storage devices may disrupt other tasks being performed (such as storage of isochronous stream). The size of this disruption may be larger for optical or flash media storage devices than faster magnetic storage devices. Therefore the test operator must use their judgment when defining 'little'.

1394 TRADE ASSOCIATION

Copyright © 2008, 1394 Trade Association. All rights reserved.

7.4.4 Common test 4

Test ID	Test Description	Windows Test Result	Macintosh Test Result
NT441	Establish base topology, without IUT, as shown in Figure 8 or Figure 9 or Figure 10.	-	-
NT442	Wait for all bus traffic to stop, except cycle start and isochronous packets.	-	-
NT443	Verify Camcorder or Windows/Mac PC is root.	-	-
NT444	Start capture of isochronous stream from camcorder to Windows/Mac PC	-	-
NT445	Start maximum (supported by Windows/Mac PC and HDD) speed and size asynchronous transfers between Windows/Mac PC and HDD.	-	-
NT445	Connect IUT to Hub and verify:		
NT446W	Did connection of IUT cause little or no noticeable interruption[3] of audio/video?	Yes or No	-
NT446M	Did connection of IUT cause little or no noticeable interruption of audio/video?	-	Yes or No
NT447W	Did connection of IUT cause little or no noticeable interruption or no disruption[4] of HDD transactions?	Yes or No	-
NT447M	Did connection of IUT cause little or no noticeable interruption or no disruption of HDD transactions?	-	Yes or No
NT448	Disconnect IUT from Hub and verify:		
NT449W	Did disconnect of IUT cause little or no noticeable interruption[2] of audio/video?	Yes or No	-
NT449M	Did disconnect of IUT cause little or no noticeable interruption of audio/video?	-	Yes or No
NT4410 W	Did disconnect of IUT cause little or no noticeable interruption or no disruption of HDD transactions?	Yes or No	-

[3] Within IEEE-1394 networks expectation of audio/video reliability are very high. However, given OS inefficiencies connection of new storage devices may disrupt other tasks being performed (such as storage of isochronous stream). The size of this disruption may be larger for optical or flash media storage devices than faster magnetic storage devices. Therefore the test operator must use their judgment when defining 'little'.

[4] Some HDDs are known to not handle bus resets during data transfer gracefully. The tester should take care to verify that the reference HDD does handle bus resets during data transfer gracefully before use in this test.

Copyright © 2008, 1394 Trade Association. All rights reserved.

NT4410M	Did disconnect of IUT cause little or no noticeable interruption or no disruption of HDD transactions?	-	Yes or No
NT4411	Repeats steps NT445 through NT4410x four more times. Did each test complete successfully?	Yes or No	Yes or No

7.4.5 Common test 5 (Only required if IUT has more than one port)

Test ID	Test Description	Windows Test Result	Macintosh Test Result
NT451	Establish base topology, with IUT, as shown in Figure 11.	-	-
NT452	Wait for all bus traffic to stop, except cycle start and isochronous packets.	-	-
NT453	Verify Camcorder or Windows/Mac PC is root.	-	-
NT454	Start capture of isochronous stream from camcorder to Windows/Mac PC	-	-
NT455	Write two giga-byte or largest possible file if media is smaller than two giga-byte file to HDD (not IUT but test HDD).	-	-
NT456W	Did isochronous transfer complete with no noticeable interruption of audio/video?	Yes or No	-
NT456M	Did isochronous transfer complete with no noticeable interruption of audio/video?	-	Yes or No
NT457W	Did HDD transfer complete with no errors?	Yes or No	-
NT457M	Did HDD transfer complete with no errors?	-	Yes or No
NT458	Repeats steps NT454 through NT457x four more times. Did each test complete successfully?	Yes or No	Yes or No

1394 TRADE ASSOCIATION

Copyright © 2008, 1394 Trade Association. All rights reserved.

7.4.6 Common test 6

Test ID	Test Description	Windows Test Result	Macintosh Test Result
NT461	With Windows/Mac PC powered off establish base topology, with IUT, as shown in Figure 8 or Figure 9 or Figure 10 and if IUT is 1394b capable Figure 12.	-	-
NT462W	Power Wndows/Mac PC was IUT correctly listed in computer's device manager/registry?	Yes or No	-
NT462W	Power Wndows/Mac PC was IUT correctly listed in computer's device manager/registry?	-	Yes or No
NT463	Verify Camcorder or Windows/Mac PC is root.	-	-
NT464	Start capture of isochronous stream from camcorder to Windows/Mac PC	-	-
NT465	Write two giga-byte or largest possible file if media is smaller than two giga-byte file to HDD (not IUT but test HDD).	-	-
NT466W	Did isochronous transfer complete with no noticeable interruption of audio/video?	Yes or No	-
NT466M	Did isochronous transfer complete with no noticeable interruption of audio/video?	-	Yes or No
NT467W	Did HDD transfer complete with no errors?	Yes or No	-
NT467M	Did HDD transfer complete with no errors?	-	Yes or No
NT468	Repeats steps NT464 through NT467x four more times. Did each test complete successfully?	Yes or No	Yes or No

7.4.7 Common test 7 (Only required if IUT has 1394b PHY)

Test ID	Test Description	Windows Test Result	Macintosh Test Result
NT471	Establish base topology, with IUT, as shown in Figure 12.	-	-
NT472	Wait for all bus traffic to stop, except cycle start and isochronous packets. Start isochronous packet stream if not already started.	-	-
NT473	Verify root is sending cycle starts packets.	-	-

1394 TRADE ASSOCIATION

Copyright © 2008, 1394 Trade Association. All rights reserved.

NT474	Write two giga-byte file to HDD (not IUT but test HDD).	-	-
NT475	Was no bus reset detected?	Yes or No	Yes or No
NT476W	Did HDD transfer complete with no errors?	Yes or No	-
NT476M	Did HDD transfer complete with no errors?	-	Yes or No
NT477	Repeats steps NT464 through NT476x four more times. Did each test complete successfully?	Yes or No	Yes or No

8 Functional Test

8.1 OHCI Registers

This section contains the OHCI Registers test. This test focuses on the register values after a soft reset (setting of the softReset bit in the HCControl register), bus reset, and normal operation conditions.

8.1.1 Soft Reset Test for all Registers

When the HCControl.softReset is set to 1, a soft reset occurs, all FIFO's are flushed and all Host Controller registers are set to their hardware reset values, unless otherwise specified. Registers outside of the Open HCI realm, i.e., host attachment registers such as those for PCI, are not affected.
***The read value of this bit shall be 1 while a soft reset or a hard reset is in progress. The read value of this bit shall be 0 when neither a soft reset nor hard reset are in progress. Software can use the value of this bit to determine when a reset has completed and the Host Controller is safe to operate.

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
RG11	Verify all register values after Soft Reset	**Variables definition:** timeout = 1 second default_reg_values = Registers hardware default value soft_reset_reg_values = Register values after Soft Reset. **Returns:** soft_reset_field_values		OHCI -5.2
		Using Open HCI implementation specification enter all registers hardware default value. (default_reg_values)		
		Write the HCControl.softReset register bit with 1b'1.		
		Wait until the HCControl.softReset register bit is 1b'0 or timeout expires		
		If timeout expires and the HCControl.softReset register bit is 1b'1: soft_reset_field_values = **FAILED** Else: continue		
		Read all Registers (soft_reset_reg_values)		
		If (default_reg_values == soft_reset_reg_values) then soft_reset_field_values = **PASSED** Else: soft_reset_field_values = **FAILED**		

1394 TRADE ASSOCIATION
Copyright © 2008, 1394 Trade Association. All rights reserved.

8.1.2 Bus Reset Test for all Registers

The following register are effect by bus reset:

Bus Management CSR Registers

- 5.5.1
-

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
RG21	Verify all register values after Soft Reset	**Variables definition:** timeout = 1 second default_reg_values = Registers hardware default value soft_reset_reg_values = Register values after Soft Reset. **Returns:** soft_reset_field_values		OHCI -5.2
		Using Open HCI implementation specification enter all registers hardware default value. (default_reg_values)		
		Write the HCControl.softReset register bit with 1b'1.		
		Wait until the HCControl.softReset register bit is 1b'0 or timeout expires		
		If timeout expires and the HCControl.softReset register bit is 1b'1: soft_reset_field_values = **FAILED** Else: continue		
		Read all Registers (soft_reset_reg_values)		
		If (default_reg_values == soft_reset_reg_values) then soft_reset_field_values = **PASSED** Else: soft_reset_field_values = **FAILED**		

8.1.3 Version Register

The Version Register shall be tested for compliance with the Open HCI specification by performing the tests defined in this section.

