# xHCI Compliance Test Specification
## eXtensible Host Controller Interface
## for Universal Serial Bus

**Date:**        August 2025

**Revision:**        1.20

**Scope of this Revision**

This revision of the specification is intended to describe the testing to be applied to hardware based on the eXtensible Host Controller Interface for Universal Serial Bus (xHCI), Revision 0.96, 1.0, 1.1 or 1.2. Tests apply to all host controller versions unless noted otherwise.

Some host controller implementations include an integrated hub. Tests that require modified testing for this implementation will be noted in the individual test.

**Revision History**

| Revision | Issue Date | Comments |
|---|---|---|
| 0.9 | 10/1/2009 | Revision 0.9 release. |
| 1.0 | 11/2/2011 | Revision 1.0 release. |
| 1.3 | 4/2/2015 | Release supporting xHC 1.1, including errata to 3-31-2015 |
| 1.7 | 12/6/18 | Remove iteration of slot speeds in TD 4.20 |
| 1.8 | 4/8/2019 | Rework of TD 4.15 to accommodate Microsoft SuperMUTT; Update TD 2.06 to allow xHC to cycle through the event ring before adding new segment |
| 1.9 | 8/19/2019 | Support for xHC 1.2. |
| 1.10 | 6/18/2020 | Update TD numbering to be of the form TD 1.01 instead of TD 1.1, for consistency with application. |
| 1.11 | 8/27/2020 | Update to TD 2.14 steps<br><br>Update TD 1.01: allow SBRN to be one of {30h, 31h, 32h} |
| 1.12 | 3/1/2021 | Fix test spec error in TD 1.02 for checking HCCPARAMS1 |
| 1.13 | 5/28/2021 | Update TD 4.12 |
| 1.14 | 3/15/2022 | Clarification to step 11 of TD 3.12 |
| 1.15 | 5/16/2022 | Fix bit checks for bits 9, 10, 11 in TD 1.2 to be in sync with code and base spec. |
| 1.16 | 8/4/2022 | Upate TD 4.01 |
| 1.17 | 4/21/2023 | Allow up to 1 second after HCRST, to allow for ports with USB-C and USB Power Delivery. |
| 1.18 | 8/22/2023 | Make allowance in tests for non-PCI xHCs. Cleanup of TD 4.02, TD 4.03, TD 4.04, TD 4.07 (no change to implementatation of these tests). |
| 1.19 | 11/5/2024 | TD 1.02 clarify that xHC 1.1 or above is required for USB-IF cert.<br><br>TD 1.05 clarify requirements for different xHC versions.<br><br>TD 4.05 add error code details for case with embedded hub. |
| 1.20 | 1/9/2025 | TD 2.14 Doorbell Test removed. |

*Third-party brands and names are the property of their respective owners.

**Significant Contributors:**

| | |
|---|---|
| Randy Aull | Microsoft Corporation |
| Vidyadhari Dharmaraju | Intel Corporation |
| David Hargrove | Microsoft Corporation |
| Mark Maszak | Microsoft Corporation |
| Steve McGowan | Intel Corporation |
| Diane Lenox | Specwerkz, LLC |

*Please send comments via electronic mail to: techadmin@usb.org or usbcompliance@usb.org*

# 1 Introduction

This document provides the compliance criteria and test descriptions for USB host controllers that conform to the eXtensible Host Controller Interface (xHCI) Specification, Revision 0.96 or 1.0. It is relevant for anyone building an xHCI-based host controller or an eXtensible Host Controller (xHC). These criteria address the functional requirements of an xHC. The electrical compliance requirements for xHCs are defined by the USB-IF. The document is divided into two major sections. The first section lists the compliance criteria and the second section lists the test descriptions used to verify an xHC's conformance to these compliance criteria.

Compliance criteria are provided as a list of assertions that describe specific characteristics or behaviors that must be met. Each assertion provides a reference to the xHCI Specification or other documents from where the assertion was derived. In addition, each assertion provides a reference to the specific test description(s) where the assertion is tested.

Test descriptions provide a high level overview of the tests that are performed to check the compliance criteria. The descriptions are provided with enough detail so that a reader can understand what the test does. The descriptions do not describe the actual step-by-step procedure to perform the test.

These tests require Low Speed, Full Speed, High Speed and Super Speed Test Devices. The HS Test Device should be installed in a separate computer from the host under test and connected via USB cable. This separate computer should be running Windows XP, have only 1 processor enabled, and only essential devices enabled for best test device operation. The LS and FS test devices require firmware to be loaded, which is done automatically by the test application. If you try to use these devices outside of the application, windows will not be able to find suitable driver for them.

Each test assertion is formatted as follows:

| Assertion | Assertion Description | Test # | Comments |
|-----------|----------------------|--------|----------|
|           |                      |        |          |

**Assertion#:** Unique identifier for each spec requirement. The identifier is in the form XHCI_SPEC_SECTION_NUMBER#X where X is a unique integer for a requirement in that section.

**Assertion Description:** Specific requirement from the specification

**Test #:** A label for a specific test description in this specification that tests this requirement. Test # can have one of the following values:

**N/A**    This item is not explicitly tested in an xHCI test description. Items can be labeled N/A for several reasons – including items that are not testable, not important to test for interoperability, or are indirectly tested by other operations performed by the xHCI compliance test.

**TD 1.X**    This item is covered by the test described in test description 1.X in this specification.

**TBD**    A test description is not yet written for this assertion. The comments field provides more details on such items.

**Comments:** Provides additional information on requirements that are marked TBD.

All tests should be run with Driver Verifier running on the compliance driver. Create a batch file with the following commands, run the batch file, and reboot the computer.

> verifier /flags 0x080

> verifier /driver xhcidrv.sys

# 2 Test Assertions

Unless otherwise noted, subsection references are to the xHCI Specification.

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| **Chapter 1 Test Assertions:** | | | |
| **Subsection reference: 1 Preface** | | | |
| **Subsection reference: 1.6 Terms and Abbreviations** | | | |
| 1.6#1 | The xHC shall set the default SOPC for an endpoint to (Max Burst Size + 1) x (Mult + 1) in the Endpoint Context. | TD 5.17 | |
| **Chapter 2 Test Assertions:** | | | |
| **Subsection reference: 2 Introduction** | | | |
| **Subsection reference: 2.2 Key features** | | | |
| 2.2#1 | xHCI shall support devices from LS to SS. | N/A | Tested by multiple TDs |
| **Chapter 3 Test Assertions:** | | | |
| **Subsection reference: 3 Architectural Overview** | | | |
| **Subsection reference: 3.1 Interface Architecture** | | | |
| 3.1#1 | The xHC shall not write directly to the Command Ring and the Transfer ring. | implied | |
| 3.1#2 | The xHC shall support a minimum page size of 4K. | N/A | |
| **Subsection reference: 3.2 xHCI Data Structures** | | | |
| **Subsection reference: 3.2.3 Slot Context** | | | |
| 3.2.3#1 | The xHC shall not update the Output Slot Context on a failure of a command targeted at the device slot. | TD 4.02 | |
| **Subsection reference: 3.2.4 Endpoint Context** | | | |
| 3.2.4#1 | The xHC shall not update the Output Endpoint Context on a failure of a command targeted at the endpoint. | TD 4.03 | |
| **Subsection reference: 3.2.7 Transfer Request Block** | | | |
| **Subsection reference: 3.2.7.1 Operation** | | | |
| 3.2.7.1#1 | The xHC shall support multi TRB TDs where each TRB can start on arbitrary page offset and have an arbitrary length. | TD 5.02, TD 5.03 | |
| 3.2.7.1#2 | The xHC shall support buffer lengths up to 64KB per TRB. | TD 5.02, TD 5.03 | |
| **Subsection reference: 3.2.9 Control Transfers** | | | |

| | | | |
|---|---|---|---|
| 3.2.9#1 | The xHC shall transfer data in the Data Stage TD in the direction indicated by the Data Stage TRB. | USB30CV Chapter 9 | |
| **Subsection reference: 3.3 Command Interface** | | | |
| **Subsection reference: 3.3.5 Configure Endpoint** | | | |
| 3.3.5#1 | The xHC shall ignore the Input Endpoint Context of an endpoint which is not added by a Configure Endpoint Command. | 0 | |
| **Subsection reference: 3.4 General Information** | | | |
| 3.4#1 | The xHC shall automatically manage all the Hub TT split transactions without software intervention. | Interop | |
| **Chapter 4 Test Assertions:** | | | |
| **Subsection reference: 4.2 Host Controller Initialization** | | | |
| 4.2#1 | The xHC shall ignore doorbell references while the RS flag is 0 in the USBCMD register. | 0 | |
| 4.2#2 | The xHC shall transmit SOFs to USB2 ports when they are in the Enabled state. | TD 6.01 | |
| **Subsection reference: 4.3 USB Device Initialization** | | | |
| 4.3#1 | The xHC shall set the CCS flag to 1 in the PORTSC register when a device is attached. | 0 0 | |
| 4.3#2 | The xHC shall set the CSC flag to 1 in the PORTSC register when a device is attached. | 0 0 | |
| 4.3#3 | The xHC shall generate a Port Status Change Event when the CSC flag transitions to 1 in the PORTSC register. | 0 0 | |
| 4.3#4 | The xHC shall set the PED flag to 1 in the PORTSC register when a SuperSpeed device is attached. | 0 | |
| 4.3#5 | The xHC shall clear the PR flag to 0 in the PORTSC register when a device is attached. | 0 0 | |
| 4.3#6 | The xHC shall set the PLS field to U0 in the PORTSC register when a SuperSpeed device is attached. | 0 | |
| 4.3#7 | The xHC shall clear the PED flag to 0 in the PORTSC register when a USB2 device is attached. | 0 | |
| 4.3#8 | The xHC shall set the PLS field to Polling in the PORTSC register when a USB2 device is attached. | 0 | |
| 4.3#9 | The xHC shall set the PRC flag to 1 in the PORTSC register after a successful port reset. | 0 | |
| 4.3#10 | The xHC shall set the PED flag to 1 in the PORTSC register after a successful port reset. | 0 | |
| 4.3#11 | The xHC shall set the PLS field to U0 in the PORTSC register after a successful port reset. | 0 | |
| 4.3#12 | The xHC shall clear the PR flag to 0 in the PORTSC register after a successful port reset. | 0 | |
| 4.3#13 | The xHC shall generate a Port Status Change Event when the PRC flag transitions to 1 in the PORTSC register. | 0 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.3#14 | The xHC shall set the SS port to the Disabled state and set the PLC flag to 1 if polling of a SS device as a result of device attach while in the Disconnected state was unsuccessful. | TBD | |
| **Subsection reference: 4.3.1 Resetting a Root Hub Port** | | | |
| 4.3.1#1 | The xHC shall set the Port Speed field to the speed of the attached device in the PORTSC register after a successful port reset. | 0<br>0 | |
| **Subsection reference: 4.3.4 Address Assignment** | | | |
| 4.3.4#1 | The xHC shall set the Slot State field to Addressed in the Slot Context upon successful completion of an Address Device Command with the BSR flag set to 0. | 0 | |
| 4.3.4#2 | The xHC shall set the Slot State field to Default in the Slot Context upon successful completion of an Address Device Command with the BSR flag set to 1. | 0 | |
| **Subsection reference: 4.3.5 Device Configuration** | | | |
| 4.3.5#1 | The xHC shall set the Slot State field to Configured in the Slot Context upon successful completion of a Configure Endpoint Command if at least one non-default endpoint is enabled. | 0 | |
| **Subsection reference: 4.4 Device Detach** | | | |
| 4.4#1 | The xHC shall clear the CCS flag to 0 in the PORTSC register when a device is detached. | 0<br>0 | |
| 4.4#2 | The xHC shall set the CSC flag to 1 in the PORTSC register when a device is detached. | 0<br>0 | |
| **Subsection reference: 4.5 Device Slot Management** | | | |
| **Subsection reference: 4.5.3 Slot States** | | | |
| **Subsection reference: 4.5.3.2 Disabled** | | | |
| 4.5.3.2#1 | The xHC shall set the Slot State field to Disabled in the Slot Context after successful completion of a Disable Slot Command. | 0 | |
| 4.5.3.2#2 | The xHC shall not generate events for a device slot that is in the Disabled state | TD 4.13 | |
| **Subsection reference: 4.5.3.3 Enabled** | | | |
| 4.5.3.3#1 | The xHC shall ignore writes to the doorbell register for a device slot in the Enabled State. | 0 | |
| **Subsection reference: 4.5.3.4 Default** | | | |
| 4.5.3.4#1 | The xHC shall set the USB Device Address field to 0 in the Output Slot Context when transitioning to the Default state. | 0<br>0 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.5.3.4#2 | The xHC shall ignore writes to the doorbell register for a device slot in the Default State except if the DB Target field is Control EP 0 Enqueue Pointer Update. | 0 | |
| **Subsection reference: 4.5.3.5 Addressed** | | | |
| 4.5.3.5#1 | The xHC shall set the Slot State field to Addressed in the Slot Context upon successful completion of a "De-configure" Configure Endpoint Command. | 0 | |
| 4.5.3.5#2 | The xHC shall not change the Slot State upon completion of an Evaluate Context Command. | 0 | |
| 4.5.3.5#3 | The xHC shall not change the Slot State upon completion of a Stop Endpoint Command. | 0 | |
| 4.5.3.5#4 | The xHC shall not change the Slot State upon completion of a Set TR Dequeue Pointer Command. | 0 | |
| 4.5.3.5#5 | The xHC shall set the Slot State field to Default in the Slot Context upon successful completion of a Reset Device Command. | 0 | |
| 4.5.3.5#6 | The xHC shall ignore writes to the doorbell register for a device slot in the Address State except if the DB Target field is Control EP 0 Enqueue Pointer Update. | 0 | |
| **Subsection reference: 4.5.3.6 Configured** | | | |
| 4.5.3.6#1 | The xHC shall not change the Slot State upon completion of a Reset Endpoint command. | 0 | |
| 4.5.3.6#2 | The xHC shall ignore writes to the doorbell register for a device slot in the Configured State except if the DB Target field indicates an enabled endpoint. | 0 | |
| **Subsection reference: 4.5.4 USB Standard Device Request to xHCI Command Mapping** | | | |
| **Subsection reference: 4.5.4.1 SET_ADDRESS Request** | | | |
| 4.5.4.1#1 | The xHC shall complete a Setup TD with a TRB Error if the bmRequestType field is set to Host-to-Device, Standard, and Device, and the bRequest field is set to SET_ADDRESS. | TD 5.25 | |
| 4.5.4.1#2 | The xHC shall not generate a Setup Transaction on the USB bus on a Setup TD with the bmRequestType field set to Host-to-Device, Standard, and Device, and the bRequest field set to SET_ADDRESS. | TD 5.25 | |
| **Subsection reference: 4.6 Command Interface** | | | |
| 4.6#1 | xHC shall return TRB Error if the command is not recognized by the xHC. | TD 4.16 | |
| 4.6#2 | The xHC shall generate a Command Completion Event on the Primary Event Ring after processing a command. | 0 | |
| **Subsection reference: 4.6.1 Command Ring Operation** | | | |
| 4.6.1#1 | The xHC shall execute commands until the Command Ring is stopped or empty upon detection of a Host Controller Command Doorbell ring. | 0 | |
| 4.6.1#2 | The xHC shall order the Command Completion Events based on the command locations in the Command Ring. | 0 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.6.1#3 | The xHC shall set the Slot ID field of the Command Completion Event to the Slot ID value in the corresponding Command TRB. | 0<br>0<br>0<br>0<br>0<br>0<br>0<br>0 | |
| 4.6.1#4 | The xHC shall set the Command TRB Pointer field of the Command Completion event to the Command TRB that initiated the event, unless not initiated by a TRB. | 0 | |
| 4.6.1#5 | The xHC shall set the Command Completion Code to Success in the Command Completion Event upon successful completion of a command. | 0<br>0<br>0<br>0 | |
| **Subsection reference: 4.6.1.1 Stopping the Command Ring** | | | |
| 4.6.1.1#1 | The xHC shall restart command execution upon Host Controller Doorbell at the TRB following the last command executed if software has not modified the Command Ring Pointer while the Command Ring is stopped. | 0 | |
| 4.6.1.1#2 | The xHC shall restart command execution upon Host Controller Doorbell at the new Command Ring Pointer if software has modified it while the Command Ring is stopped. | 0 | |
| 4.6.1.1#3 | The xHC shall generate a Command Completion Event with the Completion Code set to Command Ring Stopped when the CS flag is set to 1 in the CRCR register. | 0 | |
| 4.6.1.1#4 | The xHC shall set the Command TRB Pointer field to the current value of the Command Ring Dequeue Pointer in a Command Completion Event with the Completion Code set to Command Ring Stopped. | 0 | |
| 4.6.1.1#5 | The xHC shall complete the execution of the current command when the CS flag is set to 1 in the CRCR register. | | |
| 4.6.1.1#6 | The xHC shall generate a Command Completion Event with the Completion Code set to Command Ring Stopped only after the current executing command has been completed when the CS flag is set to 1 in the CRCR register. | | |
| **Subsection reference: 4.6.1.2 Aborting a Command** | | | |
| 4.6.1.2#1 | The xHC shall generate a Command Completion Event with the Completion Code set to Command Ring Stopped when the CA flag is set to 1 in the CRCR register. | 0 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.6.1.2#2 | The xHC shall Immediately terminate the execution of the current command when the CA flag is set to 1 in the CRCR register. | | |
| 4.6.1.2#3 | The xHC shall generate a Command Completion Event with the Completion Code set to Command Aborted if it terminates the execution of the current command when the CA flag is set to 1 in the CRCR register. | | |
| **Subsection reference: 4.6.2 No Op** | | | |
| 4.6.2#1 | The xHC shall set the Completion Code to Success in the Command Completion Event of a No Op Command. | 0 | |
| **Subsection reference: 4.6.3 Enable Slot** | | | |
| 4.6.3#1 | The xHC shall complete an Enable Slot Command with No Slots Available Error if there is no device slot available. | 0 | |
| 4.6.3#2 | The xHC shall set the Slot ID field to a value in the range of 1 to MaxSlotsEn, in the Command Completion Event for a successful Enable Slot Command. | 0 | |
| 4.6.3#3 | The xHC shall set the Slot ID field to 0 in the Command Completion Event for an Enable Slot Command with No Slots Available Error. | 0 | |
| **Subsection reference: 4.6.4 Disable Slot** | | | |
| 4.6.4#1 | The xHC shall free any bandwidth allocated for all the periodic endpoints of the device slot when processing a Disable Slot Command. | 0 | |
| 4.6.4#2 | The xHC shall terminate any USB activity targeted at the device slot when processing a Disable Slot Command. | TD 6.01 | |
| **Subsection reference: 4.6.5 Address Device** | | | |
| 4.6.5#1 | The xHC shall set the USB Device Address field to the assigned address in the range of 1 to 127 in the Slot Context upon successful completion of an Address Device Command with the BSR flag set to 0. | 0 | |
| 4.6.5#2 | The xHC shall copy all fields of the Input Slot Context to the Output Slot Context upon successful completion of an Address Device Command. | 0 | |
| 4.6.5#3 | The xHC shall copy all fields of the Input Endpoint 0 Context to the Output Endpoint 0 Context upon successful completion of an Address Device Command. | 0 | |
| 4.6.5#4 | The xHC shall set the Endpoint State to Running in the Output Endpoint 0 Context upon successful completion of an Address Device Command. | 0 | |
| 4.6.5#5 | The xHC shall assign an address that is unique for the bus instance when processing an Address Device Command with the BSR flag set to 0. | 0 | |
| 4.6.5#6 | The xHC shall ignore all Endpoint Contexts in the Input Context except for the Endpoint Context 0 when processing an Address Device Command. | 0 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.6.5#7 | The xHC shall issue a SET_ADDRESS request when processing an Address Device Command with the BSR flag set to 0. | TD 2.12<br>TD 2.14<br>TD 4.02<br>TD 4.03<br>TD 4.05<br>TD 4.06<br>TD 4.07<br>TD 4.08<br>TD 4.09<br>TD 4.11<br>TD 5.19 | |
| 4.6.5#8 | The xHC shall not issue a SET_ADDRESS request when processing an Address Device Command with the BSR flag set to 1. | TD 2.08<br>TD 2.09<br>TD 2.10<br>TD 4.02<br>TD 4.05<br>TD 6.01 | |
| 4.6.5#9 | The xHC shall set the Completion Code to USB Transaction Error on a failure of the SET_ADDRESS request upon completion of an Address Device Command with the BSR flag set to 0. | | |
| 4.6.5#10 | The xHC shall not change the Slot State field in the Slot Context on a failure of the SET_ADDRESS request upon completion of an Address Device Command with the BSR flag set to 0. | | |
| 4.6.5#11 | If the xHC detects a SET_ADDRESS request on the Default Control Endpoint Transfer Ring, it shall generate a TRB Error Completion status for the TD | TD 5.25 | |
| 4.6.5#12 | The xHC shall never forward a SET_ADDRESS request on a Default Control Endpoint Transfer Ring to a USB device | TD 5.25 | |
| 4.6.5#13 | When executing a Set Address Command with the BSR flag set to '0', if all available USB Device Addresses have been assigned for the BI that the device is associated with, then a Command Completion Event shall be generated with the Completion Code set to Resource Error. | TD 4.02 | |
| **Subsection reference: 4.6.6 Configure Endpoint** | | | |
| 4.6.6#1 | The xHC shall set the Context Entries field to 1 in the Output Slot Context when processing a Configure Endpoint Command with the DC flag set to 1. | 0 | |
| 4.6.6#2 | The xHC shall set the EP State to Disabled in the Output Endpoint Context on successful completion of a Configure Endpoint Command if the Drop Context flag is 1 and the Add context flag is 0. | 0 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.6.6#3 | The xHC shall set the EP State to Running in the Output Endpoint Context on successful completion of a Configure Endpoint Command if the Add context flag is 1. | 0 | |
| 4.6.6#4 | The xHC shall set the Slot State to Addressed in the Output Slot Context on successful completion of a Configure Endpoint Command if all endpoints except the default endpoint are in the Disabled state. | 0 | |
| 4.6.6#5 | The xHC shall set the EP State to Disabled in the Output Endpoint Context except the default endpoint on successful completion of a Configure Endpoint Command with the DC flag set to 1. | 0 | |
| 4.6.6#6 | The xHC shall ignore the Input Context Pointer field in a Configure Endpoint Command with the DC flag set to 1. | TD 4.03 | |
| 4.6.6#7 | The xHC shall set the Completion Code field to Bandwidth Error in the Command Completion Event if there is insufficient bandwidth available when executing a Configure Endpoint Command. | TD 4.03 | |
| 4.6.6#8 | When executing a Configure Endpoint Command with the DC flag set to '0', after evaluating all Drop Context flags, if the xHC determines that there is insufficient bandwidth available in the secondary Bandwidth Domain, the xHC shall set the completion code of the Command Completion Event to Secondary Bandwidth Error unless there is also insufficient bandwidth in the Primary Bandwidth Domain | TD 4.14 | |
| 4.6.6#9 | When executing a Configure Endpoint Command with the DC flag set to '0', after evaluating all Drop Context flags, if the xHC determines that there is insufficient bandwidth available in both the primary and secondary Bandwidth Domains, the xHC shall set the completion code of the Command Completion Event to either Bandwidth Error or Secondary Bandwidth Error | TD 4.14 | |
| **Subsection reference: 4.6.6.1 Exit Latency Delta (ELD)** | | | |
| 4.6.6.1#1 | The ELD shall be reported by a Configure Endpoint Command as a non-zero value in the Command Completion Parameter field in the Command Completion Event. | N/A | |
| 4.6.6.1#2 | If the Completion Code of a Configure Endpoint Command = Success, then the ELD shall define the amount of in ms by which the current Max Exit Latency value for the slot may be successfully increased and still allow the configuration to succeed. | N/A | |
| 4.6.6.1#3 | If the Completion Code of a Configure Endpoint Command = Max Exit Latency Too Large Error, the ELD shall define the amount of time in microseconds that MaxExitLatency must be reduced by to enable success. | N/A | |
| 4.6.6.1#4 | The Command Completion Parameter field shall be cleared to '0' for all other Configure Endpoint Command Completion Condition Code values besides Max Exit Latency Too Large Error. | N/A | |
| **Subsection reference: 4.6.7 Evaluate Context** | | | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.6.7#1 | The xHC shall update the Output Slot Context on a successful completion of an Evaluate Context Command. | 0 | |
| 4.6.7#2 | The xHC shall update the Output Endpoint 0 Context on a successful completion of an Evaluate Context Command. | 0 | |
| 4.6.7#3 | The xHC shall not modify any parameters in the Output Context except the ones identified by the Input Control Context when executing an Evaluate Context Command. | 0 | |
| 4.6.7#4 | On a successful completion of the Evaluate Context Command, the xHC shall set the Completion code in the Command Completion Event to Success | TD 4.04 | |
| 4.6.7#5 | When an Evaluate Context Command modifies the value of Max Exit Latency, the xHC not drop the data of any Isoch TDs, including the endpoint that received the Evaluate Context Command. | TD 5.26 | |
| **Subsection reference: 4.6.8 Reset Endpoint** | | | |
| 4.6.8#1 | The xHC shall set the EP State field to Stopped in the Endpoint Context upon successful completion of a Reset Endpoint Command. | 0 | |
| 4.6.8#2 | The xHC shall disable doorbell for a halted endpoint. | 0 | |
| 4.6.8#3 | The xHC shall not release any endpoint bandwidth when executing a Reset Endpoint Command. | TD 4.05 | |
| 4.6.8#4 | On executing a successful Reset Endpoint Command with TSP = '1', the xHC shall preserve the USB2 Data Toggle or the USB3 Sequence Number for the pipe. | TD 5.23 | |
| 4.6.8#5 | On executing a successful Reset Endpoint Command with TSP = '1', the xHC shall continue execution by retrying the last transaction the next time the doorbell is rung, if no other commands have been issued to the endpoint. | TD 5.23 | |
| 4.6.8#6 | If a successful Reset Endpoint Command is followed by a Set TR Dequeue Pointer Command, the endpoint shall start execution at the beginning of the TRB referenced by the TR Dequeue Pointer the next time the doorbell is rung. | USB30CV MSC Tests | |
| 4.6.8#7 | On executing a successful Reset Endpoint Command with TSP = '1', the xHC shall maintain any USB2 split transaction state associated with the endpoint. | N/A | |
| **Subsection reference: 4.6.9 Stop Endpoint** | | | |
| 4.6.9#1 | The endpoint shall restart execution at the beginning of the TRB referenced by the new TR Dequeue Pointer if a Set TR Dequeue Pointer Command is issued while the endpoint is stopped. | 0 | |
| 4.6.9#2 | The xHC shall generate a Transfer Event with its Completion Code set to Stopped on execution of a Stop Endpoint Command while a TD is in progress. | TD 4.15 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.6.9#3 | The xHC shall set the EP State to Stopped when executing a Stop Endpoint Command if the EP State is Running. | 0 | |
| 4.6.9#4 | The xHC shall set the Completion Code to Context State Error in the Command Completion Event for a Stop Endpoint Command if the Endpoint State is not Running. | 0 | |
| 4.6.9#5 | The xHC shall stop performing transactions from the Transfer Ring after executing a Stop Endpoint Command. | TD 4.06 | |
| 4.6.9#6 | The xHC shall start executing TRBs on the Transfer Ring when it is restarted after executing a Stop Endpoint Command. | TD 4.06 | |
| 4.6.9#7 | When executing a Stop Endpoint command, the xHC shall invalidate all cached TRBs not associated with the current TD before continuing executing of the Transfer Ring. | TD 4.15 | |
| 4.6.9#8 | If a TD was stopped during its execution due to a Stop Endpoint Command, then a Stopped Transfer Event shall be generated by the xHC. | TD 4.15 | |
| 4.6.9#9 | For a .96 host with FSE=1 in HCCPARAMS if a TD was stopped during its execution due to a Stop Endpoint Command, then a Stopped Transfer Event shall be generated by the xHC. | TD 4.15 | |
| 4.6.9#10 | For a .96 host with FSE=0 in HCCPARAMS, if a Transfer Ring was stopped due to a Stop Endpoint Command while the transfer ring is empty, then a Stopped Transfer Event shall not be generated by the xHC. | TD 4.15 | |
| 4.6.9#11 | If a Transfer Ring is stopped due to a Stop Endpoint Command, and the FSE flag in the HCCPARAMS is '0', then the xHC shall guarantee that all transfer events associated with the stopped Transfer Ring are written to system memory before the Command Completion Event is written. | TD 4.15 | |
| 4.6.9#12 | If a Transfer Ring is stopped due to a Stop Endpoint command, and the FSE flag in the HCCPARAMS is '1', then the xHC shall generate a Stopped Transfer Event, unless that Transfer Ring has been halted due to an error condition. | TD 4.15 | |
| 4.6.9#13 | If a Transfer Ring is stopped due to a Stop Endpoint command, then the xHC shall generate a Stopped Transfer Event, unless that Transfer Ring has been halted due to an error condition. | TD 4.15 | |
| 4.6.9#14 | When generating a Stopped Transfer Event due to a Stop Endpoint command, if Transfer Ring associated with the target endpoint is empty, then the event shall be generated on the event ring indicated by the Slot Context Interrupter Target field. | TD 4.15 | |
| 4.6.9#15 | When executing a Stop Endpoint command, if the Transfer Ring has been Halted due to an error condition, then no Stopped Transfer Event shall be generated. | TD 4.15 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.6.9#18 | If a Transfer Ring is stopped due to a Stop Endpoint command while the TRB type referenced by the TR Dequeue Pointer is a Transfer TRB with a Residual Length >0, the xHC shall generate a Transfer Event with the Length field set to the remaining bytes left in the transfer | TD 4.15 | |
| 4.6.9#19 | If a Transfer Ring is stopped due to a Stop Endpoint command while the TRB type referenced by the TR Dequeue Pointer is a Transfer TRB with a Residual Length >0, the xHC shall generate a Transfer Event with a Completion Code set to Stopped | TD 4.15 | |
| 4.6.9#20 | If a Transfer Ring is stopped due to a Stop Endpoint command while the TRB type referenced by the TR Dequeue Pointer is an invalid TRB, and the Chain bit in the previous TRB was set to 1, the xHC shall generate a Transfer Event on the Event Ring indicated by the Slot Context Interrupter Target field with the Completion Code set to Stopped - Length Invalid | TD 4.15 | |
| 4.6.9#21 | If a Transfer Ring is stopped due to a Stop Endpoint command while the TRB type referenced by the TR Dequeue Pointer is an invalid TRB, and the Chain bit of the previous TRB was set to 1, the xHC shall generate a Transfer Event on the Event Ring indicated by the Slot Context Interrupter Target field with the TRB Transfer Length set to 0 | TD 4.15 | |
| 4.6.9#22 | If a Transfer Ring is stopped due to a Stop Endpoint command while the TRB type referenced by the TR Dequeue Pointer is a an invalid TRB, the Chain bit of the previous TRB was set to 0 or there were no previous TRB's, and the FSE flag is set to 1, the xHC shall generate a Transfer Event with the Completion Code set to Stopped | TD 4.15 | |
| 4.6.9#23 | If a Transfer Ring has been Halted due to an error condition when a Stop Endpoint Command is received, the xHC shall not generate a Stopped Transfer Event | TD 4.15 | |
| 4.6.9#24 | Before generating a Command Completion Event for the Stop Endpoint Command, the xHC shall write the final value of the endpoint's Dequeue Pointer to the TR Dequeue Pointer field and CCS flag to the DCS field of the Output endpoint Context or Stream context associated with the stopped Transfer Ring. | TD 5.18 | |
| 4.6.9#25 | Before generating a Command Completion Event for the Stop Endpoint Command, if Stopped EDTLA Capability (SEC) = '1', then the xHC shall write the value of the EDTLA to the Stopped EDTLA field of the Stream Context associated with the stopped Transfer Ring. | N/A | |
| 4.6.9#26 | Before generating a Command Completion Event for the Stop Endpoint Command, the xHC shall also ensure that the Stream Context TR Dequeue Pointer, DCS, and Stopped ELDTA fields reflect the forward progress of any Stream that entered the Move Data state while the endpoint was in the Running state. | N/A | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.6.9#27 | If the Stop Endpoint Command is executed between TDs, then the xHC shall perform a Force Stopped Event (FSE) operation by generating a Transfer Event for the endpoint with Condition Code = Stopped – Invalid Length, TRB Pointer = current Dequeue Pointer value, and TRB Transfer Length = 0, then generate a Success Command Completion Event for the command. | TD 4.15 | |
| 4.6.9#28 | If SPC = '1' and the Stop Endpoint Command is executed, after a Short Packet condition as been detected, but before the end of the TD has been reached, then a TransferEvent TRB with its Completion Code set to Stopped – Short Packet and its TRB Transfer Length set to the value of the EDTLA shall be forced for the interrupted TRB, irrespective of whether its IOC or ISP flags are set. | TD 2.22 | |
| 4.6.9#29 | If a Transfer Event TRB and Command Completion Event TRB are both generated for a Stop Endpoint Command, the Transfer Event will always precede the Command Completion Event. | TD 5.18 | |
| 4.6.9#30 | If a TD is in progress for the pipe when the Stop Endpoint Command is executed, and a Short Packet condition has not been detected or SPC = '0' and a Short Packet has been detected, then a Transfer Event TRB with its Completion Code set to Stopped shall be forced for the interrupted TRB, irrespective of whether its IOC or ISP flags are set. | TD 5.18 | |
| 4.6.9#31 | The xHC shall wait for any partially completed USB2 split transactions to finish before completing the Stop Endpoint Command. | N/A | |
| 4.6.9#32 | The SPC bit of HCCPARAMS1 shall be set to 1 for all xHC 1.1 and xHC 1.2 compliant xHCs. | TD 1.02 | |
| 4.6.9#33 | The SEC bit of HCCPARAMS1 shall be set to 1 for all xHC 1.1 and xHC 1.2 compliant xHCs | TD 1.02 | |
| **Subsection reference: 4.6.10 Set TR Dequeue Pointer** | | | |
| 4.6.10#1 | The xHC shall copy the value of the New TR Dequeue Pointer field in the Set TR Dequeue Pointer Command TRB to the TR Dequeue Pointer field of the target Endpoint Context if the Stream ID is zero. | 0 | |
| 4.6.10#2 | The xHC shall copy the value of the New TR Dequeue Pointer field in the Set TR Dequeue Pointer Command TRB to the TR Dequeue Pointer field of the target Stream Context if the Stream ID is non-zero. | TD 4.17 | |
| 4.6.10#3 | The xHC shall copy the value of the DCS field in the Set TR Dequeue Pointer TRB to the DCS field of the target Endpoint Context if the Stream ID is zero. | 0 | |
| 4.6.10#4 | The xHC shall copy the value of the DCS field in the Set TR Dequeue Pointer TRB to the DCS field of the target Stream Context if the Stream ID is non-zero. | TD 4.17 | |
| 4.6.10#5 | If a Set TR Dequeue Pointer Command is executed that points to a TRB that had previously been partially completed TD, the xHC shall treat that TRB as the first TRB of the TD. | TD 4.15 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.6.10#6 | When the xHC receives a Set TR Dequeue Pointer Command, it shall clear any prior transfer state, e.g. setting the EDTLA to 0, clearing any partially completed USB2 split transactions, etc. | N/A | |
| **Subsection reference: 4.6.11 Reset Device** | | | |
| 4.6.11#1 | The xHC shall set the EP State field to Disabled for each Endpoint Context except for the Default Control Endpoint upon successful completion of a Reset Device Command. | 0 | |
| 4.6.11#2 | When executing the Reset Device command, the xHC shall transition the Default Control Endpoint to the Stopped or Running state. | TD 4.08 | |
| 4.6.11#3 | When executing the Reset Device command, if the Device Slot identified by the Slot ID is enabled, the xHC shall terminate any USB activity on all endpoints | TD 6.01 | |
| 4.6.11#4 | When executing the Reset Device command, if the Device Slot identified by the Slot ID is enabled, the xHC shall disable the Doorbell for all endpoints except the Default Control Endpoint. | TD 4.08 | |
| 4.6.11#5 | When executing the Reset Device command, if the Device Slot identified by the Slot ID is enabled, the xHC shall free any bandwidth allocated to any periodic endpoints | TD 4.08 | |
| 4.6.11#6 | When executing the Reset Device command, the xHC shall transition the Default Control Endpoint to the Stopped state | TD 4.08 | |
| 4.6.11#7 | When executing the Reset Device command on a slot in the Addressed or Configured state, the xHC shall set the Context Entries field fo the Slot Context to '1'. | TD 4.08 | |
| **Subsection reference: 4.6.13 Negotiate Bandwidth** | | | |
| 4.6.13#1 | The xHC shall generate Bandwidth Request Events when executing a Negotiate Bandwidth Command. | 0 | |
| 4.6.13#2 | The Slot ID field shall identify the candidate slot in a Bandwidth Request Event. | 0 | |
| 4.6.13#3 | The xHC shall set the Completion Code to Success in a Bandwidth Request Event. | 0 | |
| **Subsection reference: 4.6.15 Get Port Bandwidth** | | | |
| 4.6.15#1 | The xHC shall update the Port Bandwidth Context with the computed bandwidth available for each port based on the specified speed when executing a Get Port Bandwidth Command. | 0 | |
| 4.6.15#2 | The xHC shall set the available bandwidth to 0 for USB 3 ports when the Dev Speed is LS, FS or HS in a Get Port Bandwidth Command. | 0 | |
| 4.6.15#3 | The xHC shall set the available bandwidth to 0 for USB 2 ports when the Dev Speed is SS in a Get Port Bandwidth Command. | 0 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.6.15#4 | The xHC shall set the available bandwidth to 0 for a port assigned to the Debug Capability in a Get Port Bandwidth Command. | DbC TBD | |
| 4.6.15#5 | For a Command Completion Event of the Get Port Bandwidth command with Hub Slot ID field is NOT '0', the xHC shall set the appropriate fields in the Port Bandwidth context to the Computed Percentage of Bandwidth available for the ports of the hub specified by the Hub Slot ID based on their Speed, using the value of the Dev Speed field for ports that do not have device attached, if the dev speed is not equal to Undefined or Reserved or Unknown Speed | TD 4.11 | |
| 4.6.15#6 | The Get Port Bandwidth command shall be completed with a TRB Error Completion Code if it is not supported by the xHC. | TD 4.11 | |
| **Subsection reference: 4.6.16 Force Header** | | | |
| 4.6.16#1 | The xHC shall set the Slot ID field to 0 in the Command Completion Event of a Force Header Command. | 0 | |
| 4.6.16#2 | The xHC shall set the Completion Code to Undefined Error in the Command Completion Event of the Force Header Command if transmission of the Transaction Packet was unsuccessful. | TD 4.12 | |
| **Subsection reference: 4.6.17 Get Extended Property** | | | |
| 4.6.17#1 | If the Get/Set Extended Property Capability (GSC) bit in the HCCPARAMS2 register is '1', the xHC shall support the Get Extended Property command. | TD 1.10 | |
| 4.6.17#2 | The Get Extended Property Command shall complete with a Success Completion Code if the command is supported. | TD 1.10 | |
| 4.6.17#3 | The Get Extended Property Command shall complete with a TRB Error Completion Code if the command is not supported. | TD 1.10 | |
| **Subsection reference: 4.6.17.2 Get Extended Property** | | | |
| 4.6.17.2#1 | When a Get Extended Property Command is executed by the xHC, it shall write the required information to the Extended Property Context location specified by the Extended Property Context pointers. | TD 1.10 | |
| 4.6.17.2#2 | When a Get Extended Property Command is executed by the xHC, it shall insert a Command Completion Event TRB in the Event ring with fields described in section 4.6.17.2. | TD 1.10 | |
| | | | |
| **Subsection reference: 4.6.17.3 Enumeration of Supported Extended Capabilities** | | | |
| 4.6.17.3#1 | In response to receiving a Get Extended Property Command with the Extended Capabilites ID set to zero, the xHC shall return a Command Completion TRB with a bit set for each supported extended capability in bits [15:0] of the Command Completion Parameter field. | TD 1.10 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.6.17.3#2 | Bit 0 of Extended Capability Identifier shall indicate support for Audio Sideband Extended Capability. | Not Tested | |
| 4.6.17.3#3 | Bits 11:1 of Extended Capability Identifier are Reserved and shall be zero. | Not Tested | |
| **Subsection reference: 4.6.18 Set Extended Property** | | | |
| 4.6.18#1 | If the Get/Set Extended Property Capability (GSC) bit in the HCCPARAMS2 register is '1', the xHC shall support the Set Extended Preoprty Command. | TD 1.10 | |
| 4.6.18#2 | When a Set Extended Property Command is executed by the xHC, it shall complete with a Success Completion Code if the command is supported. | Not Tested | |
| 4.6.18#3 | When a Set Extended Property Comamnd is executed by the xHC, it shall respond with TRB Error Completion Code if the command is not supported. | Not Tested | |
| **Subsection reference: 4.8 Endpoint** | | | |
| **Subsection reference:  4.7 Doorbells** | | | |
| 4.7#1 | The xHC shall ignore doorbell references to endpoints in the Halted or Error state. | 0 | |
| **Subsection reference: 4.8.1 Endpoint Addressing** | | | |
| 4.8.1#1 | The xHC shall ignore the OUT Endpoint Context of bidirectional endpoints. | TBD | Need Compliance Device Support |
| **Subsection reference: 4.8.3 Endpoint Context State** | | | |
| 4.8.3#1 | The xHC shall set the EP State to Halted in the Endpoint Context on a halt condition or USB transaction error detected on a USB pipe. | 0 <br> TD 5.10 | |
| 4.8.3#2 | The xHC shall transition all endpoints to the Disabled state on a successful Disable Slot Command. | 0 | |
| 4.8.3#3 | The xHC shall transition an endpoint in the Stopped state to the Running state when the doorbell is rung. | 0 | |
| 4.8.3#4 | The xHC shall transition an endpoint in the Stopped state to the Running state if it is successfully modified by a Configure Endpoint command. | 0 | |
| 4.8.3#5 | The xHC shall transition an endpoint in the Error state to the Stopped state upon successful completion of a Set TR Dequeue Pointer Command. | TD 4.07 | |
| 4.8.3#6 | The xHC shall not generate Transfer Events for an endpoint in the Stopped state | TD 4.15 | |
| 4.8.3#7 | The xHC shall transition an control, bulk or interrupt endpoint to the Halted state if a tHostTransactionTimeout occurs. | N/A | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.8.3#8 | If a tHostTransactionTimeout occurs on an isochronous endpoint, thehost shall not perform any more transactions to the endpoint in the current Service Interval. | N/A | |
| 4.8.3#9 | If a tHostTransactionTimeout occurs on an isochronous endpoint the host shall not halt the endpoint. | N/A | |
| 4.8.3#10 | If a tHostTransactionTimeout occurs on an isochronous endpoint  the host shall restart the transactions to the endpoint in the next Service Interval. | N/A | |
| 4.8.3#11 | Retries are not performed for any endpoint type if a tHostTransactionTimeout occurs. | N/A | |
| **Subsection reference: 4.9 TRB Ring** | | | |
| 4.9#1 | The xHC shall consume TRBs on the Transfer Ring in order, starting at the TRB referenced by the Dequeue Pointer. | 0 | |
| 4.9#2 | The xHC shall schedule Max Packet Size transactions for all packets associated with a TD except for the last packet. | Analyzer Tests | |
| **Subsection reference: 4.9.1 Transfer Descriptors** | | | |
| 4.9.1#1 | The xHC shall set the dequeue pointer equal to the enqueue pointer when processing a partially completed TD and advance to the next TD boundary the next time the doorbell is rung. | TD 2.16 | |
| 4.9.1#2 | If the xHC receives a short packet from a device, then it shall retire the current TD. | TD 5.09 | |
| 4.9.1#3 | If the xHC receives a short DP on a SS IN pipe with the EOB flag set, the xHC shall retire the current TD and wait for an ERDY from the device before beginning IN transactions for the next TD if one exists. | TBD | Need Compliance Device support |
| 4.9.1#4 | If while advancing to the next TD due to an error in the previous TD, a TRB is encountered with the IOC bit set, the xHC shall generate a Transfer event for that encountered TRB with its Condition Code set to either Success, or the same Completion Code as the previous Transfer Event for the endpoint (the one indicating the initial error). | Implicit | |
| **Subsection reference: 4.9.2 Transfer Ring Management** | | | |
| 4.9.2#1 | The xHC shall update the DCS flag in the associated Stream Context when the Stream state is saved. | | |
| 4.9.2#2 | The xHC shall toggle the CCS of the Transfer Ring when it encounters a Link TRB with the TC flag set to 1. | Implicitly tested in loopback TDs | |
| **Subsection reference: 4.9.2.2 Pointer Advancement** | | | |
| 4.9.2.2#1 | The xHC shall process TRBs until a Transfer Ring is empty or until the endpoint is stopped. | 0 | |
| **Subsection reference: 4.9.4 Event Ring Management** | | | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.9.4#1 | The xHC shall initialize the Event Ring when the ERSTBA register is written and the RS flag is 1 in the USBCMD register. | Implicit in all transfer tests | |
| 4.9.4#2 | The xHC shall report an Event Ring Full Error when there is only room for one more entry in the Event Ring. | 0 | |
| 4.9.4#4 | If the Primary Event Ring is full, the xHC shall stop processing all Transfer and Command Rings | 0 | |
| **Subsection reference: 4.9.4.2 Shrinking an Event Ring** | | | |
| 4.9.4.2#1 | The xHC shall continue to use the removed segment until the next pass through of the Event Ring if the Event Ring Enqueue Pointer is pointing to it. | 0 | |
| **Subsection reference: 4.9.4.3 Primary and Secondary Event Rings** | | | |
| 4.9.4.3#1 | The xHC shall direct the Transfer Events to the Secondary Event Ring if the Interrupter Target is set to a non-zero value in the Transfer TRB. | 0 | |
| 4.9.4.3#2 | The xHC shall direct the Transfer Events to the Primary Event Ring if the Interrupter Target is set to 0 in the Transfer TRB. | 0 | |
| 4.9.4.3#3 | The xHC shall direct all Bandwidth Request Events to the Event Ring specified by the Interrupter Target field in the Slot Context. | 0 | |
| 4.9.4.3#4 | The xHC shall direct all Device Notification Events to the Event Ring specified by the Interrupter Target field in the Slot Context. | 0 | |
| 4.9.4.3#5 | The xHC shall not insert Event TRBs to a Secondary Event Ring except Transfer Events, Bandwidth Request Events, Device Notification Events or Vendor Defined Events. | TD 2.09, TD 2.11, TD 2.12 | |
| **Subsection reference: 4.10 Host Controller TRB Handling** | | | |
| **Subsection reference: 4.10.1 Transfer TRBs** | | | |
| 4.10.1#1 | The xHC shall generate a Transfer Event TRB upon completion of a transfer TRB with the IOC flag set. | 0 <br> TD 5.10 | |
| 4.10.1#2 | The xHC shall generate a Transfer Event TRB on the Event Ring upon detection of a short packet if the IOC flag is set in the TRB. | 0 | |
| 4.10.1#3 | The xHC shall generate a Transfer Event TRB on the Event Ring upon detection of a short packet if the ISP flag is set in the TRB. | 0 | |
| 4.10.1#4 | The xHC shall not generate a Transfer Event TRB upon detection of a short packet unless either IOC or ISP flag is set in the Transfer TRB. | 0 | |
| 4.10.1#5 | The xHC shall set the Completion Code to Short Packet when generating a Transfer Event TRB as a result of detecting a Short Packet. | 0 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.10.1#6 | The xHC shall set the Length field to the residual number of bytes not written to the Transfer TRB's data buffer when generating a Transfer Event TRB as a result of detecting a Short Packet. | 0 | |
| 4.10.1#7 | The xHC shall execute the first Event Data TRB encountered while advancing to the end of a TD after detecting a Short Packet | 0 | |
| 4.10.1#8 | The xHC shall generate a Transfer Event upon detection of an error while executing a Transfer TRB. | TD 5.10 | |
| 4.10.1#9 | The xHC shall stop on the TRB in error and not advance the Dequeue Pointer upon detection of an error requiring an endpoint to halt while executing a Transfer TRB. | 0 | |
| 4.10.1#10 | The xHC shall execute all Event Data TRBs encountered while advancing to the end of a TD after detecting a Short Packet if PAE bit is set to 1 and skip the Event Data TRBs if PAE=0, generating a corresponding Transfer Event with Completion Code set to Short Packet and TRB Transfer Length equal to the number of bytes transferred since the beginning of the TD or the previous Event Data Transfer TRB of the TD Transfer Event. | TD 5.09 and implicit | |
| **Subsection reference: 4.10.1.1 Short Transfers** | | | |
| 4.10.1.1#1 | The xHC shall not transmit any data packets greater than the Max Packet Size. | Implicit | |
| 4.10.1.1#2 | The xHC shall parse all TRBs of the TD and generate a Transfer Event for any TRB with the IOC flag set to 1 if a Short Packet condition does not occur. | TD 5.01 TD 5.02 TD 5.03 TD 5.04 IOC iterations | |
| 4.10.1.1#3 | If the PAE flag is set (1), the xHC shall parse all TRBs after a Short Packet condition occurs. | TD 2.21 | |
| 4.10.1.1#4 | If the PAE flag is clear ('0'), the xHC shall not parse any TRBs after a Short Packet condition occurs. | TD 2.21 | |
| 4.10.1.1#5 | If Event Data TRBs are not used, then the total number of received bytes for a Short Packet TD is the sum of the TRB Transfer Length fields in all transfer TRBs up to and including the one that generated the Short Packet Event, minus the residual value of the TRB Transfer Length field in the Short Packet Event. | TD 5.09 | |
| 4.10.1.1#6 | A TD Completion event is generated if the IOC flag = '1' and the TD completes successfully. | Implicit | |
| 4.10.1.1#7 | A TD Completion event is generated if a short packet occurs and the ISP or IOC flags are set. | TD 5.09 | |
| 4.10.1.1#8 | A TD completion event is generated if an error condition is detected while processing any TRB within a TD. | Implicit | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.10.1.1#9 | Events generated for a TD by TRBs encountered before the TD Completion Event shall set their Completion Code to Success. | N/A | |
| 4.10.1.1#10 | After the TD Completion Event, if any subsequent Transfer TRBs are encountered with their IOC flag set while advancing to the end of the TD then those TRBs shall also generate an Event where the Completion Code field shall return the same value as the TD Completion Event. | N/A | |
| 4.10.1.1#11 | After the TD Completion Event, if any subsequent Transfer TRBs are encountered with their IOC flag set while advancing to the end of the TD then their TRB Transfer Length shall return the same value as the TD Completion Event. | N/A | |
| **Subsection reference: 4.10.2 Errors** | | | |
| 4.10.2#1 | The xHC shall not halt an Isoch endpoint when it generates a Transfer Event with an error completion code. | TD 2.18 | |
| **Subsection reference: 4.10.2.1 Stall Error** | | | |
| 4.10.2.1#1 | The xHC shall generate a Transfer Event TRB with the Completion Code set to Stall Error upon receiving a STALL PID from a USB device. | TD 5.10 | |
| 4.10.2.1#2 | The xHC shall stop further activity on the associated Transfer Ring upon receiving a STALL PID from a USB device. | TD 4.15 | |
| **Subsection reference: 4.10.2.2 TRB Error** | | | |
| 4.10.2.2#1 | The xHC shall only set the Completion Code to TRB Error in a Command Completion Event or a Transfer Event due to an error detected on a Transfer or Command TRB. | | |
| **Subsection reference: 4.10.2.3 USB Transaction Error** | | | |
| 4.10.2.3#1 | The xHC shall complete the TRB with USB Transaction Error while executing a Setup TRB, if a high-speed device returns a NAK handshake after Cerr number of retries. | TD 5.23 | |
| 4.10.2.3#2 | The xHC shall complete the TRB with USB Transaction Error while executing a Setup TRB, if a high-speed device returns a NYET handshake after Cerr number of retries. | TD 5.23 | |
| 4.10.2.3#3 | The xHC shall complete the TRB with USB Transaction Error while executing a Setup TRB, if a low or full-speed device returns a NAK handshake after Cerr retries for a complete-split. | TD 5.23 | |
| 4.10.2.3#4 | The xHC shall complete the TRB with USB Transaction Error while executing a Setup TRB, if a SuperSpeed device returns a NRDY TP after Cerr retries to a SETUP DP. | TD 5.23 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.10.2.3#5 | The xHC shall complete the TRB with USB Transaction Error when a CRC check failure is detected while transferring data to/from a USB2 device after CErr number of retries. | 0 | |
| 4.10.2.3#6 | The xHC shall complete the TRB with USB Transaction Error when a USB3 DPP Error is detected while transferring data to/from a SuperSpeed device after Cerr number of retries. | 0 | |
| 4.10.2.3#7 | The xHC shall complete the TRB with USB Transaction Error when a Timeout condition is detected while transferring data to/from a USB2 device after CErr number of retries. | 0 | |
| 4.10.2.3#8 | The xHC shall complete the TRB with USB Transaction Error when an invalid PID is detected while transferring data to/from a USB device after CErr number of retries. | TD 5.23 | Need Compliance Device support |
| 4.10.2.3#9 | The xHC shall complete the TRB with USB Transaction Error when a timeout condition is detected while transferring data to/from a SuperSpeed device with no retry. | 0 | |
| 4.10.2.3#10 | The xHC shall only report a USB Transaction Error in a Transfer Event due to an error detected on a Transfer TRB. | Implicit | |
| 4.10.2.3#11 | When a CRC check failure is detected while transferring data to/from an isochronous endpoint on a USB2 device, the xHC shall immediately complete the TRB with USB Transaction Error | TD 2.18 | |
| 4.10.2.3#12 | When a USB3 DPP Error is detected while transferring data to/from an isochronous endpoint on a USB 3 device, the xHC shall immeidately complete the TRB with USB Transaction Error | TD 2.18 | |
| 4.10.2.3#13 | When a Timeout condition is detected while transferring data to/from a USB3 device, the xHC shall immediately complete the TRB with USB Transaction Error | TD 5.11 | |
| **Subsection reference: 4.10.2.4 Babble Detected Error** | | | |
| 4.10.2.4#1 | The xHC shall set the Completion Code field to Babble Detected Error in the Transfer Event TRB when a device sends more data than either Total Bytes to Transfer or Maximum Packet size. | 0 | |
| 4.10.2.4#2 | The xHC shall not decrement the Bus Error Counter when completing a TRB with the Completion Code field set to Babble Detected Error. | TD 5.23 | |
| 4.10.2.4#3 | The xHC shall only set the Babble Detected Error in a Transfer Event. | Implicit | |
| **Subsection reference: 4.10.2.4.1 USB2 Protocol** | | | |
| 4.10.2.4.1#1 | The xHC shall either disable Packet Babble checking for the duration of the bus transaction or do Packet Babble checking based solely on Maximum Packet Size when a data PID mismatch occurs. | TD 2.19 | Need Compliance Device support |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.10.2.4.1#2 | The xHC shall ignore the received data and respond with an ACK handshake when a data PID mismatch occurs for a USB2 device. | TD 2.19 | Need Compliance Device support |
| **Subsection reference: 4.10.3 Events** | | | |
| 4.10.3#1 | If the IOC flag is set, the xHC shall generate an event, unless the PAE flag is cleared ('0'), and a Short Packet occurs, in which case the xHC shall not generate events for Event Data TRBs after the Short Packet Completion. | | |
| **Subsection reference: 4.10.3.1 Ring Overrun and Underrun** | | | |
| 4.10.3.1#1 | The xHC shall generate a Ring Overrun Event if the Isoch ring is empty when the xHC is ready to perform a previously scheduled IN transfer. | 0 | |
| 4.10.3.1#2 | The xHC shall generate a Ring Underrun Event if the Isoch ring is empty when the xHC is ready to perform a previously scheduled OUT transfer. | 0 | |
| 4.10.3.1#3 | The xHC shall execute only one Isoch TD in each ESIT. | TD 5.20 | |
| 4.10.3.1#4 | For xHC 1.1 compliant xHCs, the TRB Transfer Length for Ring Overrun/Ring Underrun Transfer Events shall be set to '0'. | TBD | |
| 4.10.3.1#5 | For xHC 1.1 compliant xHCs, the TRB Pointer field shall point to the address of the Dequeue pointer where the Overrun/Underrun condition was detected. | TBD | |
| 4.10.3.1#6 | For xHCI 1.0 compliant xHCs, the TRB Pointer field for Overrun/Underrun Transfer Events shall be cleared to '0'. | TBD | |
| **Subsection reference: 4.10.3.2 Missed Service Error** | | | |
| 4.10.3.2#1 | The xHC shall only set the Missed Service Error on Transfer Event TRBs generated from an isochronous transfer. | N/A | |
| 4.10.3.2#2 | The xHC shall not generate a Transfer Event TRB with the completion code set to Missed Service Error if an Isoch transfer was not completed due to another error condition. | Implicit | |
| **Subsection reference: 4.10.3.3 Split Transaction Error** | | | |
| **Subsection reference:  4.10.4  IOC Flag** | | | |
| 4.10.4#1 | If the IOC flag is set, the xHC shall generate an event, unless the PAE flag is cleared ('0'), and a Short Packet occurs, in which case the xHC shall not generate events for Event Data TRBs after the Short Packet Completion. | TD 2.13 | |
| **Subsection reference: 4.11 TRBs** | | | |
| **Subsection reference: 4.11.2 Transfer TRBs** | | | |
| 4.11.2#1 | The xHC shall generate a Transfer Event upon completion of a Transfer TRB if the FE flag is 1 in the Endpoint Context. | TD 2.13 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| **Subsection reference: 4.11.2.1 Normal TRB** | | | |
| 4.11.2.1#1 | The xHC shall not place a Normal TRB on the Event Ring. | Implicit | |
| **Subsection reference: 4.11.2.2 Setup Stage, Data Stage, and Status Stage TRB** | | | |
| 4.11.2.2#1 | The xHC shall not generate an error just because a Data Stage TD follows a Setup Stage TD with the wLength set to 0. | | |
| 4.11.2.2#2 | The xHC shall not generate an error just because a Status Stage TD does not follow a Setup Stage TD with the wLength set to 0. | | |
| 4.11.2.2#3 | The xHC shall not generate an error just because a Data Stage TD does not follow a Setup Stage TD with the wLength greater than 0. | | |
| 4.11.2.2#4 | The xHC shall not generate an error just because the total size of a Data Stage TD does not equal the wLength value in the Setup Stage TD. | | |
| 4.11.2.2#5 | The xHC shall not generate an error just because the direction of the Data Stage TD is not consistent with the bmRequestType in the Setup Stage TD. | | |
| 4.11.2.2#6 | The xHC shall not generate an error just because the direction of the Status Stage TD is not consistent with the bmRequestType in the Setup Stage TD. | | |
| 4.11.2.2#7 | The xHC shall generate a Transfer Event with the Completion Code set to USB Transaction Error if an error is detected on a Setup, Data or Status Stage TD. | | |
| 4.11.2.2#8 | The xHC shall generate a Status TP to the device when executing a Status Stage TRB for a SS device. | Implicit | |
| **Subsection reference: 4.11.2.3 Isoch TRB** | | | |
| 4.11.2.3#1 | The xHC shall not change the EP State of the Isoch OUT endpoint from running state as a result of observing a Ring Underrun condition. | 0 | |
| 4.11.2.3#2 | The xHC shall not change the EP State of the Isoch IN endpoint from running state as a result of observing a Ring Overrun condition. | 0 | |
| 4.11.2.3#4 | The xHC shall not schedule any Isoch transfers when the Transfer Ring is empty for Isoch OUT endpoints. | TD 5.04 | |
| 4.11.2.3#5 | The xHC shall schedule the Isoch TD in the next match of the Frame ID with the MFINDEX register for an Isoch TRB with the SIA flag set to 0. | TD 5.04 | Not implemented |
| 4.11.2.3#6 | The xHC shall schedule the Isoch TD as soon as possible for an Isoch TRB with the SIA flag set to 1. | TD 5.04 | Not implemented |
| 4.11.2.3#7 | If ETC and ETE = 1 then TBC field supports the definition of Burst Counts up to 32 (and the TD size field is deprecated in an Isoch TRB), otherwise the TBC field supports the definition of Burst Counts up to 4 (and the TD Size field is valid). | N/A | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.11.2.3#8 | The Extended TCB Capability (ETC) shall not be enabled by an xHC implementation if the Large ESIT Payload Capability (LEC = 1) is not supported. | N/A | |
| 4.11.2.3#9 | Service Interval Boundaries are aligned.  i.e. if Interval = '1', then the Service Interval is 2 microframes long and begins with the low order bit of the MFINDEX register = 0. | N/A | |
| **Subsection reference: 4.11.2.5 Frame ID** | | | |
| 4.11.2.5#1 | If the Frame ID field value of an Isoch TD references an ESIT that already has an Isoch TD scheduled for it, then the xHC shall ignore the Frame ID and the xHC shall schedule the Isoch TD as if SIA = '1' | TD 5.04.05 | |
| 4.11.2.5#2 | If the Contiguous Frame ID Capability (CFC = '1'), then the xHC shall match the Frame ID in every Isoch TD against the Frame Index of the MFINDEX register. | N/A | |
| 4.11.2.5#3 | Contiguous Frame ID Capabiltiy support (CFC = '1') is mandatory for all xHC 1.1 and xHC 1.2 compliant xHCI implementations. | TD 1.02 | |
| 4.11.2.5#4 | If the contiguous Frame ID Capability (CFC = '0'), then the xHC shall start the Isoch data flow when the MFINDEX Frame Index matches the Frame ID value specified in the first Isoch TD and ignore the Frame ID fields in subsequent Isoch TDs until the data flow is terminated. | TBD | |
| **Subsection reference: 4.11.2.5.1 Frame ID ESIT Rules** | | | |
| 4.11.2.5.1#1 | If the Frame ID field value of an Isoch TD references an ESIT that already has an Isoch TD scheduled for it, then the Frame ID field shall be ignored and the xHC shall schedule the Isoch TD as if SIA='1'. | TBD | |
| **Subsection reference: 4.11.2.5.2 Resynchronization** | | | |
| 4.11.2.5.2#1 | If a Missed Service Error occurs then the xHC is required to advance through a Transfer Ring until it is "recynchronized" or the ring is exhausted. | TBD | |
| 4.11.2.5.2#2 | The data associated with Isoch TDs that are skipped over while attempting to resyncronize a pipe is not moved, however a Missed Service Error should be generated for every skipped Isoch TD. | TBD | |
| 4.11.2.5.2#3 | The xHC shall not drop Events associated with TRBs as it attempts to resynchronize an Isoch pipe. | TBD | |
| **Subsection reference:  4.11.3: Event TRBs** | | | |
| 4.11.3.1#1 | The xHC shall only generate a Transfer Event for a Transfer TRB for that endpoint if the IOC flag is set, a short transfer occurs with the ISP flag set, or an error occurs during the execution of the Transfer TRB. | Implicit | |
| **Subsection reference: 4.11.4 Command TRBs** | | | |
| **Subsection reference: 4.11.4.6 Evaluate Context Command TRB** | | | |
| 4.11.4.6#1 | The xHC shall not change the EP State while processing an Evaluate Context Command TRB. | 0 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| **Subsection reference: 4.11.4.12 Negotiate Bandwidth Command TRB (Optional Normative)** | | | |
| 4.11.4.12#1 | The xHC shall support a Negotiate Bandwidth Command TRBs if the BNC flag is 1 in the HCCPARAMS register. | 0 | |
| **Subsection reference: 4.11.4.13 Set Latency Tolerance Value Command TRB (Optional Normative)** | | | |
| 4.11.4.13#1 | The xHC shall support a Set Latency Tolerance Value Command if the LTC flag is 1 in the HCCPARAMS register. | 0 | |
| **Subsection reference: 4.11.5 Other TRBs** | | | |
| **Subsection reference: 4.11.5.1 Link TRB** | | | |
| 4.11.5.1#1 | The xHC shall ignore the CH flag in Link TRBs in the Command Ring. | 0 | |
| 4.11.5.1#2 | The xHC shall generate a Transfer Event for a Link TRB with the IOC flag set to 1 in a Transfer Ring. | 0 | |
| 4.11.5.1#3 | The xHC shall set the Slot ID of a Transfer Event resulting from a Link TRB to the Slot ID of the slot associated with the Transfer Ring. | 0 | |
| 4.11.5.1#4 | The xHC shall set the Endpoint ID of a Transfer Event resulting from a Link TRB to the Endpoint ID of the endpoint associated with the Transfer Ring. | 0 | |
| 4.11.5.1#5 | The xHC shall set the Length field to 0 in a Transfer Event resulting from a Link TRB. | 0 | |
| 4.11.5.1#6 | The xHC shall set the Completion Code field to Success in a Transfer Event resulting from a Link TRB. | 0 | |
| 4.11.5.1#7 | The xHC shall ignore the Interrupter Target field of Link TRBs found on the Command Ring. | 0 | |
| 4.11.5.1#8 | The xHC shall generate a Command Completion Event for a Link TRB with the IOC flag set to 1 in the Command Ring. | 0 | |
| 4.11.5.1#9 | The xHC shall set the Slot ID field to 0 in a Command Completion Event resulting from a Link TRB. | 0 | |
| 4.11.5.1#10 | The xHC shall set the Completion Code field to Success in a Command Completion Event resulting from a Link TRB. | 0 | |
| **Subsection reference: 4.11.5.2 Event Data TRB** | | | |
| 4.11.5.2#1 | The xHC shall reset the EDTLA to 0 at the beginning of a TD | 0 | |
| 4.11.5.2#2 | The xHC shall increment the EDTLA by the number of bytes transferred when a Transfer TRB is completed. | 0 | |
| 4.11.5.2#3 | The xHC shall generate an Event Data Transfer Event with the Length field set to the value of the EDTLA when it encounters an Event Data TRB with the IOC flag set to 1. | 0 | |
| 4.11.5.2#4 | The xHC shall reset the EDTLA to 0 after generating an Event Data Transfer Event. | TD 2.10 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.11.5.2#5 | The xHC shall generate an Event Data Transfer Event and set the Length field to the actual number of bytes transferred upon detection of a Short Packet for a TD that is terminated by an Event Data TRB with the IOC flag set to 1. | 0 | |
| 4.11.5.2#6 | The xHC shall set the ED flag to 1 in the Event Data Transfer Event. | 0 | |
| 4.11.5.2#7 | The xHC shall set the Completion Code field of an Event Data Transfer Event to the Completion Code of the previously executed TRB. | 0 | |
| 4.11.5.2#8 | The xHC shall set the Completion Code field of an Event Data Transfer Event to Success if there is no previously executed TRB. | 0 | |
| 4.11.5.2#9 | The xHC shall set the Parameter Component of the Transfer Event generated by an Event Data TRB to the value of the Parameter Component in the Event Data TRB. | 0 | |
| 4.11.5.2#10 | The xHC shall set the Event Data Transfer Length Accumulator for an endpoint when a Set TR Dequeue Pointer Command is executed for that endpoint. | Implicit | |
| 4.11.5.2#11 | If a Stopped Transfer Event is generated and the Condition Code = Stopped - Short Transfer ,then the TRB Transfer Length field of the Transfer Event shall contain the value of the EDLTA. | TD 2.22 | |
| **Subsection reference: 4.11.7 TD Usage Rules** | | | |
| 4.11.7#1 | The xHC shall generate an Event every time it encounters an IOC flag equal to '1', irrespective of any error events that may be forced for earlier TRBs in a TD that did not have its IOC flag set. | TD 2.15 | |
| **Subsection reference: 4.11.7.1  TD Fragments** | | | |
| 4.11.7.1#1 | The xHC shall define the IOC Interval field of the HSCPARAMS2 register to a value that allows software to implement the TD Fragments scheme defined in the 1.0 spec. | Implicitly tested | |
| **Subsection reference: 4.12 Streams** | | | |
| 4.12#1 | The SEC bit of the HCCPARAMS1 register shall be set to 1 for xHC 1.1 and xHC 1.2 compliant xHCs. | TD 1.02 | |
| **Subsection reference: 4.12.2 Stream ID Management** | | | |
| **Subsection reference: 4.12.2.1 Stream Array Bounds Checking** | | | |
| 4.12.2.1#1 | The xHC shall generate a Transfer Event with the Completion Code set to Invalid Stream ID Error if Streams are enabled and the xHC receives a USB packet with a Stream ID of 0. | TD 4.18 | |
| 4.12.2.1#2 | The xHC shall generate a Transfer Event with the Completion Code set to Invalid Stream ID Error if Linear Stream Array is enabled, and if the Stream ID is greater than the Primary Stream Array size. | TD 4.18 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.12.2.1#3 | The xHC shall generate a Transfer Event with the Completion Code set to Invalid Stream ID Error if Secondary Stream Arrays are enabled, the SCT field is 1 in the Primary Stream Context and the SSID is not 0. | TD 4.18 | |
| 4.12.2.1#4 | The xHC shall generate a Transfer Event with the Completion Code set to Invalid Stream ID Error if Secondary Stream Arrays are enabled, the SCT is 2 to 7 in the Primary Stream Context, and the SSID is 0. | TD 4.18 | |
| 4.12.2.1#5 | The xHC shall interpret the TR Dequeue Pointer field value 0 of Stream Context as an empty Transfer Ring. | TD 4.18 | |
| 4.12.2.1#6 | The xHC shall generate a Transfer Event with the Completion Code set to Invalid Stream Type Error if Streams are enabled, the TR Dequeue Pointer field is 0 and the SCT field is SSA in the Stream Context. | TD 4.18 | |
| 4.12.2.1#7 | The xHC shall generate a Transfer Event with the Completion Code set to Invalid Stream Type Error if Linear Stream Arrays are enabled, and the SCT is not Transfer Ring in the Stream Context. | TD 4.18 | |
| 4.12.2.1#8 | The xHC shall generate a Transfer Event with the Completion Code set to Invalid Stream Type Error if Secondary Stream Arrays are enabled, and the SCT is 0 in the Stream Context. | TD 4.18 | |
| 4.12.2.1#9 | The xHC shall generate a Transfer Event with the Completion Code set to Invalid Stream ID Error if Secondary Stream Arrays are enabled, the SCT is out of range in the Primary Stream Context, the SSID is not 0 and the SCT is not 0 in the Secondary Stream Context. | TD 4.18 | |
| 4.12.2.1#10 | The xHC shall generate a Transfer Event with the Completion Code set to Invalid Stream Type Error if Secondary Stream Arrays are enabled, the SCT is 2 to 7 in the Primary Stream Context, the SSID is not 0, and the SCT is not 0 in the Secondary Stream Context. | TD 4.18 | |
| **Subsection reference: 4.12.3 Evaluate Next TRB (ENT)** | | | |
| 4.12.3#1 | The xHC shall generate an Event Data Transfer Event before saving streams when it has completed executing a TRB with the CH and ENT flags set to 1, and the next TRB is an Event Data TRB. | N/A | |
| 4.12.3#2 | The xHC shall ignore the ENT flag in a TRB with the CH flag set to 0. | | |
| 4.12.3#3 | When the xHC is evaluating the next TRB in a transfer ring due to the previous TRB's ENT flag being '1', if that next TRB is a Link TRB, then the xHC shall execute the Link TRB and evaluate the TRB that the Link TRB points to before advancing to the next endpoint in the Pipe Schedule | TD 2.20 | |
| **Subsection reference: 4.14 Managing Transfer Rings** | | | |
| 4.14#1 | The xHC shall clear the PP bit to 0 in the ACK TP or DP when it detects that the Transfer Ring will be empty after executing it. | TD 5.24 | |
| 4.14#2 | The xHC shall set the PP bit to 1 in ACK TPs or DPs unless the Transfer Ring will be empty. | TD 5.24 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| **Subsection reference: 4.14.2 Periodic Transfer Ring Scheduling** | | | |
| 4.14.2#1 | The xHC shall advance the MFINDEX register every 125us while the RS flag is 1 in the USBCMD register. | 0 | |
| 4.14.2#2 | The xHC shall send a PING packet prior to initiating a periodic transfer when the Max Exit Latency greater than 0 in the Slot Context for a SS device. | TD 5.26 | |
| **Subsection reference: 4.14.2.1 Isochronous Transfer Ring Scheduling** | | | |
| 4.14.2.1#1 | The xHC shall not schedule an Isoch IN transfer on the bus when the Transfer Ring is empty. | TD 5.04 | |
| 4.14.2.1#2 | The xHC shall transmit a zero-length packet when it encounters a zero-length Isoch OUT TD on the Transfer Ring. | TD 5.04.06 | |
| **Subsection reference: 4.14.2.1.1 High Speed Endpoints** | | | |
| **Subsection reference:  4.14.3.1 Low-, Full-, and High-Speed Endpoints** | | | |
| 4.14.3.1#1 | If a Low-speed interrupt IN transaction is NAKed, then the xHC shall retry the Interrupt TD in the next ESIT | TD 5.16 | |
| 4.14.3.1#2 | If a Full-speed interrupt IN transaction is NAKed, then the xHC shall retry the Interrupt TD in the next ESIT | TD 5.16 | |
| 4.14.3.1#3 | If a High-speed interrupt IN transaction is NAKed, then the xHC shall retry the Interrupt TD in the next ESIT | TD  5.16 | |
| 4.14.3.1#4 | If a Low-speed interrupt OUT transaction is NAKed, then the xHC shall not issue another transaction for the endpoint until 1 EIST later | TD 5.16 | |
| 4.14.3.1#5 | If a Full-speed interrupt OUT transaction is NAKed, then the xHC shall not issue another transaction for the endpoint until 1 EIST later | TD  5.16 | |
| 4.14.3.1#6 | If a High-speed interrupt OUT transaction is NAKed, then the xHC shall not issue another transaction for the endpoint until 1 EIST later | TD 5.16 | |
| 4.14.3.1#7 | If a Low-speed Interrupt OUT transaction times out, then the xHC shall retry the transaction up to Cerr times for the endpoint in the same ESIT | TD 5.16 | |
| 4.14.3.1#8 | If a Full-speed Interrupt OUT transaction times out, then the xHC shall retry the transaction up to Cerr times for the endpoint in the same ESIT | TD 5.16 | |
| 4.14.3.1#9 | If a High-speed Interrupt OUT transaction times out, then the xHC shall retry the transaction up to Cerr times for the endpoint in the same ESIT | TD 5.16 | |
| 4.14.3.1#10 | For HS or FS endpoints, if the maximum number of transactions per microframe has been reached, the xHC shall retry the failed transaction at the next ESIT for the endpoint | TD 5.17 | |
| 4.14.3.1#11 | For HS or FS endpoints, the xHC shall schedule the maximum number of transactions to an endpoint per microframe if the TD Transfer Size is greater than or equal to the Max ESIT Payload | TD 5.17 | |
| 4.14.3.1#12 | For HS or FS endpoints, if the xHC retries Cerr times in a row without success, the xHC shall halt the endpoint | TD 5.16 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.14.3.1#13 | The xHC shall not transfer more than 3 DPs in an ESIT for a SuperSpeed Endpoint | TD 5.17 | |
| **Subsection reference: 4.14.4 Asynchronous Transfer Ring Scheduling** | | | |
| 4.14.4#1 | The xHC shall concatenate buffers referenced by TRBs in a TD. | Tested by multiple TDs | |
| **Subsection reference: 4.15 Suspend-Resume** | | | |
| 4.15#1 | The xHC shall not automatically transition a root hub port from the Resume or U3 state to the U0 state during a transition to D0. | TBD | |
| **Subsection reference: 4.15.1 Port Suspend** | | | |
| 4.15.1#1 | The xHC shall ignore writes to the PLS field when a port is not enabled. | 0 | |
| 4.15.1#2 | The xHC shall ignore writes to the PLS field with the LWS flag set to 0. | 0 <br><br> 0 | |
| 4.15.1#3 | U3 Entry Capability support (i.e. U3C = '1') shall be mandatory for all xHCI 1.1 and xHC 1.2 compliant xHCs. | TD 1.02 | |
| **Subsection reference: 4.15.2 Port Resume** | | | |
| 4.15.2#1 | The xHC shall set the PLS field to Resume in the PORTSC register when it detects a resume event while the port is in the U3 state. | 0 <br><br> 0 | |
| 4.15.2#2 | The xHC shall set the PLC flag to 1 in the PORTSC register when it detects a resume event while the port is in the U3 state. | 0 <br><br> 0 | |
| 4.15.2#3 | The xHC shall set the PLS field to U0 in the PORTSC register when resume signaling has completed. | 0 <br><br> 0 | |
| 4.15.2#4 | The xHC shall set the PLC flag to 1 in the PORTSC register when resume signaling has completed. | 0 <br><br> 0 | |
| 4.15.2#5 | The xHC shall generate a Port Status Change Event when the PLC flag transitions from 0 to 1. | 0 <br><br> 0 | |
| **Subsection reference: 4.15.2.1 Resume Event/Actions** | | | |
| 4.15.2.1#1 | The xHC shall transition the Port to the U0 state when software sets the PLS field to U0 and the LWS flag to 1 in the PORTSC register. | 0 | |
| 4.15.2.1#2 | When the xHC detects a resume event from a SS port while the port is in the U3 state, the xHC shall not respond with LFPS to the device until software writes a '0' to the PLS field. | TD 5.27 | |
| **Subsection reference: 4.16.2 Bandwidth Domains** | | | |
| 4.16.2#1 | The xHC shall reject a Configure Endpoint Command with a Secondary Bandwidth Error if the configuration would have exceeded the Total Available Bandwidth of the Secondary Bandwidth Domain | 0 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.16.2#2 | If the xHC does not support reporting Secondary Bandwidth Domain information, the xHC shall set the SBD flag in the HCCPARAMS to '0'. | 0 | |
| 4.16.2#3 | If the xHC set the SBD flag in the HCCPARAMS to '1', then it shall support reporting Secondary Bandwidth Domain information. | 0 | |
| 4.16.2#4 | When evaluating a Configure Endpoint Command, the xHC shall check the upstream High-speed Bandwidth Domain of a hub first. If there is enough bandwidth available in the primary Bandwidth Domain then the xHC shall check the Secondary Bandwidth Domain of the hub. | | |
| **Subsection reference: 4.17 Interrupters** | | | |
| 4.17#1 | The xHC shall set the MSI Message Capable field in the MSI Message Control register to a value equal to or less than MaxIntrs if the xHC has a MSI Capability. | 0 | |
| 4.17#2 | The xHC shall assert an interrupt if it is enabled and its associated Event Ring contains Event TRBs that require an interrupt. | TD 2.02 | |
| 4.17#3 | The xHC shall assert an interrupt if it is enabled and its associated Event Ring is not empty. | Implicit | |
| **Subsection reference: 4.17.2 Interrupt Moderation** | | | |
| 4.17.2#1 | When the xHC detects that an Interrupter is enabled, the IPE is '1', and the Event Handler is not busy (EHB = '0'), it shall set the EHB flag in the ERDP to '1' | 0 | |
| **Subsection reference: 4.17.5 Interrupt Blocking** | | | |
| 4.17.5#1 | Any TRB type that does not define a *BEI* flag always generates *Non-blocking Events*. | 0 | |
| 4.17.5#2 | If an error is detected which generates an event while processing a TRB with *BEI* = '1', then *BEI* shall be ignored and the event generated by the TRB shall be a *Non-blocking Event*. | 0 | |
| 4.17.5#3 | Any Transfer Event TRB that is not associated with a Transfer or Event Data TRB shall be a *Non-blocking Event*. | 0 | |
| **Subsection reference: 4.19 Root Hub** | | | |
| **Subsection reference: 4.19.1 Root Hub Port State Machines** | | | |
| **Subsection reference: 4.19.1.1 USB2 Root Hub Port** | | | |
| **Subsection reference: 4.19.1.1.1 Powered-off** | | | |
| 4.19.1.1.1#1 | The xHC shall transition the USB 2 root hub port to the Powered-off state on a write to the PORTSC register with the PP flag set to 0. | 0 | |
| 4.19.1.1.1#2 | The xHC shall transition the USB2 root hub port from the Powered-off state to the Disconnected state on a write to the PORTSC register with the PP flag set to 1. | 0 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.19.1.1.1#3 | The xHC shall transition the USB2 root hub port from the Powered-off state to the Test Mode state on a write to the USB2 PORTPMSC register with TestMode greater than 0. | HSETT | |
| **Subsection reference: 4.19.1.1.2 Disconnected** | | | |
| 4.19.1.1.2#1 | The xHC shall transition all root-hub ports to the Disconnected state on HCRST. | TD 6.02 | |
| 4.19.1.1.2#2 | The xHC shall set the PLS field to RxDetect when it detects disconnect of a device. | 0 <br><br> 0 | |
| **Subsection reference: 4.19.1.1.3 Disabled** | | | |
| 4.19.1.1.3#1 | The xHC shall transition a port to the Reset state on a write to the PORTSC register with the PR flag set to 1. | TD 3.01 <br><br> TD 3.02 | |
| **Subsection reference: 4.19.1.1.6 Enabled** | | | |
| 4.19.1.1.6#1 | The xHC shall transition the USB2 root-hub port to the Disabled state when software sets the PED flag to 1 in the PORTSC register. | 0 | |
| 4.19.1.1.6#2 | The xHC shall transition the USB2 root hub port to the Disconnected state when a disconnect condition is detected while the port is in the Enabled state. | 0 | |
| **Subsection reference: 4.19.1.1.12 U3** | | | |
| 4.19.1.1.12#1 | The xHC shall initiate resume signaling to the device when software writes the PORTSC register with the PLS field set to Resume and the LWS flag set to 1. | TD 3.08 | |
| **Subsection reference: 4.19.1.2 USB3 Root Hub Port** | | | |
| **Subsection reference: 4.19.1.2.1 Disabled** | | | |
| 4.19.1.2.1#1 | The xHC shall transition the USB3 root hub port from any state except Powered-off to the Disabled state on a write to the PORTSC register with the PED flag set to 1. | 0 | |
| 4.19.1.2.1#2 | The xHC shall transition the USB3 root hub port from the Disabled to the Disconnected state on a write to the PORTSC register with the PLS field set to RxDetect and the LWS flag set to 1. | 0 | |
| 4.19.1.2.1#3 | The xHC shall transition the USB3 root hub port from the Disabled to the Powered-off state on a write to the PORTSC register with the PP flag cleared to 0. | 0 | |
| **Subsection reference: 4.19.1.2.2 Powered-off** | | | |
| 4.19.1.2.2#1 | The xHC shall transition the USB3 root hub port from the Powered-off state to the Disconnected state on a write to the PORTSC register with the PP flag set to 1. | 0 | |
| **Subsection reference: 4.19.1.2.3 Disconnected** | | | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.19.1.2.3#1 | The xHC shall transition the USB3 root hub port to the Disconnected state, except from a Powered-off or Disabled State, on detecting a device disconnect. | N/A | |
| 4.19.1.2.3#2 | The xHC shall transition the port to the polling state on a device connect detect. | Implied by transition to U0 | |
| 4.19.1.2.3#3 | If a USB3 port transitioned to the Disconnected state due to the assertion of HCRST by software, then the xHC shall issue a Hot or Warm Reset when its LTSSM enters the Rx.Detect state or after a receiver detection in the Rx.Detect state. | 0 | |
| 4.19.1.2.3#4 | The assertion of HCRST = '1' shall cause the port to remain in the Disconnected state. | N/A | |
| **Subsection reference: 4.19.1.2.4 Polling** | | | |
| 4.19.1.2.4#1 | The xHC shall exchange Port Capabilities and Port Configuration LMPs while USB3 root-hub port is in the CfgExcg substate. | Link Tests TD 7.17 | |
| **Subsection reference: 4.19.1.2.4.1 Training** | | | |
| 4.19.1.2.4.1#1 | The CTC flag in the HCCPARAMS2 register shall be set to 1 for all xHCI 1.1 compliant xHCs. | TD 1.02 | |
| **Subsection reference: 4.19.1.2.4.3.2 DbC Disabled** | | | |
| 4.19.1.2.4.3.2#1 | On a write to the DCCTRL register with the DCE field set to '0' or a Disconnect Detect, while in the DbC Disabled state, the xHC shall transition the port to the Disabled state | DbC TBD | |
| **Subsection reference: 4.19.1.2.5 Reset** | | | |
| 4.19.1.2.5#1 | If a port transitions to the Reset state due to the assertion of HCRST by software, then a Hot or Warm Reset shall be issued by the port when its LTSSM enters the Rx.Detect state. | DbC TBD | |
| **Subsection reference: 4.19.1.2.9 Enabled** | | | |
| 4.19.1.2.9#1 | The xHC shall transition a USB3 port to the Reset state on a write of the PORTSC register with the WPR flag set to 1. | TD 3.06 | |
| **Subsection reference: 4.19.1.2.10 U0** | | | |
| 4.19.1.2.10#1 | The xHC shall transition the PLS field to U3 on a write to the PORTSC register with the PLS field set to U3 and the LWS flag set to 1. | 0 | |
| 4.19.1.2.10#2 | The xHC shall transition the USB3 root-hub port to the U1 substate on a U1 timeout. | USB30CV TD 9.24 | |
| 4.19.1.2.10#3 | The xHC shall transition the USB3 root-hub port to the U2 substate on a U2 timeout. | USB30CV TD 9.24 | |
| **Subsection reference: 4.19.1.2.11 U1** | | | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.19.1.2.11#1 | The xHC shall transmit an LAU and transition the USB3 root-hub port to the U1 substate when the LGO_U1 request is accepted. | TD 3.13 | |
| 4.19.1.2.11#2 | The xHC shall transmit an LXU and transition the USB3 root-hub port to the U0 substate when the LGO_U1 request is rejected. | TD 3.13 | |
| 4.19.1.2.11#3 | The xHC shall transition the USB3 root-hub port to the U2 substate when a U2 timeout occurs on a port in the U1 substate. | USB30CV TD 9.24 | |
| **Subsection reference: 4.19.1.2.12 U2** | | | |
| 4.19.1.2.12#1 | The xHC shall transition the USB3 root-hub port to the U2 substate when the LGO_U2 request is accepted and an LAU is received. | USB30CV TD 9.24 | |
| **Subsection reference: 4.19.1.2.14 Recovery** | | | |
| 4.19.1.2.14#1 | The xHC shall transition the USB3 root hub port to the U0 substate when the recovery completes successfully. | USB30CV TD 9.24 | |
| **Subsection reference: 4.19.2 Port Status Change Generation** | | | |
| 4.19.2#1 | The xHC shall only clear a change bit in the PORTSC register when software set it to 1, a Chip Hardware Reset or an HCRST. | N/A | |
| 4.19.2#2 | The xHC shall set the WRC flag only when a warm reset initiated by software's setting the WPR flag to 1 completes. | N/A | |
| 4.19.2#3 | The xHC shall set the PRC flag to 1 when a warm reset completes. | 0 | |
| **Subsection reference: 4.19.3 Connect Status Change Reporting** | | | |
| 4.19.3#1 | The xHC shall set the Port ID field in a Port Status Change Event to the port number of the Root Hub Port that detected the port status change. | 0<br>0 | |
| 4.19.3#2 | The xHC shall set the Completion Code to Success in a Port Status Change Event. | 0<br>0 | |
| **Subsection reference: 4.19.4 Port Power** | | | |
| 4.19.4#1 | The xHC shall transition the port to the Disconnected state when PP is set to 1 and if a device is not connected to a port. | 0 | |
| 4.19.4#2 | When PP = '0', CCS shall be asserted if the PP transitioned to '0' due to an over-current condition only. | N/A | |
| **Subsection reference: 4.19.9 Port Speed** | | | |
| 4.19.9#1 | If a Sublink Speed Device Notification TP is received during an Address Device Command, an SSP capable xHC shall ignore the value of the Input Slot Context:Speed field and update the Output Slot Context:Speed field with the speed reported by the Sublink Speed Device Notification TP before, it completes the Address Device Command. | | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 4.19.9#2 | An SSP capable xHC shall report the actual connect speed of a device attached to a Root Hub port in the PORTSC:Port Speed field. | | |
| 4.19.9#3 | A controller must do the SSP scheduling correctly, even if it is running under a legacy driver. | | |
| **Subsection reference: 4.19.5.1 Warm Reset** | | | |
| 4.15.5.1#1 | If a '1' to '0' transition of PR was due to a software initiated Warm Reset, or a Hot Reset that transitioned to a Warm Reset because of errors, the WRC flag and the PRC flag shall be set to '1'. | 0 | |
| **Subsection reference: 4.23 Power Management** | | | |
| **Subsection reference: 4.23.1 Power Wells** | | | |
| 4.23.1#1 | The xHC shall initialize the memory-space registers in the Aux power well to their default values on initial power-up of the Aux power well. | N/A | |
| 4.23.1#2 | The xHC shall initialize the memory-space registers in the Aux power well to their default values when a value of '1' is set in HCRST. | TD 6.02 | |
| 4.23.1#3 | The xHC shall initialize the memory-space registers in the Core power well to their default values when a value of '1' is set in HCRST | TD 6.02 | |
| 4.23.1#4 | The xHC shall initialize the memory-space registers in the Core power well to their default values on a transition from the PCI PM D3hot state to the D0 state. | N/A | |
| **Subsection reference: 4.23.2 xHCI Power Management** | | | |
| **Subsection reference: 4.23.5 USB Link Power Management** | | | |
| **Subsection reference: 4.23.5.1 Root Hub Port LPM Support** | | | |
| **Subsection reference: 4.23.5.1.1 USB2 LPM Support** | | | |