8.1.3.1 Version Register Test

The Version Register fields are all read only.

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
RG31	Read the Version Register and retrieve 4 bytes.	**Variables definition:** timeout = 1 second Version = 4 byte of Version Register Version.GUID_ROM = bit 24 of Version register Version.version = bits 23:16 of Version register Version.revision = bits 7:0 of Version register **Returns:** Version.GUID_ROM Version.version Version.revision		OHCI -5.2
		Read Version register.		
		Wait until the GUID_ROM.addrReset register bit is 1b'0 or timeout expires		
		If timeout expires then: Verdict.GUID_ROM = **NCL,** Verdict.version = **NCL,**		

Copyright © 2008, 1394 Trade Association. All rights reserved.

		Verdict.revision = **NCL** Else: continue		
		If Version.GUID_ROM == P_GUID_ROM then: Verdict.GUID_ROM = **PASSED** Else: Verdict.GUID_ROM = **FAILED**		
		If Version.version == P_Version then: Verdict.version = **PASSED** Else: Verdict.version = **FAILED**		
		If Version.revision == P_Revision then: Verdict.revision = **PASSED** Else: Verdict.revision = **FAILED**		

8.1.3.2 Version Register Test

No validation test is defined beyond those defined in the access test.

8.1.4 GUID ROM Register

The GUID ROM Register shall be tested for compliance with the Open HCI specification by performing the tests defined in this section. This test procedure shall be performed only if the Version.GUID_ROM bit is set.

8.1.4.1 GUID ROM Register Access Test

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
RG41	Reset the GUID ROM and retrieve 8 bytes.	**Variables definition:** timeout = 1 second GUID_ROM_rddata[8]= array of first 8 bytes of the GUID ROM **Returns:** GUID_ROM_addrreset_set, GUID_ROM_rdstart_set, GUID_ROM_rddata_read8		OHCI -5.3
		Write the GUID_ROM.addrReset register bit with 1b'1.		
		Wait until the GUID_ROM.addrReset register bit is 1b'0 or timeout expires		
		If timeout expires and the GUID_ROM.addrReset register bit is 1b'1: GUID_ROM_addrreset_set = **FAILED,** GUID_ROM_rdstart_set = **NCL,** GUID_ROM_rddata_read8 = **NCL** Else: continue		
		GUID_ROM_addrreset_set = **PASSED**		
		Begin loop n from 0 to 7 (8 iterations) to index GUID_ROM_rddata[n]		
		Write the GUID_ROM_rdstart register bit with 1b'1.		
		Wait until the GUID_ROM.rdstart register bit is 1b'0 or timeout expires		
		If timeout expires and the GUID_ROM. rdstart register bit is 1b'1: GUID_ROM_ rdstart _set = **FAILED,** GUID_ROM_rddata_read8 = **NCL** Else: continue		
		GUID_ROM_rdstart_set = **PASSED**		
		Read GUID_ROM.rdData register value (GUID_ROM_rddata[n])		
		Repeat loop n		
		If GUID_ROM_rddata[0:7] == P_GUID: GUID_ROM_rddata_read8 = **PASSED** **Else: GUID_ROM_rddata_read8 = FAILED**		

Copyright © 2008, 1394 Trade Association. All rights reserved.

8.1.4.2 GUID ROM Register Validation Test

No validation test is defined beyond those defined in the access test.

8.1.5 ATRetries Register

The ATRetries Register shall be tested for compliance with the Open HCI specification by performing the tests defined in this section. The secondLimit and cycleLimit are only implemented if the device supports outbound dual-phase retry.

8.1.5.1 ATRetries Register Test

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
RG51	Write to ATRetries register read/write fields to confirm registers are read/write capable and independent fields.	**Variables definition:** read_ ATRetries _reg_value = Value of the ATRetries Register after the write **Returns:** ATRetries_MaxPhysRespRetries _writeread, ATRetries_MaxATRespRetries _writeread, ATRetries_MaxATReqRetries _writeread		OHCI -5.4
		Write 0x00000FFF to ATRetries register		
		Read ATRetries register		
		If (ATRetries register == value written) then continue Else: ATRetries_MaxPhysRespRetries _writeread, ATRetries_MaxATRespRetries _writeread, ATRetries_MaxATReqRetries _writeread= **FAILED**		
		Write 0x00000A65 to ATRetries register		
		Read ATRetries register		
		If (ATRetries register == value written) then continue Else: ATRetries_MaxPhysRespRetries _writeread, ATRetries_MaxATRespRetries _writeread, ATRetries_MaxATReqRetries _writeread= **FAILED**		
		ATRetries_MaxPhysRespRetries _writeread, ATRetries_MaxATRespRetries _writeread, ATRetries_MaxATReqRetries _writeread= **PASSED**		
RG52	Write to secondLimit and cycleLimit fields to confirm registers are read only if outbound dual-phase retry is not supported and read/write capable and independent fields	Variables definition: read_ATRetries_reg_value = Value of the ATRetries Register after the write ATRetries_secondLimit_writeread, ATRetries_cycleLimit_writeread		
		Write 0xFFFF0000 to ATRetries register		
		Read ATRetries register		
		If (ATRetries register == value written) then continue Else: ATRetries_secondLimit _writeread, ATRetries_cycleLimit _writeread, = **FAILED**		
		Write 0x65550000 to ATRetries register		

Copyright © 2008, 1394 Trade Association. All rights reserved.

		Read ATRetries register		
		If (ATRetries register == value written) then continue Else: ATRetries_secondLimit _writeread, ATRetries_cycleLimit _writeread, = **FAILED**		
		ATRetries_secondLimit _writeread, ATRetries_cycleLimit _writeread, = PASSE**D**		

8.1.5.2 ATRetries Register Validation Test

No validation test is defined beyond those defined in the access test.

8.1.6 Bus Management Register

The Bus Management Registers shall be tested for compliance with the Open HCI specification by performing the tests defined in this section.

8.1.6.1 Bus Management Register Access Test

Tests of the Bus Management registers (BUS_MANAGER_ID, BANDWIDTH_AVAILABLE, CHANNELS_AVAILABLE_HI, and CHANNELS_AVAILABLE_LO) are defined in the Base 1394 Test Suite Definition with Extension for 1394b.

8.1.7 Config ROM Header Register

The Config ROM Header Register shall be tested for compliance with the Open HCI specification by performing the tests defined in this section.

8.1.7.1 Config ROM Header Access Test

Tests for the Config ROM header register are defined in the Base 1394 Test Suite Definition with Extension for 1394b.

8.1.8 Bus Identification Register

The Bus Identification register shall be tested for compliance with the Open HCI specification by performing the tests defined in this section.

8.1.8.1 Bus Identification Register Access Test

Tests for the Bus Identification register are defined in the Base 1394 Test Suite Definition with Extension for 1394b.