| 4.23.5.1.1#1 | The xHC shall set the L1S field to Success in the PORTPMSC register when software sets the PLS field to U2 in the PORTSC register of a USB2 port. | USB30CV TD 9.21 | |
| 4.23.5.1.1#2 | The xHC shall set the PLS field to U2 in the PORTSC register when it receives an ACK from a device while attempting to transition the link to the L1 state. | USB30CV TD 9.21 | |
| 4.23.5.1.1#3 | The xHC shall set the L1S field to Success in the USB2 PORTPMSC register when it receives an ACK from a device while attempting to transition the link to the L1 state. | USB30CV TD 9.21 | |
| **Subsection reference: 4.23.5.1.1.1 Hardware Controllerd LPM** | | | |
| 4.23.5.1.1.1#1 | xHC 1.1 and xHC 1.2 compliant xHCs shall have the HLE bit of the PORTPMSC register set to '1'. | TD 1.06 | |
| 4.23.5.1.1.1#2 | xHC 1.1 and xHC 1.2 compliant xHCs shall have the BLC field of rhe USB 2.0 Protocol Defined Fields set to '1'. | TD 1.05 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| **Subsection reference: 4.24.2.1 Root Hub Port to External Port Assignment** | | | |
| 4.24.2.1#1 | When integrated hubs are implemented, the xHCI Supported Protocol Capability Integrated Hub Implemented (IHI) flag shall be '1' | TD 3.14 | |
| 4.24.2.1#2 | When integrated hubs are not implemented, the xHCI Supported Protocol Capability Integrated Hub Implemented (IHI) flag shall be '0' | TD 3.14 | |
| **Subsection reference: 4.24.2.2 External Port to USB Connector Mapping** | | | |
| 4.24.2.2#1 | If USB2 and USB3 protocol ports share the same over-current detection logic, then an over-current condition shall assert OCA on both ports and transition both ports to the Powered-off state. | TD 3.15 | |
| **Chapter 5 Test Assertions:** | | | |
| **Subsection reference: 5 Register Interface** | | | |
| 5#1 | The xHC shall align the Capability Register Space on a page boundary. | 0 | |
| 5#2 | The xHC shall align the Operational Register Space on a 4-byte boundary. | 0 | |
| 5#3 | The xHC shall align the Runtime Regiter Space on a 32-byte boundary. | 0 | |
| 5#4 | The xHC shall align the Doorbell Array on a 4-byte boundary. | 0 | |
| **Subsection reference: 5.2 PCI Configuration Registers** | | | |
| **Subsection reference: 5.2.2 Class Code Registers** | | | |
| 5.2.2#1 | The Programming Interface field shall be set to 30h in the Class Code Register. | 0 | |
| 5.2.2#2 | The Sub-Class Code field shall be set to 03h in the Class Code Register. | 0 | |
| 5.2.2#3 | The Base Class field shall be set to 0Ch in the Class Code Register. | 0 | |
| **Subsection reference: 5.2.3 Serial Bus Release Number Register (SBRN)** | | | |
| 5.2.3#1 | The SBRN register shall be set to 30h for xHC 1.0. | not tested | |
| 5.2.3#2 | The SBRN register shall be set to 31h for xHC 1.1. | not tested | |
| 5.2.3#3 | The SBRN register shall be set to 32h for xHC 1.2 | not tested | |
| **Subsection reference: 5.2.4 Frame Length Adjustment Register (FLADJ)** | | | |
| 5.2.4#1 | The xHC shall not change the FLADJ register value upon receipt of a Bus Interval Adjustment Message. | TD 1.07 | |
| 5.2.4#2 | The xHC shall not modify the FLADJ register value on a hot reset. | TD 1.07 | |
| 5.2.4#3 | The xHC shall not modify the FLADJ register value on a warm reset. | TD 1.07 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 5.2.4#4 | The xHC shall not modify the FLADJ register value on a LHCRST. | TD 1.07 | |
| **Subsection reference: 5.2.5 PCI Power Management Interface** | | | |
| 5.2.5#1 | The xHC shall implement a PCI Power Management Capability in the PCI Configuration Registers. | 0 | |
| **Subsection reference: 5.2.6 Message Signaled Interrupts (MSI & MSI-X) Capability** | | | |
| 5.2.6#1 | The xHC shall implement an MSI and/or MSI-X Capabilities in the PCI Configuration Registers. | 0 | |
| **Subsection reference: 5.2.6.2 MSI-X configuration** | | | |
| 5.2.6.2#1 | The xHC shall not have more than one MSI-X Capabilities. | 0 | |
| **Subsection reference: 5.2.6.5 Accessing the MSI-X Table and MSI-X PBA** | | | |
| 5.2.6.5#1 | The MSI-X Table and MSI-X PBA shall not overlap each other. | 0 | |
| **Subsection reference: 5.3 Host Controller Capability Registers** | | | |
| **Subsection reference: 5.3.2 Host Controller Interface Version Number (HCIVERSION)** | | | |
| 5.3.2#1 | The xHC shall set the HCIVERSION register to the BCD encoding of the xHCI specification revision number it supports. | 0 | |
| 5.3.2#2 | The current available revision numbers of HCIVERSION register are 0096h, 0100h, 0110h and 0120h. | TD 1.02 | |
| **Subsection reference: 5.3.3 Structural Parameters 1 (HCSPARAMS1)** | | | |
| 5.3.3#1 | The MaxSlots field shall be set to a value in the range of 1h to FFh in the HCSPARAMS1 register. | 0 | |
| 5.3.3#2 | The MaxIntrs field shall be set to a value in the range of 1h to 400h in the HCSPARAMS1 register. | 0 | |
| 5.3.3#3 | The MaxPorts field shall be set to a value in the range of 1h to FFh in the HCSPARAMS1 register. | 0 | |
| 5.3.3#4 | The xHC shall set the MaxPorts field in the HCSPARAMS1 register to a value equal to the maximum Port Number value assigned by an xHCI Supported Protocol Capability. | TD 3.01 TD 3.02 | |
| **Subsection reference: 5.3.4 Structural Parameters 2 (HCSPARAMS2)** | | | |
| 5.3.4#1 | The SPR field shall be set to 0 in the HCSPARAMS2 register if the Max Scratchpad Buffers field is 0. | 0 | |
| 5.3.4#2 | The xHC shall set bits 13:25 in the HCSPARAMS2 register to 0 | 0 | |
| 5.3.4#3 | The xHC shall set the IOC Interval field to a value from 0-23 inclusive | 0 | |
| 5.3.4#4 | The xHC version 1.x shall set bits 8:25 in the HCSPARAMS2 register to 0 | 0 | |
| 5.3.4#5 | The xHC version 1.1 or greater shall set bits 8:20 in the HCSPARAMS2 register to 0 | 0 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 5.3.4#6 | For xHC 1.1 or greater compliant xHCs the SPR field shall be set to 0 in the HCSPARAMS2 register if the Max Scratchpad Bufs (Hi, Lo) fields are 0. | TD 1.02 | |
| **Subsection reference: 5.3.5 Structural Parameters 3 (HCSPARAMS3)** | | | |
| 5.3.5#1 | The U1 Device Exit Latency field shall be set to a value in the range of 00h to 0Ah in the HCSPARAMS3 register. | 0 | |
| 5.3.5#2 | The U2 Device Exit Latency field shall be set to a value in the range of 0000h to 07FFh in the HCSPARAMS3 register. | 0 | |
| **Subsection reference: 5.3.6 Capability Parameters (HCCPARAMS)** | | | |
| 5.3.6#1 | The MaxPSASize field shall be set to a value in the range of 1 to 15 in the HCCPARAMS register. | 0 | |
| 5.3.6#2 | The version 1.0 xHC shall set bits 10:11 of the HCCPARAMS register to 0 | TD 1.02 | |
| 5.3.6#3 | If the xHC has FSE field set in the HCCPARAMS, then xHC shall generate Stopped Transfer Event when a Transfer Ring stops between TDs. | TD 4.15 | |
| 5.3.6#4 | If the xHC does not generate Stopped Transfer Events when a Transfer Ring stops between TDs, then the xHC shall set the FSE field in the HCCPARAMS register to '0'. | TD 4.15 | |
| 5.3.6#5 | If the xHC is not capable of reporting Secondary Bandwidth Domain information, the xHC shall set the SBD field of the HCCPARAMS register to '0'. | TD 4.14 | |
| 5.3.6#6 | If the xHC sets the SBD field of the HCCPARAMS register to '1', it shall be capable of reporting Secondary Bandwidth Domain information. | TD 4.14 | |
| 5.3.6#7 | The version 1.0 xHC shall set bits 9:11 of the HCCPARAMS register to 0 | TD 1.02 | |
| 5.3.6#8 | If bit 8 (PAE) of HCCPARAMS is set, the xHC shall parse all Event Data TRBs while advancing to the next TD after a Short Packet. | TD 2.21 | |
| 5.3.6#9 | If bit 8 (PAE) of HCCPARAMS is clear, the xHC shall only parse the first Event Data after the Short Packet condition is detected. | TD 2.21 | |
| 5.3.6#10 | If bit 9 (SPC) of HCCPARAMS is set, the xHC must be capable of generating a Stopped – Short Packet Completion Code. | TD 2.22 | |
| 5.3.6#11 | If bit 10 (SEC) of HCCPARAMS is set, the xHC shall support a Stopped EDLTA field. | TD 2.22 | |
| 5.3.6#12 | If bit 11 (CFC) of HCCPARAMS is set, the xHC shall be capable of matching Frame ID of consecutive Frame IDs. | N/A | |
| 5.3.6#13 | For xHC 1.1 and xHC 1.2 compliant xHCs Stopped EDTLA Capability support (i.e. SEC = '1') in the HCCPARAMS shall be mandatory. | TD 1.02 | |
| **Subsection reference: 5.3.7 Doorbell Offset (DBOFF)** | | | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 5.3.7#1 | Bits 1:0 shall be set to 0 in the DBOFF register. | 0 | |
| **Subsection reference: 5.3.8 Runtime Register Space Offset (RTSOFF)** | | | |
| 5.3.8#1 | Bits 4:0 shall be set to 0 in the RTSOFF register. | 0 | |
| **Subsection reference: 5.3.9 Capability Parameters 2 (HCCPARAMS2)** | | | |
| 5.3.9#1 | xHC version 1.1 shall contain an HCCPARAMS2 register. | TD 1.02 | |
| 5.3.9#2 | If bit 0 (U3C) of HCCPARAMS2 is set, the xHC shall support Suspend Complete notification. | TD 2.23 | |
| 5.3.9#3 | If bit 1 (CMC) of HCCPARAMS2 is set, the xHC shall support generating Max Exit Latency Too Large Capability Error for a Configure Endpoint Command. | N/A | |
| 5.3.9#4 | If bit 2 (FSC) of HCCPARAMS2 is set, the xHC shall save any cached slot, endpoint, sream or other context info to memory when a Save State operation is initiated. | Implicit in interop | |
| 5.3.9#5 | Bits 10-31 of HCCPARAMS2 are reserved and shall be set to 0. | TD 1.02 | |
| 5.3.9#6 | The FSC field of HCCPARAMS2 shall be set to 1 for xHCI 1.1 compliant xHCs. | TD 1.02 | |
| 5.3.9#7 | The CIC field of HCCPARAMS2 shall be set to 1 for xHCI 1.1 compliant xHCs. | TD 1.02 | |
| 5.3.9#8 | The GSC field of HCCPARAMS2 shall be set to 1 if the xHC supports Get and Set Extended Property. | | |
| 5.3.9#9 | The VTC field of the HCCPARAMS2 shall be set to 1 if the xHC supports the VTIO Capability. | | |
| **Subsection reference: 5.4 Host Controller Operational Registers** | | | |
| **Subsection reference: 5.4.1 USB Command Register (USBCMD)** | | | |
| 5.4.1#1 | The xHC shall clear the RS flag of the USBCMD register to 0 on an HCRST. | 0 | |
| 5.4.1#2 | The xHC shall clear the RS flag of the USBCMD register to 0 on an LHCRST. | 0 | |
| 5.4.1#3 | The xHC shall clear the HCRST flag to 0 in the USBCMD register after completing the reset process. | 0 | |
| 5.4.1#4 | The xHC shall clear the HCRST flag of the USBCMD register to 0 on an LHCRST. | 0 | |
| 5.4.1#5 | The xHC shall not generate an interrupt when the INTE flag is 0 in the USBCMD register. | 0 | |
| 5.4.1#6 | The xHC shall clear the INTE flag of the USBCMD register to 0 on an HCRST. | 0 | |
| 5.4.1#7 | The xHC shall clear the INTE flag of the USBCMD register to 0 on an LHCRST. | 0 | |
| 5.4.1#8 | The xHC shall always return 0 when the CSS flag of the USBCMD register is read. | 0 | |
| 5.4.1#9 | The xHC shall always return 0 when the CRS flag of the USBCMD register is read. | 0 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 5.4.1#10 | The xHC shall generate an MFINDEX Wrap Event every time the MFINDEX register transitions from 3FFFh to 0 while the EWE flag is 1 in the USBCMD register. | 0 | |
| 5.4.1#11 | The xHC shall not generate MFINDEX Wrap Events while the EWE flag is 0 in the USBCMD register. | 0 | |
| 5.4.1#12 | The xHC shall clear the EWE flag of the USBCMD register to 0 on an HCRST. | 0 | |
| 5.4.1#13 | The xHC shall clear the EWE flag of the USBCMD register to 0 on an LHCRST. | 0 | |
| 5.4.1#14 | The xHC shall clear the EU3S flag of the USBCMD register to 0 on an HCRST. | 0 | |
| 5.4.1#15 | The xHC shall clear the EU3S flag of the USBCMD register to 0 on an LHCRST. | 0 | |
| 5.4.1#16 | The xHC shall clear the LHCRST flag of the USBCMD register to 0 on an HCRST. | 0 | |
| 5.4.1#17 | The xHC shall clear the LHCRST flag to 0 in the USBCMD register after completing the Light Host Controller Reset process. | 0 | |
| 5.4.1#18 | The xHC shall initiate a Hot or Warm Reset on USB3 Root Hub downstream ports when the HCRST bit in the USBCMD register gets set to '1'. | TD 6.02 | |
| 5.4.1#19 | If the CSS field of the USBCMD register is set to '0' by software, or the HCH field is '0', then the xHC shall not perform a Save State operation. | N/A | |
| 5.4.1#20 | If the R/S = '1', or HCH = '0', or when CRS is cleared to '0', the xHC shall not perform a Restore State operation. | Interop | |
| 5.4.1#21 | If bit 13 (CME) of USBCMD is set, the xHC may return a Max Exit Latency Too Large Capability Error for a Configure Endpoint Command. | N/A | |
| 5.4.1#22 | If bit 15 (TSC_EN) of USBCMD is 1, TRBSts field in the TRB updated to indicate if it is the last transfer TRB in the TD. | | |
| 5.4.1#23 | When bit 16 (VTIOE) of USBCMD is set to 1, XHCI HW will enable its VTIO capability and begin to use the information provided via that VTIO Registers to determine its DMA-ID. | | |
| **Subsection reference: 5.4.2 USB Status Register (USBSTS)** | | | |
| 5.4.2#1 | The xHC shall set the HCH flag to 1 in the USBSTS register after the RS flag is cleared to 0 in the USBCMD register. | 0 | |
| 5.4.2#2 | The xHC shall set the HCH flag to 0 in the USBSTS register after software sets the RS flag to 1 in the USBCMD register. | 0 | |
| 5.4.2#3 | Software writes to the HCH flag in the USBSTS register shall have no effect. | 0 | |
| 5.4.2#4 | The xHC shall set the HCH flag to 1 in the USBSTS register on an HCRST. | 0 | |
| 5.4.2#5 | The xHC shall set the HCH flag to 1 in the USBSTS register on an LHCRST. | 0 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 5.4.2#6 | The xHC shall clear the HSE flag to 0 in the USBSTS register on an HCRST. | 0 | |
| 5.4.2#7 | The xHC shall clear the HSE flag to 0 in the USBSTS register on an LHCRST. | 0 | |
| 5.4.2#8 | The xHC shall set the EINT flag to 1 in the USBSTS register when the IP flags of any Interrupter transitions from 0 to 1. | 0 | |
| 5.4.2#9 | A write of 0 to the EINT flag in the USBSTS register by software shall have no effect. | 0 | |
| 5.4.2#10 | A write of 1 to the EINT flag in the USBSTS register by software shall result in the value of the EINT flag to be cleared to 0. | 0 | |
| 5.4.2#11 | The xHC shall clear the EINT flag to 0 in the USBSTS register on an HCRST. | 0 | |
| 5.4.2#12 | The xHC shall clear the EINT flag to 0 in the USBSTS register on an LHCRST. | 0 | |
| 5.4.2#13 | The xHC shall set the PCD flag in the USBSTS register to 1 when any port has a change bit transition from a 0 to a 1. | 0 0 | |
| 5.4.2#14 | A write of 0 to the PCD flag in the USBSTS register by software shall have no effect. | 0 | |
| 5.4.2#15 | A write of 1 to the PCD flag in the USBSTS register by software shall result in the value of the PCD flag to being cleared to 0. | 0 0 | |
| 5.4.2#16 | The xHC shall clear the PCD flag to 0 in the USBSTS register on an HCRST. | 0 | |
| 5.4.2#17 | The xHC shall clear the PCD flag to 0 in the USBSTS register on an LHCRST. | 0 | |
| 5.4.2#18 | The xHC shall clear the SSS flag to 0 in the USBSTS register after a Save State operation completes. | 0 | |
| 5.4.2#19 | The xHC shall clear the SSS flag to 0 in the USBSTS register on an HCRST. | 0 | |
| 5.4.2#20 | The xHC shall clear the SSS flag of the USBSTS register to 0 on an LHCRST. | 0 | |
| 5.4.2#21 | A write to the SSS flag in the USBSTS register shall have no effect. | 0 | |
| 5.4.2#22 | The xHC shall clear the RSS flag to 0 in the USBSTS register after a Restore State operation completes. | 0 | |
| 5.4.2#23 | The xHC shall clear the RSS flag to 0 in the USBSTS register on an HCRST. | 0 | |
| 5.4.2#24 | The xHC shall clear the RSS flag to 0 in the USBSTS register on an LHCRST. | 0 | |
| 5.4.2#25 | A write to the RSS flag in the USBSTS register shall have no effect. | 0 | |
| 5.4.2#26 | The xHC shall clear the SRE flag to 0 in the USBSTS register on an HCRST. | 0 | |
| 5.4.2#27 | The xHC shall clear the SRE flag to 0 in the USBSTS register on an LHCRST. | 0 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 5.4.2#28 | The xHC shall clear the CNR flag to 0 in the USBSTS register on an HCRST. | 0 | |
| 5.4.2#29 | A write to the CNR flag in the USBSTS register shall have no effect. | 0 | |
| 5.4.2#30 | The xHC shall clear the HCE flag to 0 in the USBSTS register on an HCRST. | 0 | |
| 5.4.2#31 | The xHC shall clear the HCE flag to 0 in the USBSTS register on an LHCRST. | 0 | |
| 5.4.2#32 | A write to the HCE flag in the USBSTS register shall have no effect. | 0 | |
| 5.4.2#33 | If the HCH bit of the USBSTS register is '1', then the xHC shall not generate SOFs, microSOFs, or ITPs. | TD 6.01 | |
| 5.4.2#34 | If the HCH bit of the USBSTS register is '1', then the xHC shall drop any received Transaction Packets. | | |
| **Subsection reference: 5.4.3 Page Size Register (PAGESIZE)** | | | |
| 5.4.3#1 | Writes to the PAGESIZE register by software shall have no effect. | 0 | |
| **Subsection reference: 5.4.4 Device Notification Control Register (DNCTRL)** | | | |
| 5.4.4#1 | The xHC shall clear the Notification Enable field to 0 in the DNCTRL register on an HCRST. | 0 | |
| 5.4.4#2 | The xHC shall clear the Notification Enable field in the DNCTRL register on an LHCRST. | 0 | |
| 5.4.4#3 | The xHC shall generate a Device Notification Event when a Device Notification is received and the corresponding Notification Enable bit is set to 1 in the DNCTRL register. | 0 | |
| **Subsection reference: 5.4.5 Command Ring Control Register (CRCR)** | | | |
| 5.4.5#1 | The xHC shall ignore writes to the RCS flag in the CRCR register when the CRR flag is 1. | 0 | |
| 5.4.5#2 | The xHC shall use the updated RCS value on the next doorbell if software writes to the CRCR register while the command ring is stopped. | 0 | |
| 5.4.5#3 | The xHC shall use the current RCS value on the next doorbell if software does not write to the CRCR register while the command ring is stopped. | 0 | |
| 5.4.5#4 | Reads of the RCS flag in the CRCR register shall always return 0. | 0 | |
| 5.4.5#5 | The xHC shall clear the CRR flag to 0 in the CRCR register on an HCRST. | 0 | |
| 5.4.5#6 | The xHC shall clear the CRR flag to 0 in the CRCR register on an LHCRST. | 0 | |
| 5.4.5#7 | The xHC shall ignore writes to the CS flag in the CRCR register when the CRR flag in the CRCR register is 0. | 0 | |
| 5.4.5#8 | Reads of the CS flag in the CRCR shall always return 0. | 0 | |
| 5.4.5#9 | Writes of 0 to the CS flag in the CRCR register shall have no effect. | 0 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 5.4.5#10 | The xHC shall ignore writes to the CA flag in the CRCR register when the CRR flag in the CRCR register is 0. | 0 | |
| 5.4.5#11 | Reads of the CA flag in the CRCR shall always return 0. | 0 | |
| 5.4.5#12 | Writes of 0 to the CA flag in the CRCR register shall have no effect. | 0 | |
| 5.4.5#13 | The xHC shall set the CRR flag to 1 in the CRCR register when software rings the Host Controller Doorbell. | 0 | |
| 5.4.5#14 | The xHC shall clear the CRR flag to 0 in the CRCR register when the Command Ring is stopped. | 0<br>0 | |
| 5.4.5#15 | The xHC shall ignore writes to the Command Ring Pointer when the CRR flag is 1 in the CRCR register. | 0 | |
| 5.4.5#16 | Reads of the Command Ring Pointer in the CRCR register shall always return 0. | 0 | |
| 5.4.5#17 | Writes to the CRR flag of the CRCR register shall have no effect. | 0 | |
| **Subsection reference: 5.4.6 Device Context Base Address Array Pointer Register (DCBAAP)** | | | |
| 5.4.6#1 | The xHC shall clear the DCBAAP register to 0 on an HCRST. | 0 | |
| 5.4.6#2 | The xHC shall clear the DCBAAP register to 0 on an LHCRST. | 0 | |
| **Subsection reference: 5.4.7 Configure Register (CONFIG)** | | | |
| 5.4.7#1 | The xHC shall clear the MaxSlotsEn field to 0 in the CONFIG register on an HCRST. | 0 | |
| 5.4.7#2 | The xHC shall clear the MaxSlotsEn field to 0 in the CONFIG register on an LHCRST. | 0 | |
| 5.4.7#3 | The xHC shall not respond to a Doorbell Register reference for a disabled Device Slot. | TD 4.13 | |
| 5.4.7#4 | When bit 8 (U3E) of CONFIG register is set, the xHC shall assert the PLC flag when a Root Hub port transitions to the U3 State. | TD 2.23 | |
| **Subsection reference: 5.4.8 Port Status and Control Register (PORTSC)** | | | |
| 5.4.8#1 | The xHC shall not generate Port Status Change Events if the HCH flag is 1 in the USBSTS register. | TD 1.04 | |
| 5.4.8#2 | The xHC shall clear the CCS flag to 0 in the PORTSC register if the PP flag is 0. | 0 | |
| 5.4.8#3 | Software writes to the CCS flag of the PORTSC register shall have no effect. | 0 | |
| 5.4.8#4 | The xHC shall clear the CCS flag of the PORTSC register to 0 on an HCRST. | N/A | |
| 5.4.8#5 | The xHC shall clear the PED flag to 0 in the PORTSC register if the PP flag is 0, | 0 | |
| 5.4.8#6 | The xHC shall set the PED flag to 0 in the PORTSC register after a disconnect event. | 0<br>0 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 5.4.8#7 | Software writes of 0 to the PED flag of the PORTSC register shall have no effect. | 0 | |
| 5.4.8#8 | Software writes of 1 to the PED flag of the PORTSC register shall result in the xHC setting it to 0. | 0<br><br>0 | |
| 5.4.8#9 | The xHC shall clear the PED flag to 0 in the PORTSC register on an HCRST. | 0 | |
| 5.4.8#10 | Software writes to the OCA flag of the PORTSC register shall have no effect. | TD 3.12 | |
| 5.4.8#11 | The xHC shall clear the OCA flag of the PORTSC register to 0 on an HCRST. | TD 3.12 | |
| 5.4.8#12 | The xHC shall clear the PR flag of the PORTSC register to 0 on an HCRST. | 0 | |
| 5.4.8#13 | Software writes of a value of 5 to the PLS field of the PORTSC register for a USB3 protocol port shall have no effect unless the port is in the Disabled state. | TD 3.12 | |
| 5.4.8#14 | Software writes of a value of 15 to the PLS field of the PORTSC register for a USB2 protocol port shall have no effect unless the port is in the U3 state. | 0 | |
| 5.4.8#15 | The xHC shall transition the link associated with the port to U0 on a software initiated state change of the PLS field of the PORTSC register associated with a USB3 protocol port to 0. | USB30CV TD 9.24 | |
| 5.4.8#16 | The xHC shall transition the link associated with the port to U3 on a software initiated state change of the PLS field of the PORTSC register associated with a USB3 protocol port to 3. | USB30CV TD 9.24 | |
| 5.4.8#17 | The xHC shall not report device attach events for a port while the PP flag in the corresponding PORTSC register is 0. | 0 | |
| 5.4.8#18 | The xHC shall not report device detach events for a port while the PP flag in the corresponding PORTSC register is 0. | 0 | |
| 5.4.8#19 | The xHC shall not modify the PP flag of the PORTSC register on a hot reset. | TD 3.16 | |
| 5.4.8#20 | The xHC shall not modify the PP flag of the PORTSC register on a warm reset. | TD 3.06 | |
| 5.4.8#21 | The xHC shall set the PP flag of the PORTSC register to 1 on an HCRST. | 0 | |
| 5.4.8#22 | Software writes to the Port Speed field of the PORTSC register shall have no effect. | TD 3.12 | |
| 5.4.8#23 | The xHC shall set the Port Speed field to 0 in the PORTSC register when the CCS flag is 0. | 0<br><br>0 | |
| 5.4.8#24 | Software writes to the PIC field of the PORTSC register shall have no effect if the PIND flag is 0 in the HCCPARAMS register. | TD 3.12 | |
| 5.4.8#25 | The xHC shall set the PIC field to 0 in the PORTSC register if PP is 0. | TD 3.12 | |
| 5.4.8#26 | The xHC shall not modify the PIC field of the PORTSC register on a hot reset. | TD 3.12 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 5.4.8#27 | The xHC shall not modify the PIC field of the PORTSC register on a warm reset. | TD 3.12 | |
| 5.4.8#28 | The xHC shall clear the PIC field to 0 in the PORTSC register on an HCRST. | TD 3.12 | |
| 5.4.8#29 | Software reads of the LWS flag of the PORTSC register shall return 0. | 0 | |
| 5.4.8#30 | The xHC shall clear the LWS flag of the PORTSC register to 0 on an HCRST. | TD 3.12 | |
| 5.4.8#31 | The xHC shall initialize the CSC flag of the PORTSC register to 0. | TD 3.12 | |
| 5.4.8#32 | Software writes of 0 to the CSC flag of the PORTSC register shall have no effect. | 0 | |
| 5.4.8#33 | Software writes of 1 to the CSC flag of the PORTSC register shall clear it to 0. | 0 <br><br> 0 | |
| 5.4.8#34 | The xHC shall not set the CSC flag of the PORTSC register to 1 if the change was due to s/w setting PP to 0. | 0 | |
| 5.4.8#35 | The xHC shall not modify the CSC flag of the PORTSC register on a hot reset. | TD 3.12 | |
| 5.4.8#36 | The xHC shall not modify the CSC flag of the PORTSC register on a warm reset. | TD 3.12 | |
| 5.4.8#37 | The xHC shall clear the CSC flag of the PORTSC register to 0 on an HCRST. | TD 3.12 | |
| 5.4.8#38 | The xHC shall not set the PEC flag to 1 in the PORTSC register when the port is disabled due to software writes of 0 to the PP flag. | 0 | |
| 5.4.8#39 | The xHC shall clear the PEC flag of the PORTSC register to 0 on an HCRST. | TD 3.12 | |
| 5.4.8#40 | The xHC shall set the WRC flag to 1 in the PORTSC register when a warm reset processing completes. | 0 | |
| 5.4.8#41 | Software writes of a 1 to the WRC flag of the PORTSC register shall clear it to 0. | 0 | |
| 5.4.8#42 | Software writes of 0 to the WRC flag of the PORTSC register shall have no effect. | 0 | |
| 5.4.8#43 | The xHC shall not modify the WRC flag of the PORTSC register on a hot reset. | TD 3.12 | |
| 5.4.8#44 | The xHC shall clear the WRC flag of the PORTSC register to 0 on an HCRST. | TD 3.12 | |
| 5.4.8#45 | The xHC shall clear the OCC flag of the PORTSC register to 0 on an HCRST. | TD 3.12 | |
| 5.4.8#46 | Software writes of 0 to the PRC flag of the PORTSC register shall have no effect. | 0 | |
| 5.4.8#47 | The xHC shall clear the PRC flag of the PORTSC register to 0 on an HCRST. | TD 3.12 | |
| 5.4.8#48 | The xHC shall set the PLC flag of the PORTSC register of a USB2 port to 1 when L1 Resume completes (link transitions from U2 -> U0). | USB3CV TD 9.21 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 5.4.8#49 | The xHC shall not set the PLC flag to 1 in the PORTSC register if the PLS transition was due to software clearing PP to 0. | 0 | |
| 5.4.8#50 | Software writes of 1 to the PLC flag of the PORTSC register shall clear it to 0. | 0<br><br>0<br><br>0<br><br>0 | |
| 5.4.8#51 | Software writes of 0 to the PLC flag of the PORTSC register shall have no effect. | 0 | |
| 5.4.8#52 | The xHC shall clear the PLC flag of the PORTSC register to 0 on an HCRST. | TD 3.12 | |
| 5.4.8#53 | The xHC shall clear the CEC flag of the PORTSC register to 0 on an HCRST. | TD 3.12 | |
| 5.4.8#54 | The xHC shall not generate wake signalling when a device is connected to the corresponding port when the WCE flag of the PORTSC register is 0. | TD 3.12 | |
| 5.4.8#55 | The xHC shall generate resume signalling when a device is connected to the corresponding port when the WCE flag of the PORTSC register is 1, and the xHC is in a low power state. | TD 3.12 | |
| 5.4.8#56 | The xHC shall not modify the WCE flag of the PORTSC register on a hot reset. | TD 3.12 | |
| 5.4.8#57 | The xHC shall not modify the WCE flag of the PORTSC register on a warm reset. | TD 3.12 | |
| 5.4.8#58 | The xHC shall clear the WCE flag of the PORTSC register to 0 on an HCRST. | TD 3.12 | |
| 5.4.8#59 | The xHC shall not generate wake signalling when a device is disconnected from the corresponding port when the WDE flag of the PORTSC register is 0. | TD 3.12 | |
| 5.4.8#60 | The xHC shall generate resume signalling when a device is disconnected from the corresponding port when the WDE flag of the PORTSC register is 1, and the xHC is in a low power state. | TD 3.12 | |
| 5.4.8#61 | The xHC shall not modify the WDE flag of the PORTSC register on a hot reset. | TD 3.12 | |
| 5.4.8#62 | The xHC shall not modify the WDE flag of the PORTSC register on a warm reset. | TD 3.12 | |
| 5.4.8#63 | The xHC shall clear the WDE flag of the PORTSC register to 0 on an HCRST. | TD 3.12 | |
| 5.4.8#64 | The xHC shall not modify the WOE flag of the PORTSC register on a hot reset. | TD 3.12 | |
| 5.4.8#65 | The xHC shall not modify the WOE flag of the PORTSC register on a warm reset. | TD 3.12 | |
| 5.4.8#66 | The xHC shall clear the WOE flag of the PORTSC register to 0 on an HCRST. | TD 3.12 | |
| 5.4.8#67 | Software writes to the DR flag of the PORTSC register shall have no effect. | TD 3.12 | |
| 5.4.8#68 | Software writes of 0 to the WPR flag of the PORTSC register shall have no effect. | TD 3.12 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 5.4.8#69 | The xHC shall set the WPR flag of the PORTSC register to 0 for USB2 ports. | TD 3.12 | |
| 5.4.8#70 | Software reads of the WPR flag of the PORTSC register shall always return 0. | 0 | |
| 5.4.8#71 | The xHC shall clear the WPR flag of the PORTSC register to 0 on an HCRST. | TD 3.12 | |
| 5.4.8#72 | Software writes of 1 to the PRC flag of the PORTSC register shall clear it to 0. | 0<br>0 | |
| 5.4.8#73 | The xHC shall not report that a USB2 port is in the U1 state. | TD 3.12 | |
| 5.4.8#74 | The xHC shall not return a value of 12-14 in the PLS field of the PORTSC register for a USB2 port when read by software. | TD 3.12 | |
| 5.4.8#75 | The xHC shall not set the Port Speed field to a value in the range of 5 to 15 in the PORTSC register. | TD 3.12 | |
| 5.4.8#76 | The xHC shall not set the Port Speed field to 4 in the PORTSC register for a USB2 port. | TD 3.12 | |
| 5.4.8#77 | The xHC shall not set the Port Speed field to a value in the range of 1 to 3 in the PORTSC register for a USB3 port. | TD 3.12 | |
| 5.4.8#78 | The xHC shall set the Port Speed field to 0 in the PORTSC register on a HCRST. | TD 3.12 | |
| 5.4.8#79 | The xHC shall not set the PIC field to 3 in the PORTSC register. | TD 3.12 | |
| 5.4.8#80 | The xHC shall not set the WRC flag to 1 in the PORTSC register for a USB2 port. | TD 3.12 | |
| 5.4.8#81 | If the PPC field in the HCCPARAMS register is '1', then the xHC shall immediately initate a Port Power change when PP is written. | TD 3.03 | |
| 5.4.8#82 | The xHC shall set bits 28:29 of the PORTSC register to '0' | TD 3.12 | |
| **Subsection reference: 5.4.9 Port PM Status and Control Register (PORTPMSC)** | | | |
| **Subsection reference: 5.4.9.1 USB3 Protocol PORTPMSC Definition** | | | |
| 5.4.9.1#1 | The xHC shall set the U1 Timeout field to 0 in the PORTPMSC register when the PR flag is set to 1 in the PORTSC register. | TD 1.06 | |
| 5.4.9.1#2 | The xHC shall clear the U1 Timeout field to 0 in the PORTPMSC register on an HCRST. | TD 1.06 | |
| 5.4.9.1#3 | The xHC shall clear the U2 Timeout field to 0 in the PORTPMSC register on an HCRST. | TD 1.06 | |
| 5.4.9.1#4 | The xHC shall clear the FLA flag to 0 in the PORTPMSC register on an HCRST. | TD 1.06 | |
| 5.4.9.1#5 | The xHC shall set the FLA flag of the PORTPMSC register for a USB3 port to 0 when the PR flag of the PORTSC register transitions to 1. | TD 1.06 | |
| 5.4.9.1#6 | The xHC shall set the FLA flag of the PORTPMSC register for a USB3 port to 0 when the CCS flag of the PORTSC register transitions from 0 to 1. | TD 1.06 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 5.4.9.1#7 | The xHC shall ensure that the FLA flag of the PORTPMSC register for a USB3 port is 0 if the PP flag is 0 in the PORTSC register. | TD 1.06 | |
| 5.4.9.1#8 | When bit 16 (FLA) of PORTPMSC register is set to 1, the port shall generate a Set Link Function LMP with the Force Link PM Accept bit asserted ('1'). | TD 5.30 | |
| 5.4.9.1#9 | When bit 16 (FLA) of PORTPMSC register is cleared to 0, the port shall generate a Set Link Function LMP with the Force Link PM Accept bit de-asserted ('0'). | TD 5.30 | |
| **Subsection reference: 5.4.9.2 USB2 Protocol PORTPMSC Definition** | | | |
| 5.4.9.2#1 | The xHC shall set the L1S field to 1 in the PORTPMSC register when the port successfully transitioned to L1 (ACK). | TD 1.06 | |
| 5.4.9.2#2 | Software writes to the L1S field of the PORTPMSC register shall have no effect. | TD 1.06 | |
| 5.4.9.2#3 | The xHC shall clear the L1S field to 0 in the PORTPMSC register on an HCRST. | TD 1.06 | |
| 5.4.9.2#4 | The xHC shall clear the RWE flag to 0 in the PORTPMSC register on an HCRST. | TD 1.06 | |
| 5.4.9.2#5 | The xHC shall clear the HIRD/BESL field to 0 in the PORTPMSC register on an HCRST. | TD 1.06 | |
| 5.4.9.2#6 | The xHC shall clear the L1 Device Slot field to 0 in the PORTPMSC register on an HCRST. | TD 1.06 | |
| 5.4.9.2#7 | The xHC shall clear the Port Test Control field to 0 in the PORTPMSC register on an HCRST. | TD 1.06 | |
| 5.4.9.2#8 | The xHC shall clear the Port Test Control field to 0 in the PORTPMSC register on an LHCRST. | TD 1.06 | |
| 5.4.9.2#9 | The xHC shall set bits 17:27 of the PORTPMSC register for a USB2 port to '0' (xHC 1.0 only). | TD 1.06 | |
| **Subsection reference: 5.4.10 Port Link Info Register (PORTLI)** | | | |
| **Subsection reference: 5.4.10.1 USB3 Protocol PORTLI Definition** | | | |
| 5.4.10.1#1 | The xHC shall reset the Link Error Count field to 0 in the PORTLI register when the PR flag transitions from 1 to 0. | N/A | |
| 5.4.10.1#2 | The xHC shall reset the Link Error Count field to 0 in the PORTLI register when the CCS flag transitions from 0 to 1. | N/A | |
| 5.4.10.1#3 | The xHC shall clear the Link Error Count field to 0 in the PORTLI register on an HCRST. | TD 1.08 | |
| 5.4.10.1#4 | The xHC shall clear the Link Error Count field to 0 in the PORTLI register on an LHCRST. | TD 1.08 | |
| 5.4.10.1#5 | Software writes of 0 to the Link Error Count field of the PORTLI register shall reset the Link Error Count field to 0. | TD 1.08 | |
| 5.4.10.1#6 | Software writes to the Rx Lane Count field of the PORTLI register shall have no effect | TD 3.17 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 5.4.10.1#7 | Software writes to the Tx Lane Count field of the PORTLI register shall have no effect | TD 3.17 | |
| 5.4.10.1#8 | The Link Error Count field of the PORTLI register will increment by one each time a port transitions from U0 to Recovery to recover an error event. | | |
| 5.4.10.1#9 | The Link Error Count field of the PORTLI register will saturate at max. | | |
| **Subsection reference: 5.4.10.2 USB2 Protocol PORTLI Definition** | | | |
| 5.4.10.2#1 | The xHC shall set the PORTLI register for a USB2 port to 0. | TD 1.08 | |
| **Subsection reference: 5.4.11.1 USB3 Protocol PORTEXSC Definition** | | | |
| **Subsection reference: 5.4.11.1 USB2 Protocol PORTEXSC Definition** | | | |
| **Subsection reference: 5.5 Host Controller Runtime Registers** | | | |
| **Subsection reference: 5.5.1 Microframe Index Register (MFINDEX)** | | | |
| 5.5.1#1 | Software writes to the MFINDEX register shall have no effect. | 0 | |
| 5.5.1#2 | The xHC shall clear the Microframe Index field of the MFINDEX register to 0 on an HCRST. | 0 | |
| 5.5.1#3 | The xHC shall clear the Microframe Index field of the MFINDEX register to 0 on an LHCRST. | 0 | |
| 5.5.1#4 | The xHC shall not increment the MFINDEX register while the RS flag is 0 in the USBCMD register. | TD 1.04 | |
| **Subsection reference: 5.5.2 Interrupter Register Set** | | | |
| **Subsection reference: 5.5.2.1 Interrupter Management Register (IMAN)** | | | |
| 5.5.2.1#1 | The xHC shall clear the IP flag of the IMAN register to 0 on an HCRST. | 0 | |
| 5.5.2.1#2 | The xHC shall clear the IP flag of the IMAN register to 0 on an LHCRST. | 0 | |
| 5.5.2.1#3 | The xHC shall not generate interrupts when the IE flag is 0 in the corresponding IMAN register. | 0 | |
| 5.5.2.1#4 | The xHC shall clear the IE flag to 0 in the IMAN register on an HCRST. | 0 | |
| 5.5.2.1#5 | The xHC shall clear the IE flag to 0 in the IMAN register on an LHCRST. | 0 | |
| **Subsection reference: 5.5.2.2 Interrupter Moderation Register (IMOD)** | | | |
| 5.5.2.2#1 | The xHC shall set the IMODI field of the IMOD register to 4000 on an HCRST. | 0 | |
| 5.5.2.2#2 | The xHC shall set the IMODI field of the IMOD register to 4000 on an LHCRST. | 0 | |
| **Subsection reference: 5.5.2.3 Event Ring Registers** | | | |
| **Subsection reference: 5.5.2.3.1 Event Ring Segment Table Size Register (ERSTSZ)** | | | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 5.5.2.3.1#1 | The xHC shall disable a Secondary Event Ring when software sets the ERSTSZ register to 0. | 0 | |
| 5.5.2.3.1#2 | The xHC shall clear the Event Ring Segment Table Size field of the ERSTSZ register to 0 on an HCRST. | 0 | |
| 5.5.2.3.1#3 | The xHC shall clear the Event Ring Segment Table Size field of the ERSTSZ register to 0 on an LHCRST. | 0 | |
| **Subsection reference: 5.5.2.3.2 Event Ring Segment Table Base Address Register (ERSTBA)** | | | |
| 5.5.2.3.2#1 | The xHC shall set the Event Ring state machine to the Start state when software writes the ERSTBA register. | 0 | |
| 5.5.2.3.2#2 | The xHC shall clear the Event Ring Segment Table Base Address Register field of the ERSTBA register to 0 on an HCRST. | 0 | |
| 5.5.2.3.2#3 | The xHC shall clear the Event Ring Segment Table Base Address Register field of the ERSTBA register to 0 on an LHCRST. | 0 | |
| 5.5.2.3.2#4 | The xHC shall set bits 0:5 of the ERSTBA register to '0' (xHC 1.0 only). | TD 1.03 | |
| 5.5.2.3.2#5 | When s/w writes to bits 6:63 of the ERSTBA register, the xHC shall set the Event Ring state machine to the Start state (xHC 1.0 only). | Implicit | |
| **Subsection reference: 5.5.2.3.3 Event Ring Dequeue Pointer Register (ERDP)** | | | |
| 5.5.2.3.3#1 | The xHC shall clear the DESI field of the ERDP register to 0 on an HCRST. | 0 | |
| 5.5.2.3.3#2 | The xHC shall clear the DESI field of the ERDP register to 0 on an LHCRST. | 0 | |
| 5.5.2.3.3#3 | The xHC shall set the EHB flag in the ERDP register to 1 when the IP flag in the IMAN register is set to 1. | 0 | |
| 5.5.2.3.3#4 | Software writes of 1 to the EHB flag in the ERDP register shall clear this field to 0. | 0 | |
| 5.5.2.3.3#5 | The xHC shall clear the EHB flag of the ERDP register to 0 on an HCRST. | 0 | |
| 5.5.2.3.3#6 | The xHC shall clear the EHB flag of the ERDP register to 0 on an LHCRST. | 0 | |
| 5.5.2.3.3#7 | The xHC shall clear the Event Ring Dequeue Pointer field of the ERDP register to 0 on an HCRST. | 0 | |
| 5.5.2.3.3#8 | The xHC shall clear the Event Ring Dequeue Pointer field of the ERDP register to 0 on an LHCRST. | 0 | |
| **Subsection reference: 5.6 Doorbell Registers** | | | |
| 5.6#1 | The xHC shall ignore doorbell references for an endpoint with Max PStreams set to 0 if the DB Stream ID field of the DB register is non-zero. | TD 2.14 | |
| 5.6#2 | The xHC shall return a 0 when the DB Stream ID field of the Doorbell register is read. | TD 2.14 | |
| **Chapter 6 Test Assertions:** | | | |
| 6#1 | The xHC shall preserve the values of reserved and read-only fields on all data structure writes. | N/A | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| **Subsection reference: 6.2 Contexts** | | | |
| **Subsection reference: 6.2.2 Slot Context** | | | |
| 6.2.2#1 | xHC shall set the Route String field of the Output Slot Context to the value in the Input Slot Context of an Address Device Command. | 0 | |
| ~~6.2.2#2~~ | ~~xHC shall set the Speed field of the Output Slot Context to the value in the Input Slot Context of an Address Device Command.~~ | ~~0~~ | |
| 6.2.2#3 | xHC shall set the MTT flag of the Output Slot Context to the value in the Input Slot Context of an Address Device Command. | 0 | |
| 6.2.2#4 | xHC shall set the Hub flag of the Output Slot Context to the value in the Input Slot Context of an Evaluate Context Command or a Configure Endpoint Command. | 0 | |
| 6.2.2#5 | xHC shall set the Max Exit Latency field of an Output Slot Context to the value in the Input Slot Context of an Evaluate Context Command. | TD 3.11 | |
| 6.2.2#6 | xHC shall set the Root Hub Port Number field of an Output Slot Context to the value in the Input Slot Context of an Address Device Command. | 0 | |
| 6.2.2#7 | xHC shall set the Parent Hub Slot ID field of an Output Slot Context to the value in the Input Slot Context of an Address Device Command. | 0 | |
| 6.2.2#8 | xHC shall set the Parent Port Number field of an Output Slot Context to the value in the Input Slot Context of an Address Device Command. | 0 | |
| 6.2.2#9 | xHC shall set the Interrupter Target field of an Output Slot Context to the value in the Input Slot Context of an Address Device or an Evaluate Context Command. | 0 | |
| 6.2.2#10 | The xHC shall target notifications of a Ring Overrun condition at the Interrupter defined in the Interrupter Target field of a Slot Context. | 0 | |
| 6.2.2#11 | The xHC shall target notifications of a Ring Underrun condition at the Interrupter defined in the Interrupter Target field of a Slot Context. | 0 | |
| 6.2.2#12 | xHC shall set the Hub field of the output Slot Context to the copied value of the Hub field from a previous Configure Endpoint command. | 0 | |
| **Subsection reference: 6.2.3 Endpoint Context** | | | |
| 6.2.3#1 | xHC shall set the Mult field of an Output Endpoint Context to the value in the Input Endpoint Context of a Configure Endpoint Command. | 0 | |
| 6.2.3#2 | xHC shall set the MaxPStreams field of an Output Endpoint Context to the value in the Input Endpoint Context of a Configure Endpoint Command. | 0 | |
| 6.2.3#3 | xHC shall set the LSA flag of an Output Endpoint Context to the value in the Input Endpoint Context of a Configure Endpoint Command. | 0 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 6.2.3#4 | xHC shall set the Interval field of an Output Endpoint Context to the value in the Input Endpoint Context of a Configure Endpoint Command. | 0 | |
| 6.2.3#5 | xHC shall set the FE flag of an Output Endpoint Context to the value in the Input Endpoint Context of a Configure Endpoint Command (xHC .96 only). | 0 | |
| 6.2.3#6 | xHC shall set the CErr field of an Output Endpoint Context to the value in the Input Endpoint Context of a Configure Endpoint Command. | 0 | |
| 6.2.3#7 | xHC shall set the EP Type field of an Output Endpoint Context to the value in the Input Endpoint Context of a Configure Endpoint or an Evaluate Endpoint Command. | 0 | |
| 6.2.3#8 | xHC shall set the HID flag of an Output Endpoint Context to the value in the Input Endpoint Context of a Configure Endpoint Command. | 0 | |
| 6.2.3#9 | xHC shall set the Max Burst Size field of an Output Endpoint Context to the value in the Input Endpoint Context of a Configure Endpoint or an Evaluate Endpoint Command. | 0 | |
| 6.2.3#10 | xHC shall set the Max Packet Size field of an Output Endpoint Context to the value in the Input Endpoint Context of a Configure Endpoint Command. | 0 | |
| 6.2.3#11 | xHC shall set the DCS flag of an Output Endpoint Context to the value in the Input Endpoint Context of a Configure Endpoint Command. | 0 | |
| 6.2.3#12 | xHC shall set the TR Dequeue Pointer field of an Output Endpoint Context to the value in the Input Endpoint Context of a Configure Endpoint Command. | 0 | |
| 6.2.3#13 | xHC shall set the Average TRB Length field of an Output Endpoint Context to the value in the Input Endpoint Context of a Configure Endpoint Command. | 0 | |
| 6.2.3#14 | xHC shall set the Max ESIT Payload field of an Output Endpoint Context to the value in the Input Endpoint Context of a Configure Endpoint Command. | 0 | |
| 6.2.3#15 | The xHC shall not limit the number of TRB retries if the CErr field is 0 in the Endpoint Context. | TD 5.23 | |
| 6.2.3#16 | The xHC shall set the TR Dequeue Pointer field of the Output Endpoint Context to the dequeue pointer value when the endpoint enters the Halted or Stopped states if the MaxPStreams is 0. | TD 4.07 TD 5.11 | |
| 6.2.3#17 | The xHC shall not set the EP State field to a value in the range of 5 to 7 in the Endpoint Context. | TD 4.03 | |
| 6.2.3#18 | xHC shall set bit 0 of Offset 04h of an Output Endpoint Context to '0' (xHC 1.0 only). | TD 4.03 | |
| **Subsection reference: 6.2.3.7 Reset Device Command Usage** | | | |
| 6.2.3.7#1 | Upon completion of a Reset Device Command, the Output Default Control Endpoint Context (DCI = '1') Max Packet Size, EP Type, CErr, TR Dequeue Pointer, and Average TRB Length fields shall contain the same values that they contained prior to execution of the Reset Device Command. | TD 4.08 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 6.2.3.7#2 | Upon completion of a Reset Device Command, the Output Default Control Endpont Context (DCI = '1') the Mult, MaxPStreams, LSA, Interval, Max ESIT Payload, Max Burst Size, and HID shall be cleared to 0. | TD 4.08 | |
| **Subsection reference: 6.2.4.1 Stream Context** | | | |
| 6.2.4.1#1 | The xHC shall set the TR Dequeue Pointer field of a Stream Context to the dequeue pointer value when the endpoint enters the Halted or Stopped states if the MaxPStreams is not 0. | TD 4.17 | |
| **Subsection reference: 6.2.6 Port Bandwidth Context** | | | |
| 6.2.6#1 | xHC shall set the Port Bandwidth fields in the Port Bandwidth Context to values in the range of 0 and 90 for SuperSpeed or full-speed endpoints. | TD 4.11 | |
| 6.2.6#2 | xHC shall set all the Port Bandwidth fields in the Port Bandwidth Context to values in the range of 0 and 80 for high-speed endpoints. | TD 4.11 | |
| **Subsection reference: 6.4 Transfer Request Block (TRB)** | | | |
| **Subsection reference: 6.4.1 Transfer TRBs** | | | |
| 6.4.1#1 | The xHC shall ignore the value in the Data Buffer Pointer field of a TRB if the TRB Transfer Length is 0. | TD 5.04.06 | |
| **Subsection reference: 6.4.1.1 Normal TRB** | | | |
| 6.4.1.1#1 | While processing a Normal TRB, if the Interrupt On Completion field in a Normal Data TRB is set to 1, and the BEI field set to '0', the xHC shall notify the system of the completion by placing an Event TRB on the Event ring and send an interrupt at the next interrupt threshold | 0 | |
| 6.4.1.1#2 | While processing a Normal TRB, if the Interrupt On Completion field in a Normal TRB is set to 1, and the BEI field set to '1', the xHC shall notify the system of the completion by placing an Event TRB on the Event ring but shall not assert an interrupt at the next interrupt threshold | 0 | |
| **Subsection reference: 6.4.1.3 Isoch TRB** | | | |
| 6.4.1.3#1 | While processing an Isoch TRB, if the Interrupt On Completion field in an Isoch TRB is set to 1, and the BEI field set to '0', the xHC shall notify the system of the completion by placing an Event TRB on the Event ring and send an interrupt at the next interrupt threshold | 0 | |
| 6.4.1.3#2 | While processing an Isoch TRB, if the Interrupt On Completion field in an Isoch TRB is set to 1, and the BEI field set to '1', the xHC shall notify the system of the completion by placing an Event TRB on the Event ring but shall not assert an interrupt at the next interrupt threshold | 0 | |
| **Subsection reference: 6.4.2 Event TRBs** | | | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| **Subsection reference: 6.4.2.1 Transfer Event TRB** | | | |
| 6.4.2.1#1 | The xHC shall set the TRB Pointer field of a Transfer Event TRB with the ED flag set to 0 to the 64-bit address of the TRB that generated this event. | Implicit | |
| 6.4.2.1#2 | The xHC shall set the TRB Transfer Length of a Transfer Event TRB with the ED flag set to 0 to a value in the range of 0 to 10000h. | Implicit | |
| **Subsection reference: 6.4.2.2 Command Completion Event TRB** | | | |
| 6.4.2.2#1 | The xHC shall set the Slot ID field to 0 in the Command Completion Event generated by a No Op, Force Event, Set Latency Tolerance Value, Get Port Bandwidth or Force Header Command. | TD 4.12 | |
| 6.4.2.2#2 | The xHC shall set the Command Completion Parameter to 0 for commands that don't use this parameter. | TD 2.01 TD 4.01 TD 4.04 | |
| **Subsection reference: 6.4.2.7 Device Notification Event TRB** | | | |
| 6.4.2.7#1 | The xHC shall set the Notification Type field of a Device Notification Event TRB to the Notification Type field of the received Device Notification TP. | 0 | |
| 6.4.2.7#2 | The xHC shall set the Device Notification Data field of a Device Notification Event TRB to the value in bytes 05h through 0Bh of the received Device Notification TP. | 0 | |
| 6.4.2.7#3 | The xHC shall set the Completion Code field to Success in a Device Notification Event TRB. | 0 | |
| 6.4.2.7#4 | The xHC shall set the Slot ID field of a Device Notification Event TRB to the ID of the device slot that generated this event. | 0 | |
| **Subsection reference: 6.4.2.8 MFINDEX Wrap Event TRB** | | | |
| 6.4.2.8#1 | The xHC shall set the Completion Code field to Success in an MFINDEX Wrap Event TRB. | 0 | |
| **Subsection reference: 6.4.4.2 Event Data TRBs** | | | |
| 6.4.4.2#1 | While processing an Event Data TRB, if the Interrupt On Completion field in an Event Data TRB is set to 1, and the BEI field set to '0', the xHC shall notify the system of the completion by placing an Event TRB on the Event ring and send an interrupt at the next interrupt threshold | 0 | |
| 6.4.4.2#2 | While processing an Event Data TRB, if the Interrupt On Completion field in an Event Data TRB is set to 1, and the BEI field set to '1', the xHC shall notify the system of the completion by placing an Event TRB on the Event ring but shall not assert an interrupt at the next interrupt threshold | 0 | |
| **Subsection reference: 6.4.5 TRB Completion Codes** | | | |
| 6.4.5#1 | The xHC shall not set the Completion Code field of any Event TRB to 0. | Implicit | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 6.4.5#2 | The xHC shall not set the completion code field of any TRB to a value of 28 (xHC 1.0 only). | Implicit | |
| 6.4.5#3 | The xHC shall not set the completion code field of any TRB to a value of 29 (xHC .96 only). | Implicit | |
| 6.4.5#4 | The xHC shall not set the completion code field of any TRB to a value of 30 (xHC .96 only). | Implicit | |
| **Chapter 7 Test Assertions:** | | | |
| **Subsection reference: 7 xHCI Extended Capabilities** | | | |
| 7#1 | Software writes to the Capability ID field of an Extended Capability Pointer Register shall have no effect. | 0 | |
| 7#2 | Software writes to the Next xHCI Extended Capability Pointer field of an Extended Capability Pointer Register shall have no effect. | 0 | |
| 7#3 | The xHC shall not set the Capability ID field of an xHCI Extended Capability Pointer Register to 0, 6-9, 11-16, or 18-191. | TD 1.05 | |
| 7#4 | The xHC shall not set the Capability ID field of an xHCI Extended Capability Pointer Register to 0, 7-9, 11-16, or 18-191 (xHC 1.0 only). | TD 1.05 | |
| **Subsection reference: 7.1 USB Legacy Support Capability** | | | |
| **Subsection reference: 7.1.1 USB Legacy Support Capability (USBLEGSUP)** | | | |
| 7.1.1#1 | The xHC shall not reset the values in the USBLEGSUP register on a LHCRST. | N/A | |
| 7.1.1#2 | A nonzero value in the Next Capability pointer field of the USBLEGSUP register shall indicate an offset in dwords to the beginning of the next capability. | Implicit | |
| **Subsection reference: 7.1.2 USB Legacy Support Control/Status (USBLEGCTLSTS)** | | | |
| 7.1.2#1 | Software writes to the SMI on Event Interrupt flag in the USBLEGCTLSTS register shall have no effect. | 0 | |
| 7.1.2#2 | The xHC shall set the SMI on Event Interrupt flag of the USBLEGCTLSTS register to the value in the EINT flag in the USBSTS register. | 0 | |
| 7.1.2#3 | Software writes to the SMI on Host System Error flag in the USBLEGCTLSTS register shall have no effect. | 0 | |
| 7.1.2#4 | The xHC shall clear the SMI on OS Ownership Change flag to 0 in the USBLEGCTLSTS register when software writes a 1 to it. | 0 | |
| 7.1.2#5 | Software writes of 0 to the SMI on OS Ownership Change flag in the USBLEGCTLSTS register shall have no effect. | 0 | |
| 7.1.2#6 | The xHC shall set the SMI on OS Ownership Change flag to 1 in the USBLEGCTLSTS register whenever the HC OS Owned Semaphore flag transitions from 1 to 0 or 0 to 1 in the USBLEGSUP register. | 0 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 7.1.2#7 | The xHC shall clear the SMI on PCI Command flag to 0 in the USBLEGCTLSTS register when software writes a 1 to it. | 0 | |
| 7.1.2#8 | Software writes of 0 to the SMI on PCI Command flag in the USBLEGCTLSTS register shall have no effect. | 0 | |
| 7.1.2#9 | The xHC shall set the SMI on PCI Command flag to 1 in the USBLEGCTLSTS whenever the PCI Command Register is written. | 0 | |
| 7.1.2#10 | The xHC shall clear the SMI on BAR flag in the USBLEGCTLSTS register to 0 in the USBLEGCTLSTS register when software writes a 1 to it. | 0 | |
| 7.1.2#11 | Software writes of 0 to the SMI on BAR flag in the USBLEGCTLSTS register shall have no effect. | 0 | |
| 7.1.2#12 | The xHC shall set the SMI on BAR flag to 1 in the USBLEGCTLSTS whenever the BAR is written. | 0 | |
| 7.1.2#13 | The xHC shall not reset the value in the USBLEGCTLSTS register on a LHCRST. | N/A | |
| **Subsection reference: 7.2 xHCI Supported Protocol Capability** | | | |
| 7.2#1 | The xHC shall implement at least one xHCI Supported Protocol Capability. | 0 | |
| 7.2#2 | The xHC shall set the Minor Revision field to 0 in the xHCI Supported Protocol Capability if the Major Revision is 2. | 0 | |
| 7.2#3 | The xHC shall set the Major Revision field to 3 or 2 in the xHCI Supported Protocol Capability. | 0 | |
| 7.2#4 | The xHC shall set the Name String field to 20425355h in the xHCI Supported Protocol Capability. | 0 | |
| 7.2#5 | The xHC shall set the Compatible Port Offset field of the xHCI Supported Protocol Capability to a value in the range of 1 to MaxPorts. | 0 | |
| 7.2#6 | The xHC shall set the Compatible Port Count field of the xHCI Supported Protocol Capability to a value in the range of 1 to MaxPorts. | 0 | |
| 7.2#7 | The xHC shall ensure that the Compatible Port Offset + Compatible Port Count - 1 do not exceed MaxPorts. | 0 | |
| 7.2#8 | The xHC shall set the L1C flag of the xHCI Supported Protocol Capability to 0 if it is not for USB2. | 0 | |
| 7.2#9 | Software writes of the xHCI Supported Protocol Capability register shall have no effect. | 0 | |
| 7.2#10 | The xHC shall ensure that the set of ports defined by a USB3 xHCI Supported Protocol Capability shall not overlap those defined by a USB2 xHCI Supported Protocol Capability. | 0 | |
| 7.2#11 | One xHCI Supported Protocol Capability shall define a compatible Port Offset of 1. | 0 | |
| 7.2#12 | A nonzero value in the Next Capability Pointer field of an xHCI Supported Protocol Capability shall indicate an offset in dwords to the beginning of the next capability. | Implicit | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 7.2#13 | The xHCI shall set the HSO bit of the Supported Protocol Capability register to 1 if the ports described in the capability are USB 2.0 high speed only | TD 1.05 | |
| 7.2#14 | The xHCI shall set the HSO bit of the Supported Protocol Capability register to 0 if the ports described in the capability support high, full, and low speed | TD 1.05 | |
| 7.2#15 | The xHCI shall set the IHI bit of the Supported Protocol Capability register to 0 if the port mapping adheres to the default mapping described in section 4.24.2.1 | TD 3.14 | |
| 7.2#16 | The xHC shall set bits 16:27 of the Supported Protocol Capability register to 0 | TD 1.05 | |
| 7.2#17 | Software writes of the Capability ID field of an xHCI Supported Protocol Capability shall be ignored | TD 1.05 | |
| 7.2#18 | Software writes of the Next Capability Pointer field of an xHCI Supported Protocol Capability shall be ignored | TD 1.05 | |
| 7.2#19 | Software writes of the Minor Revision field of an xHCI Supported Protocol Capability shall be ignored | TD 1.05 | |
| 7.2#20 | Software writes of the Major Revision field of an xHCI Supported Protocol Capability shall be ignored | TD 1.05 | |
| 7.2#21 | Software writes of the Name String field of an xHCI Supported Protocol Capability shall be ignored | TD 1.05 | |
| 7.2#22 | Software writes of the Compatbile Port Offset field of an xHCI Supported Protocol Capability shall be ignored | TD 1.05 | |
| 7.2#23 | Software writes of the Compatbile Port Count field of an xHCI Supported Protocol Capability shall be ignored | TD 1.05 | |
| 7.2#24 | A nonzero value in the Next Capability Pointer field of an xHCI Supported Protocol Capability indicates an offset in dwords to the beginning of the next capability. | TD 1.05 | |
| 7.2#25 | If the xHC has a nonzero value for the PSIC field of the supported protocol capability field then the xHC shall provide that number of PSI registers following the supported protocol capability field register. | TD 1.09 | |
| 7.2#26 | If xHCI Protocol Extended Capability Major Revision = 02h, then LP field of PSID shall be 0. | | |
| 7.2#27 | If xHCI Protocol Extended Capability Major Revision = 03h, then LP field of PSID shall be 0 for SS. | | |
| 7.2#28 | If xHCI Protocol Extended Capability Major Revision = 03h, then LP field of PSID shall be 1 for SSP. | | |
| 7.2#29 | The xHC shall set the Minor Revision field to 10 or 0 in the xHCI Supported Protocol Capability if Major Revision is 3. | TD 1.05 | |
| **Subsection reference: 7.2.1** | | | |
| 7.2.1#1 | For each PSI dword reported by the xHC when a device is attached that operates at the bit rate defined by the PSI dword the value of the PSIV field shall be reported in the Port Speed field of the PORTSC register (xHC 1.x only) | TD 1.09 | |
| 7.2.1#2 | For each PSI dword reported by the xHC, the value of the PSIV field shall not be 0 (xHC 1.x only). | TD 1.09 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 7.2.1#3 | For each PSI dword reported by the xHC, the PLT field in the xHCI Supported Protocol Capability shall be 0 if the dword defines a symmetric bit rate (xHC 1.x only). | TD 1.09 | |
| 7.2.1#4 | For each PSI dword reported by the xHC, the PLT field in the xHCI Supported Protocol Capability shall not be 1 (xHC 1.x only) | TD 1.09 | |
| 7.2.1#5 | For each PSI dword reported by the xHC, the PLT field in the xHCI Supported Protocol Capability shall be 2 if the dword defines an asymmetric receive bit rate | | |
| 7.2.1#6 | For each PSI dword reported by the xHC, the PLT field in the xHCI Supported Protocol Capability shall be 3 if the dword defines an asymmetric transmit bit rate | | |
| 7.2.1#7 | For each PSI dword reported by the xHC, where the the PLT field in the xHCI Supported Protocol Capability reports an asymmetric receive bit rate, it shall be immediately followed by a dword reporting an asymmetric transfer rate (xHC 1.x only) | TD 1.09 | |
| 7.2.1#8 | For each PSI dword reported by the xHC, the PFD field in the xHCI Supported Protocol Capability shall be set to 0 if the link is half duplex (xHC 1.x only) | TD 1.09 | |
| 7.2.1#9 | For each PSI dword reported by the xHC, the PFD field in the xHCI Supported Protocol Capability shall be set to 1 if the link is full duplex (xHC 1.x only) | TD 1.09 | |
| **Subsection reference: 7.2.2** | | | |
| 7.2.2#1 | One xHCI Supported Protocol Capability shall define a Compatible Port Offset of 1 | | |
| 7.2.2#2 | The port numbers assigned by multiple xHCI Supported Protocol Capabilities of the same type shall not overlap | TD 1.05 | |
| **Subsection reference: 7.2.2.1** | | | |
| 7.2.2.1#1 | The set of ports defined by USB3 and USB2 xHCI Supported Protocol Capabilities shall not overlap each other | TD 1.05 | |
| **Subsection reference: 7.2.2.1.2** | | | |
| 7.2.2.1.2#1 | If the xHC set the PSIC field of the Supported Protocol Capability register to '0' then the xHC shall support the default speed mapping specified in section 7.2.2.1.1. | TD 1.09 | |
| **Subsection reference: 7.2.2.1.3.1** | | | |
| 7.2.2.1.3.1#1 | If the LSECC field of USB3 Protocol Defined fields is 1, then hardware supports reporting Link Soft Error Count through PORTEXSC register. | | |
| **Subsection reference: 7.2.2.1.3.2** | | | |
| 7.2.2.1.3.2#1 | Software writes of the HSO bit of the USB 2.0 Protocol Defined field for a USB 2.0 xHCI Supported Protocol Capability register shall be ignored | TD 1.05 | |
| 7.2.2.1.3.2#2 | Software writes of the IHI bit of the USB 2.0 Protocol Defined field for a USB 2.0 xHCI Supported Protocol Capability register shall be ignored | TD 1.05 | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 7.2.2.1.3.2#3 | Software writes of the HLC bit of the USB 2.0 Protocol Defined field for a USB 2.0 xHCI Supported Protocol Capability register shall be ignored (xHC 1.0 only). | TD 1.05 | |
| 7.2.2.1.3.2#4 | The xHCI shall set the HLC bit of the USB 2.0 Protocol Defined field for a USB 2.0 xHCI Supported Protocol Capability register to 1 if the ports described support hardware controlled USB2 link power management (xHC 1.x only). | TD 3.11 | |
| 7.2.2.1.3.2#5 | BESL LMP Capability support shall be mandatory for all xHCI 1.1 and xHC 1.2 compliant xHCs. | TD 3.11 | |
| 7.2.2.1.3.2#6 | Hardware LMP Capability support (HLC=1) shall be mandatory for all xHCI 1.1 and xHCI 1.2 compliant xHCs. | TD 3.11 | |
| **Subsection reference: 7.6 Debug Capability (DbC)** | | | |
| 7.6#1 | The Debug Capability shall be operational anytime the assigned port is not suspended and the xHC is in D0. | DbC TBD | |
| 7.6#2 | USBCMD.RS and USBCMD.LHCRST shall have no affect on the operation of the debug capability | DbC TBD | |
| 7.6#3 | The xHC shall assign the debug capability to the first root hub port that detects an attach of the downstream facing port of a SuperSpeed capable root hub or external hub | DbC TBD | |
| **Subsection reference: 7.6.1 Debugging Topologies** | | | |
| 7.6.1#1 | An xHC operating as a debug host shall be able to connect to a debug target through a USB 3.0 hub. | DbC TBD | |
| 7.6.1#2 | An xHC operating as a debug target shall only expose its USB Debug Capability through a root hub port. | DbC TBD | |
| **Subsection reference: 7.6.4 Operational Model** | | | |
| 7.6.4#1 | The DbC shall respond with a STALL TP to a SetFeature(FUNCTION_SUSPEND) setup stage request. | DbC TBD | |
| 7.6.4#2 | An xHC operating as a debug target shall only expose its USB Debug Capability through a root hub port. | DbC TBD | |
| **Subsection reference: 7.6.4.2 Event Generation** | | | |
| 7.6.4.2#1 | When set, the DCPORTSC status change bits shall remain set until software writes to the DCPORTSC register with the appropriate status change bits set to '1', a Chip Hardware Reset occurs, or the Debug Capability is disabled (DCE = '0') | DbC TBD | |
| 7.6.4.2#2 | If the assertion of a DCPORTSC status change bit results in a '0' to '1' transition of DCPSCEG, the Debug Capability shall respond by generating a Port Status Change Event | DbC TBD | |
| 7.6.4.2#3 | The xHC shall only generate Transfer Events with a Completion Code of Success, Babble Detected Error, TRB Error, Short Packet, Undefined Error, Event Ring Full Error or Vendor Defined Error | DbC TBD | |
| **Subsection reference: 7.6.4.3 Halted DbC Endpoints** | | | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 7.6.4.3#1 | When the HOT or HIT flag for an endpoint is set to '1' the xHC shall write the current value of the TR dequeue pointer for the endpoint to the endpoint context (1.0 xHC only). | DbC TBD | |
| 7.6.4.3#2 | If a bulk endpoint is transferring data when its HOT or HIT flag is set to '1' the DbC shall generate a transfer event where the TRB pointer shall reference the transfer trb that the error occurred on and the completion code shall indicate a stall error (1.x xHC only). | DbC TBD | |
| 7.6.4.3#3 | Upon reception of a ClearFeature(ENDPOINT_HALT) request, the DbC shall clear the HIT or HOT flag for the respective endpoint (1.x xHC only). | DbC TBD | |
| 7.6.4.3#4 | After reception of a ClearFeature(ENDPOINT_HALT) request, the DbC shall begin executing TRBs at the address stored in the TR Dequeue Pointer field of the endpoint context the next time the doorbell is rung (1.x xHC only). | DbC TBD | |
| **Subsection reference: 7.6.5 Port Routing and Control** | | | |
| 7.6.5#1 | When a root hub port is assigned to the Debug Capability, the associated PORTSC register shall mimic operations as if no device is attached to it. | DbC TBD | |
| 7.6.5#2 | When Debug Capability is enabled on a root hub port, all ports not assigned to the Debug Capability shall function normally | DbC TBD | |
| 7.6.5#3 | After the root hub port is assigned to the DbC, the xHC shall begin emulating a USB Debug Class device. | DbC TBD | |
| **Subsection reference: 7.6.7 The USB Debug Device** | | | |
| 7.6.7#1 | When the Debug Device is configured and the bulk endpoints are operational, the DbC Run bit in the DCCTRL register shall transiton to '1' | DbC TBD | |
| **Subsection reference: 7.6.7.2.1 Data Transfers** | | | |
| 7.6.7.2.1#1 | If a DbC Bulk pipe had previously sent an NRDY, the xHC shall generate an ERDY when the doorbell is rung | DbC TBD | |
| **Subsection reference: 7.6.8.1 Debug Capability ID Register** | | | |
| 7.6.8.1#1 | DCID.CapabilityID must be set to 10 decimal | DbC TBD | |
| 7.6.8.1#2 | A value of zero contained in the Next Capability pointer indicates the end of the capability list. | DbC TBD | |
| 7.6.8.1#3 | A nonzero value in the Next Capability pointer indicates an offset in dwords to the beginning of the next capability. | DbC TBD | |
| 7.6.8.1#4 | The DCERST Max field of the DCID register must be a value between 0 and 15. | DbC TBD | |
| 7.6.8.1#5 | Writes to the DCID register shall have no affect on the value contained in the DCID register. | DbC TBD | |
| **Subsection reference: 7.6.8.2 Debug Capability Doorbell Register** | | | |
| 7.6.8.2#1 | Reads from DCDB.DB Target return a value of zero | DbC TBD | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 7.6.8.2#2 | The default value for the DCDB register shall be 0 | DbC TBD | |
| 7.6.8.2#3 | If software writes a value of '0' to the DB Target field of the DCDB, the xHC shall Data EP 1 OUT Enqueue Pointer has been updated | DbC TBD | |
| 7.6.8.2#4 | The xHC shall set the DB Target field of the DCDB to a value of '1' when the Data EP 1 IN Enqueue Pointer has been updated | DbC TBD | |
| **Subsection reference: 7.6.8.3.1 DCERSTSZ** | | | |
| 7.6.8.3.1#1 | The default value for the DCERSTSZ register is 0 | DbC TBD | |
| **Subsection reference: 7.6.8.3.2 Debug Capability Event Ring Segment Table Base Address Reg.** | | | |
| 7.6.8.3.2#1 | The default value for the DCERSTBA register is 0 | DbC TBD | |
| **Subsection reference: 7.6.8.3.3 DCERDP** | | | |
| 7.6.8.3.3#1 | The default value for the DCERDP register is 0 | DbC TBD | |
| **Subsection reference: 7.6.8.4 DCCTRL** | | | |
| 7.6.8.4#1 | The default value for the DCCTRL register is 0 | DbC TBD | |
| 7.6.8.4#2 | Writes to the DCR bit of DCCTRL shall have no affect on the value of the DCR bit | DbC TBD | |
| 7.6.8.4#3 | A '0' to '1' transition of the DCPORTSC:PR bit shall clear the DCR bit on the DCCTRL register to '0' | DbC TBD | |
| 7.6.8.4#4 | When the DCR bit of the DCCTRL register is set to a value of '1', the Debug Capability shall accept bulk data pipe transactions | DbC TBD | |
| 7.6.8.4#5 | Setting the LSE bit of the DCCTRL register to a value of '1' shall enable the DbC to generate port status change events. | DbC TBD | |
| 7.6.8.4#6 | When DCCTRL.HOT is '1' and DCCTRL.DCR is '1' the DbC shall generate STALL TPs for all IN TPs received for the OUT TR. | DbC TBD | |
| 7.6.8.4#7 | The DbC shall clear DCCTRL.HOT when a ClearFeature(ENDPOINT_HALT) request is received for the OUT endpoint. | DbC TBD | |
| 7.6.8.4#8 | When DCCTRL.HIT is '1' and DCCTRL.DCR is '1' the DbC shall generate STALL TPs for all OUT DPs received for the IN TR. | DbC TBD | |
| 7.6.8.4#9 | The DbC shall clear DCCTRL.HIT when a ClearFeature(ENDPOINT_HALT) request is received for the IN endpoint. | DbC TBD | |
| 7.6.8.4#10 | The DbC shall clear the DRC bit in the DCCTRL register when a '1' is written to DRC. | DbC TBD | |
| 7.6.8.4#11 | The DbC shall set the DRC bit in the DCCTRL register to '1' when the DCR bit transitions from a '1' to a '0' | DbC TBD | |
| 7.6.8.4#12 | Debug Max Burst Size field of DCCTRL shall report the maximum burst size supported by the bulk endpoints for the DbC. | DbC TBD | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 7.6.8.4#13 | Writes to the Debug Max Burst Size field of the DCCTRL register shall have no affect on the value contained in the debug max burst size field. | DbC TBD | |
| 7.6.8.4#14 | The device address field shall report the USB device address assigned to the debug device during enumeration when DbC is set to '1' | DbC TBD | |
| 7.6.8.4#15 | Writes to the device address field of the DCCTRL register shall have no affect on the value contained in the device address field. | DbC TBD | |
| 7.6.8.4#16 | Setting the DCE bit of the DCCTRLR register enables the DbC operation. | DbC TBD | |
| 7.6.8.4#17 | Clearing the DCE bit of the DCCTRL register releases the root hub port assigned to the DbC. | DbC TBD | |
| **Subsection reference: 7.6.8.5 DCST** | | | |
| 7.6.8.5#1 | Writes to the DCST register shall have no affect on the value contained in the register | DbC TBD | |
| 7.6.8.5#2 | The ER bit of the DCST register shall be set to '0' when the debug capability event ring is empty. | DbC TBD | |
| 7.6.8.5#3 | The ER bit of the DCST register shall be set to '1' when the debug capability event ring contains a transfer event. | DbC TBD | |
| 7.6.8.5#4 | DCST.Debug Port Number shall be zero when the DbC is not attached to a root hub port. | DbC TBD | |
| 7.6.8.5#5 | DCST.Debug Port Number shall contain the ID of the root hub port that the DbC has been automatically attached to. | DbC TBD | |
| 7.6.8.5#6 | The default value for the DCST register is 0. | DbC TBD | |
| **Subsection reference: 7.6.8.6 DCPORTSC** | | | |
| 7.6.8.6#1 | The default value for the CCS field of the DCPORTSC register is 0. | DbC TBD | |
| 7.6.8.6#2 | The default value for the PED field of the DCPORTSC register is 0. | DbC TBD | |
| 7.6.8.6#3 | The default value for the PR field of the DCPORTSC register is 0. | DbC TBD | |
| 7.6.8.6#4 | The default value for the Port Speed field of the DCPORTSC register is 0. | DbC TBD | |
| 7.6.8.6#5 | The default value for the CSC field of the DCPORTSC register is 0. | DbC TBD | |
| 7.6.8.6#6 | The default value for the PRC field of the DCPORTSC register is 0. | DbC TBD | |
| 7.6.8.6#7 | The default value for the PLC field of the DCPORTSC register is 0. | DbC TBD | |
| 7.6.8.6#8 | The default value for the CEC field of the DCPORTSC register is 0. | DbC TBD | |
| 7.6.8.6#9 | DCPORTSC.CCS shall be zero when DCCTRL.DCE is zero | DbC TBD | |
| 7.6.8.6#10 | DCPORTSC.CCS shall be zero when no debug host is present. | DbC TBD | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 7.6.8.6#11 | DCPORTSC.CCS shall be '1' when a root hub port is connected to a debug host and assigned to the DbC. | DbC TBD | |
| 7.6.8.6#12 | DCPORTSC.PED shall be set to '1' by a '0' to '1' transition of DBC | DbC TBD | |
| 7.6.8.6#13 | DCPORTSC.PED shall be set to '1' by a '1' to '0' transition of PR | DbC TBD | |
| 7.6.8.6#14 | When DCPORTSC.PED transitions from '0' to '1', the port's link shall transition to the Rx.Detect state | DbC TBD | |
| 7.6.8.6#15 | bits 3:2 of the DCPORTSC register shall be set to '0' | DbC TBD | |
| 7.6.8.6##16 | DCPORTSC.PR shall be '1' when a bus reset sequence is detected on the root hub port assigned to the Debug Capability. | DbC TBD | |
| 7.6.8.6##17 | DCPORTSC.PR shall be '0' when port is not in reset. | DbC TBD | |
| 7.6.8.6##18 | DCPORTSC.PR shall be '0' when DCE is '0' | DbC TBD | |
| 7.6.8.6##19 | DCPORTSC.PR shall be '0' when CCS is '0' | DbC TBD | |
| 7.6.8.6##20 | DCPORTSC.PLS shall indicate the current link state | DbC TBD | |
| 7.6.8.6##21 | bit 9 of the DCPORTSC register shall be set to '0' | DbC TBD | |
| 7.6.8.6##22 | Port Speed shall contain a value of '4' when a debug host is attached (CCS = 1) | DbC TBD | |
| 7.6.8.6#23 | Port Speed shall contain a value of '0' when a debug host is not attached (CCS = 0) | DbC TBD | |
| 7.6.8.6#24 | Bits 16:14 of DCPORTSC shall be set to '0' | DbC TBD | |
| 7.6.8.6#25 | DCPORTSC.CSC shall be cleared by writing a '1' to it. | DbC TBD | |
| 7.6.8.6#26 | DCPORTSC.CSC shall be set to '1' in response to a change in current connect status for the debug capability | DbC TBD | |
| 7.6.8.6#27 | DCPORTSC.CSC shall be '0' when DCE is '0' | DbC TBD | |
| 7.6.8.6#28 | Bits 20:18 of DCPORTSC shall be set to '0' | DbC TBD | |
| 7.6.8.6#29 | DCPORTSC.PRC shall be cleared by writing a '1' to it. | DbC TBD | |
| 7.6.8.6#30 | DCPORTSC.PRC shall be set to '1' when reset processing on the port is complete. | DbC TBD | |
| 7.6.8.6#31 | DCPORTSC.PRC shall be '0' when DCE is '0' | DbC TBD | |
| 7.6.8.6#32 | DCPORTSC.PLC shall be cleared by writing a '1' to it | DbC TBD | |
| 7.6.8.6#33 | DCPORTSC.PLS shall be set to '1' in response to a U0 -> U3 transition. | DbC TBD | |
| 7.6.8.6#34 | DCPORTSC.PLS shall be set to '1' in response to a U3 -> U0 transition. | DbC TBD | |
| 7.6.8.6#35 | DCPORTSC.PLC shall be '0' when DCE is '0' | DbC TBD | |
| 7.6.8.6#36 | DCPORTSC.CEC shall be cleared by writing a '1' to it. | DbC TBD | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| **Subsection reference: 7.6.8.7 DCCP** | | | |
| | | | |
| **Subsection reference: 7.6.8.8 DCCDI1** | | | |
| 7.6.8.8#1 | Default value for DbCIC shall be 0 | DbC TBD | |
| 7.6.8.8#2 | The value of the DbC Protocol field of the DCDDI1 register shall not be greater than '1' | DbC TBD | |
| 7.6.8.8#3 | Bits 15:8 shall always return a value of '0' | DbC TBD | |
| **Subsection reference: 7.6.8.9 DCCDI2** | | | |
| 7.6.8.9#1 | Default value for DCDDI2 shall be 0 | DbC TBD | |
| 7.6.8.9#2 | While the DCE bit is '0', the xHC shall not modify the DCCTRL registers | DbC TBD | |
| **Subsection reference: 7.6.9.1 DbC** | | | |
| | | | |
| **Subsection reference: 7.7 xHCI I/O Virtualization** | | | |
| | | | |
| **Subsection reference: 7.7.1 Capability Header** | | | |
| | | | |
| **Subsection reference: 7.8 Virtualization** | | | |
| 7.8#1 | If implemented, Capability ID of xHCI Local Memory Capability shall be set to '6' | TBD | |
| 7.8#2 | If implemented, Writes to the Capability ID field of the XHCI Local memory capability register shall have no effect upon its value | TBD | |
| 7.8#3 | If implemented, the Next Capability Pointer field of te XHCI Local memory capability shall contain a value of zero if there are no additional capabilities | TBD | |
| 7.8#4 | If implemented, a nonzero value in the Next Capability Pointer field of an xHCI Local Memory Capability shall indicate an offset in dwords to the beginning of the next capability | TBD | |
| 7.8#5 | If implemented, writes to the Next Capability field of the XHCI Local memory capability register shall have no effect upon its value | TBD | |
| 7.8#6 | If implemented, Bits 31:17 of the XHCI Local Memory Capability register shall be set to 0 | TBD | |
| 7.8#7 | If implemented,, the default value of the LME bit of the XHCI Local Memory Capability shall be '0' | TBD | |
| 7.8#8 | If implemented, Setting the LME bit to '1' shall enable the local memory capability | TBD | |
| 7.8#9 | If implemented,, Size field of xHCI Local Memory Capabilty shall be in the range of 1 to 255 | TBD | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 7.8#10 | If implemented, Setting LME field of xHCI Local Memory Capability to '0' shall disable local memory | TBD | |
| **Subsection reference: 7.9 xHC Audio Sideband Capability** | | | |
| 7.9#1 | The number of sideband resources available in a given implementation shall be discoverable using the Get Extended Property Command | | |
| 7.9#2 | Each sideband resource shall support the sideband datapath for one endpoint. | | |
| 7.9#3 | Sideband resources shall be numbered from 1 to N where N is the number of sideband resources supported. | | |
| **Subsection reference: 7.9.1 Get Extended Property Commands** | | | |
| 7.9.1#1 | The Supported Values of the SubType field for Get Extended Property Command for Audio Sideband Capability are 001b and 010b. | | |
| **Subsection reference: 7.9.1.1 Get Extended Property:  Get Supported Resources Command** | | | |
| 7.9.1.1#1 | When the Get Supported Resources Command is executed by the xHC, it shall write the Supported Resource Count to the location specified by the Extended Property Conext Pointers. | | |
| 7.9.1.1#2 | When the Get Supported Resources Command is executed by the xHC, it shall insert a Command Completion Event TRB into the Event Ring with Slot ID = 0, Bit [0] of the Command Completion Parameter set to '1' and Completion Code set to Success. | | |
| **Subsection reference: 7.9.1.2 Get Extended Property:  Get Endpoint Properties Command** | | | |
| 7.9.1.2#1 | When the Get Extended Properties Command is executed by the xHC, it shall write the requested attributes to the location specified by the Extended Property Conext Pointers. | | |
| 7.9.1.2#2 | When the Get Extended Properties Command is executed by the xHC, it shall insert a Command Completion Event TRB into the Event Ring with Slot ID = Slot ID specified by the command, Bit [0] of the Command Completion Parameter set to '1', Bit[23:1] of Command Completion Parameter set to 0 and Completion Code set to Success. | | |
| **Subsection reference: 7.9.2.1 Set Extended Property:  Set Resource Assignment Command** | | | |
| 7.9.2.1#1 | When the Set Resource Assignment Command is executed by the xHC, it shall perform the appropriate internal datapath assignment to map the resource number specified in the command to the Slot/Endpoint ID specified. | | |
| 7.9.2.1#2 | When the Set Resource Assignement Command is executed by the xHC, it shall deallocate previously allocated resources if the Resource number is set to zero. | | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| 7.9.2.1#3 | When a Set Resource Assignment Command is executed by the xHC, it shall insert a Command Completion Event TRB in the Event ring with Slot ID = Slot ID specified by the command, Bit [0] of the Command Completion Parameter set to '1' and Bit [23:1] of the Command Completion parameter set to zero. | | |
| **Subsection reference: 7.10 Intel Time Stamp Correlation Capability** | | | |
| 7.10#1 | If the Intel Time Stamp Correlation Capability is supported, the xHC shall support the Get Extended Property and Set Extended Property commands, if GSC bit in HCCPARAMS2 is set. | | |
| **Subsection reference: 7.10.1 Extended Capability ID** | | | |
| 7.10.1#1 | If the Intel Time Stamp Correlation Capability is identified by bit 12 of the ECI field | | |
| **Subsection reference: 7.10.2 Get Extended Property Commands** | | | |
| 7.10.2#1 | The Time Stamp Correlation Extended Capability is identified by setting bit [12] in the Extended Capability Identified of the Get Property Command to '1'. | | |
| **Subsection reference: 7.10.3 Get Extended Property: Get Time Stamp Command** | | | |
| 7.10.3#1 | When executing a Get Time Stamp Command, the xHC shall capture a synchronous snapshot of the HW System Time and the USB Bus Interval Counter and Delta Time, and write the results to the Extended Property Context. | | |
| 7.10.3#2 | When executing a Get Time Stamp Command, the xHC shall insert a Command Completion Event TRB into the Event Ring with Slot ID = 0, Bit [12] of the Command completion Parameter set to '1' and all other bits of the Command Completion parameter set to zero. | | |
| **Appendix A Test Assertions:** | | | |
| **Subsection reference: Appendix A xHCI PCI Power Management Interface** | | | |
| A#1 | The xHC shall at least support the D0, D3hot and D3cold PCI PM states. | TBD | |
| **Subsection reference: Appendix A.1 PCI Power Management Register Interface** | | | |
| A.1#1 | The xHC shall set the Aux_Current field or Data Register to 0 when the PME_Support bit for D3cold has been disabled. | TBD | |
| **Subsection reference: Appendix A.1.1 Power State Transitions** | | | |
| A.1.1#1 | The xHC shall maintain the state of the PME_Status bit of the PMCSR when transitioning from D3hot to D0 | TBD | |
| A.1.1#2 | The xHC shall maintain the state of the PME_En bit of the PMCSR when transitioning from D3hot to D0. | TBD | |
| A.1.1#3 | The xHC shall maintain the state of the PME_Support(D3cold) bit of the PMCSR.when transitioning from D3hot to D0. | TBD | |

| Assertion # | Assertion Description | Test # | Comments |
|---|---|---|---|
| A.1.1#4 | The xHC shall not reset the USB Legacy Support Registers when transitioning from D3hot to D0. | TBD | |
| A.1.1#5 | The xHC shall not reset the PORTSC registers when transitioning from D3hot to D0. | TBD | |
| **Subsection reference: Appendix A.1.2 Power State Definitions** | | | |
| A.1.2#1 | The xHC shall set the PME_Status bit to 1 when it is programmed with Power_State of the PMCSR set to D0 and a wakeup event is detected. | TBD | |

# 3   Test Descriptions

## 3.1   Register Interface Test

For all tests program bits 6:63 (xHC 1.0 or higher) or bits 4:63 (xHC .96) with Event Ring Segment Base Address (5.5.2.3.2#5)

### TD.1.01 PCI Configuration Register Test

This test verifies that the PCI configuration space registers are set to the correct values.  This test is only applicable to xHCs using a PCI interface.

**Required Device Resource**

None

**Overview of Test Steps**

The test performs the following steps.

1    Read the Class Code Register and verify the followings:
   1.1    The *Base* Class *Code (BASEC)* field is set to 0Ch (5.2.2#3).
   1.2    The *Sub-Class Code (SCC)* field is set to 03h (5.2.2#2).
   1.3    The *Programming Interface (PI)* field is set to 30h (5.2.2#1).
2    Read the Base Address Register 0 and verify that the bits 11:4 are set to zero, i.e. the Capability Register Space aligns on a page boundary (5#1).
3    Read the Serial Bus Release Number (SBRN) Register and verify that it is set to one of the following:
   3.1    30h.
   3.2    31h.
   3.3    32h.
4    Read the Capabilities in the Capabilities List and verify the followings:
   4.1    PCI Power Management Capability exists (5.2.5#1).
   4.2    Either MSI Capability or MSI-X Capability exists (5.2.6#1).
   4.3    No more than one MSI-X Capability exists (5.2.6.2#1).
   4.4    If MSI Capability exists, the *Multiple Message Capable* field indicates a value equal to or less than the *MaxIntrs* field value in the HCSPARAMS1 register (4.17#1).
   4.5    If MSI-X Capability exists, MSI-X Table and MSI-X PBA do not overlap each other in the memory-mapped I/O register space (5.2.6.5#1).

### TD.1.02 Capability Register Test

This test verifies that the Capability Registers are set to the correct values in the memory-mapped I/O space.

**Required Device Resource**

None

**Overview of Test Steps**

The test software performs the following steps.

1    Read the CAPLENGTH register and verify that the Operational Register Space aligns on a 4-byte boundary (5#2).
2    Read the HCIVERSION register and verify:.
    2.1    The HCIVERSION is required to be greater than or equal to 0110h to qualify for USB-IF certification.
    2.2    HCIVERSION is 0096h, 0100h, 0110h or 0120h (5.3.2#2).
3    Read the HCSPARAMS1 register and verify the followings:
    3.1    The *MaxSlots* field is set to a value in the range of 1 to FFh (5.3.3#1).
    3.2    The *MaxIntrs* field is set to a value in the range of 1 to 400h (5.3.3#2).
    3.3    The *MaxPorts* field is set to a value in the range of 1 to FFh (5.3.3#3).
4    Read the HCSPARAMS2 register and verify the followings:
    4.1    The *SPR* field is set to 0 if the *Max Scratchpad Buffers* field is 0 (5.3.4#1) xHCI0.96 and 1.0.
    4.2    The SPR field is set to 0 if the Max Scratchpad Bufs (Hi, Lo) fields are 0 (5.3.4#6) xHCI 1.1 or xHCI 1.2 only.
    4.3    Bits 13-25 must be 0 xHCI Version .96 only (5.3.4#2).
    4.4    The IOC Interval field is set to a value from 0-23 inclusive, xHCI version .96 only (5.3.4#3).
    4.5    Bits 8-25 must be 0 xHCI Version 1.0 only (5.3.4#4).
    4.6    Bits 8-20 must be 0 xHCI Version 1.10 or xHCI Version 1.20 only (5.3.4#5).
5    Read the HCSPARAMS3 register and verify the followings:
    5.1    The *U1 Devce Exit Latency* field is set to a value in the range of 0 to 0Ah (5.3.5#1).
    5.2    The *U2 Device Exit Latency* field is set to a value in the range of 0 to 07FFh (5.3.5#2).
6    Read the HCCPARAMS1 register and verify the followings:
    6.1    The *MaxPSASize* field is set to a value in the range of 1 to 15 (5.3.6#1).
    6.2    Bits 10:11 are 0 (5.3.6#2) xHCI version .96 only.
    6.3    Bits 9:11 are 0 xHCI version 1.0 only. (5.3.6#7).
    6.4    Verify that bit 11 is set to 1 (CFC). xHCI Version 1.10 or later (4.11.2.5#3)
    6.5    Verify that bit 9 is set to 1 (SPC). xHCI Version 1.10 or later (4.6.9#32)
    6.6    Verify that bit 10 is set to 1 (SEC). xHCI Version 1.10 or later  (4.12#1)
7    Read the DBOFF register and verify the followings:
    7.1    The lower 2 bits are set to 0 (5.3.7#1).
    7.2    The Doorbell Array aligns on a 4-byte boundary (5#4).
8    Read the RTSOFF register and verify the followings:
    8.1    The lower 5 bits are set to 0 (5.3.8#1).
    8.2    The Runtime Register Space aligns on a 32-byte boundary (5#3).
9    Read the HCCPARAMS2 register and verify the followings, xHCI version 1.10 and 1.20 only
    9.1    xHC version 1.10 and 1.20 shall contain an HCCPARAMS2 register. (5.3.9#1)
    9.2    Verify that bit 0 is set to 1 (U3C).  (4.15.1#3)
    9.3    Verify that bit 2 is set to 1 (FSC).  (5.3.9#6)
    9.4    Verify that bit 3 is set to 1 (CTC).  (4.19.1.2.4.1#1)
    9.5    Verify that bit 5 is set to 1 (CIC).  (5.3.9#7)
    9.6    Bits 10-31 of HCCPARAMS2 are reserved and shall be set to 0. (5.3.9#5)

## TD.1.03 Register Default Value Test

This test verifies that the memory-mapped I/O registers are set to the default values after reset. It also verifies read and write attributes of each field.

**Required Device Resource**

None

**Overview of Test Steps**

The test performs the following steps.