1394 TRADE ASSOCIATION

Copyright © 2008, 1394 Trade Association. All rights reserved.

8.1.9 Bus Options Register

The Bus Options Register shall be tested for compliance with the Open HCI specification by performing the tests defined in this section.

8.1.9.1 Bus Options Register Access Test

Tests for the Bus Options register are defined in the Base 1394 Test Suite Definition with Extension for 1394b.

8.1.10 Global Unique ID Register

The Global Unique ID Register shall be tested for compliance with the Open HCI specification by performing the tests defined in this section.

8.1.10.1 Global Unique ID Register Access Test

The Bus Options Register shall be tested for compliance with the Open HCI specification by performing the tests defined in this section.

8.1.10.1.1 Global Unique ID Register Access Test Procedure

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
RG81	Retrieve Global Unique ID	**Variables definition:** read_ Global Unique ID _reg_value = value read from Global Unique ID register. **Returns:** Global Unique ID _retrieve		OHCI -5.5.5
		Read the Global Unique ID register (read_ Global Unique ID _reg_value)		
		Global Unique ID _retrieve = **PASSED**		

8.1.10.2 Global Unique ID Register Validation Test

Functional verification of these registers is tested in the Base 1394 Test Suite Definition with Extensions for 1394b.

8.1.11 Configuration ROM Mapping Register

The Configuration ROM Mapping Register shall be tested for compliance with the Open HCI specification by performing the tests defined in this section.

8.1.11.1 Configuration ROM Mapping Register Access Test

This test procedure shall verify that the read / write fields of the Configuration ROM Mapping Register can be written to and read from.

8.1.11.1.1 Configuration ROM Mapping Register Access Test Procedure

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
RG91	Write to the Configuration ROM Mapping register read/write fields to confirm registers are read/write.	**Variables definition:** default_Configuration_ROM_Mapping_reg_value = Value of Configuration_ROM_Mapping register before the write read_ Configuration_ROM_Mapping _reg_value = Value of		OHCI -5.5.6

Copyright © 2008, 1394 Trade Association. All rights reserved.

		Configuration_ROM_Mapping register after the write **Returns:** Configuration_ROM_Mapping _writeread		
		Read the Configuration_ROM_Mapping regoster (default_ Configuration_ROM_Mapping _reg_value)		
		Write (default_ Configuration_ROM_Mapping _reg_value XOR 0xFFFFFC00) to Configuration_ROM_Mapping register		
		Read the Configuration_ROM_Mapping register (read_ Configuration_ROM_Mapping _reg_value)		
		If (read_Configuration_ROM_Mapping_reg_value == (default_Configuration_ROM_Mapping_reg_value XOR 0xFFFFFC00)) : Configuration_ROM_Mapping_writeread = **PASSED** Else Configuration_ROM_Mapping_writeread = **FAILED**		

8.1.11.2 Configuration ROM Mapping Register Verification Test

Functional verification of these registers is tested in the Base 1394 Test Suite Definition with Extensions for 1394b.

8.1.12 Vendor ID Register

The Vendor ID Register shall be tested for compliance with the Open HCI specification by performing the tests defined in this section.

8.1.12.1 Vendor ID Register Access Test

8.1.12.1.1 Vendor_ID Register Access Test Procedure

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
RG111	Retrieve Vendor_ID register value to be compared to in a subsequent test	**Variables definition:** read_Vendor_ID_reg_value = value read from the Vendor_ID register. **Returns:** Vendor_ID_retrieve		OHCI -5.6
		Read the Vendor_ID register (read_Vendor_ID_reg_value)		
		Vendor_ID_retrieve = **PASSED**		

8.1.12.2 Vendor ID Register Test

Functional verification of these registers is tested in the Base 1394 Test Suite Definition with Extensions for 1394b.

8.1.13 HCControl Registers

The HCControl Registers shall be tested for compliance with the Open HCI specification by performing the tests defined in this section.

8.1.13.1 HCControl Register Access Test

The test implementation shall read the value of the HCControl Register. If the HCControl Register cannot be read the test shall have failed. If the value of the linkEnable bit of the HCControl Register is not equal to 1'b0 the test shall have failed. If the value of the LPS bit of the HCControl Register is not equal to 1'b0 the test shall have failed. If the value of the LinkEnable bit of the HCControl Register is not equal to 1'b0 the test shall have failed.

1394 TRADE ASSOCIATION

Copyright © 2008, 1394 Trade Association. All rights reserved.

8.1.13.1.1 HCControl Register Access SET/CLEAR Fields Test

The test implementation shall attempt to write a 1'b1 to the HCControl Register SET fields. The test implementation shall then read the HCControl Register and verify the contents. If the content of the HCControl Register SET Fields are not set, then the test shall have failed.

The test implementation shall attempt to write a 1'b1 to the HCControl Register CLEAR fields. The test implementation shall then read the HCControl Register and verify the contents. If the content of the HCControl Register CLEAR Fields are not cleared, then the test shall have failed.

8.1.13.1.2 HCControl Regsiter Access SET/CLEAR Fields Test Procedure

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
RG113	Write to the HCControl register SET fields to confirm registers are settable.	**Variables definition:** read_set_HCControl _reg_value = Value of HCControl register after the Set. **Returns:** HCControl_set		OHCI -5.7
		Write 32'h400E0000 to the HCControlSet register		
		Read the HCControl register (read_set_HCControl_reg_value)		
		If ((read_HCControl_reg_value AND 32'h400E0000) == 32'h400E0000) : HCControl_set = **PASSED** Else HCControl_set = **FAILED**		
RG114	Write to the HCControl register CLEAR fields to confirm registers are clearable.	Variables definition: read_clear_HCControl_reg_value = Value of HCControl register after the clear. Returns: HCControl_clear		OHCI -5.7
		Write 32'h40040000 to the HCControlClear register		
		Read the HCControl register (read_clear_HCControl_reg_value)		
		If ((read_HCControl_reg_value AND 32'h40040000) == 32'h40040000) : HCControl_clear = PASSED Else HCControl_clear = FAILED		

8.1.13.2 HCControl Register Verification Test

The bits within the HCControl register are verified in the following sections.

Field Name	Section
noByteSwapData	Implementation/OS specific
programPhyEnable	See the test below
aPhyEnhanceEnable	See the test below
LPS	Various sections within Chapter x

Copyright © 2008, 1394 Trade Association. All rights reserved.

postedWriteEnable	Tested in Physical DMA tests
linkEnable	Tested in 8.1.1
softReset	Tested in 8.1.1

8.1.13.2.1 programPhyEnable and aPhyEnhanceEnable Verification Test Procedure

The Acceleration Control request is generated by the Open HCI to enable/disable 1394a defined accelerated arbitration functions. This test verifies the operation of the acceleration control and the HCControl.programPhyEnable and HCControl.aPhyEnhanceEnable bits.

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
RG121	With IUT root and cycle master and programPhyEnable = 1 and aPhyEnhanceEnable = 0 verify AccCtrl request are not being generated for every isochronous cycle	**Variable definitions:** **Returns:** RG121		OHCI – 5.7, OHCI – 5.7.2, IEEE-1394a-2000 – 5A.3, 5A.3.1 and 5A.3.2
		Force IUT to become root and cycle master		
		Set programPhyEnable to one		
		Set aPhyEnhanceEnable to zero		

		If no accelerated ctrl request are being generated then
		RG121 = PASSED
		Else
		RG121 = FAILED		

RG122	With IUT cycle slave (not root) and programPhyEnable = 1 and aPhyEnhanceEnable = 0 verify AccCtrl request are not being generated for every isochronous cycle	**Variable definitions:** **Returns:** RG122		OHCI – 5.7, OHCI – 5.7.2, IEEE-1394a-2000 – 5A.3, 5A.3.1 and 5A.3.2
		Force IUT to not be root		
		Set programPhyEnable to one		
		Set aPhyEnhanceEnable to zero		

		If no accelerated ctrl request are being generated then
		RG122 = PASSED
		Else
		RG122 = FAILED		

RG123	With IUT root and cycle master and programPhyEnable = 1 and aPhyEnhanceEnable = 1 verify AccCtrl request are not being generated for every isochronous cycle	**Variable definitions:** **Returns:** RG123		OHCI – 5.7, OHCI – 5.7.2, IEEE-1394a-2000 – 5A.3, 5A.3.1 and 5A.3.2
		Force IUT to become root and cycle master		
		Set programPhyEnable to one		

1394 TRADE ASSOCIATION

Copyright © 2008, 1394 Trade Association. All rights reserved.