1 Set the Operational Registers as follows:
   1.1 Set the USBCMD register to FFFFFC7Ch (all ones except the *RS*, *HCRST*, *LHCRST*, *CSS* and *CRS* flags).
   1.2 Set the DNCTRL register to FFFFFFFFh.
   1.3 Set the DCBAAP register to FFFFFFFFFFFFFFFFh.
   1.4 Set the CONFIG register to FFFFFFFFh.
2 Set the Runtime Registers as follows:
   2.1 Set the IMOD register to FFFFFFFFh.
   2.2 Set the ERSTSZ register to FFFFFFFFh.
   2.3 Set the ERDP register to FFFFFFFFFFFFFFFFh.
3 Perform a Host Controller Reset by the following steps:
   3.1 Set the *HCRST* flag to 1 in the USBCMD register.
   3.2 Read the USBCMD register and wait until the *HCRST* flag is cleared to 0. Test fails if the *HCRST* flag is not cleared within 1000 ms. (5.4.1#3)
4 Read the USBCMD register and verify the followings:
   4.1 The *RS* flag is set to 0 (5.4.1#1, 5.4.1#2).
   4.2 The *HCRST* flag is set to 0 (5.4.1#4).
   4.3 The *INTE* flag is set to 0 (5.4.1#6, 5.4.1#7).
   4.4 The *LHCRST* flag is set to 0 (5.4.1#16).
   4.5 The *EWE* flag is set to 0 (5.4.1#12, 5.4.1#13).
   4.6 The *EU3S* flag is set to 0 (5.4.1#14, 5.4.1#15).
5 Read the USBSTS register and verify the followings:
   5.1 The *HCH* flag is set to 1 (5.4.2#4, 5.4.2#5).
   5.2 The *HSE* flag is set to 0 (5.4.2#6, 5.4.2#7).
   5.3 The *EINT* flag is set to 0 (5.4.2#11, 5.4.2#12).
   5.4 The *PCD* flag is set to 0 (5.4.2#16, 5.4.2#17).
   5.5 The *SSS* flag is set to 0 (5.4.2#19, 5.4.2#20).
   5.6 The *RSS* flag is set to 0 (5.4.2#23, 5.4.2#24).
   5.7 The *SRE* flag is set to 0 (5.4.2#26, 5.4.2#27).
   5.8 The *CNR* flag is set to 0 (5.4.2#28).
   5.9 The *HCE* flag is set to 0 (5.4.2#30, 5.4.2#31).
6 Read the DNCTRL register and verify that *Notification Enable* field is set to 0 (5.4.4#1, 5.4.4#2).
7 Read the CRCR register and verify that the *CRR* flag is set to 0 (5.4.5#5, 5.4.5#6).
8 Read the DCBAAP register and verify that it is set to 0 (5.4.6#1, 5.4.6#2).
9 Read the CONFIG register and verify that the *MaxSlotsEn* field is set to 0 (5.4.7#1, 5.4.7#2).
10 Read the runtime registers and verify the followings:
11 Read the MFINDEX register and verify that the *Microframe Index* field is set to 0 (5.5.1#2, 5.5.1#3).
12 Read the IMAN registers and verify the followings:
   12.1 The *IP* flag is set to 0 (5.5.2.1#1, 5.5.2.1#2).
   12.2 The *IE* flag is set to 0 (5.5.2.1#4, 5.5.2.1#5).
13 Read the IMOD registers and verify that the *IMODI* field is set to 4000 (5.5.2.2#1, 5.5.2.2#2).
14 Read the ERSTSZ registers and verify that the *Event Ring Segment Table Size* field is set to 0 (5.5.2.3.1#2, 5.5.2.3.1#3).
15 Read the ERSTBA registers and verify the followings:
   15.1 The *Event Ring Segment Table Base Address* field (6:63 bits for 1.0 only xHC and 4:63 bits for 0.96 only xHC) is set to 0 (5.5.2.3.2#2, 5.5.2.3.2#3).
   15.2 For xHCI 1.0, bits 0:5 are 0 (5.5.2.3.2#4).
16 Read the ERDP registers and verify the followings:
   16.1 The *DESI* field is set to 0 (5.5.2.3.3#1, 5.5.2.3.3#2).
   16.2 The *EHB* flag is set to 0 (5.5.2.3.3#5, 5.5.2.3.3#6).
   16.3 The *Event Ring Dequeue Pointer* field is set to 0 (5.5.2.3.3#7, 5.5.2.3.3#8).
17 If the xHC supports a Light Host Controller Reset, i.e. the *LHRC* flag is 1 in the HCCPARAMS register, repeat the steps from 1 to 17, but perform a Light Host Controller Reset in the step 3 by the following steps:
   17.1 Set the *LHCRST* flag to 1 in the USBCMD register.

17.2 Read the USBCMD register and wait until the *LHCRST* flag is cleared to 0. Test fails if the *LHCRST* flag is not cleared within 100 ms (5.4.1#17).

18 Set the USBSTS register to FFFFFFFEh (all ones except the *HCH* flag), then read the USBSTS register and verify the followings:

18.1 The *HCH* flag is set to 1 (5.4.2#3).

18.2 The *SSS* flag is set to 0 (5.4.2#21).

18.3 The *RSS* flag is set to 0 (5.4.2#25).

18.4 The *CNR* flag is set to 0 (5.4.2#29).

18.5 The *HCE* flag is set to 0 (5.4.2#32).

19 Read the PAGESIZE register and set it to FFFFFFFFh, then read it again and verify that the read value is the same as the original value (5.4.3#1).

20 Set the MFINDEX register to FFFFFFFFh, then read it again and verify that the *Microframe Index* field is set to 0 (5.5.1#1).

## TD.1.04 Command and Status Registers Test

This test verifies the operations of USBCMD and USBSTS registers.

### Required Device Resource

None

### Overview of Test Steps

The test performs the following steps.

1 Reset the xHC by setting the *HCRST* flag to 1 in the USBCMD register and wait until the *HCRST* flag is cleared to 0.

2 Initialize xHC.

3 Prompt the user to attach a USB device to a root hub port if not connected, and enable the port. Reset the port if the device connected is a 2.0 device to drive the ports to enabled state.

4 Run the xHC by the following steps:

4.1 Set the *RS* flag to 1 in the USBCMD register.

4.2 Read the USBSTS register and wait until the *HCH* flag is cleared to 0. Test fails if the *HCH* flag is not cleared to 0 within 100 ms (5.4.2#2).

5 Read the MFINDEX register periodically (every 1 second) and verify that the xHC is incrementing it every 125 us (4.14.2#1).

6 Stop the xHC by the following steps:

6.1 Clear the *RS* flag to 0 in the USBCMD register.

6.2 Read the USBSTS register and wait until the *HCH* flag is set to 1. Test fails if the *HCH* flag is not set to 1 within 100 ms (5.4.2#1).

7 Read the MFINDEX register periodically (every 1 second) and verify that the xHC is not incrementing it (5.5.1#4).

If there are exposed root ports:

8 Prompt user to attach or detach device then verify that there are not any port status change events. (5.4.8#1)

9 Save the current state by the following steps:

9.1 Set the *CSS* flag to 1 in the USBCMD register.

9.2 Read the USBCMD register and verify that the *CSS* flag is 0.

9.3 Read the USBSTS register and wait until the *SSS* flag is cleared to 0. Test fails if the *SSS* flag is not cleared within 100 ms (5.4.2#18).

10 Restore the previous state by the following steps:

10.1 Set the *CRS* flag to 1 in the USBCMD register.

10.2 Read the USBCMD register and verify that the *CRS* flag is 0.

10.3 Read the USBSTS register and wait until the *RSS* flag is cleared to 0. Test fails if the *RSS* flag is not cleared within 100 ms (5.4.2#22).

### TD.1.05 Extended Capabilities Registers Test

This test verifies the values and operations of the xHCI Extended Capability Registers.

**Required Device Resource**

None

**Overview of Test Steps**

The test performs the following steps.

1　Reset the xHC by setting the *HCRST* flag to 1 in the USBCMD register.
2　Get the pointer to the first Extended Capability Register by reading the *xECP* field in the HCCPARAMS register.
3　Write FFh to the *Capabilities ID* field, then read it and verify that it is set to the original value (7#1).
4　Write FFh to the *Next xHCI Extended Capability Pointer* field, then read it and verify that it is set to the original value (7#2).
5　If the *Capabilities ID* field is set to 1, take the following steps:
　5.1 Write the USBLEGCTLSTS register with setting the *SMI on Event Interrupt* and the *SMI on Host System Error* flags to 1.  All other bits in this register will be set to 0.
　5.2 Read the USBLEGCTLSTS register and verify the followings:
　　5.2.1　The *SMI on Event Interrupt* flag is set to 0 (7.1.2#1).
　　5.2.2　The *SMI on Host System Error* flag is set to 0 (7.1.2#3).
　5.3 Verify the xHC's operation on the *SMI on OS Ownership Change* flag as follows:
　　5.3.1　Set the *HC OS Owned Semaphore* flag to 1, then verify that the *SMI on OS Ownership Change* flag is set to 1 (7.1.2#6).
　　5.3.2　Write a 0 to the *SMI on OS Ownership Change* flag, and verify that it is still set to 1 (7.1.2#5).
　　5.3.3　Clear the *SMI on OS Ownership Change* flag by writing 1 to this flag.
　　5.3.4　Clear the *HC OS Owned Semaphore* flag to 0, then verify that the *SMI on OS Ownership Change* flag is set to 1 (7.1.2#6).
　　5.3.5　Write a 0 to the *SMI on OS Ownership Change* flag, and verify that it is still set to 1 (7.1.2#5).
　　5.3.6　Clear the *SMI on OS Ownership Change* flag by writing 1 to this flag.
　5.4 If xHC is PCI-based, verify the xHC's operation on the *SMI on PCI Command* flag as follows:
　　5.4.1　Write the PCI Command Register in the PCI Configuration Space with the current value and verify that the *SMI on PCI Command* flag is set to 1 in the USBLEGCTLSTS register (7.1.2#9).
　　5.4.2　Write a 0 to the *SMI on PCI Command* flag and verify that it is still set to 1 (7.1.2#8).
　　5.4.3　Write a 1 to the *SMI on PCI Command* flag and verify that it is cleared to 0 (7.1.2#7).
　5.5 If xHC is PCI-based, verify the xHC's operation on the *SMI on BAR* flag as follows:
　　5.5.1　Write the Base Address Register in the PCI Configuration Space with the current value and verify that the *SMI on BAR* flag is set to 1 in the USBCTLREG register (7.1.2#12).
　　5.5.2　Write a 0 to the *SMI on BAR* flag and verify that it is still set to 1 (7.1.2#11).
　　5.5.3　Write a 1 to the *SMI on BAR* flag and verify that it is cleared to 0 (7.1.2#10).
6　If the *Capabilities ID* field is set to 2, take the following steps:
　6.1 Read the xHCI Supported Protocol Capability and verify the followings:
　　6.1.1　The *Minor Revision* field is set to 0 if the Major Revision is 2(7.2#2).
　　6.1.2　The Minor Revision field is set to 10 or 0 if the Major Revision is 3 (7.2#29).
　　6.1.3　The *Major Revision* field is set to 3 or 2 (7.2#3).
　　6.1.4　The *Name String* field is set to 20425355h (7.2#4).
　　6.1.5　The *Compatible Port Offset* field is set to a value in the range of 1 to *MaxPorts* (7.2#5).
　　6.1.6　The *Compatible Port Count* field is set to a value in the range of 1 to *MaxPorts* (7.2#6).
　　6.1.7　The *Compatible Port Offset* + *Compatible Port Count* – 1 is less than or equal to *MaxPorts* (7.2#7).
　　6.1.8　The *L1C* flag is set to 0 if the *Major Revision* field is set to 3 (7.2#8).
　　6.1.9　Write *HSO* bit and verify that write operation has no effect (7.2.2.1.3.2#1).
　　6.1.10　Write *IHI* bit and verify that write operation has no effect (7.2.2.1.3.2#2).

6.1.11 Write *HLC* bit and verify that write operation has no effect (7.2.2.1.3.2#3) (xHCI 1.0 only).

6.1.12 The *Protocol Defined* field (bits 27:16) is set to 0 for USB3 protocol (xHCI 1.00).

6.1.13 The *Protocol Defined* field (bits 23:16) is set to 0 for USB3 protocol (xHCI 1.10 and xHCI 1.20).

6.1.14 In the *USB 2.0 Protocol Defined fields* verify that bit 20 is 1 (BLC). (4.23.5.1.1.1#2) (xHCI 1.10)

6.2 Write all 1s to the xHCI Supported Protocol Capability, then read it again and verify that registers are set to the original values (7.2#9).

6.3 In case *HSO* flag is zero in USB 2.0 Supported Protocol Capability and the xHC has at least one removable root USB 2.0 port, attach FS or LS device to a port and verify that *CCS* = 1 and *Port Speed* = FS or LS on at least one removable port (7.2#13).

6.4 In case *HSO* flag is one in USB 2.0 Supported Protocol Capability:

6.4.1 Attach FS or LS device to a port if the xHC has at least one removable port.

6.4.2 Verify that all the connected ports have Port Speed field set to HS (7.2#14).

7 If the *Next xHCI Extended Capability Pointer* field is set to a non-zero value, get the pointer to the next Extended Capability Register and repeat the steps from 3 to 7.

8 Test fails if the xHC has no xHCI Supported Protocol Capability (7.2#1).

9 Test fails if more than one xHCI Supported Protocol Capabilities declare the same port (7.2#10, 7.2.2#1).

10 Test fails if there is no xHCI Supported Protocol Capability with the *Compatible Port Offset* field set to 1 (7.2#11).

11 For xHCI 0.96, verify that *Capability ID* field is not 0, 6-9, 11-16 or 18-191 (7#3).

12 For xHCI 1.0 and 1.10, verify that *Capability ID* field is not 0, 7-9, 11-6 or 18-191 (7#4).

13 Confirm that there are no overlaped USB2/USB3/USB3.1 ports (7.2.2.1#1)

## TD.1.06 PORTPMSC Register Test

This test verifies the PORTPMSC Register behavior.

**Required Device Resource**

LPM-capable USB 2.0 device, USB 3.0 device, USB 3.1 device

**Overview of Test Steps**

The test performs the following steps.

1 Reset the xHC by setting the *HCRST* flag to 1 in the USBCMD register and wait until the *HCRST* flag is cleared to 0.

2 Read the PORTPMSC Register for the USB 2 Port under test.

3 RSVD field testing:

3.1 Bits 17:27 of the PORTPMSC register are 0 for xHC 1.0 and xHC 1.10

3.2 Bits 16:27 of the PORTPMSC register are 0 for xHC 0.96 only

3.3 Write all 1's into bits 16:27 for xHC 0.96 only and into bits 17:27 for xHC 1.0 only and verify that the bits remain 0 (5.4.9.2#9).

If there are exposed 2.0 root ports:

4 Prompt user to attach a L1 capable USB 2.0 device to the 2.0 port under test if no device is attached, then wait for a Port Status Change Event. Test fails if it does not receive a Port Status Change Event.

5 In case of xHCI 1.0 or xHCI 0.96 with L1C bit set:

5.1 Write the PORTSC register with setting the *PP* flag to 1, the *PLS* field to 2 (L1 LPM) and the *LWS* to 1.

5.2 Read PORTSC register after 300ms and verify the followings:

5.2.1 *PP* = 1

5.2.2 CCS = 1

5.2.3 *PED* = 1
5.2.4 *PR* = 0
5.2.5 *PLS* = 2 (L1S) (4.23.5.1.1#2)
5.2.6 *LWS* = 0 (5.4.8#29)
5.3 Read PORTPMSC register.
5.4 Verify that the L1S =1 (Success) indicating that Port successfully transitioned into L1 (ACK) (5.4.9.2#1).
6 Reset the xHC by setting the *HCRST* flag to 1 in the USBCMD register and wait until the *HCRST* flag is cleared to 0.
7 Read PORTPMSC and verify the following:
7.1 L1S = 0 (5.4.9.2#3)
7.2 RWE = 0 (5.4.9.2#4)
7.3 HIRD = 0 (5.4.9.2#5)
7.4 L1 Device Slot = 0 (5.4.9.2#6)
7.5 Port Test Control = 0 (5.4.9.2#7)
8 If the Host supports LHCRST repeat steps 4 through 8 and do LHCRST instead of HCRST and verify that Port Test Control = 0 (5.4.9.2#8)
9 Write L1S=1 in PORTPMSC register and after allowing 300ms delay read back the value to verify that L1S field remains 0 (5.4.9.2#2)
Verify FLA flag behavior on connect:
10 Set the FLA flag to one in all the PORTSC register for USB 3 ports.
11 Prompt a user to attach a SuperSpeed Gen1 device to a USB 3.0 port and wait for connection.
12 Verify that FLA flag is reset to zero (5.4.9.1#6).
13 Set U1 Timeout field and FLA flag to1 in the PORTPMSC register.
14 Issue hot reset by setting the PR flag in the PORTSC register and verify the followings:
14.1 U1 timeout is reset to zero (5.4.9.1#1).
14.2 FLA flag is reset to zero (5.4.9.1#5).
15 Set U1 Timeout field, U2 Timeout field and FLA flag to 1 in the PORTPMSC register.
16 Issue HCRST and verify the followings:
16.1 U1 timeout is reset to zero (5.4.9.1#2).
16.2 U2 timeout is reset to zero (5.4.9.1#3).
16.3 FLA flag is reset to zero (5.4.9.1#4).
17 Set FLA flag to1 in the PORTPMSC register.
18 Clear the PP flag to zero and verify that the FLA flag is reset to zero (5.4.9.1#7).
19 Repetition: Repeat steps 10-18 with a SS Gen2 device, if supported by xHC

## TD.1.07 FLADJ Register Test

This test verifies the host controller's operation on FLADJ register.

### Required Device Resource

SS compliance device (with Bus Interval Adjustment Message support – currently not supported)

### Overview of Test Steps

The test performs the following steps.
1 Connect an SS compliance device.
2 Set the FLADJ register to 21h.
3 Set the DNCTRL register to 8.
4 Make the compliance device to send a Bus Interval Adjustment Message.
5 Wait for a Device Notification Event with Bus Interval Adjustment Message, read the FLADJ register and verify that it is still set to 21h.
6 Issue a host reset, read the FLADJ register and verify that it is still set to 21h.
7 Issue a warm reset, read the FLADJ register and verify that it is still set to 21h.
8 If LHRC flag is 1 in HCCPARAMS register, issue light host controller reset, read the FLADJ register and verify that it is still set to 21h.

### TD 1.08  PORTLI Register Test

This test verifies the host controller's operation on PORTLI register.

#### Required Device Resource

SS compliance device (with link error generation support – currently not supported)

#### Overview of Test Steps

The test performs the following steps.
1.  Connect an SS compliance device.
2.  Make the compliance device to generate link error.
3.  Perform transfer the device.
4.  Issue HCRST, read the PORTLI register and verify that it is reset to zero (5.4.10.1#3).
5.  If LHRC flag is 1 in HCCPARAMS register:
    5.1    Perform transfer to the device.
    5.2    Issue LHCRST, read the PORTLI register and verify that it is reset to zero (5.4.10.1#4).
6.  Set the PORTLI register to a non-zero value, read it and verify that it is set to zero (5.4.10.1#5).
7.  Read all the PORTLI registers for USB 2.0 ports and verify that they are set to zero (5.4.10.2#1).


### TD 1.09  Protocol Speed ID Test

This test verifies the Protocol Speed ID fields are correctly implemented. This test is applicable to xHCI 1.0, 1.1 and 1.2 host controllers.

#### Required Device Resource

LS, FS, HS ,SS ,and SSP device

#### Overview of Test Steps

The test performs the following steps.
1.  Find xHCI Supported Protocol Capability Registers and if the PSIC field is set to a non-zero value, verify the followings for all the PSI fields.
    1.1    PSIV field is not set to 0 (7.2.1#2).
    1.2    PLT field is set to 0 for USB protocol (7.2.1#3).
    1.3    PLT field is not set to 1 (7.2.1#4).
    1.4    PLT field is set to 3 if the previous PSI's PLT field is set to 2 (7.2.1#7).
    1.5    PLT field is not set to 3 if the previous PSI's PLT field is not set to 2 (7.2.1#7).
    1.6    PFD field is 0 for USB 2.0 protocol (7.2.1#8).
    1.7    PFD field is 1 for USB 3.0 and 3.1 protocol (7.2.1#9).
    1.8    For USB 3.1 protocol, PSIC field is 2. For USB 3.0 protocol, PSIC field is 1. For USB 2.0 protocol, PSIC field is 3 if HSO = 0 and 1 if HSO = 1 (7.2#25).
    1.9    If PSIV is 5 and PSIM is 10, verify LP=1 (7.2#28), else LP=0 (7.2#27)

2.  In case the xHC exposes at least one removable port, instruct the user to attach FS, LS, HS, SS, SSP device and wait for Port Status Change Event.
    2.1    If PSIC is 0, Port Speed shall be set to the following values in PORTSC register (7.2.2.1.2#1).
        2.1.1   1 for FS
        2.1.2   2 for LS
        2.1.3   3 for HS
        2.1.4   4 for SS
        2.1.5   5 for SSP
    2.2    If PSIC is not 0, Port Speed shall be set to the value in PSI (7.2.1#1).

**TD 1.10 Extended Property Test**

This test verifies xHC Get Extended Property

**Required Device Resource**

**Overview of Test Steps**

This test applies only to xHC version 1.2

The test performs the following steps.
1. Reset the xHC by setting the *HCRST* flag to 1 in the USBCMD register and wait until the HCRST flag is cleared to 0.
2. Initialize the Event Ring 0 by taking the following steps:
    a. Allocate buffers for Event Ring Segments.
    b. Allocate a buffer for the Event Ring Segment Table (ERST) and initialize it by the addresses and the sizes of the segments.
    c. Set the ERSTSZ register to the number of segments.
    d. Set the ERDP register to the base address of the first Event Ring Segment.
    e. Set the ERSTBA register to the base address of the ERST.
3. Initialize the Command Ring by taking the following steps:
    a. Allocate buffers for Command Ring Segments.
    b. Set the *Command Ring Pointer* field to the base address of the first Command Ring Segment and the *RCS* flag to 1 in the CRCR register.
4. Set the *RS* flag to 1 in the USBCMD register and wait until the *HCH* flag is cleared to 0 in the USBSTS register.
5. Read the HCCPARAMS2 register.
    a. Read bit 8 to determine GSC.
    b. Read bit 9 to determine VTC.
6. Insert a GetExtended Property Command into the Command Ring with these parameters:
    a. Slot ID = 0
    b. EP ID = 0
    c. Subtype = 0
7. Ring the Host Controller Doorbell.

8. Wait for Command Completion Event and verify the following:

    a. Completion code field is set to Success or TRBError (4.6.17#3)

    b. If GSC=1 *Completion Code* must be Success (4.6.17#1)

    c. If GSC=0 *Completion Code* must be TRBError (4.6.17#1)

9. If *Completion Code* = Success, verify that bits 11:1 of Extended Capability Identifier are 0. (4.6.17.3.#3)

## 3.2   TRB Ring Test

**TD.2.01 Command and Event Ring Test**

This test verifies the operations of Command Ring and Event Ring handling.

**Required Device Resource**

None

**Overview of Test Steps**

The test performs the following steps.

1     Reset the xHC by setting the *HCRST* flag to 1 in the USBCMD register and wait until the HCRST flag is cleared to 0.

2     Initialize the Event Ring 0 by taking the following steps:

     2.1     Allocate buffers for Event Ring Segments.

     2.2     Allocate a buffer for the Event Ring Segment Table (ERST) and initialize it by the addresses and the sizes of the segments.

     2.3     Set the ERSTSZ register to the number of segments.

     2.4     Set the ERDP register to the base address of the first Event Ring Segment.

     2.5     Set the ERSTBA register to the base address of the ERST.

3     Initialize the Command Ring by taking the following steps:

     3.1     Allocate buffers for Command Ring Segments.

     3.2     Set the *Command Ring Pointer* field to the base address of the first Command Ring Segment and the *RCS* flag to 1 in the CRCR register.

4     Set the *RS* flag to 1 in the USBCMD register and wait until the *HCH* flag is cleared to 0 in the USBSTS register.

5     Repeat the following steps until both Command Ring and Event Ring wrap around twice, i.e. Cycle State transitions from 1 to 0 and transitions back to 1 again.

     5.1     Insert a No Op Command TRB into the Command Ring and ring the Host Controller Doorbell.

     5.2     Wait for a Command Completion Event and verify the followings:

         5.2.1   A Command Completion Event is received within 100 ms.

         5.2.2   The *Completion Code* field is set to Success (4.6.2#1).

         5.2.3   The *Command TRB Pointer* field is set to the pointer to the No Op Command TRB (4.6.1#4).

         5.2.4   The *Command Completion Parameter* is set to 0 (6.4.2.2#2)

6     Repeat the following steps 3 times:

     6.1     Insert 200 No Op Command TRBs into the Command Ring without ringing the Host Controller Doorbell.

     6.2     Ring the Host Controller Doorbell.

     6.3     Wait for Command Completion Events for the commands and verify the followings:

         6.3.1   Command Completion Events for all the No Op Commands are placed in the Event Ring (4.6.1#1).

         6.3.2   Command Completion Events are placed in order (4.6.1#2).

7     Set the *EWE* flag to 1 in the USBCMD register and verify the followings:

     7.1     MFINDEX Wrap Events within 3 seconds (5.4.1#10).

     7.2     The *Completion Code* field is set to Success in MFINDEX Wrap Events (6.4.2.8#1).

8     Set the *EWE* flag to 0 in the USBCMD register. Test fails if it receives an MFINDEX Wrap Event (5.4.1#11).

9     Clear the *RS* flag to 0 in the USBCMD register and wait until the *HCH* flag becomes 1 in the USBSTS register.

**Repetitions**

Repeat for any possible combination of the following Command Ring and Event Ring structures:

- Command Ring
  - ➢   1 segment: 4096 bytes
  - ➢   2 segments: 144 bytes and 4000 bytes
  - ➢   3 segments: 2096 bytes, 64 bytes and 1280 bytes
- Event Ring
  - ➢   1 segment: 4096 bytes
  - ➢   2 segments: 464 bytes and 3024 bytes

## TD.2.02 Interrupter Test

This test verifies the operations of interrupter handling.

**Required Device Resource**

None

**Overview of Test Steps**

The test performs the following steps.

1   Reset the xHC by setting the *HCRST* flag to 1 in the USBCMD register and wait until the *HCRST* flag is cleared to 0.
2   Initialize the Command Ring with a single 256-byte segment.
3   Initialize the Primary Event Ring with a single 1024-byte segment.
4   Set the *RS* flag to 1 and the *INTE* flag to 1 in the USBCMD register then wait until the *HCH* flag is cleared to 0 in the USBSTS register.
5   If there are integrated devices, clear any port status changes for these devices only.
6   Insert a No Op Command TRB into the command ring, ring the Host Controller Doorbell and verify that the xHC does not generate an interrupt (5.5.2.1#3).
7   Clear the *INTE* flag to 0 in the USBCMD register and set the *IE* flag to 1 in the IMAN register then verify that the xHC does not generate an interrupt if message interrupt is used (5.4.1#5).
8   Set the *INTE* flag to 1 in the USBCMD register and verify the followings:
    8.1   An interrupt is received.
    8.2   The *EINT* flag is set to 1 in the USBSTS register (5.4.2#8).
    8.3   The *EHB* flag is set to 1 in the ERDP register (5.5.2.3.3#3).
    8.4   A Command Completion Event is received in the Event Ring.
    8.5   If the xHC has a USB Legacy Support Capability, the *SMI on Event Interrupt* flag is set to 1 in the USBLEGCTLSTS register (7.1.2#2).
9   Update the *Event Ring Dequeue Pointer* field in the ERDP register without clearing the *EHB* flag.
10  Write the USBSTS register with the *EINT* flag set to 0, read the USBSTS register and verify that it is still set to 1 (5.4.2#9).
11  Write the USBSTS register with the *EINT* flag set to 1, read the USBSTS register and verify that it is cleared to 0 (5.4.2#10).
12  Insert a No Op Command TRB into the command ring, ring the Host Controller Doorbell and verify that it does not receive an interrupt.
13  Clear the *EHB* flag by writing one to this flag in the ERDP register.
14  Wait for an interrupt and verify the followings:
    14.1  An interrupt is received.
    14.2  The *EINT* flag is set to 1 in the USBSTS register (5.4.2#8).
    14.3  The *EHB* flag is set to 1 in the ERDP register (5.5.2.3.3#3).
    14.4  A Command Completion Event is received in the Event Ring.
15  Update the *Event Ring Dequeue Pointer* field and set the *EHB* flag to one in the ERDP register.
16  Read the ERDP register and verify that the *EHB* flag is zero (5.5.2.3.3#4).

## TD.2.03 Command Ring Control Register Test

This test verifies the operations of Command Ring Control Register.

**Required Device Resource**

None

**Overview of Test Steps**

The test performs the following steps.

1   Reset the xHC.
2   Initialize the primary Event Ring and the primary interrupter.
3   Initialize the Command Ring by the following steps:
    3.1   Allocate a Command Ring Segment and initialize it by zero.
    3.2   Set the *Command Ring Pointer* field to the base address of the Command Ring Segment and set the *RCS* flag to 1 in the CRCR register.
    3.3   Read the CRCR register and verify the followings:
        3.3.1  *RCS* = 0 (5.4.5#4)
        3.3.2  *Command Ring Pointer* = 0 (5.4.5#16)
4   Start the xHC by setting the *RS* flag to one in the USBCMD register.

5     Insert a No Op Command TRB (C = 1) into the Command Ring and ring the Host Controller Doorbell,
      then verify that a Command Completion Event is generated.
6     Read the CRCR register and verify that the CRR flag is 1 (5.4.5#13).
7     Set the *Command Ring Pointer* field to the base address of the Command Ring Segment and set the *RCS*
      flag to 1 in the CRCR register.
8     Ring the Host Controller Doorbell and verify that the xHC does not generate a Command Completion
      Event (5.4.5#15).
9     Update the enqueue pointer to the base address of the Command Ring Segment + 64, then set the
      *Command Ring Pointer* field to the enqueue pointer value and set the *RCS* flag to 0 in the CRCR register.
10    Insert a No Op Command TRB (C = 0) into the Command Ring and ring the Host Controller Doorbell,
      then verify that the xHC does not generate a Command Completion Event (5.4.5#1).

## TD.2.04 Command Ring Stop Test

This test verifies the Command Ring stop operations.

**Required Device Resource**

None

**Overview of Test Steps**

The test performs the following steps.

1     Initialize the xHC.
2     Insert a No Op Command into the Command Ring and ring the Host Controller Doorbell then verify that
      the xHC generates a Command Completion Event.
3     Set the *CS* flag to 1 in the CRCR register.
4     Read the CRCR register and verify that the *CS* flag is 0 (5.4.5#8).
5     Wait for a Command Completion Event and verify the followings:
      5.1     *Completion Code* = Command Ring Stopped
      5.2     *TRB Pointer* is the base address of the command ring + 16.
      5.3     *Command Completion Parameter* is 0 (6.4.2.2#2).
6     Read the CRCR register and verify that the *CRR* flag is set to 0 (5.4.5#14).
7     Write a value of 1 to CRR field in CRCR register.
8     Read back the CRCR register and verify that the *CRR* flag remains set to 0 (5.4.5#17).
9     Set the *CS* flag to 1 in the CRCR register and verify that the xHC does not generate a Command
      Completion Event (5.4.5#7).
10    Set the *Command Ring Pointer* field to the base address of the Command Ring Segment and the *RCS* flag
      to 0 in the CRCR register.
11    Change the Cycle Bit of the NO Op Command to 0 to verify Host will update its CCS flag to 0.
12    Change the cycle bits of the other TRBs to 1.
13    Ring the Host Controller Doorbell and verify the following:
      13.1    The xHC generates a Command Completion Event.
      13.2    Command TRB Pointer = the base address of the Command Ring Segment (5.4.5#2).
14    Toggle Cycle bit of the No Op Command back to 1.
15    Set the *CS* flag to 1 in the CRCR register to stop the Ring again.
16    Wait for a Command Completion Event and verify the followings:
      16.1    *Completion Code* = Command Ring Stopped
      16.2    *TRB Pointer* is the base address of the command ring + 16.
17    Change the cycle bits of the other TRBs to 0.
18    Set the *Command Ring Pointer* field to the base address of the Command Ring Segment and the *RCS* flag
      to 1 in the CRCR register.
19    Ring the Host Controller Doorbell and verify that the followings:
      19.1    The xHC generates a Command Completion Event.
20    Command TRB Pointer = the base address of the Command Ring Segment (4.6.1.1#2), (5.4.5#2).Write
      the CRCR register with the *CS* flag set to 0 and verify that the xHC does not generate a Command
      Completion Event (5.4.5#9).
21    Repeat the following steps 8 times.

21.1 Set the *CS* flag to 1 in the CRCR register and verify the followings:
    21.1.1 The xHC generates a Command Completion Event.
    21.1.2 Completion Code = Command Ring Stopped (4.6.1.1#3) (5.4.5#3)
    21.1.3 The Command Ring Pointer field = the current command ring pointer (4.6.1.1#4).
21.2 Insert a No Op Command TRB into the Command Ring, ring the Host Controller Doorbell then verify that the xHC generates a Command Completion Event and the Command TRB Pointer field points to the No Op Command TRB (4.6.1.1#1).

## TD.2.05 Command Ring Abort Test

This test verifies the Command Ring abort operations.

**Required Device Resource**

SS device**Overview of Test Steps**

The test performs the following steps.

1 Initialize the xHC.
2 Insert a No Op Command into the Command Ring and ring the Host Controller Doorbell then verify that the xHC generates a Command Completion Event.
3 Set the *CA* flag to 1 in the CRCR register.
4 Read the CRCR register and verify that the *CA* flag is 0 (5.4.5#11).
5 Wait for a Command Completion Event and verify the followings:
    5.1 *Completion Code* = Command Ring Stopped (4.6.1.2#1)
    5.2 *TRB Pointer* is the base address of the command ring + 16.
    5.3 *Command Completion Parameter* = 0 (6.4.2.2#2).
6 Read the CRCR register and verify that the *CRR* flag is set to 0 (5.4.5#14).
7 Set the *CA* flag to 1 in the CRCR register and verify that the xHC does not generate a Command Completion Event (5.4.5#10).
8 Set the *Command Ring Pointer* field to the base address of the Command Ring Segment and the *RCS* flag to one in the CRCR register.
9 Ring the Host Controller Doorbell and verify that the followings:
    9.1 The xHC generates a Command Completion Event.
    9.2 *Command TRB Pointer* = the base address of the Command Ring Segment (4.6.1.1#2).
10 Write the CRCR register with the *CA* flag set to 0 and verify that the xHC does not generate a Command Completion Event (5.4.5#12).

## TD.2.06 Event Ring Size Change Test

This test verifies the event ring size change operations.

**Required Device Resource**

None

**Overview of Test Steps**

The test performs the following steps.

1 Initialize the xHC.
2 Initialize the Command Ring.
3 Initialize the Event Ring with a single segment as follows:
    3.1 Allocate a 256-byte buffer for an Event Ring Segment and initialize it by zero.
    3.2 Allocate a 32-byte buffer for an Event Ring Segment Table.
    3.3 Initialize the Event Ring Segment Table for segment 0 as follows:
        3.3.1 Set the Event Ring Base Address field to the base address of the segment.
        3.3.2 Set the Ring Segment Size field to 16.
    3.4 Set the ERSTSZ register to 1.
    3.5 Set the ERDP register to the base address of the Event Ring Segment.
    3.6 Set the ERSTBA register to the base address of the Event Ring Segment Table.
4 If there are integrated devices, count the number of events on the event ring.

5    Perform 17 (minus events counted in step 4) No Op Commands.
6    Add a new segment as follows.
    6.1    Allocate a 256-byte buffer and initialize it by zero.
    6.2    Initialize the Event Ring Segment Table for segment 1 as follows:
        6.2.1   Set the Event Ring Base Address field to the base address of the segment.
        6.2.2   Set the Ring Segment Size field to 16.
    6.3    Set the ERSTSZ register to 2.
7    Perform 16 No Op Commands.
8    If the Nop Ops Command Completion Events are received in segment 0, perform 16 No Op Commands
9    Verify that the last Command Completion Event is received in the Segment 1.
10   Remove the Segment 1 by setting the ERSTSZ register to 1.
11   Perform 15 No Op Commands and verify that Command Completion Events are received in the Segment 1 (4.9.4.2#1).
12   Perform 17 No Op Commands.
13   Verify that the last Command Completion Event is received in the Segment 0.

### TD.2.07 Event Ring Full Test

This test verifies the event ring size change operations.

**Required Device Resource**

HS Device

**Overview of Test Steps**

The test performs the following steps.

1    Initialize the xHC.
2    Create a Command Ring.
3    Initialize the Event Ring 0 with a single segment for 16 event TRBs.
4    If there are integrated devices, count the number of events on the event ring.
5    Insert 16 No Op Command TRBs into the Command Ring and then ring the Host Controller Doorbell.
6    Verify that 15 (minus the events counted in step 3) Command Completion Events for No Op Commands are received and then a Host Controller Event is placed with the *Completion Code* set to Event Ring Full Error (4.9.4#2).
7    Update the Dequeue Pointer by writing the ERDP register and verify that the Command Completion Event is received for the last No Op Command.
8    Dequeue all the 16 events on the Event Ring. This action will empty the Event Ring and update the ERDP at the same time.

    If there are no exposed ports, skip the remainder of test.

9    Prompt the user to attach a USB HS device to root hub port if not connected, and enable the port.
10   Perform an Enable Slot Command.
11   Initialize a Control Transfer Ring.
12   Perform an Address Device Command with BSR=0.
13   Count the number of events that got generated on the Event Ring (count all events, not just the port status change events).
14   Insert 16 No Op Command TRBs minus the events counted in step 13 into the Command Ring and then ring the Host Controller Doorbell.
15   Verify that 15 (minus the events counted in step 13) Command Completion Events for No Op Commands are received and then a Host Controller Event is placed with the *Completion Code* set to Event Ring Full Error (4.9.4#2) (4.9.4#4)
16   Issue a control transfer to GetStatus() from the device.
17   Read the Event on the Event ring and verify it is a Host Controller Event with Event Right Full Error and not the transfer completion event for GetStatus() (4.9.4#4)

### TD.2.08 Transfer Ring Test

This test verifies the operations of transfer ring.

**Required Device Resource**

USB device

**Overview of Test Steps**

The test performs the following steps.

1   Initialize the xHC
2   Prompt the user to attach a USB device to a root hub port if not connected, and enable the port.
3   Perform an Enable Slot Command.
4   Initialize a Transfer Ring.
5   Perform an Address Device Command with the *BSR* flag set to 1.
6   Insert a No Op TRB into the Transfer Ring with the *IOC* flag set to 1 and ring the doorbell.
7   Test fails if it does not receive a Transfer Event within 1000 ms (4.10.1#1).
8   Insert a No Op TRBs into the Transfer Ring with the *IOC* flag set to 0 and ring the doorbell.
9   Test fails if a Transfer Event is notified (4.10.1#4).
10  Insert 10 No Op TRBs into the Transfer Ring with the *IOC* flag set to 1 and ring the doorbell.
11  Wait for Transfer Events and verify the followings:
    11.1   10 Transfer Events are placed in the Event Ring (4.9.2.2#1).
    11.2   The Transfer Events are placed in order (4.9#1).
12  Repeat the above steps by changing the Interrupter Target.

### TD.2.09 Secondary Event Ring Test

This test verifies the operations of transfer ring.

**Required Device Resource**

USB device

**Overview of Test Steps**

The test performs the following steps by changing the Secondary Event Ring to be tested.

1   Initialize the xHC
2   Attach a USB device to a downstream port.
3   Perform an Enable Slot Command.
4   Set the *Interrupter Target* field to the Secondary Interrupter value in the Input Slot Context.
5   Initialize a Transfer Ring.
6   Perform an Address Device Command with the *BSR* flag set to 1.
7   Initialize a Secondary Event Ring and enable it.
8   Enable the Secondary Interrupter by setting the *IE* flag to 1 in the IMAN register.
9   Insert No Op TRBs with the *IOC* flag set to 1 and the *Interrupter Target* field identifying the secondary interrupter.
10  If message interrupt (MSI or MSI-X) is enabled, verify that the xHC generates an interrupt from the interrupter specified by the *Interrupter Target* field.
11  Verify that Transfer Events are received in the target Event Ring (4.9.4.3#1).
12  Only Transfer Events, Bandwidth Request Events, Device Notification Events and Vendor Defined Events are allowed in the Secondary Event Ring.  Test fails if any other events are generated (4.9.4.3#5)
13  Insert No Op TRBs with the *IOC* flag set to 1 and the *Interrupter Target* field set to 0.
14  Verify that Transfer Events are received in the Event Ring 0 (4.9.4.3#2).
15  Issue a No Op Command and verify that the Command Completion Event is received in the Event Ring 0 (4.6#2).
16  Only Transfer Events, Bandwidth Request Events, Device Notification Events and Vendor Defined Events are allowed in the Secondary Event Ring.  Test fails if any other events are generated (4.9.4.3#5)
17  Disable the Secondary Event Ring by setting the ERSTSZ register to 0.

**Repetitions**

Repeat for all the assigned interrupters supported by the xHC.

## TD.2.10 Event Data TRB Test

This test verifies the operations of Event Data TRB handling.

### Required Device Resource

USB device

### Overview of Test Steps

The test performs the following steps:

1  Initialize the xHC.
2  Attach a USB device to a port.
3  Perform an Enable Slot Command.
4  Perform an Address Device Command with the *BSR* flag set to 1.
5  Insert following TDs into the Transfer Ring and ring the doorbell.
    5.1    Setup Stage TD
        5.1.1  Setup Stage TRB: *bmRequestType* = 80h, *bReqeust* = 6, *wValue* = 0100h, *wIndex* = 0,
            *wLength* = 4096, *IOC* = 0.
    5.2    Data Stage TD
        5.2.1  Data Stage TRB: *TRB Transfer Length* = 4096, *ENT* = 1, *ISP* = 0, *CH* = 1, *IOC* = 0, *DIR* =
            1.
        5.2.2  Event Data TRB: *ENT* = 0, *CH* = 0, *IOC* = 1, *Event Data (Lo)* = 01234567h, *Event Data*
            *(Hi)* = 89ABCDEFh.
    5.3    Status Stage TD
        5.3.1  Status Stage TRB: *ENT* = 1, *CH* = 1, *IOC* = 0, *DIR* = 0
        5.3.2  Event Data TRB: *ENT* = 0, *CH* = 0, *IOC* = 1, *Event Data (Lo)* = 02468ACEh, *Event Data*
            *(Hi)* = 13579BDFh.
6  Wait for Transfer Events and verify the followings:
    6.1    2 Transfer Events are received.
    6.2    For the 1st Transfer Event:
        6.2.1  *ED* = 1 (4.11.5.2#6)
        6.2.2  *TRB Transfer Length* = 18 (4.11.5.2#5).
        6.2.3  *Completion Code* = Short Packet (4.11.5.2#7).
        6.2.4  *Parameter* field = 01234567h (Lo) and 89ABCDEFh (Hi) (4.11.5.2#9).
    6.3    For the 2nd Transfer Event:
        6.3.1  *ED* = 1 (4.11.5.2#6)
        6.3.2  *TRB Transfer Length* = 0 (4.11.5.2#3)
        6.3.3  *Completion Code* = Success (4.11.5.2#8)
        6.3.4  *Parameter* field = 02468ACEh (Lo) and 13579BDFh (Hi) (4.11.5.2#9).
7  Repeat the steps 5 and 6 with changing the *wLength* and TRB Transfer Length to 18. In the 1st Transfer
   Event, verify that *Completion Code* = Success.
8  Insert a TD which only contains an Event Data TRB with the *IOC* flag set to 1.
9  Wait for a Transfer Event and verify that the followings:
    9.1    *TRB Transfer Length* = 0 (4.11.5.2#1)
    9.2    *Completion Code* = Success (4.11.5.2#8)
10 Construct the following TD to verify EDTLA resets to 0 after generating an Event Data Transfer Event:
    10.1   TRB1 with Chain bit=1, with buffer size=1K
    10.2   Event Data TRB with Chain bit =1, IOC=1
    10.3   TRB2 with Chain bit=1 with buffer size=1K
    10.4   Event Data TRB with Chain bit =0, IOC=1
11 Verify the following:
    11.1   1st Event Data Transfer event will report a transferred length of 1K (4.11.5.2#4)
    11.2   2nd Event Data Transfer event will report a transferred length of 1K (4.11.5.2#4)

**Repetitions**

None

## TD.2.11 Command Ring Link TRB Test

This test verifies the operations of Link TRB handling in the Command Ring.

**Required Device Resource**

None

**Overview of Test Steps**

The test performs the following steps by changing the Secondary Event Ring to be tested.

1. Initialize the xHC.
2. Insert a Link TRB with the *IOC* flag set to 1 and the *CH* flag set to 1 into the Command Ring and ring the Host Controller Doorbell.
3. Wait for a Command Completion Event and verify the followings:
   3.1. A Command Completion Event is received within 100 ms (4.11.5.1#8).
   3.2. A Command Completion Event is received on the Primary Event Ring (4.11.5.1#7).
   3.3. The *Slot ID* field is set to 0 (4.11.5.1#9).
   3.4. The *Completion Code* field is set to Success (4.11.5.1#10).
4. Insert a Link TRB with the *IOC* flag set to 0 and the *CH* flag set to 1 into the Command Ring and ring the Host Controller Doorbell.
5. Verify that a Command Completion Event is not received for the Link TRB.
6. Only Transfer Events, Bandwidth Request Events, Device Notification Events and Vendor Defined Events are allowed in the Secondary Event Ring. Test fails if any other events are generated (4.9.4.3#5)

**Repetitions**

Repeat for *CH* flag value 0 and 1.
And repeat for all the *Interrupter Target* values.

## TD.2.12 Transfer Ring Link TRB Test

This test verifies the operations of Link TRB handling in the Command Ring.

**Required Device Resource**

USB device

**Overview of Test Steps**

The test performs the following steps by changing the Secondary Event Ring to be tested.

1. Initialize the xHC. The *MaxSlotsEn* field of the CONFIG register is set to the *MaxSlots* field value in the HCSPARAMS1 register.
2. Prompt a user to attach a USB device to a root hub port if it is not attached.
3. Perform an Enable Slot Command and an Address Device Command (*BSR* = 0).
4. Perform a Configure Endpoint Command.
5. Insert a TD with the following TRBs.
   5.1. Link TRB with the *IOC* flag set to 1 and the *CH* flag set to 1.
   5.2. No Op TRB with the *IOC* flag set to 0 and the *CH* flag set to 0.
6. Wait for a Transfer Event and verify the followings:
   6.1. A Transfer Completion Event is received within 100 ms on an Event Ring specified by the *Interrupter Target* field (4.11.5.1#2).
   6.2. The *Slot ID* field identifies the current slot (4.11.5.1#3).
   6.3. The *Endpoint ID* field identifies the current endpoint (4.11.5.1#4).
   6.4. The *Length* field is set to 0 (4.11.5.1#5).
   6.5. The *Completion Code* field is set to Success (4.11.5.1#6).

6.6    Only Transfer Events, Bandwidth Request Events, Device Notification Events and Vendor Defined Events are allowed in the Secondary Event Ring.  Test fails if any other events are generated (4.9.4.3#5)

7    Repeat the above steps by changing *Endpoint ID* and *Interrupter Target* values.
8    Reset the root hub port.
9    Perform a Reset Device Command.
10   Repeat the steps 3 through 9 without disabling the slot.

**Repetitions**

Repeat for all the possible *Slot IDs* supported by the xHC (1 to *MaxSlots*).
And repeat for all the non-default *Endpoint ID* values (2 to 31).
And repeat for all the *Interrupter Target* values.

## TD.2.13 Test Removed


## TD.2.14 Test Deprecated



## TD.2.15 Interrupt Blocking Test

This test verifies the xHC's operations of interrupt blocking functions. This test is only applicable to xHCI 1.0 host controllers.

**Required Device Resource**

USB compliance device

**Overview of Test Steps**

The test performs the following steps.

1    Initialize the xHC.
2    Prompt a user to attach a USB compliance device to any port if it is not connected.
3    Enumerate the device. The Interrupter Target field is set to 0 in the Slot Context.
4    Configure the device with including an Isochronous endpoint and a Bulk or Interrupt endpoint.
5    Issue a transfer to Isoch OUT endpoints with the following TD settings.
     5.1    Isoch TRB (IOC = 1, BEI = 1, Interrupter Target = 1)
6    After 100 ms, verify the following:
     6.1    A Transfer Event is received but no interrupt is generated from the Interrupter specified by the Interrupter Target (6.4.1.3#2).
     6.2    An Isoch Ring Underrun Event is received and interrupt is generated from the Interrupter specified by the Interrupter Target field in the Slot Context (4.17.5#1).
7    Repeat the steps 5 and 6 with the following TD settings (6.4.1.1#2).
     7.1    Isoch TRB (IOC = 0, BEI = 0)
     7.2    Normal TRB (IOC = 1, BEI = 1, Interrupter Target = 1)
8    Repeat the steps 5 and 6 with the following TD settings (6.4.4.2#2).
     8.1    Isoch TRB (IOC = 0, BEI = 0)
     8.2    Normal TRB (IOC = 0, BEI = 0)
     8.3    Event Data TRB (IOC = 1, BEI = 1, Interrupter Target = 1)
9    Repeat the steps from 5 to 8 with setting the BEI flag to zero and verify that interrupt is generated from the Interrupter specified by the Interrupter Target (6.4.1.3#1, 6.4.1.1#3, 6.4.4.2#1).
10   Configure the compliance device's Bulk or Interrupt endpoint  to always return STALL
11   Issue a transfer to the Bulk or Interrupt endpoint with a Normal TRB (IOC = 1, BEI = 1).
12   After one second, verify that interrupt is received specified by the Interrupter Target field (4.17.5#2).
13   Repeat for all supported interrupters except the Primary Interrupter.

### TD.2.16 Partial TD

This test verifies the xHC's operations regarding partial TDs.

**Required Device Resource**

USB compliance device

**Overview of Test Steps**

The test performs the following steps.

1 Initialize the xHC.
2 Connect a compliance device with Bulk OUT endpoint.
3 Create the following TRBs:
   3.1    Normal TRB (Length = 1024, IOC = 0, CH = 1)
   3.2    Normal TRB (Length = 1024, IOC = 1, CH = 1)
4 Ring the doorbell.
5 Wait for a Transfer Event.
6 Issue a Stop Endpoint Command to assure that xHC updates the dequeue pointer in the endpoint context.
7 Verify that the dequeue pointer indicates the current enqueue pointer in the Endpoint Context (4.9.1#1).
8 Create the remaining TRBs:
   8.1    Normal TRB (Length = 1024, IOC = 0, CH = 1)
   8.2    Normal TRB (Length = 1024, IOC = 1, CH = 0)
9 Ring the doorbell.
10 Wait for Transfer Event.

### TD.2.17 EOB Test

This test verifies the xHC's handling of EOB.

**Required Device Resource**

SS compliance device, bus analyzer.

**Overview of Test Steps**

The test performs the following steps.

1 Initialize the xHC.
2 Connect an SS compliance device and configure it with loopback pair of bulk OUT and bulk IN endpoints.
3 Issue a bulk OUT transfer (length = 500 bytes).
4 Start analyzer.
5 Insert two TDs (length = 1024 bytes) into transfer ring of bulk IN endpoint then ring the doorbell.
6 Wait for transfer event for the first TD. Test fails if no transfer event is received or Completion Code is not Short Packet.
7 Stop analyzer.
8 Verify that the xHC did not send ACK packet after receiving DP with EOB flag (4.9.1#3).

### TD.2.18 Isoch Error Completion Test

This test verifies the xHC's handling of error completion in Isoch transfers.

**Required Device Resource**

USB compliance device

**Overview of Test Steps**

The test performs the following steps.

1 Initialize the xHC.
2 Connect a compliance device and configure an Isoch IN endpoint with invalid CRC.

3    Insert Isoch TDs in the transfer ring for the Isoch IN endpoint then ring the doorbell.
4    Verify that transfer events are generated with USB Transaction Error (4.10.2.3#11 for USB 2.0, 4.10.2.3#12 for USB 3.0).
5    Verify that EP state is not Halted in the endpoint context (4.10.2#1).

## TD.2.19 PID Mismatch Test

This test verifies the xHC's handling PID mismatch.

### Required Device Resource

USB compliance device – currently not supported

### Overview of Test Steps

The test performs the following steps.

1    Initialize the xHC.
2    Connect a compliance device and configure a loopback pair of Bulk OUT and Bulk IN endpoints. Also configure it so that it returns two DATA0 packets in the consecutive transactions.
3    Issue a Bulk OUT transfer of 1000-byte length.
4    Start analyzer.
5    Issue a Bulk IN transfer of 1000-byte length.
6    Stop analyzer.
7    Verify that transfer does not result in Babble Detected Error (4.10.2.4.1#1)
8    Verify that the xHC sends ACK after receiving the second DATA0 packet (4.10.2.4.1#2).

## TD 2.20  Evaluate Next TRB Test

This test verifies the host controller correctly understands ENT flag in the Transfer TRB and processes the following Event Data TRB.

### Required Device Resource

SS compliance device, it shall set EOB to one in the last DP of an IN transfer so that endpoint will be in flow control.

### Overview of Test Steps

The test performs the following steps.

1    Connect an SS compliance device.
2    Configure the compliance device with loopback pair of Bulk OUT and Bulk IN endpoint.
3    Issue a 2000-byte Bulk OUT transfer.
4    Generate a TD for a Bulk IN transfer as follows:
    4.1    Normal TRB (Lengt=4096, IOC=0, CH=1, ENT=1)
    4.2    Link TRB (CH=1)
    4.3    Event Data TRB (IOC=1, CH=0)
5    Verify that transfer event is received for the Event Data TRB (4.12.3#3).

## TD.2.21 Short Packet PAE Test

This test verifies the host controller's operation on PAE flag.

### Required Device Resource

SS, HS and FS compliance device.

### Overview of Test Steps

The test performs the following steps.

1.    Initialize the xHC.

2. Connect a compliance device.
3. Get the Parse All Event Data flag by reading the PAE field in the HCCPARAMS1 register.
4. Configure the compliance device for bulk loopbacks.
5. Create the following TRBs for OUT:
   5.1. Bulk-Out Normal TRB
       5.1.1. Length = MaxPacketSize – 1
       5.1.2. IOC=0
       5.1.3. ChainBit=1
   5.2. Event Data TRB
       5.2.1. IOC=1
       5.2.2. ChainBit=0

6. Ring the doorbell.
7. Create the following TRBs for IN:
   7.1. Bulk-In Normal TRB
       - Length = MaxPacketSize
       - IOC=0
       - ChainBit-1
       - ISP-0
   7.2. Event Data TRB
       - IOC=0
       - ChainBit=1


   7.3. Bulk-In Normal TRB
       - Length = MaxPacketSize
       - IOC=0
       - ChainBit-1
       - ISP-0
   7.4. Event Data TRB
       - IOC=1
       - ChainBit=0
8. Ring the doorbell for both transfer rings.
9. Wait for the Transfer Events.
10. If PAE is set, verify that 2 Transfer Events were received for the IN endpoint (4.10.1.1#3)
    – Transfer length should be MaxPacketSize – 1 for first event.
    - Transfer length should be 0 for second event.
11. If PAE is clear, verify that 1 Transfer Event was received for the IN endpoint (4.10.1.1#4)
       - Transfer length should be MaxPacketSize – 1 for the event.




## TD.2.22 Stopped-Short Packet Test

This test verifies the host controller's ability to generat a Completion Code of Stopped - Short Packet.

### Required Device Resource

SS compliance device.

### Overview of Test Steps

The test performs the following steps.

1. Initialize the xHC.
2. Connect an SS compliance device.
3. Read the HCCPARAMS1 register.

3.1. Get the Stopped - Short Packet Capability flag from SPC field.

4. Create the following TRBs:
   4.1. Bulk-Out transfer normal TRB (Length = Max Packet Size - 1, IOC = 0, ChainBit = 1)
   4.2. Event Data TRB (IOC = 1, ChainBit = 0)
5. Ring the doorbell.
6. Create the following TRBs:
   6.1. Bulk-In transfer normal TRB (Length = Max Packet Size, IOC = 0, ChainBit = 1, ISP=0)
7. Ring the doorbell.
8. Wait for the Transfer Events.
9. Verify that the PAE flag indicates the host controller behavior after a Short Packet.
   9.1. If bit 9 (SPC) of HCCPARAMS is set, the Completion Code shall be Stopped - Invalid Length (4.6.9#20).
   9.2. If bit 9 (SPC) of HCCPARAMS is clear, the Completion Code shallbe Stopped (4.6.9#2).
   9.3. The TRB Transfer Length field of the Transfer Event shall contain the value of the EDLTA (4.11.5.2#11).
10. Create the following TRBs:
    10.1. Bulk-Out transfer normal TRB (Length = Max Packet Size - 1, IOC = 0, ChainBit = 1)
    10.2. Event Data TRB (IOC = 1, ChainBit = 0)
11. Ring the doorbell.
12. Create the following TRBs:
    12.1. 16 Bulk-In transfer normal TRB (Length = Max Packet Size, IOC = 0, ChainBit = 1, ISP=0)
    12.2. Link TRB with ChainBit = 1, ToggleCycle = 0.pointing to the first Bulk IN TRB
13. Ring the doorbell.
14. Wait for the Transfer Events.
15. Verify that the PAE flag indicates the host controller behavior after a Short Packet.
    15.1. If bit 9 (SPC) of HCCPARAMS is set, the Completion Code shall be Stopped - Short Packet (5.3.6#10).
    15.2. If bit 9 (SPC) of HCCPARAMS is clear, the Completion Code shallbe Stopped (4.6.9#2).
    15.3. The TRB Transfer Length field of the Transfer Event shall contain the value of the EDLTA (4.11.5.2#11).

### TD.2.23 U3 Entry Capability Test

Test support for U3 entry port status change.

#### Required Device Resource

LS, FS, HS, SS Gen 1 and SS Gen 2 devices

#### Overview of Test Steps

The test performs the following steps.

1   Initialize the xHC.
2   Attach a device.
Verify behavior with U3EntryEnable clear:
3   Read the Config register.
4   Set bit 8 (U3EntryEnable) to 0.
5   Suspend the attached device:
    5.1    Set PP = 1
    5.2    Set PLS = 3
    5.3    Set LWS = 1
    5.4    Write to the PORTSC register
6   Resume the port:
    6.1    Set PP=1
    6.2    Set PLS = 15
    6.3    Set LWS = 1
    6.4    Write to the PORTSC register
7   Verify that port resumed to U0.
8   Verify that no port status change occurred.

If U3C is supported, verify behavior with U3EntryEnable set:
9   Read the Config register.
10   Set bit 8 (U3EntryEnable) to 1.
11   Suspend the device:
    11.1   Set PP = 1
    11.2   Set PLS = 3
    11.3   Set LWS = 1
    11.4   Write to PORTSC register
12   Wait for port status change event
13   Verify that port status change occurred.(5.3.9#3, 5.4.7#4)
14   Resume the port:
    14.1   Set PP=1
    14.2   Set PLS = 15
    14.3   Set LWS = 1
    14.4   Write to the PORTSC register
15   Verify that port resumed go U0.
16   Verify that no port status change occurred.

17   Clear port status change.
18   Repeat for all speed devices. (LS, FS, HS, SS Gen 1, SS Gen 2)


## 3.3   Root Hub Test

### TD.3.01 SuperSpeed Device Attach and Detach Test

This test verifies that the xHC correctly detects device attach and detach on SuperSpeed ports.

**Required Device Resource**

USB 3.0 device at root port only, not downstream of integrated hubs.

**Overview of Test Steps**

The test performs the following steps with no device attached to SuperSpeed ports.

1   Initialize the xHC.
2   Prompt a user to attach a device to a port if it is not attached.
3   Wait for a Port Status Change Event and verify the followings:
    3.1   A Port Status Change Event is received (4.3#3).
    3.2   The *Port ID* field is set to a valid port number of a USB3 port (4.19.3#1).
    3.3   The *Completion Code* field is set to Success (4.19.3#2).
4   Read the PORTSC register specified by the *Port ID* field in the Port Status Change Event and verify the followings:
    4.1   *PP* = 1
    4.2   *CCS* = 1 (4.3#1)
    4.3   *PED* = 1 (4.3#4)
    4.4   *PR* = 0
    4.5   *PLS* = 0 (U0) (4.3#6)
    4.6   *CSC* = 1 (4.3#2)
    4.7   *Port Speed* = 4 (SS) (4.3.1#1)
5   Read the USBSTS register and verify that the *PCD* flag is set to 1 (5.4.2#13).
6   Write the USBSTS register with setting the *PCD* flag to 0, then read the USBSTS register again and verify that the *PCD* flag is still set to 1 (5.4.2#14).
7   Write the USBSTS register with setting the *PCD* flag to 1, then read the USBSTS register again and verify that the *PCD* flag is cleared to 0 (5.4.2#15).

8    Write the PORTSC register with the *PP* flag set to 1 and all the other fields set to 0, then read the
     PORTSC register again and verify the followings:
     8.1    *CCS* = 1 (5.4.8#3)
     8.2    *PED* = 1 (5.4.8#7)
     8.3    *CSC* = 1 (5.4.8#32).
9    Write the PORTSC register with the *PP* flag set to 1 and the *CSC* flag set to 1, then read the PORTSC
     register again and verify that the *CSC* flag is cleared to 0 (5.4.8#33).
10   Detach the device from the port.
11   Wait for a Port Status Change Event and verify the followings:
     11.1    A Port Status Change Event is received (4.3#3).
     11.2    The *Port ID* field is set to the port number of the current port (4.19.3#1).
     11.3    The *Completion Code* field is set to Success (4.19.3#2).
12   Read the PORTSC register and verify the followings:
     12.1    *PP* = 1
     12.2    *CCS* = 0 (4.4#1)
     12.3    *PED* = 0 (5.4.8#6)
     12.4    *PR* = 0
     12.5    *PLS* = 5 (RxDetect) (4.19.1.1.2#2)
     12.6    *CSC* = 1 (4.4#2)
     12.7    *Port Speed* = 0 (Undefined) (5.4.8#23)
13   Read the USBSTS register and verify that the *PCD* flag is set to 1 (5.4.2#13).
14   Write the USBSTS register with setting the *PCD* flag to 1, then read the USBSTS register again and
     verify that the *PCD* flag is cleared to 0 (5.4.2#15).
15   Write the PORTSC register with the *PP* flag set to 1 and the *CSC* flag set to 1, then read the PORTSC
     register again and verify that the *CSC* flag is cleared to 0 (5.4.8#33).
     Repetitions:
     For 1.10 xHC, repeat with SS Gen2 device.  (Port Speed = 5)


**Repetitions**

Repeat for all the USB3 ports exposed by the xHC.

## TD.3.02 USB 2.0 Device Attach and Detach Test

This test verifies that the xHC correctly detects device attach and detach on USB 2.0 ports.

**Required Device Resource**

USB 2.0 devices at root ports only, not downstream of integrated hubs.

**Overview of Test Steps**

The test performs the following steps with no device attached to USB 2.0 ports.

1    Initialize the xHC.
2    Prompt a user to attach a device to a port if it is not attached.
3    Wait for a Port Status Change Event and verify the followings:
     3.1    A Port Status Change Event is received (4.3#3).
     3.2    The *Port ID* field is set to a valid port number of a USB2 port (4.19.3#1).
     3.3    The *Completion Code* field is set to Success (4.19.3#2).
4    Read the PORTSC register specified by the *Port ID* field in the Port Status Change Event and verify the
     followings:
     4.1    *PP* = 1
     4.2    *CCS* = 1 (4.3#1)
     4.3    *PED* = 0 (4.3#7)
     4.4    *PR* = 0
     4.5    *PLS* = 7 (Polling) (4.3#8)
     4.6    *CSC* = 1 (4.3#2)
5    Read the USBSTS register and verify that the *PCD* flag is set to 1 (5.4.2#13).

6  Write the USBSTS register with setting the *PCD* flag to 1, then read the USBSTS register again and verify that the *PCD* flag is cleared to 0 (5.4.2#15).

7  Write the PORTSC register with setting the *PP* flag to 1 and the *CSC* flag to 1, then read the PORTSC register again and verify that the *CSC* flag is cleared to 0 (5.4.8#33).

8  Write the PORTSC register with setting the *PP* flag to 1 and the *PR* flag to 1.

9  Wait for a Port Status Change Event and verify the followings:
    9.1  A Port Status Change Event is received within 200 ms (4.3#13).
    9.2  The *Port ID* field is set to a valid port number of a USB2 port (4.19.3#1).
    9.3  The *Completion Code* field is set to Success (4.19.3#2).

10  Read the PORTSC register and verify the followings:
    10.1  *PP* = 1
    10.2  *CCS* = 1
    10.3  *PED* = 1 (4.3#10)
    10.4  *PR* = 0 (4.3#12)
    10.5  *PLS* = 0 (U0) (4.3#11)
    10.6  *PRC* = 1 (4.3#9)
    10.7  *Port Speed* = 1 (FS), 2 (LS) or 3 (HS) depending on the attached device. (4.3.1#1)

11  Write the PORTSC register with setting the *PP* flag to 1 and the *PRC* flag to 1, then read the PORTSC register again and verify that the *PRC* flag is cleared to 0 (5.4.8#72).

12  Detach the device from the port.

13  Wait for a Port Status Change Event. Test fails if a Port Status Change Event is not received.

14  Read the PORTSC register and verify the followings:
    14.1  *PP* = 1
    14.2  *CCS* = 0 (4.4#1)
    14.3  *PED* = 0 (5.4.8#6)
    14.4  *PR* = 0
    14.5  *PLS* = 5 (RxDetect) (4.19.1.1.2#2)
    14.6  *CSC* = 1 (4.4#2)
    14.7  *Port Speed* = 0 (Undefined) (5.4.8#23)

15  Read the USBSTS register and verify that the *PCD* flag is set to 1.

16  Write the USBSTS register with setting the *PCD* flag to 1, then read the USBSTS register again and verify that the *PCD* flag is cleared to 0 (5.4.2#15).

17  Write the PORTSC register with setting the *PP* flag to 1 and the *CSC* flag to 1, then read the PORTSC register again and verify that the *CSC* flag is cleared to 0 (5.4.8#33).

**Repetitions**

Repeat for all the USB2 root ports exposed by the xHC.
Repeat for HS, FS and LS devices.

**TD.3.03 Port Power Test**

This test verifies that the xHC correctly controls port power.

**Required Device Resource**

USB device

**Overview of Test Steps**

The test performs the following steps for each port.

1  Initialize the xHC.
2  Read HCCPARAMS register.  Skip this test if PPC=0.
3  Set the *PP* flag to 0 in all the PORTSC registers.
4  Read the PORTSC registers and verify the followings:
    4.1  *PP* = 0 (4.19.1.1.1#1)
    4.2  *CCS* = 0
    4.3  *PED* = 0

4.4      *PR* = 0

4.5      *PLS* = 4 (Disabled)

4.6      *PIC* = 0

5    Prompt a user to attach a device to a port if no device is attached. Test fails if a Port Status Change Event is received (5.4.8#17).

6    Read the PORTSC registers and verify the followings:

6.1      *PP* = 0

6.2      *CCS* = 0 (5.4.8#2)

6.3      *PED* = 0 (5.4.8#5)

6.4      *PR* = 0

6.5      *PLS* = 4 (Disabled)

6.6      *CSC* = 0 (5.4.8#34)

7    Set the *PP* flag to 1 in the PORTSC registers.

8    Wait for a Port Status Change Event. Test fails if it does not receive a Port Status Change Event.

9    Read the PORTSC register and verify the followings:

9.1      *PP* = 1

9.2      *CCS* = 1(5.4.8#81)

9.3      *PR* = 0

9.4      *CSC* = 1

9.5      For SuperSpeed port:

       9.5.1   *PED* = 1

       9.5.2   *PLS* = 0 (U0)

9.6      For USB 2.0 port:

       9.6.1   *PED* = 0

       9.6.2   *PLS* = 7 (Polling)

10   Read the USBSTS register and verify that the *PCD* flag is set to 1.

11   Set the *PP* flag to 0 in the PORTSC registers. Test fails if it receives a Port Status Change Event.

12   Read the PORTSC register and verify the followings:

12.1    *PP* = 0

12.2    *CCS* = 0 (5.4.8#2)

12.3    *PED* = 0 (5.4.8#5)

12.4    *PR* = 0

12.5    *PLS* = 4 (Disabled)

12.6    *PEC* = 0 (5.4.8#38)

12.7    *PLC* = 0 (5.4.8#49)

13   Detach the device from the port. Test fails if it receives a Port Status Change Event (5.4.8#18).

14   Read the PORTSC register and verify the followings:

14.1    *PP* = 0

14.2    *CCS* = 0

14.3    *PED* = 0

14.4    *PR* = 0

14.5    *PLS* = 4 (Disabled)

15   Set the *PP* flag to 1 in the PORTSC register.

16   Read the PORTSC register and verify the followings:

16.1    *PP* = 1

16.2    *CCS* = 0

16.3    *PED* = 0

16.4    *PR* = 0

16.5    *PLS* = 5 (RxDetect) (4.19.1.1.1#2, 4.19.1.2.2#1)

16.6    *CSC* = 1

17   Clear the *CSC* flag in the PORTSC register.

**Repetitions**

Repeat for all the root ports exposed by the xHC.

### TD.3.04 SuperSpeed Port Disable Test

This test verifies that the xHC correctly disables SuperSpeed ports.

**Required Device Resource**

USB 3.0 device

**Overview of Test Steps**

The test performs the following steps for each port.

1  Initialize the xHC.
2  Attach a USB 3.0 device to a port.
3  Wait for a Port Status Change Event. Test fails if it does not receive a Port Status Change Event.
4  Read the PORTSC register and clear the *CSC* flag by setting it to 1.
5  Write the PORTSC register with setting the *PP* flag to 1 and the PED flag to 1.
6  Wait for a Port Status Change Event. Test fails if it does not receive a Port Status Change Event.
7  Read the PORTSC register and verify the followings:
   7.1   $PP = 1$
   7.2   $CCS = 0$
   7.3   $PED = 0$ (5.4.8#8)
   7.4   $PR = 0$
   7.5   $PLS = 4$ (Disabled) (4.19.1.2.1#1)
   7.6   $CSC = 1$
   7.7   $PEC = 0$
   7.8   $LWS = 0$
8  Clear the *CSC* flag in the PORTSC register by setting it to 1.
9  Read the USBSTS register and verify that the PCD flag is set to 1.
10  Clear the *PCD* flag in the USBSTS register by setting it to 1.
11  Set the *PP* flags to 0 in all the PORTSC registers (including USB2 ports).
12  Read the PORTSC register and verify that the *PP* flags are cleared to 0 (4.19.1.2.1#3).
13  Insert some delay so that the device's upstream port can transition to SS.Disabled.
14  Set the *PP* flag to 1 in the PORTSC register.
15  Wait for a Port Status Change Event. Test fails if it does not receive a Port Status Change Event.
16  Read the PORTSC register and verify that the port is enabled.
17  Write the PORTSC register with setting the *PP* flag to 1 and the *CSC* flag to 1.
18  Repeat the steps from 5 to 8.
19  Detach the USB 3.0 device from the port.
20  Read the PORTSC register and verify that the *PLS* field is set to 4 (Disabled).
21  Write the PORTSC register with setting the *PP* flag to 1, the *PLS* field to 5 (RxDetect) and the *LWS* flag to 1.
22  Read the PORTSC register and verify the followings:
   22.1   $PP = 1$
   22.2   $CCS = 0$
   22.3   $PED = 0$
   22.4   $PR = 0$
   22.5   $PLS = 5$ (RxDetect) (4.19.1.2.1#2)

### TD.3.05 USB 2.0 Port Disable Test

This test verifies that the xHC correctly disables USB 2.0 ports.

**Required Device Resource**

USB 2.0 device

**Overview of Test Steps**

The test performs the following steps for each port.

1  Initialize the xHC.

2    Prompt a user to attach a USB 2.0 device to a port if no device is attached.
3    Wait for a Port Status Change Event. Test fails if it does not receive a Port Status Change Event.
4    Read the PORTSC register and clear the *CSC* flag by setting it to 1.
5    Write the PORTSC register with setting the *PP* and *PR* flags to 1.
6    Wait for a Port Status Change Event. Test fails if it does not receive a Port Status Change Event.
7    Read the PORTSC register and clear the *PRC* flag by setting it to 1.
8    Write the PORTSC register with setting the *PP* and *PED* flags to 1.
9    Read the PORTSC register and verify the followings:

     9.1    $PP = 1$
     9.2    $CCS = 1$
     9.3    $PED = 0$ (5.4.8#8)
     9.4    $PR = 0$
     9.5    $PLS = 7$ (Polling)
     9.6    $PEC = 0$

10   Write the PORTSC register with setting the *PP* and *LWS* flags to 1 and the *PLS* field to 3, then read the PORTSC register and verify that the *PLS* field is set to 7 (4.15.1#1).

## TD.3.06 SuperSpeed Port Warm Reset Test

This test verifies the warm reset operation of SuperSpeed ports.

### Required Device Resource

USB 3.0 device

### Overview of Test Steps

The test performs the following steps for each port.

1    Initialize the xHC.
2    Prompt a user to attach a USB 3.0 device to a port if no device is attached.
3    Wait for a Port Status Change Event, read the PORTSC register and verify that the port is successfully enabled.
4    Write the PORTSC register with setting the *PP* and *WPR* flags to 1.
5    Read the PORTSC register and verify that the *WPR* flag is set to 0 (5.4.8#70).
6    Wait for a Port Status Change Event. Test fails if the xHC does not generate a Port Status Change Event.
7    Read the PORTSC register and verify the followings:

     7.1    $PR = 0$
     7.2    $PRC = 1$ (4.19.2#3)
     7.3    $WPR = 0$
     7.4    $WRC = 1$ (5.4.8#40)
     7.5    $PP = 1$ (5.4.8#20)

8    Read the USBSTS register and verify that the *PCD* flag is set to 1.
9    Clear the *PCD* flag in the USBSTS register by setting it to 1.
10   Write the PORTSC register with the *PP* flag set to 1, then read the PORTSC register and verify the followings:

     10.1   $PRC = 1$ (5.4.8#46)
     10.2   $WRC = 1$ (5.4.8#42)

11   Write the PORTSC register with the *PP*, *PRC* and *WRC* flags set to 1.
12   Read the PORTSC register and verify the followings:

     12.1   $PRC = 0$ (5.4.8#72)
     12.2   $WRC = 0$ (5.4.8#41)

## TD.3.07 SuperSpeed Port Suspend and Resume Test

This test verifies suspend and resume operations of SuperSpeed ports.

### Required Device Resource

USB 3.0 device

**Overview of Test Steps**

The test performs the following steps for each port.

1. Initialize the xHC.
2. Prompt a user to attach a USB 3.0 device to a port if no device is attached, then wait for a Port Status Change Event. Test fails if it does not receive a Port Status Change Event.
3. Write the PORTSC register with setting the *PP* flag to 1 and the *PLS* field to 3 (U3).
4. Read the PORTSC register after 300 ms and verify that the *PLS* field is set to 0 (U0) (4.15.1#2).
5. Write the PORTSC register with setting the *PP* flag to 1, the *PLS* field to 3 (U3) and the *LWS* to 1.
6. Read the PORTSC register after 300 ms and verify the followings:
   6.1  *PP* = 1
   6.2  CCS = 1
   6.3  *PED* = 1
   6.4  *PR* = 0
   6.5  *PLS* = 3 (U3) (4.19.1.2.10#1)
   6.6  *LWS* = 0 (5.4.8#29)
7. Write the PORTSC register with setting the *PP* flag to 1, the *PLS* field to 0 (U0) and the *LWS* flag to 1.
8. Wait for a Port Status Change Event. Test fails if it does not receive a Port Status Change Event.
9. Read the PORTSC register and verify the followings:
   9.1  *PP* = 1
   9.2  *CCS* = 1
   9.3  *PED* = 1
   9.4  *PR* = 0
   9.5  *PLS* = 0 (U0) (4.15.2#3)
   9.6  *PLC* = 1 (4.15.2#4)
   9.7  *LWS* = 0 (5.4.8#29)
10. Read the USBSTS register and verify that the *PCD* flag is set to 1.
11. Write clear the *PCD* flag in the USBSTS register by setting it to 1.
12. Write the PORTSC register with setting the *PP* flag to 1, then read the PORTSC register and verify that the *PLC* flag is still set to 1 (5.4.8#51).
13. Write the PORTSC register with setting the *PP* and *PLC* flags to 1, then read the PORTSC register and verify that the *PLC* flag is cleared to 0 (5.4.8#50).
14. Detach the USB 3.0 device from the root hub port.

## TD.3.08 USB 2.0 Port Suspend and Resume Test

This test verifies suspend and resume operations of USB 2.0 ports.

**Required Device Resource**

USB 2.0 device

**Overview of Test Steps**

The test performs the following steps for each port.

1. Initialize the xHC.
2. Attach a USB 2.0 device to a root hub port and wait for a Port Status Change Event.
3. Reset the root hub port and transition the port to the Enabled state.
4. Write the PORTSC register with the PP flag set to 1 and the PLS field set to 3 (U3).
5. Read the PORTSC register after 300 ms and verify that the PLS field is set to 0 (U0) (4.15.1#2).
6. Write the PORTSC register with the PP flag set to 1, PLS field set to 3 (U3) and LWS flag set to 1.
7. Read the PORTSC register after 300 ms and verify the followings:
   7.1  PP = 1
   7.2  CCS = 1
   7.3  PED = 1
   7.4  PR = 0
   7.5  PLS = 3 (U3)
   7.6  LWS = 0 (5.4.8#29)

8    Write the PORTSC register with the PP flag set to 1, PLS field set to 15 (Resume) and LWS flag set to 1.
9    Read the PORTSC register after 300 ms and verify the followings:
    9.1    PP = 1
    9.2    CCS = 1
    9.3    PED = 1
    9.4    PR = 0
    9.5    PLS = 15 (Resume)
    9.6    PLC = 0
    9.7    LWS = 0 (5.4.8#29)
10   After 1000 ms, write the PORTSC register with the PP flag set to 1, PLS field set to 0 (U0) and LWS flag set to 1.
11   Wait for a Port Status Change. Test fails if the xHC does not generate a Port Status Change Event (4.15.2#5).
12   Read the PORTSC register and verify the followings:
    12.1   PP = 1
    12.2   CCS = 1
    12.3   PED = 1
    12.4   PR = 0
    12.5   PLS = 0 (U0) (4.15.2#3)
    12.6   PLC = 1 (4.15.2#4)
    12.7   LWS = 0 (5.4.8#29)
13   Read the USBSTS register and verify that the PCD flag is set to 1.
14   Write clear the PCD flag.
15   Write the PORTSC register with setting the PP and PLC flags to 1, then read the PORTSC register and verify that the PLC flag is cleared to 0 (5.4.8#50).
16   Detach the USB 2.0 device from the root hub port.

## TD.3.09 SuperSpeed Port Remote Wakeup Test

This test verifies remote wakeup operations of SuperSpeed ports.

### Required Device Resource

USB 3.0 compliance device

### Overview of Test Steps

The test performs the following steps for each port.

1    Initialize the xHC.
2    Attach a USB 3.0 compliance device to a port and enumerate it.
3    Perform a VEN_Set_RMTWK_TO request with the SleepTime set to 3 seconds.
4    Suspend the port.
5    Wait for a Port Status Change Event.
6    Read the PORTSC register and verify the followings:
    6.1    PLS = Resume (4.15.2#1)
    6.2    PLC = 1 (4.15.2#2)
7    Write the PORTSC register with setting the PP and PLC flags to 1, then read the PORTSC register and verify that the PLC flag is cleared to 0 (5.4.8#50).
8    Write the PORTSC register with setting the PP flag to 1, the PLS field to 0 (U0) and the LWS flag to 1.
9    Wait for a Port Status Change Event.
10   Read the PORTSC register and verify the followings:
    10.1   PLS = U0
    10.2   PLC = 1
11   Write the PORTSC register with setting the PP and PLC flags to 1, then read the PORTSC register and verify that the PLC flag is cleared to 0 (5.4.8#50).

### Repetitions

Repeat for all the USB3 ports exposed by the xHC.

### TD.3.10 USB 2.0 Port Remote Wakeup Test

This test verifies remote wakeup operations of USB 2.0 ports.

**Required Device Resource**

USB 2.0 compliance device

**Overview of Test Steps**

The test performs the following steps for each port.

1   Initialize the xHC.
2   Attach a USB 2.0 compliance device to a port and enumerate it.
3   Perform a VEN_Set_RMTWK_TO request with the SleepTime set to 3 seconds.
4   Suspend the port.
5   Wait for a Port Status Change Event.
6   Read the PORTSC register and verify the followings:
    6.1   PLS = Resume (4.15.2#1)
    6.2   PLC = 1 (4.15.2#2)
7   Write the PORTSC register with setting the PP and PLC flags to 1, then read the PORTSC register and verify that the PLC flag is cleared to 0 (5.4.8#50).
8   Wait 20 ms.
9   Write the PORTSC register with setting the PP flag to 1, the PLS field to 0 (U0) and the LWS flag to 1.
10  Wait for a Port Status Change Event.
11  Read the PORTSC register and verify the followings:
    11.1   PLS = U0
    11.2   PLC = 1
12  Write the PORTSC register with setting the PP and PLC flags to 1, then read the PORTSC register and verify that the PLC flag is cleared to 0 (5.4.8#50).

**Repetitions**

Repeat for all the USB2 ports exposed by the xHC.

### TD.3.11 USB2 Hardware LPM Test

This test verifies hosts ability to do Hardware LPM L1 on USB2 Ports. This test is only applicable to xHCI 1.0 host controllers with Hardware LPM capability, i.e. HLC flag is 1 in USB 2.0 xHCI Supported Protocol Capability register.

**Required Device Resource**

USB 2.0 device capable of doing L1 LPM

**Overview of Test Steps**

The test performs the following steps for each port.

1   Initialize the xHC.
2   Iterate through all USB 2.0 Protocol Defined fields in the Supported Protocols.
3   For xHC >= 1.10 verify the following:
    3.1   Bit 19 (HLC) is set to 1 (4.23.5.1.1.1#3)
    3.2   Bit 20 (BLC) is set to 1 (4.23.5.1.1.1#2)
4   Attach a USB 2.0 device capable of doing LPM L1 to a port and enumerate it.
5   Read the HLC value in USB 2 xHCI Supported Protocol Capability register.
6   If the HLC=1 perform the following steps to Enable Hardware LPM.
7   Write USB2 PORTHLPMC with following values
    7.1   HIRD Deep(HIRDD) = 1ms
    7.2   HIRD Mode
    7.3   L1 Timeout

8    Write the following values in USB2 PORTPMSC
     8.1    L1 Device Slot = Slot ID of the device attached.
     8.2    HIRD less that HIRDD and RWE fields.
9    Set the HLE bit in the USB2 PORTPMSC to enable Hardware LPM on the xHC.
10   Read PORTPMSC and verify that HLE=1.   (4.23.5.1.1.1#1)
11   Perform Evaluate Context Command to set Max Exit Latency.

12   Max Exit Latency in the Slot context field must be set to non-zero value (6.2.2#5).
13   Wait 100 ms and verify that the PLS field is set to U2 in the PORTSC register (7.2.2.1.3.2#8).

### Repetitions

Repeat for all the USB 2.0 ports exposed by the xHC.

## TD.3.12 PORTSC Register Test

This test verifies PORTSC registers.

### Required Device Resource

USB device

### Overview of Test Steps

The test performs the following steps for each port.

1    Initialize the xHC.
2    Write the PORTSC register with the PP flag set to 1.
3    Attach a USB device to a root hub port and wait for a Port Status Change Event.
4    If HS, FS or LS device is connected, reset the root hub port and transition the port to the Enabled state.
     PED = 1
5    Perform Host Controller reset by the following steps:
     5.1    Set the HCRST flag to 1 in the USBCMD register.
     5.2    Read the USBCMD register and wait until the HCRST flag is cleared to 0.  Test fails if the
            HCRST flag is not cleared within 1000 ms.  (5.4.1#3)
     5.3    Set up the Command and Event Rings and the Device Slots again after HCRST.
6    In case HS, FS or LS device is connected, verify that the PED flag is reset to 0 (5.4.8#39).
7    Write the PORTSC register with the PP flag set to 1.
8    Wait for Port Status Change Event
9    Verify CCS = 1.
10   Clear CSC bit.
11   If HS, FS or LS device is connected, verify PED = 0 (5.4.8#9)
12   Verify PR = 0 (5.4.8#12)
13   In case HS, FS or LS device is connected, reset the root hub port and transition the port to the Enabled
     state.  PED = 1.
14   Clear PRC bit.
15   PLS must be 0.
16   If testing a USB2.0 device:
     16.1   Write 15 to PLS field.
     16.2   Verify that PLS is still 0.  (5.4.8#14)
17   Write OCA flag of PORTSC
18   Verify write of OCA flag has no effect (5.4.8#10)
19   If testing a Super Speed device:
     19.1  Write 5 to the PLS field and verify that it is still set to 0 (5.4.8#13)
20   Write Port Speed field and verify that it has no effect (5.4.8#22).
21   In case PIND flag is 0 in the HCCPARAMS register:
     21.1   Write PIC field to 1.
     21.2   Verify that PIC field is still set to 0 (5.4.8#24).
22   In case PIND flag is 1 in the HCCPARAMS register:
     22.1   Write PIC field to 1.

22.2 Set the PP flag to zero and verify that the PIC field is reset to 0 (5.4.8#25).

22.3 Write PIC field to 1 then issue host reset and verify that the PIC field is not changed (5.4.8#26).

22.4 In case of SuperSpeed port, write PIC field to 1, issue warm reset and verify that the PIC field is not changed (5.4.8#27).

22.5 Write PIC field to 2 then issue HCRST and verify that the PIC field is reset to 0 (5.4.8#28).

23 Detach a device and re-attach it.

24 Set the WCE, WDE and WOE flags to one.

25 In case of SS, issue warm reset and verify the followings (do not clear WRC):

25.1 CSC flag is not changed (5.4.8#36).

25.2 WCE flag is not changed (5.4.8#57).

25.3 WDE flag is not changed (5.4.8#62).

25.4 WOE flag is not changed (5.4.8#65).

26 Issue hot reset and verify the followings (do not clear PRC):

26.1 CSC flag still set to one (5.4.8#35).

26.2 WCE flag is not changed (5.4.8#56).

26.3 WDE flag is not changed (5.4.8#61).

26.4 WOE flag is not changed (5.4.8#64).

26.5 In case of SS, WRC flag is not changed (5.4.8#43).

27 Detach the device.

28 Issue HCRST and verify the followings:

28.1 The CSC flag is reset to 0 (5.4.8#37).

28.2 WCE flag is reset to 0 (5.4.8#58).

28.3 WDE flag is reset to 0 (5.4.8#63).

28.4 WOE flag is reset to 0 (5.4.8#66).

28.5 WRC flag is reset to 0 (5.4.8#44).

28.6 PRC flag is reset to 0 (5.4.8#47).

28.7 Port Speed is reset to 0 (5.4.8#78).

29 Issue HCRST and verify that the OCC flag is reset to 0 (5.4.8#45).

30 Issue HCRST and verify that the PLC flag is reset to 0 (5.4.8#52).

31 Issue HCRST and verify that the CEC flag is reset to 0 (5.4.8#53).

32 Write DR flag and verify that it has no effect (5.4.8#67).

33 Set WPR flag to 0 and verify that it has no effect (5.4.8#68).

### TD.3.13 U1/U2 Test

This test verifies.

**Required Device Resource**

USB 3.0 device

**Overview of Test Steps**

The test performs the following steps for each port.

1 Steps for 4.19.1.2.11#1, 4.12.1.2.11#2.

2 Connect a USB 3.0 device.

3 Start capture.

4 Perform a Set Feature (U1_ENABLE) request to the device.

5 Stop capture and have user verify that the xHC returned LXU upon reception of LGO_U1 (4.19.1.2.11#2).

6 Start capture.

7 Set the U1 timeout field to FFh in the PORTPMSC register.

8 Stop capture and have user verify that the xHC returned LAU upon reception of LGO_U1 (4.19.1.2.11#1).

### TD.3.14 Integrated Hub Test

This test verifies.

**Required Device Resource**

**Overview of Test Steps**

The test performs the following steps for each port.

1    For all USB 2.0 hubs that are non-removable, verify that Supported Protocol Capability field IHI (Integrated Hub Implemented) Flag matches hub and non-removable host topology (4.24.2.1#1, 4.24.2.1#2). User will be prompted to indicate whether the hub is embedded or integrated.

## TD.3.15 Overcurrent Test

This test verifies.

**Required Device Resource**

Over-current fixture

**Overview of Test Steps**

The test performs the following steps for each port.

Steps for 4.24.2.2#1

## TD.3.16 Hot Port Reset Test

This test verifies.

**Required Device Resource**

**Overview of Test Steps**

1    Connect a USB device.
2    Set the PR flag and the PP flag to one in the PORTSC register.
3    Wait for a Port Status Change Event. Test fails if no event is received.
4    Read the PORTSC register and verify the followings:
5    PP = 1 (5.4.8#19)

## TD.3.17 Field Test

Verify that Lane Count fields are read-only

**Required Device Resource**

SuperSpeed Gen1 and Gen2 devices

**Overview of Test Steps**

If the HC version is 1.1, perform the following steps:
1.    Connect and enumerate a SuperSpeed Gen1 device
2.    Read the PORTLI register.
3.    Write (Rx Lane Count + 1) back to the PORTLI register.
4.    Read PORTLI register again.
5.    Verify that the Rx Lane Count field has not changed. (5.4.10.1#6)
6.    Write (Tx Lane Count + 1) back to the PORTLI register.
7.    Read PORTLI register again.
8.    Verify that the Tx Lane Count field has not been changed (5.4.10.1#7)
Repetitions:
Repeat with SuperSpeed Gen2 device

## 3.4 Command Interface Test

### TD.4.01 Enable and Disable Slot Command Test

This test verifies that the xHC correctly enables and disables device slots.

**Required Device Resource**

None

**Overview of Test Steps**

The test performs the following steps by changing the number of slots enabled (*MaxSlotsEn*) from 1 to MaxSlots.

1    Reset the xHC by setting the *HCRST* flag to 1 in the USBCMD register and wait until the *HCRST* flag is cleared to 0.
2    Set the *MaxSlotsEn* field to the number of slots enabled in the CONFIG register.
3    Initialize the Command Ring, the primary Event Ring and the primary interrupter.
4    Start the xHC by setting the *RS* flag to 1 in the USBCMD register.
5    Repeat the following steps for the *MaxSlotsEn* times.
   5.1     Perform an Enable Slot Command.
   5.2     Wait for a Command Completion Event and verify the followings:
       5.2.1  The *Completion Code* field is set to Success.
       5.2.2  The *Slot ID* field is set to a value in the range of 1 to *MaxSlotsEn* (4.6.3#2).
       5.2.3  Assigned *Slot ID* is unique.
       *5.2.4  Command Completion Parameter = 0 (6.4.2.2#2)*
   5.3     Allocate Device Context
6    Perform another Enable Slot Command and verify the followings:
   6.1     The *Completion Code* field is set to No Slots Available Error (4.6.3#1).
   6.2     The *Slot ID* field is set to 0 (4.6.3#3).
   6.3     Allocate Device Context
7    Perform a Disable Slot Command by changing the Slot ID from 1 to *MaxSlotsEn* and verify the followings:
   7.1     The *Completion Code* field is set to Success.
   7.2     The *Slot ID* field is set to the Slot ID value of the disabled slot (4.6.1#3).
8    Perform another Disable Slot Command by changing the Slot ID from 1 to *MaxSlotsEn*. Test fails if the *Completion Code* field is not Slot Not Enabled Error.
9    Stop the xHC by setting the *RS* flag to 0 in the USBCMD register.

**Repetitions**

Repeat for all the *MaxSlotsEn* values from 1 to MaxSlots.

### TD.4.02 Address Device Command Test

This test verifies the xHC's operation of an Address Device Command.

**Required Device Resource**

USB device

**Overview of Test Steps**

The test performs the following steps.

1    Initialize and start the xHC.
2    Prompt a user to attach a USB device to a port if no device is attached.
3    Reset the port if the port is not enabled.
4    Perform an Enable Slot Command.
5    Allocate a buffer for the Output Device Context and initialize it by zero.
6    Write the pointer to the Output Device Context in the DCBAA.
7    Allocate an Input Context and initialize it as follows:

7.1     Slot Context
      7.1.1   *Speed* = PORTSC.Port Speed
      7.1.2   *Context Entries* = 1
      7.1.3   *Root Hub Port Number* = XXX
7.2     Input Control Context
      7.2.1   *A0, A1* = 1

7.3     Endpoint 0 Context
      7.3.1   *CErr* = 3
      7.3.2   *EP Type* = 4 (Control)
      7.3.3   *Max Packet Size* = 512 (SS) or 64 (HS) or 8 (FS, LS)
      7.3.4   *TR Dequeue Pointer* = base address of the Transfer Ring
      7.3.5   *DCS* = 1
7.4     All the other Input Contexts are initialized by non-zero values.