		Set aPhyEnhanceEnable to one		

		If accelerated ctrl requests are not being generated then
		RG123 = PASSED
		Else
		RG123 = FAILED		

RG124	With IUT cycle slave (not root) and programPhyEnable = 1 and aPhyEnhanceEnable = 1 verify AccCtrl request are being generated for every isochronous cycle	**Variable definitions:** **Returns:** RG124		OHCI – 5.7, OHCI – 5.7.2, IEEE-1394a-2000 – 5A.3, 5A.3.1 and 5A.3.2
		Force IUT to not be root		
		Set programPhyEnable to one		
		Set aPhyEnhanceEnable to zero		

		If accelerated ctrl requests with Accelerated set to zero for each cycle sync are being generated and accelerated ctrl requests with Accelerated set to one after each cycle start packet is observed then
		RG124 = PASSED
		Else
		RG124 = FAILED		

RG125	With IUT cycle slave (not root) and programPhyEnable = 1 and aPhyEnhanceEnable = 1 and transmitting isochronous streams verify AccCtrl request to disable acceleration are being generated for every isochronous cycle but AccCtrl request to enable are not.	**Variable definitions:** **Returns:** RG125		OHCI – 5.7, OHCI – 5.7.2, IEEE-1394a-2000 – 5A.3, 5A.3.1 and 5A.3.2
		Force IUT to not be root		
		Set programPhyEnable to one		
		Set aPhyEnhanceEnable to zero		
		Have IUT send isochronous stream		

		If accelerated ctrl requests with Accelerated set to zero for each cycle sync are being generated and no Accelerated set to one after transmitted isochronous packet is observed then
		RG125 = PASSED
		Else
		RG125 = FAILED		

8.1.14 Fairness Control Register

The Fairness Control Register shall be tested for compliance with the Open HCI specification by performing the tests defined in this section.

8.1.14.1 Fariness Control Register Access Test

This test procedure shall verify that the Open HCI implementation correctly sets value written to the Fairness Control Register and returns that value when read.

8.1.14.1.1 Fairness Control Register Access Test Procedure

Copyright © 2008, 1394 Trade Association. All rights reserved.

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
RG131	Write and Read Fairness Control register value and compared	**Variables definition:** timeout = 1 second read_Fairness_Control_reg_value = value read from the Fairness Control register. write_Fairness_Control_reg_value **Returns:** Fairness_Control_retrieve		OHCI -5.8
		Write the Fairness Control register (write_Fairness_Control_reg_value = 0x000000FF)		
		Read the Fairness Control register (read_Fairness_Control_reg_value		
		If Fairness_Control_retrieve == write_Fairness_Control_reg_value Then RG131 = **PASSED** Else RG132 = **FAILED**		
RG132		Write the Fairness Control register (write_Fairness_Control_reg_value = 0x00000000)		
		Read the Fairness Control register (read_Fairness_Control_reg_value		
		If Fairness_Control_retrieve == write_Fairness_Control_reg_value Then RG132 = **PASSED** Else RG132 - **FAILED**		

8.1.14.2 Fairness Control Register Verification Test

There is no test defined beyond those tests defined in the access test section.

8.1.15 LinkControl Register

The LinkControl Register shall be tested for compliance with the Open HCI specification by performing the tests defined in this section.

8.1.15.1 LinkControl SET Register Access Test

The test implementation shall attempt to write a 1’b1 to the LinkControl Register SET fields. The test implementation shall then read the LinkControl Register and verify the contents. If the content of the LinkControl Register SET Fields are not set, then the test shall have failed.

8.1.15.1.1 LinkControl SET Register Access Test Procedure

Copyright © 2008, 1394 Trade Association. All rights reserved.

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
RG141	Write to the LinkControl register SET fields to confirm registers are settable.	**Variables definition:** read_set_LinkControl_reg_value = Value of LinkControl register after the Set. **Returns:** LinkControl_set		OHCI -5.10
		Write 32'h00300600 to the LinkControlSet register		
		Read the LinkControl register (read_set_LinkControl_reg_value)		
		If ((read_LinkControl_reg_value AND 32'h00300600) == 32'h00300600): LinkControl_set = **PASSED** Else LinkControl_set = **FAILED**		

8.1.15.2 LinkControl CLEAR Register Access Test

The test implementation shall attempt to write a 1'b1 to the LinkControl Register CLEAR fields. The test implementation shall then read the LinkControl Register and verify the contents. If the content of the LinkControl Register CLEAR Fields are not cleared, then the test shall have failed.

8.1.15.2.1 LinkControl CLEAR Regsiter Access Test Procedure

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
RG142	Write to the LinkControl register CLEAR fields to confirm registers are clearable.	**Variables definition:** read_clear_LinkControl_reg_value = Value of LinkControl register after the clear. **Returns:** LinkControl_clear		OHCI -5.10
		Write 32'h00300600 to the LinkControlClear register		
		Read the LinkControl register (read_clear_LinkControl_reg_value)		
		If ((read_LinkControl_reg_value AND 32'h00300600) == 0x00000000) : LinkControl_clear = **PASSED** Else LinkControl_clear = **FAILED**		

8.1.15.3 LinkControl Register Validation Test

This test procedure shall verify that the Link Control Register can be used to configure and enable the link core protocol portions of the Open HCI implementation.

8.1.15.3.1 LinkControl.cycleSource Field

The LinkControl.cycleSource field enables the use of an external clock source for cycleTimer.cycleCount. This IUT does not implement an external cycle source therefore LinkControl.cycleSource is read-only and always cleared.

8.1.15.3.2 LinkControl.cycleMaster Field

Functional verification of these registers is tested in the Base 1394 Test Suite Definition with Extensions for 1394b.

8.1.15.3.3 LinkControl.cycleTimerEnable Field

When LinkControl.cycleTimerEnabled is set, the cycleTimer is enabled to count 24.576 MHz clocks, 125us isochronous periods, and seconds. When LinkControl.cycleTimerEnabled is cleared, the cycleTimer is disabled.

1394 TRADE ASSOCIATION

Copyright © 2008, 1394 Trade Association. All rights reserved.

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
RG1431	With the IUT root and cycleMaster, verify cycleTimerEnable starts and stop cycleTimer	**Variables definition:** **Returns:** RG1431		OHCI -5.9, 5-10, 5-11, 5-12
		Have IUT and Tester clear LinkControl.cycleMaster		
		Have IUT and Tester set LinkControl.cycleTimerEnable.		
		Have the IUT set the PHY Root Holdoff Bit (RHB) in PHY base register 1 using the PHY control register.		
		Initiate a bus reset		
		AT IntEvent.selfIDComplete IF NodeID.root == 1 THEN AT IntEvent.cycle64Seconds Set IntEventClear.cycle64Seconds FOR SECONDS = 1 to 64 FOR COUNT = 1 to 8000 AT IntEvent.cycleSynch IF IntEventSet.cycle64Seconds == 1 THEN RG31 = FAIL ELSE CONTINUE END FOR COUNT END FOR SECONDS IF IntEventSet.cycle64Seconds == 1 THEN RG31 = PASS ELSE RG31 = FAIL ELSE RG31 = FAIL		
		Have IUT clear LinkControl.cycleTimerEnable.		
		IF IntEvent.cycleSynch or IntEventSet.cycle64Seconds occurs THEN RG1431 = FAIL ELSE RG1431 = PASS		

8.1.15.3.4 LinkControl.rcvPhyPkt Field

The LinkControl.rcvPhyPkt field is covered in section ???.

8.1.15.3.5 LinkControl.rcvSelfID Field

This field is implicitly test in all DMA tests

8.1.16 Node Identification and Status Register

The Node Identification and Status Register shall be tested for compliance with the Open HCI specification by performing the tests defined in this section.

8.1.16.1 Node Identification and Status Register Access Test

This test procedure shall verify that the read only fields of the Node Identification and Status Register cannot be written to and that the set fields of the Node Identification and Status Register can be set.

8.1.16.1.1 Node Identification and Status Register Access Test Procedure

Test	Test Description	Procedure	Pass or	Specification

Copyright © 2008, 1394 Trade Association. All rights reserved.

Number			Fail	Reference
RG151	Write to the Node Identification and Status register read/write fields to confirm registers are read/write.	**Variables definition:** default_Node_Identification_and_Status _reg_value = Value of Node Identification and Status register before the write read Node_Identification_and_Status _reg_value = Value of Node Identification and Status register after the write **Returns:** Node_Identification_and_Status _writeread		OHCI -5.12
		Read the Node_Identification_and_Status register (default_Node_Identification_and_Status _reg_value)		
		Write (default_Node_Identification_and_Status _reg_value XOR 0x0000FFC0) to Node Identification and Status register		
		Read the Node Identification and Status register (read_Node_Identification_and_Status _reg_value)		
		If (read_Node_Identification_and_Status _reg_value == (default_Node_Identification_and_Status _reg_value XOR 0x0000FFC0)) : Node_Identification_and_Status _writeread = **PASSED** Else Node_Identification_and_Status _writeread = **FAILED**		

8.1.16.2 Node Identification and Status Register Validation Test

Functional verification of these registers is tested in the Base 1394 Test Suite Definition with Extensions for 1394b.

8.1.17 PHY Control Register

The PHY Control Register shall be tested for compliance with the Open HCI specification by performing the tests defined in this section.

8.1.17.1 PHY Control Register Access

There is no test defined beyond those tests defined in the validation test section.

8.1.17.2 PHY Control Register Validation Test

8.1.17.2.1 PHY Control Register Validation Test Procedure

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
RG163	Read all base PHY registers and write PHY register 1	**Variables definition:** default_PHY_Control_reg_value = Value of PHY Control register before the write read PHY_Control_reg_value = Value of PHY Control register after the write writePHY_Control_reg_value = Value to be written to PHY Control register PHY_Register [0:15] = Array used to hold the PHY Register contents default_PHY_Register [0:15] = Known PHY register contents PHY_Register_write = **Returns:**		OHCI -5.13

Copyright © 2008, 1394 Trade Association. All rights reserved.

		PHY_Control_read PHY_Control_write		
		Read the PHY register 0 through 15 using PHY_Control Register (PHY_Register[])		
		If PHY_Register[] == default_PHY_Register [] Then PHY_Control_read = PASSED Else PHY_Control_read = FAILED		
		Write PHY register 1 gap_count field to value invert to the value read above (PHY_Register_write = !PHY_Register.gap_count[1]) using PHY_Control register		
		Read the PHY register 1 (PHY_Regsiter[1])		
		If PHY_Register[1] = PHY_Register_write Then PHY_Control_write = PASSED Else PHY_Control_write = FAILED		

8.1.18 Isochronous Cycle Timer Register

The Isochronous Cycle Time Register shall be tested for compliance with the Open HCI specification by performing the tests defined in this section.

8.1.18.1 Isochronous Cycle Timer Register Access

This test procedure shall verify that the read only fields of the Isochronous Cycle Time Register cannot be written to and that the set fields of the Isochronous Cycle Time Register can be set.

8.1.18.1.1 Isochronous Cycle Timer Register Access Test Procedure

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
RG171	Write to the Isochronous Cycle Time register read/write fields to confirm registers are read/write.	**Variables definition:** default_Isochronous_Cycle_Time_reg_value = Value of Isochronous Cycle Time register before the write read Isochronous_Cycle_Time_reg_value = Value of Isochronous Cycle Time register after the write **Returns:** Isochronous_Cycle_Time_writeread		OHCI -5.13
		Read the Isochronous Cycle Time register (default_Isochronous_Cycle_Time_reg_value)		
		Write (default_Isochronous_Cycle_Time_reg_value XOR 0xFFFFFFFF) to Isochronous Cycle Time register		
		Read the Isochronous Cycle Time register (read_Isochronous_Cycle_Time_reg_value)		
		If (read_Isochronous_Cycle_Time_reg_value == (default_Isochronous_Cycle_Time_reg_value XOR 0xFFFFFFFF)) : Isochronous_Cycle_Time_writeread = **PASSED** Else Isochronous_Cycle_Time_writeread = **FAILED**		

8.1.18.2 Isochronous Cycle Timer Register Validation Test

The Isochronous Cycle Time register is functionally verified in Base 1394 Test Suite Definition.

8.1.19 Asynchronous Request Filters Registers

The Asynchronous Request Filters Register shall be tested for compliance with the Open HCI specification by performing the tests defined in this section.

1394 TRADE ASSOCIATION

Copyright © 2008, 1394 Trade Association. All rights reserved.

8.1.19.1 AsynchronousRequestFilter Registers Access Test

The test implementation shall read the values of the AsynchronousRequestFilter Registers. If the AsynchronousRequestFilter Registers cannot be read the test shall have failed. If the values of the AsynchronousRequestFilter Registers are not 32'h0000_0000 the test shall have failed.

8.1.19.1.1 AsychronousReqeustFilter Registers Access SET

The test implementation shall attempt to write a 1'b1 to the AsynchronousRequestFilter Register SET fields. The test implementation shall then read the AsynchronousRequestFilter Register and verify the contents. If the content of the AsynchronousRequestFilter Register SET Fields are not set, then the test shall have failed.

8.1.19.1.2 AsychronousReqeustFilter Registers Access SET Test Procedure

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
RG183	Write to the Asynchronous_Request_Filters register SET fields to confirm registers are settable.	**Variables definition:** read_set_Asynchronous_Request_Filters_reg_value = Value of Asynchronous_Request_Filters register after the Set. **Returns:** Asynchronous_Request_Filters_set		OHCI -5.14
		Write 32'hFFFFFFFF to the Asynchronous_Request_FiltersSet register		
		Read the Asynchronous_Request_Filters register (read_set_Asynchronous_Request_Filters_reg_value)		
		If (read_Asynchronous_Request_Filters_reg_value == 0xFFFFFFFF) : Asynchronous_Request_Filters_set = **PASSED** Else Asynchronous_Request_Filters_set = **FAILED**		

8.1.19.1.3 AsychronousReqeustFilter Registers Access CLEAR

The test implementation shall attempt to write a 1'b1 to the AsynchronousRequestFilter Register CLEAR fields. The test implementation shall then read the AsynchronousRequestFilter Register and verify the contents. If the content of the AsynchronousRequestFilter Register CLEAR Fields are not cleared, then the test shall have failed.