8   Perform an Address Device Command with the *BSR* flag set to 1 and verify the followings:
    8.1     The *Slot ID* field is set to the value identifying the target device slot in the Command Completion Event (4.6.1#3).
    8.2     The *Completion Code* field is set to Success in the Command Completion Event (4.6.1#5)
    8.3     The Output Slot Context is updated with the parameters specified in the Input Slot Context (4.6.5#2).
    8.4     The Output Endpoint 0 Context is updated with the parameters specified in the Input Endpoint 0 Context (4.6.5#3).
    8.5     The *Slot State* field is set to Default in the Output Slot Context (4.3.4#2).
    8.6     The *USB Device Address* field is set to 0 in the Output Slot Context (4.5.3.4#1).
    8.7     The *EP State* field is set to Running in the Output Endpoint 0 Context (4.6.5#4).
    8.8     All the other Output Endpoint Contexts are set to 0 (4.6.5#6).

9   Perform an Address Device Command with the *BSR* flag set to 0 and verify the followings:
    9.1     The *Slot ID* field is set to the value identifying the target device slot in the Command Completion Event (4.6.1#3).
    9.2     The *Completion Code* field is set to Success in the Command Completion Event (4.6.1#5)
    9.3     The *Slot State* field is set to Addressed in the Slot Context (4.3.4#1).
    9.4     The *USB Device Address* field is set to a value in the range of 1 to 127 (4.6.5#1).
    9.5     The *EP State* field is set to Running in the Output Endpoint 0 Context (4.6.5#4).

10   Perform an Address Device Command with the BSR flag set to 1 and verify that the command completes with Completion Code set to Context State Error.  Also verify that the no field is changed in the output context (3.2.3#1).

11   Reset the port.

12   Perform an Enable Slot Command.

13   Allocate a buffer for the Output Device Context and initialize it by zero.

14   Write the pointer to the Output Device Context in the DCBAA.

15   Perform an Address Device Command with the BSR flag set to 0 and verify the followings:
    15.1     The *Slot ID* field is set to the value identifying the target device slot in the Command Completion Event (4.6.1#3).
    15.2     The *Completion Code* field is set to Success in the Command Completion Event (4.6.1#5)
    15.3     The *Slot State* field is set to Addressed (4.3.4#1).
    15.4     The *USB Device Address* field is set to a value in the range of 1 to 127 (4.6.5#1).
    15.5     The *USB Device Address* field is different from the previously assigned values (4.6.5#5).
    15.6     The *EP State* field is set to Running in the Output Endpoint 0 Context (4.6.5#4).

16   Repeat the step 10 to 14 until an Enable Slot Command fails with No Slots Available Error or until the Address Device Command with BSR=0 returns ResourceError Command CompletionEvent (4.6.5#13).

17   Reset the port.

18   Perform Disable Slot Commands for all the enabled slots and verify the followings:
    18.1     The *Slot State* field is set to Disabled in the Slot Contexts (4.5.3.2#1).
    18.2     The *EP State* field is set to Disabled in the Endpoint 0 Context (4.8.3#2).

### TD.4.03 Configure Endpoint Command Test

This test verifies the xHC's operation of a Configure Endpoint Command.

**Required Device Resource**

SuperSpeed USB device

**Overview of Test Steps**

The test performs the following steps.

1    Initialize and start the xHC.
2    Attach a SuperSpeed device to a root hub port if it is not connected and enable the port.
3    Perform an Enable Slot Command.
4    Perform an Address Device Command with the *BSR* flag set to 0.
5    Perform a Configure Endpoint Command with the following Input Context settings, to configure endpoint 1 OUT (DCI=2) as a Bulk Out EP, and endpoint 1 IN (DCI=3) as a 4 stream Bulk In EP:
    5.1      Input Control Context:
         5.1.1  *A2, A3* = 1
    5.2      Input Slot Context
         5.2.1  *Context Entries* = 3 // numerical maximum of (DCI of OUT Endpoint, DCI of IN Endpoint)
    5.3      Input Endpoint 1 OUT Context (DCI=2):
         5.3.1  *EP Type* = 2 // Bulk Out
         5.3.2  *CErr Count = 3*
         5.3.3  *Max Packet Size* = 1024
         5.3.4  *TR Dequeue Pointer* = base address of the Transfer Ring
         5.3.5  *DCS* = 1
         5.3.6  FE = 1 (xHCI 0.96 only)
    5.4      Input Endpoint 1 IN Context (DCI=3):
         5.4.1  EP Type = 6 // Bulk In
         5.4.2  CErr = 3
         5.4.3  Max Packet Size = 1024
         5.4.4  TR Dequeue Pointer = base address of Stream Context.
         5.4.5  MaxPStreams = 1 // 4 Streams
         5.4.6  LSA = 1
         5.4.7  HID = 1
    5.5      Other Input Endpoint Contexts are set to all 1s.
6    Test fails if the Completion Code field is not set to Success in the Command Completion Event (4.6.1#5).
7    Read the Output Context and verify the followings:
    7.1      Output Slot Context:
         7.1.1  *Slot State* = Configured (4.3.5#1).
    7.2      Output Endpoint 1 OUT Context (DCI=2):
         7.2.1  Parameters specified in the Input Context are copied. (EPType 6.2.3#7) (CErr Count 6.2.3#6), (MaxPacketSize 6.2.3#10), (TR Dequeue Pointer 6.2.3#12), (DCS flag 6.2.3#11) , (FE 6.2.3#5, xHCI 0.96 only)
         7.2.2  Bit 0 of Offset 04h is set to 0 (xHC 1.0 only) (6.2.3#18)
         7.2.3  *EP State* is Running (4.6.6#3).
    7.3      Output Endpoint 1 IN Context (DCI=3):
         7.3.1  Parameters specified in the Input Context are copied. (EPType 6.2.3#7) (CErr Count 6.2.3#6), ( MaxPacketSize 6.2.3#10), (TR Dequeue Pointer 6.2.3#12), (MaxPStreams 6.2.3#2), (LSA 6.2.3#3), (HID 6.2.3#8)
         7.3.2  *EP State* is Running (4.6.6#3).
    7.4      Other Output Endpoint Contexts are set to all 0s (3.3.5#1).
8    Perform a Configure Endpoint Command with the following Input Context settings, to reconfigure endpoint 1 OUT (DCI=2) as an Interrupt Out EP and disable endpoint 1 OUT (DCI=3):
    8.1      Input Control Context
         8.1.1  *A2* = 1
         8.1.2  *D2,D3* = 1

8.2　　Input Slot Context
　　8.2.1　*Context Entries* = 3 // DCI of IN Endpoint
8.3　　Input Endpoint 1 OUT Context (DCI=2):
　　8.3.1　*EP Type* = 3 // Interrupt Out
　　8.3.2　*Max Packet Size* = 1024
　　8.3.3　*Interval* = 3
　　8.3.4　*Max ESIT Payload* = Max Packet Size
　　8.3.5　*Average TRB Length* = Max Packet Size
8.4　　Input Endpoint 1 IN Context (DCI=3):
　　8.4.1　No initialization needed. Content ignored by xHC Drop operation.
9　Test fails if the Completion Code field is not set to Success in the Command Completion Event (4.6.1#5).
10　Test fails if the Output Context is not updated as follows:
　10.1　　Output Slot Context
　　10.1.1　*Slot State* = Configured (4.3.5#1).
　10.2　　Output Endpoint 1 Context
　　10.2.1　*EP State* = Running (4.6.6#3).
　　10.2.2　*EP Type* = 3 (6.2.3#7)
　　10.2.3　*Max Packet Size* = 1024 (6.2.3#10)
　　10.2.4　*Interval* = 3 (6.2.3#4)
　　10.2.5　*Max ESIT Payload* = Max Packet Size (6.2.3#14)
　　10.2.6　*Average TRB Length* = Max Packet Size (6.2.3#13)
　　10.2.7
11　Perform a Configure Endpoint Command with the following Input Context settings, to disable endpoint 1 OUT (DCI=2):
　11.1　　Input Control Context
　　11.1.1　*D2* = 1
　11.2　　Input Slot Context
　　11.2.1　*Context Entries* = 2
12　Test fails if the Completion Code field is not set to Success in the Command Completion Event (4.6.1#5).
13　Test fails if the Output Context is not updated as follows:
　13.1　　Output Slot Context
　　13.1.1　*Slot State* = Addressed (4.6.6#4).
　　13.1.2　*Context Entries* = 1
　13.2　　Output Endpoint 1 OUT Context
　　13.2.1　*EP State* is Disabled (4.6.6#2).
14　Perform a Configure Endpoint Command with the following Input Context settings, to configure endpoint 1 OUT (DCI=2) as a Bulk EP:
　14.1　　Input Control Context:
　　14.1.1　*A2* = 1
　14.2　　Input Slot Context
　　14.2.1　*Context Entries* = DCI of OUT Endpoint
　14.3　　Input Endpoint 1 OUT Context:
　　14.3.1　*EP Type* = 2 // Bulk Out
　　14.3.2　*Max Packet Size* = 1024
　　14.3.3　*TR Dequeue Pointer* = base address of the Transfer Ring
　　14.3.4　*DCS* = 1
15　Test fails if the Completion Code field is not set to Success in the Command Completion Event (4.6.1#5).
16　Perform a Configure Endpoint Command with the *Deconfigure* (*DC)* flag set to 1 and the Input Context Pointer filed set to 0 then verify the followings, to deconfigure the Device Slot:
　16.1　　Completion Code is set to Success in the Command Completion Event (4.6.6#6).
　16.2　　Output Slot Context is updated as follows:
　　16.2.1　*Slot State* = Addressed (4.5.3.5#1).
　　16.2.2　*Context Entries* = 1 (4.6.6#1).
　16.3　　Output Endpoint 1 OUT Context is updated as follows:
　　16.3.1　*EP State* = Disabled (4.6.6#5).

17  Perform a Configure Endpoint Command with the following Input Context Settings, to configure endpoint 1 OUT (DCI=2) as an Isochronous Out EP:
    17.1   Input Control Context:
        17.1.1 A2 = 1
    17.2   Input Slot Context
        17.2.1 Context Entries = 2
    17.3   Input Endpoint 1 OUT Context (DCI=2):
        17.3.1 EP Type = Isochronous Out = 1
        17.3.2 Max Packet Size = 1024
        17.3.3 Max Burst Size = 15
        17.3.4 Mult = 2
        17.3.5 TR Dequeue Pointer = base address of Transfer Ring
        17.3.6 DCS = 1
        17.3.7 Average TRB Length = 4096
        17.3.8 Max ESIT Payload = Max Packet Size * (Max Burst Size + 1) * (Mult + 1)  // 48KB

18  If the Configure Endpoint Command fails with Bandwidth Error, retry the command with smaller Mult or MaxBurstSize.

19  Test fails if the Completion Code field is not set to Success in the Command Completion Event (4.6.1#5).

20  Verify that Output Endpoint 1 OUT Context is updated as follows:
    20.1   EP Type = Isochronous Out = 1 (6.2.3#7)
    20.2   Max Packet Size = 1024 (6.2.3#10)
    20.3   Max Burst Size = value in the Input Endpoint Context (6.2.3#9)
    20.4   Mult = value in the Input Endpoint Context (6.2.3#1)
    20.5   TR Dequeue Pointer = base address of Transfer Ring (6.2.3#12)
    20.6   DCS = 1 (6.2.3#11)

21  Perform a Configure Endpoint Command with the following Input Context Settings. to configure all endpoints as Isochronus Out or In EPs, and test that a Bandwidth Error is detected:
    21.1   Input Control Context
        21.1.1 A2, A3, …, A31 = 1
        21.1.2 D2 = 1 // Reconfigure endpoint 1 OUT (DCI=2), which is still configured after the previous test.
    21.2   Input Slot Context
        21.2.1 Context Entries = 31
    21.3   Input Endpoint Contexts, (DCI = 2-31) are set as follows:
        21.3.1 EP Type = Isochronous Out (1) for OUT endpoints (even DCIs) and Isochronous In (5) for IN endpoints (odd DCIs)
        21.3.2 Max Packet  Size = 1024
        21.3.3 Max Burst Size = 15
        21.3.4 Mult = 2
        21.3.5 TR Dequeue Pointer = base address of Transfer Ring
        21.3.6 DCS = 1
        21.3.7 Average TRB Length = 4096
        21.3.8 Max ESIT Payload = Max Packet Size * (Max Burst Size + 1) * (Mult + 1)

22  Test fails if Completion Code filed is not set to Bandwidth Error in the Command Completion Event (4.6.6#7)

23  Verify that the Output Endpoint Contexts are not updated (3.2.4#1).

## TD.4.04 Evaluate Context Command Test

This test verifies the xHC's operation of an Evaluate Context Command.

**Required Device Resource**

USB device

**Overview of Test Steps**

The test performs the following steps.

1     Initialize and start the xHC.
2     Attach a FS or SS USB device to a downstream port.
3     Perform an Enable Slot Command.
4     Perform an Address Device Command with the *Max Packet Size* field as follows in the Input Endpoint 0 Context, to update the Max Packet Size of the Control endpoint.
    4.1     For FS, 8 bytes.
    4.2     For SS, 512 bytes
5     Perform an Evaluate Context Command with the Input Context fields set as follows.
    5.1     Input Control Context
        5.1.1  *A1 = 0.*
    5.2     Input Slot Context
        5.2.1  *Max Exit Latency* = 10.
        5.2.2  *Interrupter Target* = 1 (only when the xHC supports secondary interrupters).
        5.2.3  *Context Entries = 1.*
    5.3     Input Endpoint 0 Context
        5.3.1  For FS, *Max Packet Size = bMaxPacketSize0* in the Device Descriptor.
        5.3.2  For SS, *Max Packet Size* = $2^{bMaxPacketSize0}$.
    5.4     Set all the other Input Endpoint Contexts to all 1s.
6     Test fails if the *Completion Code* field is not set to Success in the Command Completion Event (4.6.1#5, 4.6.7#4).
7     Test fails if the *Command Completion Parameter* is not 0 (6.4.2.2#2).
8     Count the number of integrated devices.
9     Test fails if the *Slot ID* is not set to 1 + (number of integrated devices) in the Command Completion Event (4.6.1#3).
10   Read the Output Slot Context and verify the followings:
    10.1    The above fields are correctly updated (4.6.7#1).
    10.2    The *Slot State* field is not changed (4.5.3.5#2).
11   Read the Output Endpoint 0 Context and verify the followings:
    11.1    The above field is correctly updated (4.6.7#2).
    11.2    The *EP State* field is not changed (4.11.4.6#1).
12   Read the other Output Endpoint Contexts and verify that they are set to all 0s (4.6.7#3).

**Repetitions**

Repeat for FS and SS speeds.

## TD.4.05 Reset Endpoint Command Test

This test verifies the xHC's operation of a Reset Endpoint Command.

**Required Device Resource**

USB device

**Overview of Test Steps**

The test performs the following steps.

1     Initialize and start the xHC.
2     Prompt the user to attach a USB device to a root hub port if not connected, and enable the port.
3     Perform an Enable Slot Command.
4     Perform an Address Device Command with the *BSR* flag set to 1.
5     Perform an Address Device Command with the *BSR* flag set to 0.
6     Perform a Configure Endpoint Command with the following Endpoint Context settings.
    6.1     Endpoint 1 OUT Context
        6.1.1  *EP Type* = 2 (Bulk OUT).
7     Issue Get Port Bandwidth for all speeds for the Device attached. Save the value if xHC supports this command.

8     Detach the USB device from the port.

9   Insert two transfer TRBs into the Transfer Ring for the EP1 and ring the doorbell.
    9.1    If device is attached to a root port verify that the first transfer fails with USB Transaction Error.
    9.2    If device is attached behind an embedded hub verify that the first transfer fails with USB Transaction Error or STALL error.

10  Read the Endpoint 1 OUT Context and verify the followings:
    10.1   The *EP State* field is set to Halted (4.8.3#1).
    10.2   The *TR Dequeue Pointer* field is set to the pointer to the first Transfer TRB (4.10.1#9).
11  Ring the doorbell and verify that the xHC does not generate a Transfer Event (4.6.8#2).
12  Perform a Reset Endpoint Command with the *TSP* flag set to 0.
13  Read the Command Completion Event and verify that the *Slot ID* field is set to the same value in the command (4.6.1#3).
14  Verify that the *Command Completion Parameter* is 0 (6.4.2.2#2).
15  Read the Slot Context and verify that the *Slot State* is Configured (4.5.3.6#1).
16  Read the Endpoint 1 OUT Context and verify that the *EP State* is Stopped (4.6.8#1).
17  If this xHC supports Get Port Bandwidth:
    17.1   Issue Get Port Bandwidth for all speeds for the Device attached.
    17.2   This value must be the same as the Get Port Bandwidth value before issuing Reset Endpoint Command was issued in step 17. (4.6.8#3)
18  Perform a Disable Slot Command.

## TD.4.06 Stop Endpoint Command Test

This test verifies the xHC's operation of a Stop Endpoint Command.

**Required Device Resource**

USB compliance device

**Overview of Test Steps**

The test performs the following steps.

1   Initialize and start the xHC.
2   Prompt a user to attach a USB device to port if no device is attached.
3   Enumerate the device and transition it to the Default State.
4   Perform a Stop Endpoint Command for the default endpoint.
5   Read the Device Context and verify the followings:
    5.1    The *Slot State* field is set to Default in the Slot Context (4.5.3.5#3).
    5.2    The *EP State* field is set to Stopped in the Endpoint 0 Context (4.6.9#3).
6   Perform a Stop Endpoint Command for the default endpoint and verify that the *Completion Code* is set to Context State Error (4.6.9#4).
7   Verify that the *Command Completion Parameter* is set to 0 (6.4.2.2#2).
8   Ring the doorbell for the default endpoint, then read the Endpoint 0 Context and verify that the *EP State* has transitioned to Running (4.8.3#3).
9   Perform an Address Device Command with the *BSR* flag set to 0.
10  Perform a Stop Endpoint Command for the default endpoint.
11  Read the Device Context and verify the followings:
    11.1   The *Slot State* is set to Addressed in the Slot Context (4.5.3.5#3).
    11.2   The *EP State* is set to Stopped in the Endpoint 0 Context (4.6.9#3).
12  Perform a Stop Endpoint Command for the default endpoint and verify that the *Completion Code* is set to Context State Error (4.6.9#4).
13  Verify that the Command Completion Parameter is set to 0 (6.4.2.2#2).
14  Perform a Configure Endpoint Command to enable the bulk IN endpoint.
15  Configure the compliance device for a bulk IN endpoint to always return NAK or NRDY, then issue a bulk IN transfer.
    15.1   For USB2, perform a VEN_Change_EP request with the *bFixedRespMode* field set to 02h.
    15.2   For USB3, perform a VEN_Set_NRDY_Rate request with the *Packet Count* set to 1.

16   Perform a Stop Endpoint Command for the bulk IN endpoint and verify that a Command Completion Event is received with the *Completion Code* field set to Success.
17   Verify that the Command Completion Parameter is 0 (6.4.2.2#2).
18   Read the Device Context and verify the followings:
    18.1   The *Slot State* is set to Configured in the Slot Context (4.5.3.5#3).
    18.2   The *EP State* is set to Stopped in the Endpoint Context for the bulk IN endpoint (4.6.9#3).
19   Configure the compliance device for a bulk IN endpoint to normally return data.
    19.1   For USB2, perform a VEN_Change_EP request with the *bFixedRespMode* field set to 00h.
    19.2   For USB3, perform a VEN_Change_EP request.
20   Ring the doorbell and reactivate the bulk IN endpoint, then verify that the transfer completes successfully. (4.6.9#6)
21   Read the Device Context and verify that the EP State is set to Running in the Endpoint Context for the bulk IN endpoint (4.8.3#3).

**Repetitions**

Repeat for HS and SS speeds.

## TD.4.07 Set TR Dequeue Pointer Command Test

This test verifies the xHC's operation of a Set TR Dequeue Pointer Command.

**Required Device Resource**

USB Compliance device

**Overview of Test Steps**

The test performs the following steps.

1    Initialize and start the xHC.
2    Attach a USB device to a root hub port and enable the port.
3    Perform an Enable Slot Command.
4    Perform an Address Device Command with the *BSR* flag set to 0.
5    Perform a Configure Endpoint.
6    Insert a No Op TRB with the *IOC* flag set to 1 and ring the doorbell then verify that a Transfer Event is received.
7    Perform a Stop Endpoint Command with the *Endpoint ID* set to 1.
8    Verify that TR Dequeue Pointer in Output Endpoint Context points to enqueue position for EP. (6.2.3#16)
9    Perform a Set TR Dequeue Pointer Command with the *Endpoint ID* set to 1, the *New TR Dequeue Pointer* set to the base address of another Transfer Ring Segment and the *DCS* flag set to 1.
10   Read the Device Context and verify the followings:
    10.1   The *Slot State* field is set to Configured in the Slot Context (4.5.3.5#4).
    10.2   The *EP State* is set to Stopped in the Endpoint 0 Context.
    10.3   The *TR Dequeue Pointer* field is set to the specified value (4.6.10#1).
    10.4   The *DCS* flag is set to the specified value (4.6.10#3).
11   Insert a No Op TRB with the *IOC* flag set to 1 into the new Transfer Ring Segment and ring the doorbell then verify that a Transfer Event are received (4.6.9#1).
12   Perform the following steps to verify that an EP in Error State is put to Stopped State when a Set TR Dequeue Pointer is issued:
    12.1   Configure the Bulk OUT Endpoint of the Compliance device to always Timeout.
    12.2   Prepare a TD with 1 TRB to transfer 1KB worth data.
    12.3   Ring the doorbell.
    12.4   Verify the host returns a Transfer Event with Completion code of USB Transaction Error.
    12.5   Reset endpoint.
    12.6   Perform Set TR Dequeue Pointer command targeting the Bulk OUT EP (Endpoint ID=2, Endpoint address=1 with Direction=OUT), with the TR Dequeue Pointer pointing to the base of the Bulk OUT Transfer Ring.

12.7    Read the Device Context and verify that the
        12.7.1 EP State is set to Stopped in the Bulk Endpoint Context.(4.8.3#5)

**Repetitions**

Repeat for the DCS flag value 1 and 0.

## TD.4.08 Reset Device Command Test

This test verifies the xHC's operation of a Reset Device Command.

**Required Device Resource**

FS, HS or SS USB device

**Overview of Test Steps**

The test performs the following steps.

1   Initialize and start the xHC.
2   Prompt a user to attach a USB device to a root hub port if not connected and enable the port.
3   Perform an Enable Slot Command.
4   Perform an Address Device Command with the *BSR* flag set to 0.
5   Perform Port Bandwidth Command for speed of attached device.
    5.1     Save Available Bandwidth
6   Perform a Configure Endpoint Command with the following Input Context settings:
    6.1     Input Control Context:
        6.1.1 *A0, A2* = 1
    6.2     Input Slot Context
        6.2.1 *Context Entries* = 2
    6.3     Input Endpoint 1 OUT Context:
        6.3.1 *EP Type* = 2
        6.3.2 *Max Packet Size* = 1024 (SS) or 512 (HS) or 64 (FS)
        6.3.3 *TR Dequeue Pointer* = base address of the Transfer Ring
        6.3.4 *DCS* = 1
7   Perform Port Bandwidth Command for speed of attached device.
    7.1     Compare available bandwidth to value saved in 5.1
    7.2     Test fails if available bandwidth is not less than or equal to the first value. (4.6.15#1)
8   Save the values from the following fields in the Endpoint Context for EP0:
    8.1     MaxPacketSize
    8.2     EP Type
    8.3     CErr
    8.4     TRDequeuePointer
    8.5     Average TRB Length
9   Reset the port.
10  Perform a Reset Device Command.
11  Read the Device Context and verify the followings:
    11.1    The *Slot State* field is set to Default in the Slot Context (4.5.3.5#5).
    11.2    The *USB Device Address* field is set to 0 in the Slot Context (4.5.3.4#1).
    11.3    The Contet Entries field in the Slot Context is set to 1 (4.6.11#7).
12  Read the Endpoint Context for EP 0and verify the following:
    12.1    The Endpoint State is 3, Stopped or 1, Running (4.6.11#2)
    12.2    The following fields are the same as before the Reset Device Command: (6.2.3.7#1)
        12.2.1 MaxPacketSize
        12.2.2 EP Type
        12.2.3 CErr
        12.2.4 TRDequeuePointer
        12.2.5 Average TRB Length
    12.3    The following fields have been reset to 0: (6.2.3.7#2)
        12.3.1 Mult

12.3.2 MaxPStreams
12.3.3 LSA
12.3.4 Interval
12.3.5 Max ESIT Payload
12.3.6 Max Burst Size
12.3.7 HID

13  If xHC supports GetPortBandwidth:
    13.1    Perform Port Bandwidth Command for speed of attached device.
        13.1.1 Compare available bandwidth to value saved in step 5.1
        13.1.2 Test fails if available bandwidth is not the same as the value saved in step 5.1 (4.6.11#5
14  Prepare a TRB in the Transfer Ring for Endpoint 1 OUT, ring the doorbell specifying this endpoint and
    verify that no event is generated for this transfer (4.6.11#4).
15  Perform an Enable Slot Command.
16  Perform an Address Device Command with the *BSR* flag set to 0.
17  Perform a Configure Endpoint Command.
18  Perform a Reset Device Command.
19  Read the Device Context and verify the followings:
    19.1    The *Slot State* field is set to Default in the Slot Context (4.5.3.5#5).
    19.2    The *USB Device Address* field is set to 0 in the Slot Context (4.5.3.4#1).
    19.3    The *EP State* field is set to Disabled in all the non-default Endpoint Contexts (4.6.11#1).

### TD.4.09 Negotiate Bandwidth Command Test

This test verifies the xHC's operation of a Negotiate Bandwidth Command.

**Required Device Resource**

USB device

**Overview of Test Steps**

The test performs the following steps if the xHC supports bandwidth negotiation, i.e. the *BNC* flag is set to 1
in the HCCPARAMS register.

1    Initialize and start the xHC.
2    Prompt a user to attach a device to a port if it is not attached.
3    Repeat the following steps for *MaxSlots* number of times or until the Configure Endpoint Command fails
     due to Bandwidth Error or Resource Error:
    3.1     Perform an Enable Slot Command.
    3.2     Perform an Address Device Command with the *BSR* flag set to 0.
    3.3     Perform a Configure Endpoint Command with one or more periodic endpoints except for the first
            slot, i.e. Slot ID 1.
    3.4     Reset the root hub port this device is attached to.
4    Issue a Negotiate Bandwidth Command with the *Slot ID* field set to 1 and verify the followings:
    4.1     The Negotiate Bandwidth Command succeeds (4.11.4.12#1).
    4.2     One or more Bandwidth Request Events are generated (4.6.13#1).
    4.3     The *Slot ID* field is set to an assigned value in the Bandwidth Request Events (4.6.13#2).
    4.4     The *Completion Code* field is set to Success in the Bandwidth Request Events (4.6.13#3).
    4.5     The Bandwidth Request Event is generated on the Event Ring specified by the *Interrupter Target*
            field in the Slot Context for the target device slot (4.9.4.3#3).

**Repetitions**

Repeat for all the interrupters supported by the xHC.

### TD.4.10 Set Latency Tolerance Value Command Test

This test verifies the xHC's operation of a Set Latency Tolerance Value Command.

**Required Device Resource**

None

**Overview of Test Steps**

The test performs the following steps.

1    Initialize and start the xHC.
2    If the *LTC* flag is 1 in the HCCPARAMS register, perform the following steps:
    2.1    Insert a Set Latency Tolerance Value Command into the Command Ring and ring the Host Controller Doorbell.
    2.2    Wait for a Command Completion Event and verify the followings:
        2.2.1    A Command Completion Event is received within 100 ms.
        2.2.2    The *Completion Code* field is set to Success (4.11.4.13#1).
        2.2.3    The *Slot ID* field is set to 0 (6.4.2.2#1).
        2.2.4    The *Command Completion Parameter* is set to 0 (6.4.2.2#2).

**Repetitions**

Repeat for all the possible BELT values.

## TD.4.11 Get Port Bandwidth Command Test

This test verifies the xHC's operation of a Get Port Bandwidth Command.

**Required Device Resource**

USB Device

**Overview of Test Steps**

The test performs the following steps by changing the device speed and the root hub or external hub port number.  If there are integrated devices,

1    Initialize and start the xHC.
2    Perform Get Port Bandwidth Commands with the *Dev Speed* field set to LS, FS and HS
3    If Command is completed with TRB Error Completion, test is complete for current speed.
4    If Command did not complete with TRB Error then verify the following:

    4.1    For USB3 ports, Port Bandwidth Context is set to 0 (4.6.15#2).
    4.2    For USB2 ports, without any integrated USB2 devices:
        4.2.1    When the *Dev Speed* is HS, Port Bandwidth Context is set to 80 or 0[1].
        4.2.2    When the *Dev Speed* is FS or LS, Port Bandwidth Context is set to 90 or 0.
    4.3    For USB2 ports, with an integrated USB2 device:
        4.3.1    When the *Dev Speed* is HS and the integrated device is HS, Port Bandwidth Context is less than or equal to 80.
        4.3.2    When the *Dev Speed* is FS or LS and the integrated device is LS or FS, Port Bandwidth Context is less than or equal to 90.
5    Perform Get Port Bandwidth Commands with the *Dev Speed* field set to SS and verify the followings.
    5.1    For USB2 ports, Port Bandwidth Context is set to 0 (4.6.15#3).
    5.2    For USB3 ports, Port Bandwidth Context is set to 90 or 0.
6    Prompt a user to attach a device to a port if no device is attached to the current root hub port.
7    Perform an Enable Slot Command.
8    Perform an Address Device Command (*BSR* = 0).
9    Perform a Configure Endpoint Command with Input Context including periodic endpoints.
10   Perform Get Port Bandwidth Commands with the *Dev Speed* field set to LS, FS and HS then verify the followings.
    10.1    For USB3 ports, Port Bandwidth Context is set to 0 (4.6.15#2).
    10.2    For USB2 ports:

---

[1] The xHC may consider no bandwidth is available if a USB3 device is attached to the physical connector this USB2 port is wired to.

10.2.1 When the *Dev Speed* is HS, Port Bandwidth Context is set to a value less than 80 if this is the current port and the current speed is HS, otherwise it is set to a value less than or equal to 80.

10.2.2 When the *Dev Speed* is FS or LS, Port Bandwidth Context is set to a value less than 90 if this is the current port and the current speed is FS or LS, otherwise it is set to a value less than or equal to 90.

11  Perform Get Port Bandwidth Commands with the *Dev Speed* field set to SS and verify the followings.

11.1    For USB2 ports, Port Bandwidth Context is set to 0 (4.6.15#3).

11.2    For USB3 ports, Port Bandwidth Context is set to a value less than 90 if this is the current port or there is an integrated USB3 device, otherwise it is set to a value less than or equal to 90.

12  Perform a Disable Slot Command.

13  Perform Get Port Bandwidth Commands with the *Dev Speed* field set to LS, FS and HS then verify the followings.

13.1    For USB3 ports, Port Bandwidth Context is set to 0 (4.6.15#2).

13.2    For USB2 ports:

13.2.1 When the *Dev Speed* is HS, Port Bandwidth Context is set to 80 (4.6.4#1) or 0.

13.2.2 When the *Dev Speed* is FS or LS, Port Bandwidth Context is set to 90 (4.6.4#1) or 0.

14  Perform Get Port Bandwidth Commands with the *Dev Speed* field set to SS and verify the followings.

14.1    For USB2 ports, Port Bandwidth Context is set to 0 (4.6.15#3).

14.2    For USB3 ports, Port Bandwidth Context is set to 90 (4.6.4#1) or 0.

**Repetitions**

Repeat for all ports, both root hub and external hub ports.
And repeat for LS, FS, HS and SS.

## TD.4.12 Force Header Command Test

This test verifies the xHC's operation of a Force Header Command.

**Required Device Resource**

USB 3.0 Device

**Overview of Test Steps**

The test performs the following steps.

1    Initialize and start the xHC.
2    Attach a USB 3.0 device to a root hub port.
3    Wait for a Port Status Change Event then read the PORTSC register and verify that the port is successfully enabled
4    For the following Packet Types (LMP, TP, DPH):

4.1    Insert a Force Header Command into the Command Ring and ring the Host Controller Doorbell.

4.2    Wait for a Command Completion Event and verify the followings:

4.2.1  The *Completion Code* field is set to Success (4.6.1#5).

4.2.2  The *Slot ID* field is set to 0 (4.6.16#1, 6.4.2.2#1).

4.2.3  The *Command Completion Parameter* is set to 0 (6.4.2.2#2).

**Repetitions**

Repeat for all the USB3 root hub ports.

## TD.4.13 Disable Slot Test

This test verifies the xHC's operation of Disable Slot

**Required Device Resource**

USB 3.0 Device

**Overview of Test Steps**

The test performs the following steps.

1. Initialize and start the xHC

2. Prompt the user to attach a USB device to a port if no device is attached.

3. Reset the port if the port is not enabled

4. Allocate an Input Context and initialize it as follows:
   a. Slot Context
      i. *Speed* = PORTSC.Port Speed
      ii. *Context Entries* = 1
      iii. *Root Hub Port Number* = XXX
   b. Endpoint 0 Context
      i. *CErr* = 3
      ii. *EP Type* = 4 (Control)
      iii. *Max Packet Size* = 512 (SS) or 64 (HS) or 8 (FS, LS)
      iv. *TR Dequeue Pointer* = base address of the Transfer Ring
      v. *DCS* = 1
   c. All the other Input Contexts are initialized by non-zero values.
5. Perform an Address Device Command with the *BSR* flag set to 1.
6. Disable the slot

7. Perform an Address Device Command with BSR flag set to 0.

8. Verify that the xHC does not send the address device command (4.5.3.2#2, 5.4.7#3)

## TD.4.14 Secondary Bandwidth Domain Test

This test verifies the xHC's operation of secondary bandwidth domain management.

**Required Device Resource**

High-speed hub, two full-speed devices and high-speed device

**Overview of Test Steps**

The test performs the following steps.

1 Initialize and start the xHC.
2 Attach a high-speed hub to a root hub port or a downstream port of an integrated USB 2.0 hub. No device is attached to downstream ports of the high-speed hub.
3 Enumerate the high-speed hub. Note that the hub must be operating as single-TT regardless of multi-TT capable or not.
4 Allocate a Port Bandwidth Context that can contain bandwidth information for all the downstream ports of the hub and all the root hub ports, whichever the more.
5 Perform a Get Port Bandwidth command with the Hub Slot ID field set to the slot ID assigned to the hub and the Dev Speed field set to full-speed then verify the followings:
   5.1 Command succeeds.
   5.2 In case of xHCI 1.0 or xHCI 0.96 with SBD = 1, available bandwidth is 90% for all the ports (4.16.2#3).
   5.3 In case of xHCI 0.96 with SBD = 0, available bandwidth is 0 % for SS root hub ports (4.16.2#2). Note that following steps are skipped for xHCs without secondary bandwidth reporting capability.
6 Attach a full-speed device to a downstream port of the high-speed hub.
7 Perform the following commands.
   7.1 Enable Slot
   7.2 Address Device (BSR = 0)
   7.3 Verify the Output Slot Context is updated with the parameters specified in the Input Slot Context (including the TTPortNumber & TTHubSlotID) (6.2.2#1, 6.2.2#2, 6.2.2#3, 6.2.2#6, 6.2.2#7, 6.2.2#8, 6.2.2.#9)

7.4   Configure Endpoint with the following endpoint settings.
    7.4.1   Endpoint 1
        7.4.1.1   EP Type = 1 (Isoch OUT)
        7.4.1.2   Max Packet Size = 1023
        7.4.1.3   Interval = 3 (1 ms)
    7.4.2   Endpoint 2
        7.4.2.1   EP Type = 5 (Isoch IN)
        7.4.2.2   Max Packet Size = 1023
        7.4.2.3   Interval = 3 (1 ms)
7.5   Verify that the command fails with completion code set to Secondary Bandwidth Error (4.6.6#8, 4.16.2#1).
7.6   Get Port Bandwidth
    7.6.1   Hub Slot ID = slot ID assigned to the hub
    7.6.2   Dev Speed = full-speed.
7.7   Verify that available bandwidth is 90% for all the ports (4.16.2#3).
7.8   Configure Endpoint with the following endpoint settings.
    7.8.1   Endpoint 1
        7.8.1.1   EP Type = 1 (Isoch OUT)
        7.8.1.2   Max Packet Size = 1023
        7.8.1.3   Interval = 3 (1 ms)
7.9   Verify that the command succeeds.
7.10   Get Port Bandwidth
    7.10.1 Hub Slot ID = slot ID assigned to the hub
    7.10.2 Dev Speed = full-speed.
7.11   Verify that available bandwidth is less than 90% for the port the device is connected to (4.16.2#3).

8   Attach the other full-speed device to another downstream port of the high-speed hub.
9   Perform the following commands.
9.1   Enable Slot
9.2   Address Device (BSR = 0)
9.3   Configure Endpoint with the following endpoint settings.
    9.3.1   Endpoint 1
        9.3.1.1   EP Type = 5 (Isoch IN)
        9.3.1.2   Max Packet Size = 1023
        9.3.1.3   Interval = 3 (1 ms)
9.4   Verify that the command fails with completion code set to Secondary Bandwidth Error (4.6.6#8).
9.5   Get Port Bandwidth
    9.5.1   Hub Slot ID = slot ID assigned to the hub.
    9.5.2   Dev Speed = full-speed
    9.5.3   Verify that available bandwidth is the same as the value returned in step 6.10 (4.16.2#3).

10   Attach a high-speed device to the high-speed hub.
11   Perform the following commands.
11.1   Enable Slot
11.2   Address Device (BSR = 0)
11.3   Configure Endpoint with the following endpoint settings.
    11.3.1 Endpoint 1
        11.3.1.1 EP Type = 1 (Isoch OUT)
        11.3.1.2 Max Packet Size = 1024
        11.3.1.3 Interval = 0 (125 us)
        11.3.1.4 Max Burst Size = 2
11.4   If the Configure Endpoint Command results in bandwidth error, decrease max packet size and retry the command until it succeeds.
12   Perform the following steps targeting the second full-speed device.
12.1   Perform a Configure Endpoint Command with the following endpoint settings.
    12.1.1 Endpoint 1
        12.1.1.1 EP Type = 5 (Isoch IN)

12.1.1.2 Max Packet Size = 1023
12.1.1.3 Interval = 3 (1 ms)
12.2 Verify that the completion code is set to Bandwidth Error or Secondary Bandwidth Error (4.6.6#9).

## TD.4.15 Force Stopped Event Test

This test verifies the xHC's operation of Force Stopped Event of generating a Stopped Transfer Event.

### Required Device Resource

LS Compliance Device or SS Compliance Device (Microsoft SuperMUTT)

### Overview of Test Steps

The test performs the following steps.

1   For each test case, perform the following
2   Initialization:
    2.1   Initialize and start the xHC.
    2.2   Prompt the user to attach a Compliance Device in the first run of this test if it is not already attached.
    2.3   Perform an Enable Slot Command.
    2.4   Perform an Address Device Command with the *BSR* flag set to 0.
    2.5   Perform a Configure Endpoint.
3   Cleanup:
    3.1   Deconfigure all Endpoints
    3.2   Stop host controller and clean up all resources

Test Case 1
1.   Configure Interrupt IN and OUT Endpoints for loopback with MaxPacketSize = 8.
2.   Issue Stop Endpoint Command for Interrupt OUT EP (LS compliance device) or IN EP (other speeds compliance device)
3.   Verify that Transfer Event is received with these characteristics:
    a.   Completion Code = TrbStoppedLengthInvalid (4.6.9#27)
    b.   TRB Transfer Length = 0 (4.6.9#27)
    c.   Interrupter equal to the interrupter specified in the Slot Context
    d.   TRBPointer = address of enqueue pointer TRB (4.6.9#27)
4.   Verify that Command Completion event is received with these characteristics:
    a.   Completion Code = TrbSuccess (4.6.9#27)
    b.   Interrupter = 0

Test Case 2
1.   Configure Interrupt IN and OUT Endpoints for loopback with MaxPacketSize = 8.
2.   Queue Interrupt OUT TD that is large enough to still be processing when endpoint is stopped.
3.   Ring Doorbell.
4.   Issue Stop Endpoint Command for Interrupt OUT EP.
5.   Verify that Transfer Event is received with these characteristics:
    a.   Completion Code = TrbStopped (4.6.9#8, 4.6.9#30 or 4.6.9#2)
    b.   TRB Transfer Length <= TD transfer size
    c.   Interrupter equal to the interrupter specified in the Interrupter Target field (4.9.4.3#1)
6.   Verify that Command Completion event is received with these characteristics:
    a.   Completion Code = TrbSuccess
    b.   Interrupter = 0
7.   Verify that there are no additional events generated (4.6.9#11, 4.8.3#6)

Test Case 3 – LS compliance device
1.   Configure Interrupt IN and OUT Endpoints for loopback with MaxPacketSize = 8.
2.   Configure Interrupt OUT EP to always NAK

3. Queue an 8-byte TD to the Interrupt OUT EP.
4. Queue a 32-byte TD to the Interrupt OUT EP.
5. Ring Doorbell.
6. Issue Stop Endpoint Command for Interrupt OUT EP.
7. Verify that Transfer Event is received with these characteristics;
    a. Completion Code = TrbStopped (4.6.9#8, 4.6.9#30 or 4.6.9#2)
    b. TRB Transfer Length = 8 (4.6.9#18)
    c. Interrupter equal to iunterrupter specified in Interrupter Target field (4.9.4.3#1)
8. Verify that Command Completion Event is received with these characteristics:
    a. Completion Code = TrbSuccess
    b. Interrupter = 0
9. Verify that no further events are generated (4.6.9#11, 4.6.10#5)
10. Change Interrupt OUT EP back to normal operation
11. Reverse Cycle bit on 2$^{nd}$ TRB, invalidating location in ring
12. Ring Doorbell
13. Verify that no events are generated (4.6.9#7)

Test Case 3 – FS/HS/SS compliance device
1. Configure Interrupt IN and OUT Endpoints for loopback with MaxPacketSize = 8.
2. Queue several 8-byte TDs for Interrupt OUT EP.
3. Ring Doorbell.
4. Issue Stop Endpoint command for Interrupt OUT EP.
5. Verify that one Transfer Event is received with these characteristics:
    a. Completion Code = TrbStopped (4.6.9#8, 4.6.9#30 or 4.6.9#2)
    b. Transfer Length <= 8 (4.6.9#18)
    c. Interrupter equal to interrupter specified in Interrupter Target field (4.9.4.3#1)
6. Verify that a Command Completion Event is received with these characteristics:
    a. Completion Code = TrbSuccess
    b. Interrupter = 0
7. Verify that no additional events are generated after Command Completion Event (4.6.9#11, 4.6.10#5)
8. Based on the number of Transfer Events received, reverse the cycle bit on the next TD on the Interrupt OUT EP ring.
9. Ring Doorbell
10. Verify that no events are generated (4.6.9#7)

Test Case 4
1. Configure Interrupt IN and OUT Endpoints for loopback with MaxPacketSize = 8.
2. Configure Interrupt IN and OUT Endpoints to STALL.
3. Queue 3-32-byte TDs on Interrupt OUT EP (LS) or Interrupt IN EP (FS/HS/SS)
4. Ring Doorbell
5. Verify that a Transfer Event is received with these characteristics:
    a. Completion Code = TrbStallError
6. Issue Stop Endpoint Command for OUT EP (LS) or IN EP (FS/HS/SS)
7. Verify that a Command Completion Event is received with these characteristics (4.6.9#10):
    a. Completion Code = TrbContextStateError
    b. Interrupter = 0
    c. Verify that the Output Context state for the EP is not 1 (4.6.9#4)
8. Verify that a Transfer Event I sreceived with these characteristics:
    a. If (SPC)
        i. Completion Code is TrbStoppedShortPacket (4.6.9#28)
    b. Else
        i. Completion Code is TrbStopped
    c. Interrupter equal to interrupter specified in Interrupter Target field (4.9.4.3#1)
9. Verify that no events are received after Stop Endpoint command is completed (4.6.9#11)

Repeat for all supported interrupters.
Repeat for LS and SS compliance device.

### TD.4.16 Invalid Command Test

This test verifies the xHC behavior for unrecognized command.

**Required Device Resource**

**Overview of Test Steps**

The test performs the following steps.

1    Initialize and start the xHC.
2    Initialize a Command TRB with TRB type 40 (Reserved)
3    Ring Doorbell
4    Verify that xHC returns TRB Error (4.6#1)

### TD.4.17 Set TR Dequeue Pointer - Streams

Verify Set TR Dequeue Pointer on endpoint with streams enabled.

**Required Device Resource**

SS Compliance Device (with stream support – currently not supported)

**Overview of Test Steps**

The test performs the following steps.

1.    Connect an SS compliance device.
2.    Configure the device slot with Stream Bulk endpoint.
3.    Perform a Set TR Dequeue Pointer Command.
4.    Verify that Stream Context is updated with the specified dequeue pointer (4.6.10#2) and DCS value (4.6.10#4).
5.    Perform a Stream bulk transfer.
6.    Perform a Stop Endpoint Command and verify that Stream Context is updated with the current dequeue pointer (6.2.4.2#1).

### TD.4.18 Stream Array Bounds Checking

Streams Array test.

**Required Device Resource**

SS Compliance Device (with stream support – currently not supported)

**Overview of Test Steps**

The test performs the following steps.

1.    TBD

### TD.4.19 Configure All Endpoints Test

Verify that all endpoints can be configured for bulk, interrupt and isochronous endpoint types.

**Required Device Resource**

SS Compliance Device

**Overview of Test Steps**

The test performs the following steps.