8.1.19.1.4 AsychronousReqeustFilter Registers Access CLEAR Test Procedure

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
RG184	Write to the Asynchronous_Request_Filters register CLEAR fields to confirm registers are clearable.	**Variables definition:** read_clear_Asynchronous_Request_Filters_reg_value = Value of Asynchronous_Request_Filters register after the clear. **Returns:** Asynchronous_Request_Filters_clear		OHCI -5.14
		Write 32'hFFFFFFFF to the Asynchronous_Request_FiltersClear register		
		Read the Asynchronous_Request_Filters register (read_clear_Asynchronous_Request_Filters_reg_value)		
		If (read_Asynchronous_Request_Filters_reg_value == 0x00000000) : Asynchronous_Request_Filters_clear = **PASSED** Else Asynchronous_Request_Filters_clear = **FAILED**		

1394 TRADE ASSOCIATION

Copyright © 2008, 1394 Trade Association. All rights reserved.

8.1.19.2 AsynchronousRequestFilter Registers Validation Test

The AsychronousRequestFilter registers are tested in the asynchronous DMA tests.

8.1.20 PhysicalRequestFilter Registers

The PhysicalRequestFilter Register shall be tested for compliance with the Open HCI specification by performing the tests defined in this section.

8.1.20.1 PhysicalRequestFilter Registers Access Test

The test implementation shall read the values of the PhysicalRequestFilter Registers. If the PhysicalRequestFilter Registers cannot be read the test shall have failed. If the values of the PhysicalRequestFilter Registers are not 32'h0000_0000 the test shall have failed.

8.1.20.1.1 PhysicalRequestFilter Register Access SET Test

The test implementation shall attempt to write a 1'b1 to the PhysicalRequestFilterRegister SET fields. The test implementation shall then read the PhysicalRequestFilterRegister and verify the contents. If the content of the PhysicalRequestFilterRegister SET Fields are not set, then the test shall have failed.

8.1.20.1.2 PhysicalRequestFilter Regsiter Access SET Test Procedure

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
RG188	Write to the PhysicalRequestFilter register SET fields to confirm registers are settable.	**Variables definition:** read_set_PhysicalRequestFilter_reg_value = Value of PhysicalRequestFilter register after the Set. **Returns:** PhysicalRequestFilter_set		OHCI -5.14.2
		Write 32'hFFFFFFFF to the PhysicalRequestFilterSet register		
		Read the PhysicalRequestFilter register (read_set_PhysicalRequestFilter_reg_value)		
		If (read_PhysicalRequestFilter_reg_value == 0xFFFFFFFF) : PhysicalRequestFilter_set = **PASSED** Else PhysicalRequestFilter_set = **FAILED**		

8.1.20.1.3 PhysicalRequestFilter Register Access CLEAR Test

The test implementation shall attempt to write a 1'b1 to the PhysicalRequestFilter Register CLEAR fields. The test implementation shall then read the PhysicalRequestFilter Register and verify the contents. If the content of the PhysicalRequestFilter Register CLEAR Fields are not cleared, then the test shall have failed.

8.1.20.1.4 PhysicalRequestFilter Regsiter Access CLEAR Test Procedure

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
RG189	Write to the PhysicalRequestFilter register CLEAR fields to confirm registers are clearable.	**Variables definition:** read_clear_PhysicalRequestFilter_reg_value = Value of PhysicalRequestFilter register after the clear. **Returns:** PhysicalRequestFilter_clear		OHCI -5.14.2
		Write 32'hFFFFFFFF to the PhysicalRequestFilterClear register		

Copyright © 2008, 1394 Trade Association. All rights reserved.

		Read the PhysicalRequestFilter register (read_clear_PhysicalRequestFilter_reg_value)		
		If (read_PhysicalRequestFilter_reg_value == 0x00000000) : PhysicalRequestFilter_clear = **PASSED** Else PhysicalRequestFilter_clear = **FAILED**		

8.1.20.1.5 Functional Verification

The PhysicalRequestFilter registers are tested in the physical DMA tests.

8.1.21 Physical Upper Bound Registers

The Physical Upper Bound Register shall be tested for compliance with the Open HCI specification by performing the tests defined in this section. These tests shall only be performed if this optional register is implemented in the Open HCI implementation.

8.1.21.1 Physical Upper Bound Registers Access Test

This test procedure shall verify that the read only fields of the Physical Upper Bound Register cannot be written to and that the set fields of the Physical Upper Bound Register can be set.

8.1.21.1.1 Physical Upper Bound Registers Access Procedure

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
RG193	Wrtie to the Physical Upper Bound register read/write fields to confirm registers are read/write.	**Variables definition:** default_Physical_Upper_Bound_reg_value = Value of Physical Upper Bound register before the write read Physical_Upper_Bound_reg_value = Value of Physical Upper Bound register after the write **Returns:** Physical_Upper_Bound_writeread		OHCI -5.14
		Read the Physical Upper Bound register (default_Physical_Upper_Bound_reg_value)		
		Write (default_Physical_Upper_Bound_reg_value XOR 0xFFFFFFFF) to Physical Upper Bound register		
		Read the Physical_Upper_Bound register (read_Physical_Upper_Bound_reg_value)		
		If (read_Physical_Upper_Bound_reg_value == (default_Physical_Upper_Bound_reg_value XOR 0xFFFFFFFF)) : Physical_Upper_Bound_writeread = **PASSED** Else Physical_Upper_Bound_writeread = **FAILED**		

Copyright © 2008, 1394 Trade Association. All rights reserved.

9 Asynchronous Transmit DMA

This section contains the tests for the Asynchronous Transmit and Receive DMA. The Asynchronous Transmit DMA contains two contexts: **AT request** and **AT response** and one hardware DMA engine, Physical response. This section only covers the AT request and AT response DMAs.

9.1 Asynchronous Requests and Responses Test

The test implementation shall allocate a 1394 address range in initial memory space of the IUT and the Tester, to be accessed by the IUT and Tester node. The allocated address range must be large enough to hold the largest packet that is capable of being transmitted by the Tester node and received by the OHCI Implementation.

The test shall consist of the following:

Step 1: At least two minutes of Asynchronous Write and Read Requests from the IUT to the Tester node at maximum packet size resulting in at least 30% bus utilization.

Step 2: At least two minutes of Asynchronous Write and Read Requests from the Tester node to the IUT at maximum packet size resulting in at least 30% bus utilization.

Step 3: At least two minutes of Asynchronous Write and Read Requests to/from the IUT/Tester node at maximum packet size resulting in at least 50% bus utilization.

If any asynchronous read or write request/response does not complete successfully the test shall have failed. If the data payload transmitted in the asynchronous read response is not identical to the data payload sent in the asynchronous write request the test shall have failed.

9.1.1 Length

The test implementation shall cause write requests and read response to be generated at the maximum size allowed by the implementation and 1394 bus speed.

As specified by IEEE-1394 the maximum asynchronous packet size as a function of 1394 data rate is:

- S100 – 512 bytes
- S200 – 1024 bytes
- S400 – 2048 bytes
- S800 – 4096 bytes
- S1600 – 8192 bytes
- S3200 – 16384 bytes

9.1.2 Alignment

The test does not attempt to control the byte alignment of the memory location the packet is fetched from. This implementation will be OS specific.

1394 TRADE ASSOCIATION

Copyright © 2008, 1394 Trade Association. All rights reserved.

9.1.3 Interrupts

The test does not attempt to control how or when OUTPUT_LAST or OUTPUT_LAST_Immediate descriptor generate interrupts. This implementation will be OS specific.

9.1.4 Speed

The test implementation shall run each test step at each supported 1394 bus speed.

9.1.5 Retried request shall not block responses.

This test does not explicitly attempt to create the situation where by retried requests could block a response packet. However, this situation may occur in the in the multiple DMA test found later in this document.

1394 TRADE ASSOCIATION

Copyright © 2008, 1394 Trade Association. All rights reserved.

9.1.6 Asynchronous Requests and Responses Test Procedure

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
ATRR1	Verify Asynchronous Request Transmit DMA functionality	Have IUT setup context for asynchronous transmit request and asynchronous receive response. Variables: SUPPORTED_SPEED = {100, 200, 400, 800, 1600} Timeout = 50,000 requests WRITE_DATA = write_data_buffer		OHCI-6 OHCI-7 OHCI-8
		Have Tester allocate asynchronous resource.		
		FOR: SUPPORTED_SPEED FOR: Timeout dataLength = MAX_AT_DATALENGTH Have IUT transmit Block Write Request IF Write Request is not successful THEN ATRR1 = FAIL Have IUT transmit Block Read Request IF Read Request is not successful THEN ATRR1 = FAIL IF WRITE_DATA != READ_DATA THEN ATRR1 = FAIL END FOR: Iterations END FOR: SUPPORTED_SPEED IF ATRR1 != FAIL THEN ATRR1 = PASS		
ATRR2	Verify Asynchronous Response Transmit DMA functionality	Have IUT setup context for asynchronous transmit response and asynchronous receive request. Variables: SUPPORTED_SPEED = {100, 200, 400, 800, 1600} Timeout = 50,000 responses WRITE_DATA = write_data_buffer Write_Address != Physical Address Read_Address != Physical Address		OHCI-6 OHCI-7 OHCI-8
		Have Tester allocate asynchronous resource.		
		FOR: SUPPORTED_SPEED FOR: Timeout dataLength = MAX_AT_DATALENGTH Have Tester transmit Block Write Request IF Write Request is not successful THEN ATRR2 = FAIL Have Tester transmit Block Read Request IF Read Request is not successful THEN ATRR2 = FAIL IF WRITE_DATA != READ_DATA THEN ATRR2 = FAIL END FOR: Iterations END FOR: SUPPORTED_SPEED IF ATRR2 != FAIL THEN ATRR2 = PASS		
ATRR3	Verify Asynchronous Request and Response Transmit DMA functionality	Have IUT setup context for asynchronous transmit request and response and asynchronous receive request and response. Variables: SUPPORTED_SPEED = {100, 200, 400, 800, 1600} Timeout = 50,000 requests and responses WRITE_DATA = write_data_buffer Write_Address != Physical Address Read_Address != Physical Address		OHCI-6 OHCI-7 OHCI-8
		Have Tester allocate asynchronous resource.		
		FOR: SUPPORTED_SPEED FOR: Timeout dataLength = MAX_AT_DATALENGTH Have IUT and Tester transmit Block Write Request IF Write Requests are not successful THEN ATRR3 = FAIL		

Copyright © 2008, 1394 Trade Association. All rights reserved.

		Have IUT and Tester transmit Block Read Request IF Read Request are not successful THEN ATRR3 = FAIL IF WRITE_DATA != READ_DATA THEN ATRR3 = FAIL END FOR: Iterations END FOR: SUPPORTED_SPEED IF ATRR3 != FAIL THEN ATRR3 = PASS		

Copyright © 2008, 1394 Trade Association. All rights reserved.

10 Asynchronous Receive DMA

The Asynchronous Receive DMA is tested in Chapter 9 along with the Asynchronous Transmit DMA.

Copyright © 2008, 1394 Trade Association. All rights reserved.

11 Isochronous Transmit DMA

This test procedure shall test the ability of the IUT to transmit isochronous data as defined in the OHCI specification. The test shall test an individual isochronous context and then incrementally increase the number of context until IT_CONTEXT_NUMBER is utilized.

11.1.1 Length

The test implementation shall cause the IUT to transmit isochronous packets. The test implementation must ensure that a Tester node is receiving the isochronous packets being sent. After every isochronous packet is sent, the Tester node shall verify that the data payload received is identical to the data payload of the isochronous packet sent by the IUT.

As specified by IEEE-1394 the maximum isochronous packet size as a function of 1394 data rate is:

- S100 – 1024 bytes
- S200 – 2048 bytes
- S400 – 4096 bytes
- S800 – 8192 bytes
- S1600 – 16384 bytes
- S3200 – 32768 bytes

The test implementation must also comprehend the number of isochronous context/packets being sent per isochronous cycle and shall not violation the 100usec specified by the standard.

11.1.2 Alignment

The test does not attempt to control the byte alignment of the memory location the packet is fetched from. This implementation will be OS specific.

11.1.3 Interrupts

The test does not attempt to control how or when OUTPUT_LAST or OUTPUT_LAST_Immediate descriptor generate interrupts. This implementation will be OS specific.

11.1.4 Speed

The test implementation shall run each test step at each supported 1394 bus speed.

11.2 Channel

The test does not attempt to control how or which isochronous channels are allocated for the test. This implementation will be OS specific.

1394 TRADE ASSOCIATION

Copyright © 2008, 1394 Trade Association. All rights reserved.

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
IT1	Verify isochronous transmit on one context at a time.	Have IUT setup context for isochronous transmit. Variables: SUPPORTED_SPEED = {100, 200, 400, 800, 1600} Test_Length = 500,000 packets TRANSMIT_DATA = isoch_data_buffer		OHCI-9
		Have Tester setup and run context for isochronous receive. RECEIVE_DATA = TRANSMIT_DATA		
		FOR: SUPPORTED_SPEED FOR: Test_Length dataLength = MAX_AT_DATALENGTH Have IUT run IT context IF Tester node validates received packet = isoch_data_buffer THEN IT1 = PASS ELSE IT1 = FAIL		
		IT_CONTEXT_NUMBER		
IT2	Verify isochronous transmit on all context.	Have IUT setup context for isochronous transmit. Variables: IT_CONTEXT_NUMBER = {4,5,6,7,8,9,n} SUPPORTED_SPEED = {100, 200, 400, 800, 1600} Test_Length = 500,000 packets TRANSMIT_DATA = RECEIVE_DATA		OHCI-9
		Have Tester setup and run n context for isochronous receive. RECEIVE_DATA = TRANSMIT_DATA		
		FOR: IT_CONTEXT FOR: SUPPORTED_SPEED FOR: Test_Length dataLength = MAX_AT_DATALENGTH Have IUT run IT contexts IF Tester node validates received packet = isoch_data_buffer THEN IT2 = PASS ELSE IT2 = FAIL		

Copyright © 2008, 1394 Trade Association. All rights reserved.

12 Isochronous Receive DMA

This test procedure shall test the ability of the IUT to receive isochronous data as defined in the OHCI specification. The test shall test an individual isochronous receive context and then incrementally increase the number of context until IR_CONTEXT_NUMBER is utilized.

12.1.1 Length

The test implementation shall cause the IUT to receive isochronous packets. The test implementation must ensure that a Tester node is transmitting the isochronous test packets. After every isochronous packet is sent, the IUT shall verify that the data payload received is identical to the data payload of the isochronous packet sent by the Tester.

As specified by IEEE-1394 the maximum isochronous packet size as a function of 1394 data rate is:

- S100 – 1024 bytes
- S200 – 2048 bytes
- S400 – 4096 bytes
- S800 – 8192 bytes
- S1600 – 16384 bytes
- S3200 – 32768 bytes

The test implementation must also comprehend the number of isochronous context/packets being sent per isochronous cycle and shall not violation the 100usec specified by the standard.

12.1.2 Alignment

The test does not attempt to control the byte alignment of the memory location the packet is fetched from. This implementation will be OS specific.

12.1.3 Interrupts

The test does not attempt to control how or when INPUT_LAST or INPUT_MORE descriptor generate interrupts. This implementation will be OS specific.

12.1.4 Speed

The test implementation shall run each test step at each supported 1394 bus speed.