1. Connect an SS compliance device.
2. For each endpoint numbers, 1 to 15:
    a. Create input context for bulk IN endpoint
    b. Create input context for bulk OUT endpoint
    c. Set slot context Context Entries to DCI of IN endpoint
    d. For all endpoints not being test in this iteration, set context to 0xFF.
    e. Issue Configure Endpoint command.
    f. Verify that command event completion code is Success (4.6.1#5)
    g. Read the output slot context and verify the following:
        i. Slot State is Configured (4.3.5#1)
        ii. Endpoint type is bulk(6.2.3#7)
        iii. Error Count is same as input context (6.2.3#6)
        iv. MaxPacketSize is same as input context (6.2.3#10)
        v. Dequeue pointer is at top of TRB ring (6.2.3#12)
        vi. DCS is same as input context (6.2.3#11)
        vii. Endpoint state is running (4.6.6#3)
        viii. LSA field is same as input context (6.2.3#3)
        ix. HID field is same as input context (6.2.3#8)
    h. Verify that output contexts for all endpoints not being tested in this iteration are 0 (3.3.5#1)
    i. Set the cooresponding bits for the bulk IN and OUT endpoints for this iteration in the Drop Context flag structure.
    j. Issue Configrue Endpoint command
    k. Verify that command event completion is Success.
    Repeat for interrupt and Isochronous endpoints.


## TD.4.20 Slot Speed Test

Test slot speed for SS Gen1 and Gen2 devices.

**Required Device Resource**

SS Gen1 and Gen2 devices

**Overview of Test Steps**

The test performs the following steps.

SuperSpeed Gen1

1. Connect an SS Gen1 device.
    a. Enable slot for device using the port speed reported in PORTSC
    b. Write 1 to bit 5 of DNCTRL register to enable Sublink Speed notifications
    c. Issue SetAddress (BSR=0)
    d. Verify that no dev notifications were received
    e. Reset device
    f. Disable slot.
2. Detach device

SuperSpeed Gen2
3. Connect a SS Gen2 device
    a. Enable slot for device using the port speed reported in PORTSC
    b. Issue SetAddres (BSR=0)
    c. Verify that 2 dev notifications were received
    d. Verify for first dev notification:
        i. TRB type is DeviceNotificationEvent (5.4.4#3)
        ii. Interrupter is 0

        iii.   NotificationType is USB_NOTIFICATION_SUBLINK_SPEED.
        iv.   Link is Symmetic (0)
         v.   Direction is RX (0)
        vi.   LSE is 3 (GBS)
       vii.   Lane Count is 0.
      viii.   LP is 1 (SSP)
        ix.   LSM matches value for $2^{nd}$ TRB
         x.   Slot ID is for connected device (6.4.2.7#4)

e.  Verify for second dev notification:
         i.   TRB type is DeviceNotificationEvent (5.4.4#3)
        ii.   Interrupter is 0
        iii.   NotificationType is USB_NOTIFICATION_SUBLINK_SPEED
        iv.   Link is Symmetric (0)
         v.   Direction is TX (1)
        vi.   LSE is 3 (GBS)
       vii.   Lane Count is 0.
      viii.   LP is 1 (SSP)
        ix.   LSM matches value for $1^{st}$ TRB
         x.   Slot ID is for connected device (6.4.2.7#4)

f.  Reset device.

g.  Disable slot

4.  Detach device

## 3.5  Transfer Tests

- Ports should be used for transfer tests on a round-robin basis, so that all ports get used for testing.  It is not necessary to run all transfer tests on all ports.

- Loopback tests allow for LS/FS/HS device errors and will try to reset the device or prompt the user to manually unplug and reset the test device.  This retry behavior is only valid if the error is caused by a device error.  If the error is caused by the host, this will result in a test failure.

### TD.5.01 Control Loopback Test

This test performs functional testing of control transfers.

**Required Device Resource**

USB 2.0 and 3.0 compliance devices

**Overview of Test Steps**

The test performs the following steps.

1    Initialize and start the xHC.
2    Prompt a user to attach a USB compliance device to a port of the host if it is not attached.
3    Configure the compliance device for control endpoint loopback.
4    Schedule a control write transfer to the control endpoint using compliance device vendor command.
5    Check the result of the transfer.
6    Schedule a control read transfer to the control endpoint using compliance device vendor command.
7    Check the result of the transfer.
8    Verify that the OUT data matches the IN data (same data in the same order).

**Repetitions**

Repeat for all allowed control transfer sizes (from 1 to EP0 MaxPacketSize).

And repeat for all allowed max packet size for control transfers:

- For LS: 8 bytes
- For FS: 8, 16, 32, and 64 bytes
- For HS: 64 bytes
- For SS: 512 bytes

And repeat for LS, FS, HS and SS compliance device.
And repeat for all byte repeating data patterns.
Run test with and without Event Data TRB.

## TD.5.02 Bulk Loopback Tests

This test performs functional testing of bulk transfers.

### Required Device Resource

USB 2.0 and 3.0 compliance devices

### Overview of Test Steps

The test performs the following steps.

1. Initialize and start the xHC.
2. Prompt a user to attach a USB compliance device to a port of the host if it is not attached.
3. Configure the compliance device for bulk endpoint loopback.

### TD 5.02.01 Bulk Loopback using Scatter-Gather List

4. Perform loopback using scatter-gather list of data buffers.
    - 4.1 Schedule a bulk OUT transfer to the bulk OUT endpoint.
    - 4.2 Check the result of the transfer.
    - 4.3 Schedule a bulk IN transfer to the bulk IN endpoint.
    - 4.4 Check the result of the transfer.
    - 4.5 Repeat for 10 iterations.
    - 4.6 Verify that the OUT data matches the IN data (same data in the same order).

### TD 5.02.02 Bulk Loopback using Contiguous Data Buffer

5. Perform loopback using a physically contiguous data buffer.
    - 5.1 Schedule a bulk OUT transfer to the bulk OUT endpoint.
    - 5.2 Check the result of the transfer.
    - 5.3 Schedule a bulk IN transfer to the bulk IN endpoint.
    - 5.4 Check the result of the transfer.
    - 5.5 Repeat for 10 iterations.

### Repetitions

Transfer size for scatter-gather iterations are 16K for SS, 6K for FS and HS.
Transfer size for contiguous buffer iteration is 64K.
And repeat for all possible page offset (from 0 to 4095) for scatter-gather iterations.
And repeat for all allowed max packet size for bulk transfers:

- For FS: 8, 16, 32, and 64 bytes
- For HS: 512 bytes
- For SS: 1024 bytes

And repeat for all allowed max burst size for SS bulk transfers (from 1 to 16).
And repeat for FS, HS and SS compliance device
And repeat for all byte repeating data patterns.
Run test with and without Event Data TRB.

## TD.5.03 Interrupt Loopback Test

This test performs functional testing of interrupt transfers.

**Required Device Resource**

USB 2.0 and 3.0 compliance devices

**Overview of Test Steps**

The test performs the following steps.

1   Initialize and start the xHC.
2   Prompt a user to attach a USB compliance device to a port of the host if it is not attached.
3   Configure the compliance device for interrupt endpoint loopback.

### TD 5.03.01 Interrupt Loopback using Scatter-Gather List

4   Perform loopback using scatter-gather list of data buffers.
    4.1   Schedule an interrupt OUT transfer to the interrupt OUT endpoint.
    4.2   Check the result of the transfer.
    4.3   Schedule an interrupt IN transfer to the interrupt IN endpoint.
    4.4   Check the result of the transfer.
    4.5   Verify that the OUT data matches the IN data (same data and in the same order).
    4.6   Repeat for 10 iterations.

### TD 5.03.02 Interrupt Loopback using Contiguous Data Buffer

5   Perform loopback using physically contiguous data buffer.
    5.1   Schedule a bulk OUT transfer to the bulk OUT endpoint.
    5.2   Check the result of the transfer.
    5.3   Schedule a bulk IN transfer to the bulk IN endpoint.
    5.4   Check the result of the transfer.
    5.5   Repeat for 10 iterations.

**Repetitions**

Transfer size for scatter-gather iterations are 16K for SS, 6K for FS and HS.
Transfer size for contiguous buffer iteration is 64K.
And repeat for all possible page offset (from 0 to 4095) for scatter-gather iterations.
And repeat for all allowed max packet size for interrupt transfers (1 to 1024, dependent on burst size).
And repeat for all allowed max burst size for SS or HS interrupt transfers (1 to 3).
And repeat for LS, FS, HS and SS compliance device
And repeat for all byte repeating data patterns.
Run test with and without Event Data TRB.

## TD.5.04 Isochronous Loopback Test

This test performs functional testing of isochronous transfers.

**Required Device Resource**

USB 2.0 and 3.0 compliance devices

**Overview of Test Steps**

The test performs the following steps.

1   Initialize and start the xHC.
2   Prompt a user to attach a USB compliance device to a port of the host if it is not attached.

### TD 5.04.01 Test isochronous OUT.

    2.1   Configure compliance device for isochronous endpoint OUT looped back to bulk IN.
    2.2   Write data buffer of MaxPacketSize * Burst.  This will be contained in 1 isochronous TD.
    2.3   Read data buffer back.
    2.4   Verify that the OUT completed successfully.  xHCI *CompletionCode* field should be set to TrbSuccess.

2.5  Verify that the IN completed successfully.

2.6  Verify that the OUT data matches the IN data (same data and in same order).

2.7  Verify that the host generated a Transfer Event with *CompletionCode* set to Ring Underrun for the OUT endpoint (4.10.3.1#2).

2.8  Verify that the EP State field is set to Running in the EndpointContext for the OUT endpoint (4.11.2.3#1).

2.9  Repeat for 10 iterations.

2.10  Add steps for 4.11.2.3#5 and 4.11.2.3#6

### TD 5.04.02 Test Isochronous IN.

2.11  Configure compliance device for isochronous endpoint IN looped back to bulk IN.

2.12  Write data buffer of MaxPacketSize * Burst. This will be contained within 1 isochronous TD.

2.13  Read data buffer back.

2.14  Verify that the OUT completed successfully.

2.15  Verify that the IN completed successfully.  xHCI *CompletionCode* field should be set to TrbSuccess.

2.16  Verify that the OUT data matches the IN data (same data and in same order).

2.17  Verify that the host generated a Transfer Event with *CompletionCode* set to Ring Overrun for the IN endpoint (4.10.3.1#1).

2.18  Verify that the EP State field is set to Running in the EndpointContext for the IN endpoint (4.11.2.3#1).

2.19  Repeat for 10 iterations

2.20  Add steps for 4.11.2.3#5 and 4.11.2.3#6

2.21

### TD 5.04.03 Test Isochronous Streaming OUT

2.22  Configure compliance device for isochronous endpoint OUT looped back to bulk IN.

2.23  Stream continuous isochronous traffic of MaxPacketSize * Burst for 30 minutes.

2.24  Verify that the Isochronous OUT completed successfully.

2.25  Verify that bulk INT data matches the IN data.

2.26  Test fails if any Ring Overrun or Ring Underrun events are generated.

2.27  After last Isochronous packet verify that a Ring Underrun is generated.

### TD 5.04.04 Test Isochronous Streaming IN

2.28  Configure compliance device for isochronous endpoint IN looped back to bulk OUT.

2.29  Stream continuous isochronous traffic of MaxPacketSize * Burst for 30 minutes.

2.30  Verify that the Isochronous IN completed successfully.

2.31  Verify that bulk OUT data matches the IN data.

2.32  Test fails if any Ring Overrun or Underrun events are generated.

2.33  After last Isochronous packet verify that a Ring Overrun is generated.

### TD 5.04.05 Isochronous ESIT Test

2.34  Start analyzer.

2.35  Issue an Isoch TD with SIA set to 0 and Frame ID set to the current MFINDEX register value minus one.

2.36  Stop analyzer.

2.37　Verify that the transfer is executed successfully without waiting for the next opportunity MFINDEX register matches the specified Frame ID.

### TD 5.04.06 Isochronous 0-Length Packet Test

1　Verify 6.4.1#1

2　Start analyzer.

3　Issue an Isoch TD with TRB transfer length set to zero.

4　Stop analyzer.

5　Verify that zero-length transaction was executed (4.11.2.1#2).

#### .Repetitions

Repeat for transfer size of (MaxPacketSize * Burst / 2) + 1.
And repeat for all allowed max packet size for isochronous transfers:
And repeat for all allowed max burst size for SS or HS isochronous transfers (1 to 16 for SS and 1 to 3 for HS).
And repeat for all allowed number of bursts (mult) for SS.
And repeat for HS and SS compliance device
And repeat for FS with isochronous out but no loopback, because of compliance device limitations.
Run test with and without Event Data TRB.

## TD.5.05 Control Loopback behind High Speed Hub Test

This test performs functional testing of control transfers behind a high-speed hub.

### Required Device Resource

USB 2.0 compliance devices

### Overview of Test Steps

The test performs the 0 with the following changes.

● Connect the compliance device behind 1 high-speed hub to the xHC.

### Repetitions

Repeat for LS, FS and HS compliance device.

## TD.5.06 Bulk Loopback behind High Speed Hub Test

This test performs functional testing of bulk transfers behind a high-speed hub.

### Required Device Resource

USB 2.0 compliance devices

### Overview of Test Steps

The test performs the 0 with the following changes.

● Connect the compliance device behind 1 high-speed hub to the xHC.

### Repetitions

Repeat for FS and HS compliance device.

## TD.5.07 Interrupt Loopback behind High Speed Hub Test

This test performs functional testing of interrupt transfers behind a high-speed hub.

### Required Device Resource

USB 2.0 compliance devices

**Overview of Test Steps**

The test performs the 0 with the following changes.

● Connect the compliance device behind high-speed hubs to the xHC.

**Repetitions**

Repeat for LS, FS and HS compliance device.

## TD.5.08 Isochronous Loopback behind High Speed Hub Test

This test performs functional testing of isochronous transfers behind a high-speed hub.

**Required Device Resource**

USB 2.0 compliance devices

**Overview of Test Steps**

The test performs the 0 with the following changes.

● Connect the compliance device behind 1 high-speed hub to the xHC.

**Repetitions**

Repeat for FS and HS compliance device.

## TD.5.09 Short Packet Test

This test performs functional testing of short packet.

**Required Device Resource**

USB compliance devices

**Overview of Test Steps**

The test performs the following steps.

1    Initialize and start the xHC.
2    Prompt a user to attach a USB compliance device to a port if it is not attached.
3    Configure the compliance device for bulk transfer loopback.
4    Initialize the OUT buffer to a known state.
5    Initialize 2 IN buffers to 0.
6    Initiate 2 IN transfers of length 1024.
7    Perform an OUT transfer of length 1.
8    Wait for first IN transfer to complete, then verify the following:
    8.1    The xHC generates a TransferEvent (4.10.1#7).
    8.2    The *Completion Code* field is set to Short Packet.
    8.3    The *TRB Transfer Length* is set to the OUT transfer size.
    8.4    The IN data matches the OUT data
9    Perform a second OUT transfer length of 1.
10   Wait for the second IN transfer to complete, then verify the following:
    10.1    The xHC generates a TransferEvent (4.10.1#7).
    10.2    The *Completion Code* field is set to Short Packet.
    10.3    The *TRB Transfer Length* is set to the OUT transfer size.
    10.4    The IN data matches the OUT data.

**Repetitions**

Repeat for OUT transfer sizes from 1 to 1024, skipping multiples of MaxPacketSize.

And repeat for IN transfer size of 6144 bytes, OUT transfer sizes from 1 to 6144, skipping multiples of MaxPacketSize.

Repeat for FS, HS and SS compliance device.

Repeat using Interrupt loopback for LS, FS, HS and SS compliance device.

Repeat using Isoch loopback for FS, HS and SS compliance device


### TD.5.11 USB Transaction Error Test

This test performs functional testing of USB transaction errors.

**Required Device Resource**

USB compliance devices

**Overview of Test Steps**

The test performs the following steps.

1. Initialize and start the xHC.
2. Prompt a user to attach a USB compliance device to a port if it is not attached.
3. Configure the compliance device response for a bulk OUT endpoint to always generate a CRC error:
   - 3.1 For HS, set the *bFixedRespMode* field to 08h in the endpoint change data when performing a VEN_Change_EP request.
   - 3.2 For SS, perform a VEN_Set_LCW_CRC_Error request with the *Packet Count* set to 1.
4. Perform an OUT transfer with a TD which only contains a Normal TRB with *IOC* = 0 and verify the followings.
   - 4.1 The xHC generates a Transfer Event (4.10.1#8).
   - 4.2 The *Completion Code* field is set to USB Transaction Error (4.10.2.3#5 for HS, 4.10.2.3#6 for SS).
5. Read the Endpoint Context and verify that the *EP State* field is Halted (4.8.3#1).
6. Verify that TR Dequeue Pointer in Output Endpoint Context points to enqueue position for EP. (6.2.3#16)

7. Perform a Reset Endpoint Command.
8. Configure the compliance device response for a bulk OUT endpoint with no response.
   - 8.1 For HS, set the *bFixedRespMode* field to 04h in the endpoint change data when performing a VEN_Change_EP request.
   - 8.2 For SS, perform a VEN_Set_TO_Rate request with the *Packet Count* set to 1.
9. Perform an OUT transfer with a TD which only contains a Normal TRB with *IOC* = 0 and verify the followings.
   - 9.1 The xHC generates a Transfer Event (4.10.1#8).
   - 9.2 The *Completion Code* field is set to USB Transaction Error (4.10.2.3#7 for HS, 4.10.2.3#9 for SS).
10. Read the Endpoint Context and verify that the *EP State* field is Halted (4.8.3#1).
11. If the device attached is a Super Speed Compliance device, Stop the Endpoint.
12. Set TR Dequeue Pointer back to the base of the Bulk transfer ring but with IOC=1 on the Normal TRB.
13. Configure the SS Compliance device bulk OUT EP to always timeout with appropriate Vendor defined Command.
14. Ring the Doorbell.
15. Start a timer in the test application immediately after the transfer is posted. Stop the timer once an Interrupt is received and verify the following.
    - 15.1 Verify that the timeout measured by the application is nearly same as tHostTransactionTimeout which is between 32 and 5032 micro seconds (4.8.3#7). Note that it is not possible to maintain such a small timing value at the application level. Tester may need to request CPU times when transfer is submitted and when the interrupt is received and take the time difference. Alternate methods like requesting the driver to measure the time may also be used.

15.2     A transfer event is generated with completion code field set to USB Transaction Error (4.10.2.3#13).

15.3     Read the Output Endpoint context and verify that the End Point State is set to Halted (4.8.3#7)

**Repetitions**

Repeat test for HS and SS compliance devices.
And repeat for Bulk and Interrupt endpoints.

## TD.5.12 Babble Detected Error Test

This test performs functional testing of babble detected errors.

**Required Device Resource**

USB compliance devices

**Overview of Test Steps**

The test performs the following steps.

1     Initialize and start the xHC.
2     Prompt a user to attach a USB compliance device to a port of the host if no device is attached.
3     Configure the compliance device for bulk endpoint loopback.
4     Schedule an OUT and an IN transfers with the following transfer sizes.
     4.1     For HS, OUT transfer size is 512 bytes and IN transfer size is 256 bytes.
     4.2     For SS, OUT transfer size is 1024 bytes and IN transfer size is 512 bytes.
5     Verify that the IN transfer completes with a Babble Detected Error (4.10.2.4#1).
6     Read the Endpoint Context and verify that the *EP State* field is set to Halted (4.8.3#1).

**Repetitions**

Repeat test for HS and SS compliance devices.

## TD.5.13 Bulk Stream Protocol Test

This test performs functional testing of bulk stream protocol.

**Required Device Resource**

USB 3.0 compliance devices

**Overview of Test Steps**

1     Initialize and start the xHC.
2     Prompt a user to attach a USB compliance device to a port if it is not attached.
3     Configure the compliance device for bulk endpoint loopback.
4     Perform a VEN_Set_Streams requests to the bulk OUT and IN endpoints.
5     Schedule an OUT and an IN transfers.
6     Check the result of the transfer.
7     Verify that the OUT data matches the IN data (same data in the same order).

**Repetitions**

Repeat for the following sets of stream data arrays.
- Pattern 1
  - OUT: (SID=1, Length=100)
  - IN: (SID=2, Length=100)
- Pattern 2
  - OUT: (SID=1, Length=200), (SID=2, Length=500), (SID=3, Length=300)
  - IN: (SID=3, Length=200), (SID=1, Length=500), (SID=2, Length=400)
- Pattern 3

> ➢ Set the number of stream data entries (*NumStreams*) to the smaller value of the maximum stream data array size, i.e. *wMaxNumOfStreams*, and the maximum number of streams supported by xHC.
> ➢ OUT: SID = 1 to *NumStreams*, Length=3000
> ➢ IN: SID= *NumStreams* to 1, Length=3000

## TD.5.14 Immediate Data Transfer Test

This test performs functional testing of immediate data transfer.

### Required Device Resource

USB compliance devices

### Overview of Test Steps

The test performs the following steps:

1   Initialize and start the host controller.
2   Prompt a user to attach a USB compliance device to a port.
3   Configure the compliance device for control, bulk or interrupt endpoint loopback.
4   Schedule an OUT transfer with the *IDT* flag set to 1 in the Transfer TRB.
5   Set Data Buffer Pointer field of TRB to the value <>
6   Ring the doorbell for the OUT endpoint.
7   Test fails if the OUT transfer fails.
8   Schedule an IN transfer of 8 bytes with IDT set to 0 and using a standard data buffer.
9   Ring the doorbell for the IN transfer.
10  Test fails if the IN transfer fails..
11  Verify that the OUT data matches the IN data (same data in the same order).

### Repetitions

Repeat for Control, Bulk and Interrupt transfers.
And repeat for the transfer data sizes from 1 byte to 8 bytes.
And repeat for SS, HS and FS compliance devices.

## TD.5.15 Device Notification Test

This test performs functional testing of device notifications.

### Required Device Resource

USB 3.0 compliance devices supporting VEN_Set_LTM_Value command (currently not supported).

### Overview of Test Steps

The test performs the following steps:

1   Initialize and start the host controller.
2   Prompt a user to attach a USB 3.0 compliance device to a port.
3   Perform an Enable Slot Command.
4   Perform an Address Device Command with the *BSR* flag set to 0.
5   Write the DNCTRL register with setting the bit 2 (N2) to 0.
6   Perform a VEN_Set_LTM_Value request to the USB 3.0 compliance device with the *wValue* set to TBD and verify that a Device Notification Event is not received.
7   Write the DNCTRL register with setting the bit 2 (N2) to 1.
8   Perform a VEN_Set_LTM_Value request to the USB 3.0 compliance device with the *wValue* set to TBD and verify the followings:
   8.1   A Device Notification Event is received (5.4.4#3).
   8.2   A Device Notification Event is received in the Event Ring specified by the *Interrupter Target* field in the Slot Context (4.9.4#3).
   8.3   The *Notification Type* field is set to 2 in the Device Notification Event (6.4.2.7#1).
   8.4   The BELT value is set the XXXX in the *Device Notification Data* of the Device Notification Event (6.4.2.7#2).

8.5    The *Completion Code* field is set to Success in the Device Notification Event (6.4.2.7#3).

8.6    The *Slot ID* field is set to the ID of the current slot in the Device Notification Event (6.4.2.7#4).

9    Reset the downstream port the compliance device is attached to.

10    Perform a Reset Device Command.

11    Repeat the above steps from step 3 until the Enable Slot Command fails with No Slots Available Error.

**Repetitions**

Repeat for all the interrupters supported by the host controller.
And repeat for all the Slot IDs supported by the host controller.
And repeat for all the possible BELT values.

## TD.5.16 Interrupt Interval Test

This test performs functional testing of interrupt transfer intervals.

**Required Device Resource**

USB 2.0 and 3.0 compliance devices

**Overview of Test Steps**

The test performs the following steps:

1    Initialize and start the xHC.

2    Prompt a user to attach a USB compliance device to a port of the host if it is not attached.

3    Configure the compliance device for an interrupt OUT endpoint.

4    Set bInterval to 1 (endpoint interval 0).

5    Prompt the user to start an analyzer.

6    Put an 8 byte Interrupt OUT on the transfer ring and ring the doorbell.

7    Prompt the user to stop the analyzer.

8    Have user verify that interval between OUT packets matches the Interval used for this test iteration.

9    Have user verify that only 1 TD is executed within each ESIT.

10    Put the Interrupt EP in Timeout Mode.

11    Prompt user to start the analyzer.

12    Put an 8 byte Interrupt OUT on the transfer ring and ring the doorbell.

13    Prompt user to stop the analyzer.

14    Have the user verify that xHC will retry the same transaction Cerr times.  A LS or FS device must schedule the retries in the next ESITs. HS device may retry in the same ESIT. (4.14.3.1#7) for LS, (4.14.3.1#8) for FS, (4.14.3.1#9) for HS.

15    Configure the compliance device for an interrupt IN endpoint.

16    Prompt user to start the analyzer.

17    Put an 8 byte Interrupt IN on the transfer ring and ring the doorbell.

18    Prompt user to stop the analyzer.

19    Have the user verify that xHC will retry the same transaction Cerr times.  A LS or FS device must schedule the retries in the next ESITs. HS device may retry in the same ESIT.

20    Configure the compliance device for an interrupt OUT endpoint.

21    Put interrupt EP in NAK mode.

22    Prompt user to start analyzer.

23    Put an 8 byte Interrupt OUT on the transfer ring and ring the doorbell.

24    Prompt user to stop the analyzer.

25    Have the user verify that xHC will retry same TD in next ESIT. No other transaction is issued for that Interrupt OUT EP until 1 ESIT later. (4.14.3.1#4) for LS, (4.14.3.1#5) for FS, (4.14.3.1#6) for HS.

26    Configure the compliance device for an interrupt IN endpoint.

27    Prompt user to start analyzer.

28    Put an 8 byte Interrupt IN on the transfer ring and ring the doorbell.

29    Prompt user to stop the analyzer.

30    Have the user verify that xHC will retry same TD in next ESIT. No other transaction is issued for that Interrupt OUT EP until 1 ESIT later. (4.14.3.1#1) for LS, (4.14.3.1#2) for FS, (4.14.3.1#3) for HS.

Repeat for LS, FS, HS and SS compliance device.
And repeat for all supported intervals for device speeds.

## TD.5.17 Isochronous Interval Test

This test performs functional testing of isochronous transfer intervals.

**Required Device Resource**

USB 2.0 and 3.0 compliance devices

**Overview of Test Steps**

The test performs the following steps:

1   Initialize and start the xHC.
2   Prompt a user to attach a USB compliance device to a port of the host if it is not attached.
3   Configure the compliance device for an isochronous OUT endpoint.
4   Set bInterval to 1 (endpoint interval 0).
5   Put 4 8-byte transfers on the transfer ring.
6   Prompt the user to start an analyzer.
7   Ring the doorbell for the transfer ring so all 4 TDs will be executed.
8   Prompt the user to stop the analyzer.
9   Have user verify that interval between OUT packets matches the bInterval used for this test iteration.
10  Have user verify that only 1 TD is executed within each ESIT (4.10.3.1#3)
11  Have user verify that number of packets transferred in each service interval is (Max Burst Size + 1) x (Mult + 1) (1.6#1).

**Repetitions**

Repeat for FS, HS and SS compliance device.
And repeat for all supported intervals for device speed and transfer type.
And repeat for all valid max burst size and mult values for device speed.

## TD.5.18 Stopped Transfer Event Test

This test verifies the functionality of xHC when TR Dequeue pointer is pointing to TRB with residual length>0 or to an invalid TRB and when the Endpoint is Stopped.

**Required Device Resource**

USB 2.0 and 3.0 compliance devices

**Overview of Test Steps**

The test performs the following steps:

1   Initialize and start the xHC.
2   Prompt a user to attach a USB compliance device to a port of the host if it is not attached.
3   Configure the compliance device for Bulk Endpoint IN and  OUT endpoint pairs with Maxpacket Size=1K and setting both endpoints in NAK mode.
4   TRB settings (regarding short packet –SPE, etc)
5   Perform a loopback transfer with Transfer Size = 1K, ie., Bulk OUT sending out 1K and Requested Size = 2K, i.e., Bulk IN requesting 2K.

6    Compliance Device will respond with 1K and then starts NAKing the rest of the Data while Host is waiting for the rest of the 1K. (At this time Host TR Dequeue pointer is pointing to 2$^{nd}$ TRB on the IN TD, whose residual length > 0

7    Issue Stop Endpoint Command to Bulk OUT Endpoint.

8    Verify  that the following events are received:
     8.1    xHC will generate a transfer event with completion code set to Stopped (4.6.9.#19)
     8.2    Length field is set to remaining bytes to transfer (1KB) (4.6.9#18)
     8.3    Output context Dequeue Pointer is pointing to the TRB to next be processed.  (4.6.9#24)

9    Verify that Stop Endpoint Command Completion event is received after Transfer Event (4.6.9#29)

10   TR Dequeue pointer is pointing to invalid TRB (Cycle bit Mismatch)
     10.1    Place 2 TRBs of each 1KB each on the bulk OUT Transfer Ring with following settings:
        10.1.1 TD Size to 2KB on the 1$^{st}$ TRB. Cycle Bit = 1(valid), Chain Bit=1
        10.1.2 Set TD Size=1K on the 2$^{nd}$ TRB. Cycle Bit = 0 (invalid), Chain Bit=0

11   Ring the Doorbell.

12   Issue Stop Endpoint Command. Host would have stopped on the 2$^{nd}$ TRB and waiting for it to become valid.

13   Verify that the host generates Transfer Event with Length field set to 0 (4.6.9#21) and Completion code set to Stopped (4.6.9#20)

14   Verify that Stop Endpoint Command Completion is received after Transfer Event (4.6.9#29)

## TD.5.19 Mixed Traffic Test

This test generates concurrent traffic of different types and speeds.

### Required Device Resource

USB 2.0 and 3.0 compliance devices

### Overview of Test Steps

The test performs the following steps:

1    Initialize and start the xHC.
2    Prompt a user to attach a LS, FS, HS and SS compliance devices  below a SS hub.
3    Concurrently run the following loopback tests:
     3.1    TD 5.1 Control Loopback Test on LS compliance device
     3.2    TD 5.2 Bulk Loopback Test on FS compliance device
     3.3    TD 5.3 Interrupt Loopback Test on HS compliance device – subset of full test
     3.4    TD 5.4 Isochronous Loopback Test on SS compliance device – subset of full test
4    Test fails if any of these loopback tests fail.

### Repetitions

Repeat with FS: Interrupt loopback, HS: Isochronous Loopback, SS: Bulk Loopback.
Repeat with FS: Isochronous loopback, HS:  Bulk Loopback, SS: Interrupt Loopback.

## 3.6   Analyzer Tests

## TD.6.01 Bus Traffic Analyzer Test

Testing for SOFs

1    Initialize and start the xHC.
2    Prompt user to attach a USB 2.0 device downstream of the host behind an analzyer.
3    Reset the port if the port is not enabled.
4    Perform an Enable Slot Command.
5    Allocate a buffer for the Output Device Context and initialize it by zero.

6    Write the pointer to the Output Device Context in the DCBAA.
7    Allocate an Input Context and initialize it as follows:
     7.1    Slot Context
         7.1.1   Speed=PORTSC.Port Speed
         7.1.2   Context Entries = 1
         7.1.3   Root Hub Port Number = XXX
     7.2    Endpoint 0 Context
         7.2.1   CErr = 3
         7.2.2   EP Type = 4(Control)
         7.2.3   Max Packet Size = 512(SS) or 64(HS) or 8(FS, LS)
         7.2.4   TR Dequeue Pointer = base address of Transfer Ring
         7.2.5   DCS=1
     7.3    All other Input Contexts are initialized by non-zero values.
8    Prompt user to start analyzer.
9    Perform Address Device Command with the BSR Flag set to 1.
10   Have user verify there was no SetAddress packet present on the bus.
11   Test fails if there was a SetAddress(4.6.5#8)
12   Start Analyzer
13   Halt host controller by writing 0 to USBCMD.RS.
14   Prompt user to stop analyzer
15   Test fails if SOF/ITPs did not stop.  (5.4.2#33)
16   Repeat steps 3-7 to enable communication to the device.
17   Initialize bulk OUT endpoint to always NAK.
18   Submit Bulk OUT request.
19   Prompt user to start analyzer.
20   Issue Reset Device Command
21   Prompt user to stop analyzer.
22   Test fails if bus traffic did not stop (4.6.11#3)
23   Repeat steps 3-7 to enable communication with device again
24   Initialize bulk OUT endpoint to always NAK.
25   Submit Bulk Out request
26   Prompt user to start analyzer
27   Issue Disable Slot Command.
28   Prompt user to stop analyzer
29   Test fails if bus traffic does not stop (4.6.4#2)

### TD.6.02 HCRST Analyzer Test

This test verifies that a host sends a host or warm reset to a device.

**Required Device Resource**

USB 3.0 device

**Overview of Test Steps**

The test performs the following steps:


1    Initialize and start the xHC
2   Prompt user to plug in a SS device behind an analyzer if one is not already plugged in.
3   Verify that the device shows it is connected and enabled.
4   Write 0 to USBCMD.RS to stop the host controller.
5   Prompt user to start USB analyzer trace.
6   Write 1 to USBCMD.HCRST
7   Verify that PLS field is set to RxDetect, Polling or U0 in all the SS PORTSC registers (4.19.1.1.2#1).
8   Prompt user to stop USB analyzer trace.
9   Test fails if user does not see a hot or warm reset on the bus (5.4.1#18)

10  Verify that memory-space registers in Aux power well are set to their default values (4.23.1#2)
11  Verify that memory-space registers in the core power well are set to their default values (4.23.1#3)
12  Verify that OCA flag of PORTSC registers is 0 (5.4.8#11)

## TD 5.10 Stall Error Test

This test performs functional testing of stall error.

### Required Device Resource

USB compliance devices

### Overview of Test Steps

The test performs the following steps.

1  Initialize and start the xHC.
2  Prompt User to attach an analyzer to any of the ports.
3  Prompt a user to attach a USB compliance device to a port if it is not attached.
4  Configure device response for a interrupt OUT endpoint to always return STALL:
    4.1    For LS, FS & HS, set the *bFixedRespMode* field to 01h in the endpoint change data when performing a VEN_Change_EP request.
    4.2    For SS, perform a VEN_Set_STALL_Rate request with the *Packet Count* set to 1.
5  Perform an OUT transfer with a TD which only contains a Normal TRB with IOC = 0 and verify the followings.
    5.1    The xHC generates a Transfer Event (4.10.1#8).
    5.2    The *Completion Code* field is set to Stall Error (4.10.2.1#1).
6  Read the Endpoint Context and verify that the *EP State* field is Halted (4.8.3#1).
7  Prompt the user to take an analyzer trace and verify that no USB traffic to or from the device (10.4.2.1#2)
8  Remove the Compliance device from STALL mode and bring it back to normal mode (use appropriate Vendor command).
9  Prompt user to start capturing a trace.
10  Perform an OUT again and verify that the transfer does not go out on the analyzer trace (10.4.2.1#2).
11

### Repetitions

Repeat test for LS, FS, SS and HS compliance devices.
Repeat for FS, SS & HS compliance device using bulk OUT instead of interrupt OUT.

## TD 5.23 CErr Retries Test

This test performs functional testing of Soft Retry by the 1.0 host. This test also tests various cases under which a USB Transaction is expected after CErr retries attempt.

### Required Device Resource

HS and SS Compliance Devices

1  Initialize and start xHC.
2  Prompt the user to attach Compliance device if it is not attached.
3  Prompt the user to start a trace.
4  Perform Enable Slot.
5  Create Input Context, Slot Context and Endpoint 0 Context.
6  Perform Address Device Command with BSR=0.
7  Perform Configure Endpoint Command to enable Bulk IN Endpoint.
8  Configure the Compliance device for a Bulk IN endpoint to always Timeout when a bulk transfer is issued.
9  For USB2, perform VEN_Change_EP with uTimeout field set to 1.

10  For SS Compliance device, issue command to Compliance device to do a CRC Error (not implemented in the SS compliance device yet)
11  Read the Transfer completion event. The Completion field must be set to USB transaction Error.
12  Reset Endpoint with TSP=1.
13  Ring the Doorbell.
14  Prompt the user to end the trace.
15  Verify the Following on the Analyzer:
  15.1  Test fails if the Analyzer trace captured does not show that the last transaction has been retired CErr times. (4.6.8#5)
  15.2  Test fails if xHC does not preserve USB2 Data Toggle or USB3 Sequence Number
16  Perform the following steps to verify the Host returns a USB Transaction Error if a device returns NYET for Setup TRB:
  16.1  Set up the HS Compliance device to always return a NYET.
  16.2  Perform a GetStatus() Control Transfer by making a Get Status Setup TRB, Data Stage TRB and Status Stage TRB.
  16.3  Prompt user to start the trace.
  16.4  Ring the Doorbell for the Control EP0.
  16.5  After a second prompt user to end the trace.
  16.6  Verify the following:
    16.6.1  The captured trace shows that the Host retried the Setup Stage for CErr times = 3 times.
    16.6.2  Host returns a Transfer Event with Completion Code set to USB Transaction Error.(4.10.2.3#2, 4.10.2.3#1)
  16.7  If the compliance device supports sending an invalid PID (a PID value out of the 4 defined handshake values), perform the following steps:
    16.7.1  Configure the Bulk IN EP on the Compliance device such that the device always returns an invalid PID value out of the range of the 4 defined handshake PID values.
    16.7.2  Prompt the user to start the trace.
    16.7.3  Submit a Bulk IN transfer by queuing a Normal TRB on the Bulk IN Ring.
    16.7.4  Prompt user to Stop the trace.
    16.7.5  Verify the following:
      16.7.5.1  Trace must show that the Host retired the transfer CErr=3 number of times.
      16.7.5.2  Transfer Event is returned with a completion code set to USB Transaction Error. (4.10.2.3#8)

17  Perform the following steps to verify the Host returns a USB Transaction Error if a device returns NAK for Setup TRB:
  17.1  Prompt the user to detach the HS Compliance device.
  17.2  Attach a high-speed single-TT hub to a root hub port or a downstream port of an integrated USB 2.0 hub. No device is attached to downstream ports of the high-speed single-TT hub.
  17.3  Enumerate the high-speed single-TT hub.
  17.4  Attach a full-speed device to a downstream port of the high-speed single-TT hub.
  17.5  Perform the following commands on the device.
    17.5.1  Enable Slot
    17.5.2  Address Device (BSR = 0)
  17.6  Configure Endpoint 0 with the following endpoint settings.
    17.6.1  Endpoint 0
      17.6.1.1  EP Type = Control
      17.6.1.2  Max Packet Size = 512(SS) or 64(HS) or 8(FS/LS)
      17.6.1.3  CErr Count = 3
  17.7  Set up the FS Compliance device to always NAK
  17.8  Issue a GetStatus() Control Transfer to the FS Compliance device by making a Get Status Setup TRB, Data Stage TRB and Status Stage TRB.
  17.9  Prompt user to start the trace.
  17.10  Ring the Doorbell for the Control EP0.
  17.11  After a second prompt user to end the trace.
  17.12  Verify the following:

   17.12.1   The captured trace shows that the Host retried the Setup Stage for CErr times = 3 times.

   17.12.2   Host returns a Transfer Event with Completion Code set to USB Transaction Error.(4.10.2.3#3)

18 Perform the following steps to verify the Host returns a USB Transaction Error if a device returns an NRDY for Setup TRB:

 18.1 Prompt the user to detach the FS Compliance device and attach an SS Compliance device.

 18.2 Perform Enable Slot.

 18.3 Create Input Context, Slot Context and Endpoint 0 Context.

 18.4 Perform Address Device Command with BSR=0.

 18.5 Perform Configure Endpoint Command to enable Bulk IN Endpoint.

 18.6 Configure the Compliance device for the Control Endpoint to always return NRDY when a control transfer is issued.

 18.7 Perform a GetStatus() Control Transfer by making a Get Status Setup TRB, Data Stage TRB and Status Stage TRB.

 18.8 Prompt user to start the trace.

 18.9 Ring the Doorbell for the Control EP0.

 18.10 After a second prompt user to end the trace.

 18.11 Verify the following:

   18.11.1   The captured trace shows that the Host retried the Setup Stage for CErr times = 3 times.

   18.11.2   Host returns a Transfer Event with Completion Code set to USB Transaction Error.(4.10.2.3#4)

 18.12 Stop the Endpoint.

 18.13 Set TR Dequeue Pointer back to the base of Bulk IN Transfer Ring

 18.14 Configure the Bulk IN endpoint on the device to always Timeout the transfer using appropriate vendor defined Command.

 18.15 Prompt user to start the trace.

 18.16 Ring the Doorbell of the Bulk IN transfer ring.

 18.17 After 1 second prompt the user to stop the trace.

 18.18 Verify on the trace that the host did not retry the transfer CErr times. (Assert# TBD)

19 Perform the following steps to verify that the xHC does not retry transactions when babble error is encountered.

 19.1 Configure the compliance device with a loopback pair with a Bulk OUT and Bulk IN endpoints.

 19.2 Perform a 1024-byte bulk OUT transfer.

 19.3 Start analyzer.

 19.4 Insert a TD with length set to 1023-byte into the transfer ring for the Bulk IN endpoint then ring the doorbell.

 19.5 Wait for a Transfer Event for the Bulk IN transfer and verify that the Completion Code is set to Babble Detected Error.

 19.6 Stop analyzer.

 19.7 Verify that the xHC only performs one Bulk IN transaction (no retry was made) (4.10.2.4#2).

20 Perform the following steps to verify that the xHC does not limit the number of retries when CErr is set to 0.

 20.1 Configure the compliance device with a Bulk endpoint Configure the Compliance device for a Bulk IN endpoint to always Timeout when a bulk transfer is issued.

 20.2 Perform Configure Endpoint Command to enable Bulk IN Endpoint with Cerr field set to 0.

 20.3 Start analyzer.

 20.4 Issue a bulk IN transfer.

  Stop analyzer and verify that the xHC continues retry of transaction (6.2.3#15).


**Repetitions**

Repeat test for HS and SS compliance devices.
And repeat for Bulk and Interrupt endpoints.

### TD 5.24 Packets Pending Analyzer Test

This test verifies correct processing of the Packets Pending bit.

#### Required Device Resource

USB 3.0 compliance devices, USB analyzer

#### Overview of Test Steps

The test performs the following steps:

1. Initialize and start the xHC.
2. Prompt a user to attach a USB compliance device to a port of the host if it is not attached.
3. Configure the compliance device for bulk endpoint loopback.
4. Prompt user to start USB analyzer trace.
5. Perform 1 loopback packet of size 8.
6. Prompt user to stop USB analyzer trace.
7. Test fails if PP bit in ACK TP for Bulk IN is 1 (4.14#1)
8. Test fails if PP bit in DP for Bulk OUT is 1 (4.14#1)
9. Perform 2 loopback packets of size 8.
10. Test fails if PP bit in the first ACK TP for Bulk IN or the first DP for Bulk OUT is 0 (4.14#2).
11. Test fails if PP bit in the second ACK TP for Bulk IN or the second DP for Bulk OUT is 1 (4.14#1)


### TD 5.25 SetAddress Test

This test verifies correct processing SetAddress.

#### Required Device Resource

USB 2.0 and 3.0 compliance devices, USB analyzer

#### Overview of Test Steps

The test performs the following steps:

1. Connect a device
2. Initialization of xHC and default endpoint
3. Start analyzer
4. Create control transfer to initiate SetAddress control transfer
   - 4.1 Setup Stage TRB
     - 4.1.1 bmRequestType = 0
     - 4.1.2 bRequest = 5
     - 4.1.3 wValue = 1
     - 4.1.4 wIndex = 0
     - 4.1.5 wLength = 0
     - 4.1.6 Transfer Type (TRT) = 0
   - 4.2 Status Stage TRB
5. Ring Doorbell for default endpoint 0
6. Verify that xHC completes this request with TRB Error Completion (4.6.5#11, 4.5.4.1#1)
7. Stop Analyzer
8. Have tester verify that a SetAddress control transfer was not seen on the bus (4.6.5#12, 4.5.4.1#2)


### TD 5.26 Ping Test

Test USB 3.0 PING functionality.

#### Required Device Resource

USB 3.0 compliance devices, USB analyzer

**Overview of Test Steps**

The test performs the following steps:

1. Connect a SS compliance device and configure it with Isoch endpoints.
2. Perform an Evaluate Context Command with Max Exit Latency field set to a non-zero value.
3. Set U1 timeout to 1 in PORTPMSC register.
4. Start analyzer.
5. Start Isoch data streaming.
6. Stop analyzer.
7. Have a user verify that the xHCI transmits PING packets (4.14.2#2). Also verify that PING packets are transmitted at least Max Exit Latency time before Isoch packets (4.6.7#5).

**Repetitions**

Repeat the steps with different Max Exit Latency values.


## TD 5.27 Resume LFPS Test

Test LFPS resume signaling.

**Required Device Resource**

SS compliance device (with remote wake – currently not supported)

USB 3.0 analyzer

**Overview of Test Steps**

The test performs the following steps:

1. Connect an SS compliance device.
2. Start analyzer.
3. Enable remote wake from the device after TBD ms.
4. Transition the port to U3.
5. Wait for a Port Status Change Event then read the PORTSC register and verify that PLS is set to Resume.
6. Stop analyzer.
7. Verify that the xHC does not respond with LFPS to the device (4.15.2.1#2).


## TD 5.28 Isochronous Analyzer Test

Test Isochronous scheduling.

**Required Device Resource**

USB compliance device

**Overview of Test Steps**

The test performs the following steps:

1. Connect a compliance device and configure it with Isochronous OUT endpoint.
2. Start analyzer.
3. Issue an Isoch TD.
   - 3.1  SIA = 0
   - 3.2  Frame ID = XXX
4. Stop analyzer.
5. Verify that Isoch transaction is executed in the frame specified in the Isoch TRB (4.11.2.3#5).

**TD 5.29 Interrupt on Short Packet with no IOC Test**

**TD 5.30 Force Link PM Accept Analyzer Test**

Test ability to clear FLA for xHC 1.10 hosts

**Required Device Resource**

SS Gen1 and Gen2 devices

**Overview of Test Steps**

The test performs the following steps:

1    Connect and enumerate a SuperSpeed Gen1 device

<u>Set</u>

2    Start analyzer.
3    Set bit 16 (FLA) of PORTPMSC register to 1.
4    Stop analyzer.
5    Verify that an LMP was sent by the host Set Link Function bit 1 set to 1. (5.4.9.1#8)

<u>Clear</u>

6    Start analyzer.
7    Clear bit 16 (FLA) of PORTPMSC register to 0.
8    Stop analyzer.
9    Verify that an LMP was sent by the host Set Link Function bit 1 set to 0. (5.4.9.1#9)

Repetitions:

Repeat with SuperSpeed Gen2 device