12.2 Channel

The test implementation does not attempt to control how or which isochronous channels are allocated for the test. This implementation will be OS specific.

1394 TRADE ASSOCIATION

Copyright © 2008, 1394 Trade Association. All rights reserved.

12.3 Context Node

The test implementation does not attempt to control which context mode (Packet-per-Buffer or Buffer fill) is used. This implementation will be OS specific.

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
IR1	Verify isochronous receive on one context at a time.	Have IUT setup context for isochronous receive. Variables: SUPPORTED_SPEED = {100, 200, 400, 800, 1600} Test_Length = 500,000 packets RECEIVE_DATA = TRANSMIT_DATA		OHCI-9
		Have Tester setup and run context for isochronous receive. TRANSMIT_DATA = isoch_data_buffer		
		FOR: SUPPORTED_SPEED FOR: Test_Length dataLength = MAX_AT_DATALENGTH Have IUT run IR context IF IUT node validates received packet = RECEIVE_DATA THEN IR1 = PASS ELSE IR1 = FAIL		
		IR_CONTEXT_NUMBER		
IR2	Verify isochronous receive on all context.	Have IUT setup context for isochronous receive. Variables: IR_CONTEXT_NUMBER = {4,5,6,7,8,9,n} SUPPORTED_SPEED = {100, 200, 400, 800, 1600} Test_Length = 500,000 packets RECEIVE_DATA = TRANSMIT_DATA		OHCI-9
		Have Tester setup and run n context for isochronous receive. TRANSMIT_DATA = isoch_data_buffer		
		FOR: IR_CONTEXT FOR: SUPPORTED_SPEED FOR: Test_Length dataLength = MAX_AT_DATALENGTH Have IUT run IR context IF Tester node validates received packet = isoch_data_buffer THEN IR2 = PASS ELSE IR2 = FAIL		

Copyright © 2008, 1394 Trade Association. All rights reserved.

13 Physical Request DMA

The Physical Request DMA engine shall be tested for compliance with the Open HCI Specification by performing the tests defined in this section.

For the purposes of testing the Physical request DMA and the corresponding Physical response DMA operation the Open HCI requirements are broken down into the following categories:

- Physical Requests

- Posted Writes

- Responses

- Filtering Physical Requests

Using the categories list above this test specification defines both explicit and implicit test of these functions.

13.1 Physical DMA Test Procedure

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
PDMA1	ROM Access			
		ROM Access is tested in the Base 1394 Test Suite Definition with Extensions for 1394b.		
PDMA2	Bus Management CSR Registers			
		Bus Management CSR Register is tested in the Base 1394 Test Suite Definition with Extensions for 1394b.		

Copyright © 2008, 1394 Trade Association. All rights reserved.

PDMA3	Physical Addressing without posted writes	Have IUT create buffers in physical memory. Physical Address Variables: SUPPORTED_SPEED = {100, 200, 400, 800, 1600} Timeout = 50,000 write/read iterations postedWriteEnable = 0 PhysicalRequestFilterHi = 0 PhysicalRequestFilterLo = Tester Node Id WRITE_DATA = write_data_buffer Write_Address != Physical Address Read_Address != Physical Address		
		Have Tester allocate asynchronous resource.		
		FOR: SUPPORTED_SPEED FOR: Timeout dataLength = MAX_AT_DATALENGTH Have Tester transmit Block Write Request IF Write Request is not successful THEN PDMA3 = FAIL Have Tester transmit Block Read Request IF Read Request is not successful THEN PDMA3 = FAIL IF WRITE_DATA != READ_DATA THEN PDMA3 = FAIL END FOR: Iterations END FOR: SUPPORTED_SPEED IF PDMA3 != FAIL THEN PDMA3 = PASS		
PDMA4	Physical Addressing with posted writes	Have IUT create buffers in physical memory. Physical Address Variables: SUPPORTED_SPEED = {100, 200, 400, 800, 1600} Timeout = 50,000 write/read iterations postedWriteEnable = 1 PhysicalRequestFilterHi = 0 PhysicalRequestFilterLo = Tester Node Id WRITE_DATA = write_data_buffer Write_Address != Physical Address Read_Address != Physical Address		
		Have Tester allocate asynchronous resource.		
		FOR: SUPPORTED_SPEED FOR: Timeout dataLength = MAX_AT_DATALENGTH Have Tester transmit Block Write Request IF Write Request is not successful THEN PDMA3 = FAIL Have Tester transmit Block Read Request IF Read Request is not successful THEN PDMA3 = FAIL IF WRITE_DATA != READ_DATA THEN PDMA3 = FAIL END FOR: Iterations END FOR: SUPPORTED_SPEED IF PDMA3 != FAIL THEN PDMA3 = PASS		
PDMA5	Responses	Physical DMA responses are tested in PDMA1 - 4		
PDMA6	Physical Request Filters	Run test PDMA3 with PhysicalRequestFilterHi = 0 PhysicalRequestFilterLo = !Tester Node Id IF Write and Read request are handled by the AR Request DMA THEN PDMA6 = PASS ELSE PDMA6 = FAIL		

Copyright © 2008, 1394 Trade Association. All rights reserved.

14 Multiple DMA

14.1 Multiple DMA Engine

The tests defined in this section shall verify that separate DMA engines can be exercised at the same time. The tests in this section require that the tests defined in Chapters 9 through 13 be performed simultaneously. In each test defined in this Chapter, the appropriate tests referred to in sections 9 through 13 shall begin execution simultaneously. The tests shall not complete until all of the individual tests have completed. The tests defined in this section shall have failed if any of the tests in sections 9 through 13 fail.

14.2 Asynchronous and Isochronous DMA

The test implementation shall perform the test procedures described in section ATRR3, IT2 and IR2 simultaneously. This exercises the AT Request, AT Response, AR Request, AR Response, IT Transmit, and IT Receive DMA engines of the IUT concurrently. If any of these tests fails, the test shall have failed.

14.3 Asynchronous, Isochronous, and Physical DMA

The test implementation shall perform the test procedures described in section ATRR3, IT2, IR2 and PDMAxx simultaneously. This exercises the AT Request, AT Response, AR Request, AR Response, IT Transmit, IT Receive, and Physical DMA engines of the IUT concurrently. If any of these tests fails, the test shall have failed.

Test Number	Test Description	Procedure	Pass or Fail	Specification Reference
MDMA1	Exercise the AT Request, AT Response, AR Request, AR Response, IT Transmit, and IT Receive DMA engines of the IUT concurrently.	Perform the test procedures described in ATRR3, IT2 and IR2 simultaneously.		
		IF: Any of these tests fail, THEN: MDMA1 = FAIL ELSE: MDMA1 = PASS		
MDMA2	Exercise the AT Request, AT Response, AR Request, AR Response, IT Transmit, IT Receive, and Physical DMA engines of the IUT concurrently.	Perform the test procedures described in ATRR3, IT2, IR2 and PDMA3 simultaneously. .		
		IF: Any of these tests fail, THEN: MDMA2 = FAIL ELSE: MDMA2 = PASS		

1394 TRADE ASSOCIATION

Copyright © 2008, 1394 Trade Association. All rights reserved.

Annex A
(informative)

Bibliography

[B1] IEEE Std 1212-2001, Standard for a Control and Status Registers (CSR) Architecture for microcomputer buses

[B2] IEEE Std 1394-1995, Standard for a High Performance Serial Bus

[B3] IEEE Std 1394a-2000, Standard for a High Performance Serial Bus—Amendment 1

[B4] IEEE Std 1394b-2002, Standard for a High Performance Serial Bus—Amendment 2

[B5] ISO/IEC 9899:1990, Programming Languages—C

[B6] 1394 Open Host Controller Interface Specification – Release 1.1

Copyright © 2008, 1394 Trade Association. All rights reserved.